ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка методу та програмних засобів підтримки інклюзивної взаємодії на основі жестового інтерфейсу з використанням методів машинного навчання»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
(код, найменування спеціальності)
освітньо-професійної програми «Інженерія програмного забезпечення»
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Андрій ПАНІБРАТОВ
(підпис)

Виконав: здобувач вищої освіти група ПДМ-62
Андрій ПАНІБРАТОВ

Керівник: Оксана ЗОЛОТУХІНА
к.т.н., доцент

Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Панібратову Андрію Івановичу

1. Тема кваліфікаційної роботи: Розробка методу та програмних засобів підтримки інклюзивної взаємодії на основі жестового інтерфейсу з використанням методів машинного навчання

керівник кваліфікаційної роботи Оксана ЗОЛОТУХІНА к.т.н., доцент,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «19»10.2023р. №145.

2. Строк подання кваліфікаційної роботи «29» грудня 2023 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, параметри системи, очікування користувачів до розроблюваного методу підвищення інклюзивності, результати тестів використання системи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Дослідження принципів роботи нейронних мереж.
2. Аналіз штучних нейронів та їх можливостей.

3. Розробка вимог до системи жестового керування, побудованої з використанням нейронних мереж.

5. Перелік графічного матеріалу: *презентація*

1. Мета, об’єкта та предмет дослідження.
2. Обгрунтування вибору нейронної мережі
3. Алгоритм оформлення замовлення з використанням жестового інтерфейсу.
4. Навчання нейронної мережі.
5. Математична можель визначення положення рук.
6. Схема імплементації системи розпізнавання жестів у вже існуючі додатки.
7. Технічні вимоги до системи інклюзивногї взаємодії на основі жестового інтерфейсу.
8. Визначення жесту.
9. Надача переваги руці, що робить розпізнаний жест.
10. Результати тестування.

6. Дата видачі завдання«19» жовтня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	19.10-05.11.23	
2	Вивчення матеріалів для аналізу розвитку нейронних мереж	06.11-12.11.23	
3	Дослідження питання необхідності інклюзивності	13.11-19.11.23	
4	Аналіз особливостей використання нейронних мереж у створенні методів керування	20.11-26.11.23	
5	Дослідження технологій машинного навчання	27.11-03.12.23	
6	Застосування машинного навчання в жестовому інтерфейсі	04.12-10.12.23	
7	Оформлення роботи: вступ, висновки, реферат	11.12-20.12.23	
8	Розробка демонстраційних матеріалів	21.12-29.12.23	

Здобувач вищої освіти _____ Андрій ПАНІБРАТОВ
(підпис)

Керівник
кваліфікаційної роботи _____ Оксана ЗОЛОТУХІНА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 83 стор., 6 табл., 29 рис., 32 джерел.

Мета роботи – підвищення рівня інклюзивності взаємодії «людина-комп'ютер» на основі жестового інтерфейсу за рахунок використання методів машинного навчання.

Об'єкт дослідження – інклюзивна взаємодія «людина-комп'ютер» на основі жестового інтерфейсу.

Предмет дослідження – методи та засоби забезпечення інклюзивної взаємодії «людина-комп'ютер» на основі жестового інтерфейсу. *Короткий зміст роботи*: У роботі було розроблено метод підтримки інклюзивності для користувачів з обмеженими можливостями на основі жестового інтерфейсу з використанням методів машинного навчання. Визначено потенційних користувачів такого методу та їх очікування до нього. Виконано аналіз дизайнерських та програмних рішень, що можуть бути використані для досягнення очікувань користувачів.

Було створено програмне забезпечення для реалізації розробленого методу з метою використання в галузі закладів швидкого харчування. Проведено аналіз результатів використання на основі зібраної статистики використання.

КЛЮЧОВІ СЛОВА: НЕЙРОННІ МЕРЕЖІ, CNN МЕРЕЖІ, МАШИННЕ НАВЧАННЯ, СТАТИСТИЧНА ІНФОРМАЦІЯ, ЖЕСТОВЕ КЕРУВАННЯ, TENSORFLOW МОДЕЛЬ, ІНКЛЮЗИВНІСТЬ

ABSTRACT

Text part of the master's qualification work:
83 pages, 6 pictures, 29 table, 32 sources.

The purpose of the work – to increase availability of technological devices usage during pandemics and for users with limited possibilities through the usage of a gesture interface built with the help of machine learning methods.

Object of research – usage of neural networks to read and interpret gestures.

Subject of research – method of reading and interpreting gestures of the user with the usage of optical filming devices.

Summary of the work: During the completion of scientific work I have researched the question of usage of neural networks in gesture recognition systems with the goal of improving customer inclusivity. Final product, gesture control system, is planned to be used in fast-food restaurants with the goal of increasing inclusivity of customers with special needs and during pandemics. In order to achieve this goal a number of steps were completed.

First step included in itself analysis of the neural networks as a definition, their types and components. Special attention was given to the most basic component of neural networks – artificial neurons. Analysis of artificial neurons included their mathematical explanation, including an example of data processing by a neuron. Acquired data was examined in its role as an instrument of customer inclusivity increasing.

Second step included collection of customer expectations towards the gesture control system and analysis of technological possibilities of terminals that are commonly used by fast-food restaurant chains. Results were collected and organized into a list of requirements towards developed system of gesture control. Based on list of formed requirement, I have determined software and neural networks that will be used during creation of the system.

Third step included development of the planned system. This process includes creation of required gestures list, training of neural networks to satisfy system need, software solution implementation and testing of created system. This part of research also includes calculations of system work correctness and comparison of acquired results towards initial goals and expectations.

Lastly, developed system was tested by different groups of people, including general customers, customers with special needs that may benefit the most from the gesture control system, and customers who can, potentially, have problems with the usage of the system. Results were collected and analyzed, forming a picture of creating potentially successful gesture recognition system aimed at customer inclusivity increase.

KEYWORDS: NEURAL NETWORKS, CNN NETWORKS, MACHINE LEARNING, STATISTIC INFORMATION, GESTURE CONTROL, TENSORFLOW MODEL, INCLUSIVENESS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ЗНМ – згорткова нейронна мережа;

ML – Machine Learning;

ОС – операційна система;

ПЗ – Програмне забезпечення;

ІТ – інформаційні технології.

Framework – інфраструктура програмних рішень, що полегшує розробку складних систем;

API (application programming interface) - це набір чітко визначених методів для взаємодії різних компонентів;

CNN (convolutional neural networks) – згорткові нейронні мережі;

DCNN (deep convolutional neural networks) – глибокі згорткові нейронні мережі;

Кластер (англ. cluster) – група однакових або подібних елементів, зібраних разом або близько розташованих один до одного;

Набір даних (англ. data set) – колекція однотипних даних, що застосовується в задачах машинної обробки даних

ВСТУП

Актуальність магістерської роботи зумовлена підвищенням кількості категорій населення, що потребують особливих умов при використання технічних засобів. До таких категорій можна відности людей з фізичними вадами, такими як проблеми з моторикою, зором, імунітетом, тощо, особи з психологічними потребами та люди похилого віку. Необхідність розробки методу підвищення інклюзивнотсі посилюється і маштабними кризами, що існують на цей день – пандемією covid-19, що створює умови за яких користувачам варто ставитись до гігієни з особливою ретельністю, та війною в Україні, що призводить до підвищення кількості людей з особливими потребами.

Об`єкт дослідження – інклюзивна взаємодія «людина-комп'ютер» на основі жестового інтерфейсу.

Предмет дослідження – методи та засоби забезпечення інклюзивної взаємодії «людина-комп'ютер» на основі жестового інтерфейсу.

Мета роботи - підвищення рівня інклюзивності взаємодії «людина-комп'ютер» на основі жестового інтерфейсу за рахунок використання методів машинного навчання.

Методи дослідження – метод машинного навчання, методи використання нейронних систем, метод штучних нейронів, методи проектування та розробки програмного забезпечення, технології об'єктно-орієнтованого програмування.

Практична значущість результатів полягає в використанні розробленого методу та програмних засобів в якості системи для підвищення доступності технологічних засобів для користувачів з обмеженими можливостями у сфері мереж їжі швидкого приготування.

Для досягнення мети вирішувалися наступні завдання

1) Дослідження нейронних мереж, їх особливостей, типів, складових, задач, що такі мережі здатні вирішувати. Аналіз потенціалу нейронних мереж у сфері підвищення інклюзивності користувачів. Розгляд штучних нейтронів

як головної складової штучних мереж. Дослідження математичного опису штучного нейрона, його вхідних та вихідних даних.

2) Формування вимог користувачів та бізнесів що до системи жестового керування спрямованої на підвищення інклюзивності користувачів у процесі використання терміналів самообслуговування.

3) Створення графічного опису системи жестового керування на основі нейронних мереж з використанням методів машинного навчання.

4) Розробка моделі розпізнавання жестів на основі нейронних мереж.

5) Аналіз результатів взаємодії користувачів з розробленою системою, формування статистичних результатів використання системи особами з особливими потребами.

Для розв'язання поставленої задачі у світі використовуються високий спектр різноманітних методів. Такі методи зазвичай підвищують доступність до застосунку шляхом додавання додаткових опцій до додатку – наприклад режим для людей з проблемами розпізнавання кольору, режим великих літер, можливість модифікувати яскравість, чіткість, гучність, тощо. Проте з ходом технічного прогресу, людство почало задумуватись над використанням відносно нової розробки, що може бути корисною для підвищення доступності додатків для користувача – штучного інтелекту, а саме нейронних мереж.

Серед існуючих методів, що використовують нейронні мережі можна відзначити системи голосового управління, що на даний день є доволі поширеними у світі. Можливість голосового управління існує у багатьох застосунках на сучасних телефонах, використовується у деяких терміналах надання послуг та навіть вбудовуються у керування розумними будинками та авто.

Необхідність використання нейронних мереж зумовлено потужними засобами керування додатками, які вони надають. Існує велика кількість нейронних мереж, що можуть бути натреновані для виконання широкого діапазону функцій, імплементація якого без використання нейронних мереж може бути неможливою у зв’язку з занадто високими вимогами до часу розробки,

об’єму пам’яті який така система займе та проблеми з використанням такої програми на слабких системах.

Необхідність створення методу підвищення інклюзивності саме на основі жестового керування можна пов’язати з перевагами, що така система може надати користувачам. До них можна віднести:

- Відсутність необхідності дотикатися до екрану, що значно підвищить гігієнічність.
- Можливість використання у місцях з високим рівнем шуму, що робить використання голосового керування неможливим.
- Можливість тримати дистанцію від терміналу, що може призвести до зниження рівню розповсюджування інфекційних захворювань

На основі розглянутих переваг було обрано галузь мереж швидкого харчування як середу, в якій така система буде найбільш корисною для підвищення показників інклюзвиності користувачів. Мережі швидкого харчування мають широкий ринок користувачів, що включає в себе людей з різними особливими потребами. Велика кількість відвідувачів викликає необхідність у підвищених рівнях гігієни у таких закладах. Ця проблема посилюється під час пандемічної небезпеки – що можна було побачити під час пандемії covid-19.

З урахуванням особливостей ситуації в Україні, важливість імплементації системи тільки зростає. Через війну, складний економічний стан, демографічні особливості та поширеність інфекційних захворювань, кількість людей з особливим потребами знаходиться у стані активного рост. Це вказує на необхідність створення систем, які здатні підвищити доступність технологічних приладів для різних груп користувачів.

1 ОГЛЯД НЕЙРОННИХ МЕРЕЖ

1.1. Нейронна мережа як поняття

Нейронні мережі – це надзвичайно потужний інструмент для вирішення комплексних задач, що потребують виконання великої кількості роботи пов’язаної з управлінням, класифікацією, розрахунками та розпізнаванням даних. Загалом, нейронні мережі можна назвати продвинутою формою штучного інтелекту, що оперує у спосіб, схожий на роботу мозку людини.

Головна перевага над звичайним штучним інтелектом полягає у можливості виконання нелінійних завдань. У той час, як більш традиційний штучний інтелект здатний працювати лише з простими задачами, проблеми у яких знаходяться у формі “задача-рішення” (так звані “лінійні завдання”), нейронні мережі здатні працювати з набагато більш комплексними задачами, що можуть мати надзвичайно велику кількість вхідних даних – і при цьому видавати вірний результат з високою вірогідністю.

Прикладами таких завдань можна назвати:

- Сучасні розумні будинки, в яких нейронні мережі здатні самостійно аналізувати показники з датчиків розташованих у середині, та приймати рішення на основі цих показників.

- Розумні авто, що завдяки аналізу інформації з камер автомобілю здатні самостійно вести автомобіль по дорозі у режимі автопілоту, реагувати на можливі перешкоди, активувати підсистеми авто за необхідності, допомагати водію та автоматично паркуватися у складних ситуаціях.

- Аналітичні застосунки, здатні прогнозувати курс ринку, зріст чи падіння цін, шанс успіху нового товару на ринку, тощо.

- Чат-боти, здатні симулювати процес спілкування з реальною людиною на основі інформації отриманою від аналізу справжніх співбесід, застосунки для

автоматичного написання історій, перевірки правопису, перекладу на іноземні мови.

Можливість отримання позитивних результатів у таких прикладах, що мають десятки тисяч вхідних даних, досягається лише за допомогою нейронних мереж, натренованих для роботи у своїй власній сфері. Тренування нейронних мереж схоже на тренування мозку звичайної людини – шляхом надавання прикладів, що можуть як мати лише саме завдання, так і завдання з прописаними результатами (навчання з вчителем) [1].

1.2 Математичне пояснення роботи нейронних мереж

Як можна зрозуміти з минулого розділу, можливість виконання спеціалізованих комплексних завдань можливо за допомогою навчання нейронних мереж на основі тестових даних. Проте для повного розуміння самого процесу, необхідно звернути увагу на найменшу частину нейронної мережі, з якої мережа, власне, і складається – штучний нейрон.

Аналог нейрону, що є складовою біологічного мозку - штучний нейрон, є схожим за своєю побудовою та функцією до свого органічного варіанту. Функціонування штучного нейрону можна описати за наступною формулою:

$$y = \Psi(\varphi(w, x)) \tag{1.1}$$

Де y - вихідне значення нейрону, ψ - функція активації, φ - дискримінантна функція, w – порогове значення системи, x - вектор вхідного сигналу (сигналів). Порогове значення визначається пам'яттю системи, а отже нейрон можна розглядати як примітивний обчислювальний пристрій, процесор.

Дискримінантні функції та функції активації залежать від задачі, поставленої перед системою. В залежності від задачі, у якості дискримінантної функції використовуються зважена сума, зважений добуток та зважена відстань.

Функція активації є важливою складовою системи, що у великій мірі визначає направлення обрахунку. Вона розраховує вхідний рівень нейрону, що в подальшому передається іншим нейронам. В ситуації, коли обчислення будуть вестись над розпізнаванням руки як об’єкту та жестів, що рука виконує, нам знадобляться такі функції:

1) Порогова функція, або функція одиничного стрибка. Така функція застосовується, коли нейрон працює з задачею, що може мати лише два значення – максимальне та мінімальне, бінарними задачами, так/ні дилемами, що можуть мати лише чітку, визначену відповідь.

$$y = \begin{cases} 0, S < \theta \\ 1, S \geq \theta \end{cases} \quad (1.2)$$

Де y - вихідне значення функції, S – це середньозважене значення суми, яке було отримане під час першого етапу обчислення вихідного значення нейрону, θ – поріг спрацювання функції, її задана точність.

2) Сигмоїдальна функція, що є нелінійною функцією з насиченням. Вона використовується, коли відповідь не можна описати чіткою відповіддю, проте відповідь може бути описана невід’ємним числом. Вона здатна вносити нелінійність у роботу системи, змушуючи нейронну мережу перевіряти варіанти схожі на лінійну відповідь системи. При цьому, ця функція не вносить значних збоїв чи помилок у роботу системи, адже відповіді будуть перевірені нейронами, що знаходяться далі у системі – де хибні відповіді будуть відкинуті.

$$y = \frac{1}{1+e^{-2\alpha S}} \quad (1.3)$$

Де y - вихідне значення функції, S - середньозважене значення суми, отримане під час попереднього етапу обчислення вихідного значення нейрону, α

– параметр варіативності відповідей функції, значення що відповідає за її крутизну на графічному представленні.

Наведені вище функції є головними інструментами роботи нейронів під час визначення руки на зображенні, екранній формі чи відео потоці. У окремих випадках можуть бути використані такі функції як лінійна функція з константою, що використовується для варіації значень без насичення та порогова біполярна, що може приймати негативні значення у якості нижньої межі.

3) Біполярна функція гіперболічного тангенсу, що здатна працювати з кутами об’єктів. Хоча така функція не має особливого застосування під час пошуку самої руки, вона є надзвичайно важливою під час розпізнання жестів на основі положення пальців руки.

$$\Psi(x) = \tanh(Kx) = \frac{e^{Kx} - e^{-Kx}}{e^{Kx} + e^{-Kx}} \quad (1.4)$$

Де К – це константа, визначена на основі попередніх операцій розрахунків проведених нейронами чи, як в нашому випадку, визначена заздалегідь у даних, що надаються програмі.

У випадку з необхідністю розрахунків положення пальців відносно зап’ястя, наступні дані вважаються базовими для початкової точки пальцю (положення з відкритими пальцями):

- Для великого пальцю: 150
- Для вказівного пальцю: 120
- Для середнього пальцю: 100
- Для безіменного пальцю: 80
- Для мізинця: 60

Так як людина має дві руки, що є дзеркальними одна до одної, варто не забувати про можливість появи від’ємних при використанні іншої руки.

Для визначення жесту на основі положення пальцю нам знадобляться і інші точки перевірки куту, 1 для великого пальця, та 2 для інших, відповідно кількості точок, де людина може зігнути свій палець.

Загалом, роботу нейрону у будь-які ситуації можна описати графічним виглядом, що є вірним для проведення будь-якої можливої операції. Це графічне зображення можна побачити на рисунку 1.1

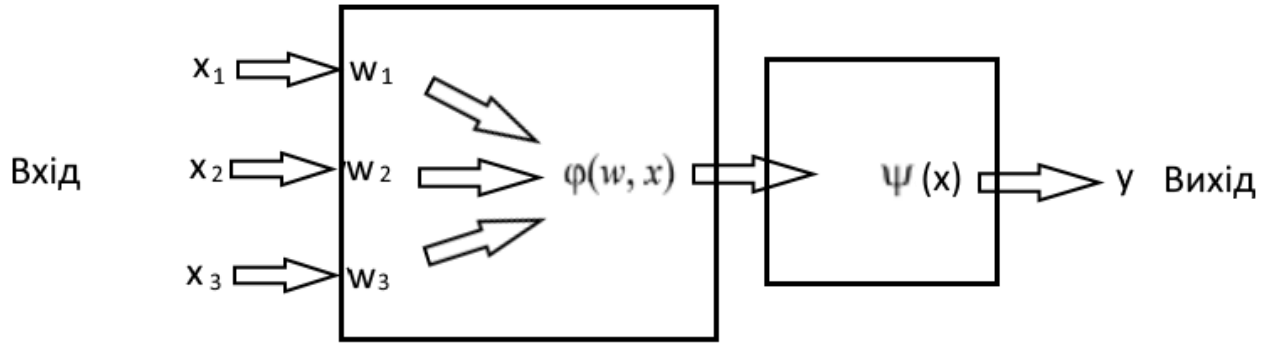


Рис. 1.1 Схема роботи нейрону

Як приклад, можна розглянути застосування штучного нейрону з використанням сигмоїдальної функції. Беручи x як вхідні дані системи, E – функцію втрати точності, а w - як ваги у системі, ми можемо виконати одну з головних задач, що стоїть перед розробниками нейронних мереж – зменшити кількість помилок та неточностей у системі. Для цього буде використано ланцюгову функцію.

$$f(x) = (g * h)(x) = g[h(x)] \quad (1.5)$$

$$f'(x) = g'[h(x)] * h'(x)$$

Беручи суму як S = x1W1 + x2W2, а в якості активатора – сигмоїдальну функцію A = $e^x/(1 + e^x)$. Ця функція обрана, адже очікувану кількість помилок можна зобразити у виді сигмоїда.

$$\frac{dE}{dW1} = \frac{dE}{dA} * \frac{dA}{dS} * \frac{dS}{dW1} \quad (1.6)$$

$$\frac{dE}{dW2} = \frac{dE}{dA} * \frac{dA}{dS} * \frac{dS}{dW2} \quad (1.7)$$

Використання цього правила можливе за умови виконання зворотнього розповсюдження, коли нам вже відомо вхідні дані, ваги, та їх вихідні дані. Результат дасть число помилки, в той час, як наша мета на цьому кроці – знайти таке W, при якому помилка буде якомога меншою [2]. Для досягнення цієї мети можна використати:

$$W1new = W1old - \eta \frac{dE}{dW1} \quad (1.8)$$

$$W2new = W2old - \eta \frac{dE}{dW1}$$

Знаходження точності – лише один з прикладів можливостей нейронних мереж. Саме пропуск інформації через штучні нейрони і дає можливість нейронній мережі видавати кінцевий результат. Після проведення обрахунку вхідних даних, кожен нейрон передає результати далі через так званий синапс – точку з'єднання нейронів між собою.

Кількість штучних нейронів у мережі залежить від кількості шарів, проте загалом вона може бути визначена таким чином:

Вхідний шар – кількість штучних нейронів відповідає кількості показників, що мають вхідні дані.

Вихідний шар – регресивні моделі мають 1 нейрон на виході, класифікаційні моделі мають однакову кількість нейронів з кількістю показників вихідних даних.

Приховані шари – кількість нейронів у таких шарах назвати важко, проте загалом вона відповідає одному з трьох наступних правил, розташованих за частотою використання [3]:

1) Кількість нейронів у кожному шарі менше ніж 2/3 сумісної кількості нейронів в вхідному та вихідному шарі.

2) Кількість нейронів знаходиться між кількістю нейронів у вхідному та вихідному шарів.

3) Кількість нейронів у кожному прихованому шарі становить ½ від нейронів у вхідному шарі.

Як можна побачити, загалом кількість нейронів у прихованих шарах є меншою від кількості нейронів у вхідному шарі, проте у цього правила є виключення – коли мова йде про надзвичайно комплексні нейронні мережі, наприклад автопілот розумного авто. В таких мережах приховані шари можуть мати більше нейронів, виконуючих обчислення ніж кількість нейронів, що працюють з вхідними даними.

1.3 Класифікація нейронних мереж

Нейронні мережі, хоч і мають схожу базову структуру, використовуючи нейрони для проведення усіх розрахунків та виконання логічних дій, відрізняються за своєю структурою, кількістю логічних шарів, методом навчання, налаштуванням ваг та типом вхідної інформації.

За характером навчання, нейронні мережі можна поділити на:

- Навчання без вчителя: Навчання без вчителя використовує навчальну базу, що складається лише з вхідних даних певної форми. Процес навчання відповідає за виділення статистичних властивостей навчальної множини і групування подібних вхідні векторів у класи чи кластери. Через принцип роботи навчання без вчителя, використання достатньо близьких вхідних векторів видасть однакові виходи.
- Навчання з вчителем: Навчання з вчителем передбачає наявність навчальної бази, що містить правильні, визначені приклади вхідних та вихідних даних. Вхідні дані, та відповідючі їх вихідні дані називаються парою. Мережа навчається на певній кількості таких пар. Для мінімізації похибки значення ваг

змінюються в залежності від алгоритму навчання. Приклади навчальної множини пред'являються послідовно, обчислення похибки і ваги підлаштовуються для всіх прикладів використаних пар. Завершення такого тренування та суму його результатів називають епохою. Мережа проходить кроки навчання до моменту, коли похибка стане мінімальною.

- Навчання з підкріпленням: Навчання з підкріпленням – метод схожий на навчання з вчителем, проте, на відміну від навчання з вчителеме, використовує штучне середовище для оцінювання ваг, а не оцінки отримані від людей. Таким чином вчителем є середовище, його власна модель. Сигнали, отримані під час процесу навчання називають сигналами підкріплення. Деякі правила підкріплення базуються на неявних вчителів, що робить їх схожими до навчання без вчителя.

За налаштуванням ваг мережі можна поділити на:

- Мережі зі статичними зв’язками: вагові коефіцієнти нейронної мережі встановлюються виходячи з умов задачі, ще на етапі до завантаження вхідних даних у систему.

- Мережі з динамічними зв’язками: Для таких мереж налаштування ваг відбувається не в процесі розгляду умов задачі, а в процесі роботи над вихідними даними. Значення ваг можуть змінюватися у процесі роботи, залежно від вхідних даних.

За типом вхідної інформації мережі діляться на:

- Аналогові: усі вхідні дані надходять у формі дійсних чисел, без інших обмежень. Такі системи мають більше можливостей і з ними легше працювати, проте вони, зазвичай, більш вимогливі до обчислювальних можливостей технічного забезпечення.

- Двійкова: усі вхідні дані представлено у формі нулів та одиниць. Така система більш обмежена у своїх можливостях, проте працює швидше.

За шляхом з’єднання мережі діляться на:

- Цілісні: усі нейрони у мережі пов’язані між шарами. Зазвичай такий шлях поєднання нейронів зустрічається у простіших мережах.

- Частково поєднані: не всі нейрони з одного шару мають з’єднання з нейронами наступного шару, зазвичай формуючи пари з’єднаних нейронів або групи.

Тип нейронної мережі є доволі широким поняттям, а тому в роботі буде розглянуто саме тип, найбільш ефективний для аналізу екранних зображень та пошуку об’єктів на них [4].

1.4 Згорткова нейронна мережа

1.4.1 Головна ідея методу

Згорткова мережа, також відома як Convolutional Neural Networks (CNN) – це однин з типів сучасних нейронних мереж, що з’явився відносно нещодавно, проте вже набрав шаленої популярності в усіх сферах життя. Якщо розглядати нейронну мережу як мозок, то CNN мережі нагадують зорову кору у ньому.

Такий тип мережі працює, поділяючи вхідне зображення на сектори, кожному з яких надається свій коефіцієнт відповідності об’єкту який мережа шукає. Приклад роботи такої мережі можна побачити на рисунку 1.2.

Кожен нейрон у мережі типу CNN відповідає за розпізнавання окремих форм, елементів зображення чи деталей, які є частиною загального об’єкта, який система намагається знайти. Кожній області аналізу надається своя власна вага, яка позначає, наскільки ймовірним є знаходження об’єкту у цій області [5].

Такі системи здатні до самонавчання, покращуючи базовий результат з кожним вірним опрацюванням зображення проведеним у режимі тренування. Проте простіші види CNN мереж спираються на вже проведені раніше аналізи зображень, зазвичай отримані в режимі навчання з вчителем, де шуканий об’єкт є чітко виділеним для нейронної мережі за допомогою встановлених ваг.

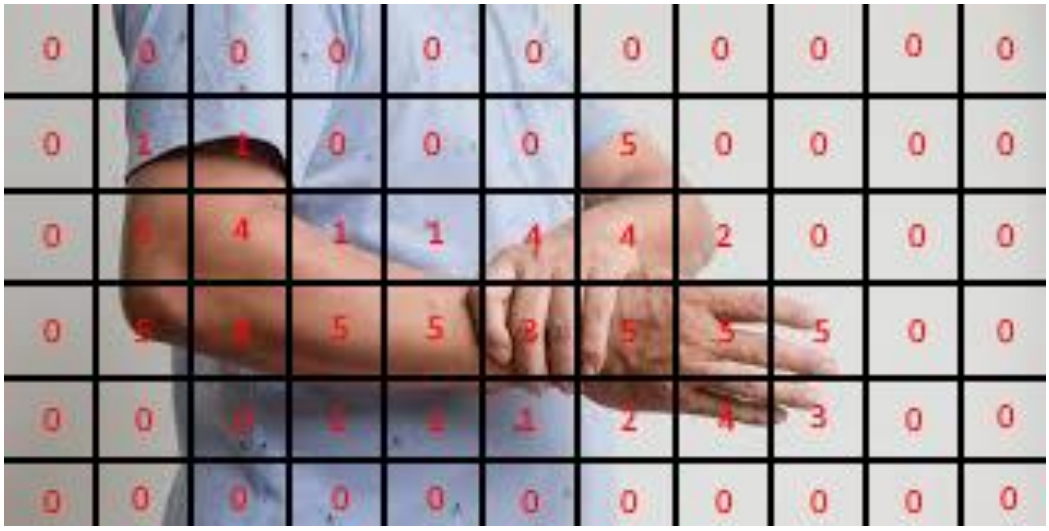


Рис. 1.2 Матриця ймовірностей наявності руки з результатами

1.4.2 ReLu та Pooling

Для роботи сучасних CNN систем використовують сімейство функцій активації ReLu. Найбільш використовуваними функціями з цього сімейства є функції згортки та функція пулінгу.

На прикладі пошуку руки можна провести наступний алгоритм роботи: для початку зображення розділяється на квадрати та береться зона аналізу розміром 12x12 у якій проводиться аналіз зображення. Присвоєне значення вказує на схожість блоку до об’єкту який ми шукаємо. Це визначається, завдяки деталям присутнім у блоці, наприклад: кольори власні шкірі людини, яке направлення об’єктів домінує в кожному - горизонтальне, вертикальну або одну з діагональних, чи відповідають пропорції об’єкту у блоці пропорціям руки людини [6].

На виході отримується кілька масивів ймовірностей, які є простими ознаками наявного образу руки на зображенні. При знаходженні якостей, наявних у руці людини, алгоритм знову вибере зону 12x12 з квадратів, значення яких було присвоєне на основі аналізу минулого набору даних з використанням функцій середнього пошуку або пулінгу, в залежності від умов використання. Загальний алгоритм роботи CNN систем зображено на рисунку 1.3.

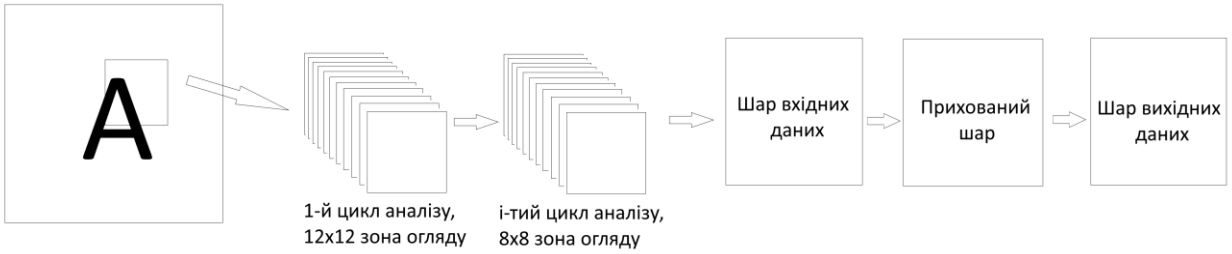


Рис. 1.3 Алгоритм роботи нейронної мережі виду CNN

Методи пулінгу та середніх значень дозволяють знизити зону зображень, що потребують аналізу між ітераціями.

Метод пулінгу базується на використанні вікна аналізу (фільтру), що розділяє отримані результати точності на блоки та бере елемент з найбільшим значенням у кожному із них, збираючі лише ті частини зображення, де вірогідність знаходження об’єкту який ми шукаємо є найбільшою. Простими словами, система орієнтується на даних, що ми шукаємо, ігноруючи все інше.

Цей метод має наступні переваги:

- Зменшення параметрів, які має використовувати модель за допомогою down-sampling (зменшення масштабу)

- Підвищує точність результатів, особливо якщо зображення постійно змінюється.

- Наочність результатів у процесі тренування мережі

- Можливість використання для аналізу графічних та звукових вхідних даних.

Розмір вихідного зображення з використанням методу пулінгового методу ReLu можна розрахувати за формулою 1.4.

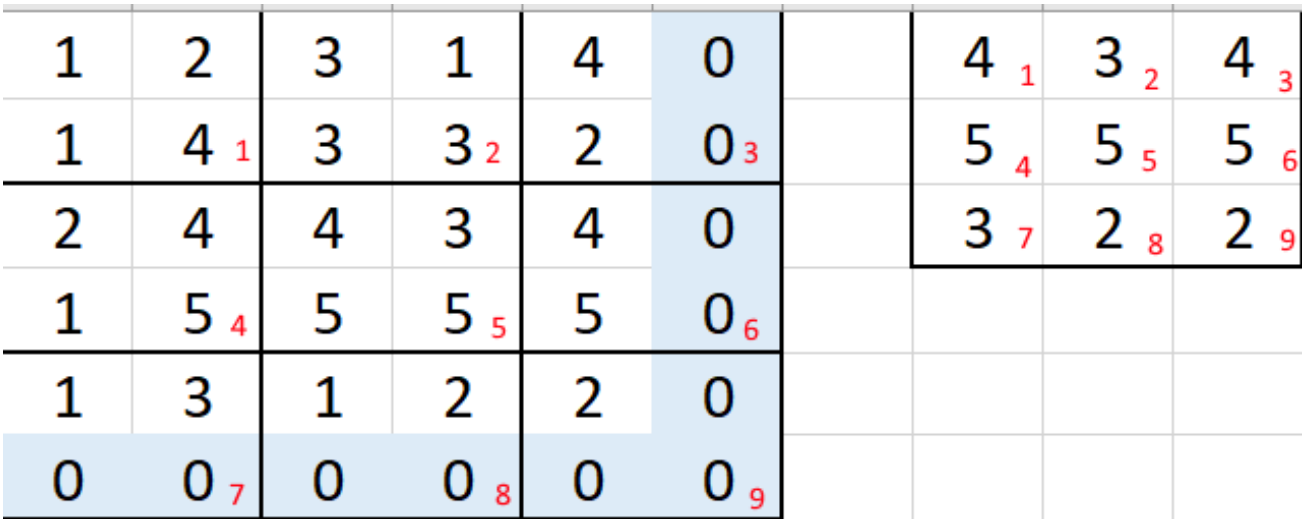


Рис. 1.4 Алгоритм пулінгу, блакитні клітини - паддінгові

$$W_{out} = \frac{(W_{in} - F + 2P)}{S} + 1 \qquad (1.9)$$

Де W(in) — ширина входу, F — розмір фільтру, P — паддінг, S — крок роботи методу. Під паддингом розуміється розширення зони аналізу шляхом додавання фіктивного горизонтального та вертикального ряду клітин, що перетворює зону аналізу 6x6 у 7x7. Ці клітини мають значення 0 і необхідні для коректної роботи методу, наприклад у ситуації коли пулінгове вікно 3x3 має працювати з кроком 5,5.

Крок – це частина зображення, яка аналізується у поточний момент. Наприклад, на рисунку 1.5 показано крок 5,5 для пулінгового вікна 2x2, де синім виділено минуле вікно, червоним – поточне [7].

1	2	3	1	4	0
1	4	3	3	2	0
2	4	4	3	4	0
1	5	5	5	5	0
1	3	1	2	2	0
0	0	0	0	0	0

Рис 1.5 Приклад кроку методу ReLu з використанням пулінгу, блакитні клітини – паддингові

Метод згортки використовується у ситуаціях коли системі не потрібно чітко розуміти пропорції об’єкту, а лише бачити його приблизний силует. Роботу методу зображено на рисунку 1.6.

2	3	1		4	6
0	3	5		2.5	7.5
0	2	5			

Рис 1.6 Приклад роботи методу згортки

Обчислення виконуються за формулою 1.10, де y – значення вихідної матриці, а x – клітинки вхідної матриці, що знаходиться у вікні аналізу. Використання цього методу є важливим не тільки для полегшення роботи

нейромережі, а й для можливості отримання результатів загалом. Як і в методі пулінгу, система проходить декілька ітерацій методу згортки, в залежності від поставленої задачі.

$$y_1 = 0.5 * (x_1 + x_2 + x_3 + x_4) \quad (1.10)$$

Від кількості ітерацій аналізу зображення, розміру вікна аналізу та кількості зон, на яке ділиться зображення, залежить якість вихідних даних та час, який система витрачає на аналіз зображення. На результат також впливають комплексність об’єкту що ми шукаємо, де велика кількість елементів не завжди є поганою для якості (адже кожна деталь може використовуватися нейронною мережею під час аналізу зображення) та кількість блоків, на яке ділиться зображення. Загалом, висока кількість ітерацій призводять до кращої якості аналізу зображень, ніж високий розмір зон, що ми аналізуємо у кожному з методів. Сучасні CNN системи не використовують вікна аналізу більше за 24x24, і навіть такий розмір присутній лише у надзвичайно рідкісних випадках. Звичайний розмір вікна знаходиться у проміжку 2x2 та 6x6.

1.5 Нейронні мережі як засіб підвищення інклюзивності

Використання нейронних мереж для підвищення інклюзивності має широкий спектр можливих застосувань. Завдяки широким можливостям нейронних мереж у галузях зчитування та аналізу інформації, можна відзначити такі можливі напрямки:

- Аналіз даних використання технологічного засобу користувачами з обмеженими можливостями для визначення показників, в яких такі користувачі мають проблеми з користуванням. Наприклад – швидкість використання, швидкість розуміння інформації на дисплеї, можливість користування функціоналом технічного засобу, тощо. Цей спосіб є популярним і широко

використовується компаніями по всьому світу для підвищення рівнів інклюзивності.

- Системи голосового керування, що спрямовані на допомогу користувачам з проблемами моторики. Такі системи використовують голос користувача, що є простим у розумінні методом керування технічними засобами. Такі системи також є поширеними у світі, проте мають обмеження у вигляді можливості використання лише за умов часткової тиші у приміщенні.

- Системи керування жестами, що спрямовані на допомогу користувачам з слабким імунітетом та для підвищення рівню гігієни під час епідемічної загрози. У той час, як керування жестами є складнішим у розумінні для людини, воно не залежить від зовнішніх умов та не потребує від користувача відкривати себе до можливості захворювання. Системи цього типу підвищення інклюзивності не є поширеними, маючи високий потенціал для можливого вдосконалення.

Нейронні мережі можуть бути використані і інших шляхах, що можуть підвищити доступність технологічних засобів для населення, проте наведені типи є найбільш поширеними у середі надання послуг [8,9].

1.6 Можливі проблеми у використанні нейронних мереж

Під час роботи з нейронними мережами варто звертати увагу на недоліки, що вони несуть з собою – як технологічні, так і соціальні. Метод, розроблений у цій роботі має підвищувати інклюзивність певних груп користувачів, але це не може бути досягнуто за рахунок зменшення зручності користування технологічним засобом [10].

1.6.1 Технічні обмеження

Під час роботи над CNN мережею варто пам’ятати, що вона, згідно початковому плану, скоріш за все буде використовуватись на малопотужних терміналах мереж швидкого харчування та кафе.

Для кращого розуміння ситуації варто вказати, що 50-шарова згорткова мережа вимагає процесор Intel Core i7-7700 CPU (чи аналог), 16 GB оперативної пам’яті відеокарту кращу чи аналогічну 8 GB NVIDIA GTX 1080 GPU. У той же час, термінал Self-Checkout Kiosk 23.6, що можна побачити у мережах KFC та McDonalds має лише 2 GB оперативної пам’яті та процесор Rk3288, що не тільки значно менш потужний, але й ще й вимушений працювати у ролі вбудованої відеокарти.

Як можна побачити, потужні CNN нейронні мережі значно перевищують можливості середньостатистичного терміналу – проте це не є проблемою, яку неможливо вирішити. Загалом, можна виділити два шляхи, якими можна піти для вирішення проблеми:

1) Використовувати слабкішу нейронну мережу з меншою кількістю шарів.
2) Розгорнути нейронну мережу на сервері, поєднуючому термінали, звідки вона зможе приймати вхідні сигнали від терміналів, аналізувати дані, та повертати відповідь до них.

1.6.2 Недовіра до нейронних мереж

Недовіра користувачів до штучного інтелекту та, як результат, і до штучних мереж, є серйозною проблемою на шляху до імплементації таких мереж з метою полегшення доступу до технічних засобів. Адже якщо користувачі сформують негативний погляд на розроблюваний метод, вони можуть перенести свої негативні погляди на людей, що потребують жестовий інтерфейс для ефективного користування технічним засобом. Це може призвести до:

- Морального тиску на громадян з обмеженими можливостями
- Зниження кількості користувачів, охочих використовувати метод
- Підвищення невдоволення користувачів, навіть якщо метод працює вірно
- Репутаційної шкоди для закладів, що використовують метод і, як наслідок, зниження зацікавленості зі сторони надавачів послуг.

Негативне ставлення до нейронних мереж є відносно поширеною тенденцією, і це можуть підтвердити статистичні дослідження проведені у 2019-2023 роках. Так, наприклад, дослідження Pew Research Center проведене на території США показало наступні результати, продемонстровані на рисунках 1.7, 1.8, 1.9 [11]:

Рис. 1.7 Реакція публіки на імплементацію нейронних мереж у сканери обличчя

Більшість користувачів, а саме 46%, негативно ставились до використання нейронних мереж у системах, спрямованих на ідентифікацію обличчя. Цей результат було здебільшого викликано турбуванням що до конфіденційності користувачів, боязні постійного знаходження під наглядом камер та боязні втрати робочих місць.

У той же час, використання нейронних мереж у системах, здатних розпізнавати перешкоди на шляху авто та автоматично пілотувати його, показує зовсім іншу картину:

Рис. 1.8 Реакція публіки на імплементацію нейронних мереж у розумних авто

44% користувачів позитивно ставляться до імплементації систем такого плану, відзначаючи значне підвищення зручності. Не зважаючи на поточні турбування що до безпечності таких систем, відсоток користувачів з негативним ставленням зменшився до 26%, що є набагато більш допустимим порогом недовіри.

Цікаво, що думка користувачів показувала незначну варіацію при розгляді як повного керування авто нейронною системою, так і при використанні штучного інтелекту у якості асистента водія.

Варто зазначити ще один напрямок дослідження – використання нейронних мереж у пошуку хибної інформації. Згідно з результатами дослідження, лише 31% користувачів мав позитивне ставлення до імплементації штучних мереж цим шляхом, у той час, як 39% користувачів мали негативне ставлення, вважаючі що штучний інтелект не здатний надати достатньо високого рівня надійності для використання без нагляду системи людиною.

Рис. 1.9 Реакція публіки на імплементацію нейронних мереж у аналізі хибної інформації

З дослідження можна зробити наступні висновки:

- Користувачі здатні закрити очі на проблеми системи якщо вона підвищує зручність використання продукту

- Думка користувачів що до системи значно погіршується, якщо на їх погляд вона не здатна надавати достатньо надійний результат.

- Користувачі не люблять системи, що можуть порушувати їх право на конфіденційність.

Ці питання варто ретельно розглянути під час планування системи та взяти їх до уваги, для створення максимально інклюзивного продукту.

2 ДОСЛІДЖЕННЯ ВИМОГ ДО СИСТЕМИ

2.1 Визначення потенційних користувачів системи

Для поставлення вимог до системи, спочатку необхідно визначити базу користувачів, зацікавлених у розроблюваному методі. Групи таких користувачів можна визначити розглянувши сильні сторони системи.

Можливість дистанційного керування без необхідності відкривати обличчя (наприклад, знімати медичну маску) надає можливість працювати з технічним забезпеченням, не піддаючи себе небезпеці інфекційних захворювань. Така перевага, в першу чергу, важлива для користувачів з послабленим імунітетом, що включає у себе осіб похилого віку [12].

Наведена перевага може бути важлива і для звичайних користувачів у період епідемій, а тому система здатна бути корисною для звичайного, повсякденного користувача. Такі користувачі можуть бути зацікавлені системою і зі сторони її новизни та незвичності, за умови, що вона буде ефективно працювати на рівні з стандартною тач-скрін системою керування.

Додаткова група людей, які можуть бути зацікавлені у жестовій системі керування – люди з захворюваннями шкіри, що не можуть використовувати стандартні методи керування з гігієнічних міркувань [13].

Гігієнічна сторона системи має доволі зрозумілу вимогу – кількість дотиків до технічного засобу має дорівнювати 0. Це буде важливо для користувачів з імунними проблемами. Система, на додачу, має бути легкою в засвоюванні для користувачів похилого віку, що можуть мати проблеми з розумінням сучасних технологій [14].

Після визначення груп клієнтів, варто проаналізувати, чи які проблеми можуть виникнути у визначених груп людей на додачу до стандартних проблем, що містить метод жестового керування, а саме швидкодії та вірності результатів, надавши опитуваним запропонувати свої власні ідеї для кращого розуміння вимог.

2.2 Визначення бізнесів, зацікавлених у системі

Бізнесом, на який орієнтується система, було визначено мережі закладів харчування, що використовують термінали для можливості замовлення харчів. В своїй суті, розроблена мережа може бути використана будь-якими закладами, що використовують термінали, комп'ютери чи інші технічні засоби для оформлення замовлення, проте мережі закладів харчування обрані як головний пріоритет з наступних причин:

- Заклади такого типу підпадають під категорію місць особливої небезпеки отримання захворювання під час карантину. Імплементація жестового інтерфейсу дозволить зменшити ризик передачі захворювання [15].

- Термінали, що використовуються для оформлення замовлення – це пристрій, який використовують більшість клієнтів які відвідують заклад. Під час карантину, за неможливості отримання замовлень через персонал, такі термінали і взагалі стануть обов'язковими для використання зі сторони клієнтів закладу охочих зробити замовлення [16].

- Відсутність значного ризику при помилкових діях системи. На відміну від секторів, що вимагають точність роботи 99.99%+ (наприклад медичного), незначні помилки у роботі системи оформлення замовлення не несуть значної шкоди та є легкими у подоланні зі сторони клієнтів [17].

- Бізнеси такого типу є достатньо популярними місцями відвідування для визначеної користувацької бази.

- Ця галузь активно модернізує заклади терміналами, що є чудовим моментом для імплементації системи.

Розглянуті вище причини є достатньо вагомими для спеціалізації саме на мережах швидкого харчування, проте варто розглядати можливість розширення і в інші комерційні сфери, наприклад – автоматичні кіоски, термінали обслуговування у супермаркетах, тощо [18].

2.3 Збір інформації що до побажань користувачів

Для розуміння вимог користувачів до продукту недостатньо лише припустити їх вимоги. Для отримання більш чітких вимог для системи, було проведено опитування користувачів, як з груп людей на які орієнтується система, так і звичайних повсякденних користувачів, що можуть бути зацікавлені у роботі з системою. Для отримання більш вірогідних результатів було взято вибірку з 80 користувачів, що надали детальні відповіді, та результати з опитувань у соціальних мережах проведені в групах, що асоціюються з потенційною користувацькою базою тим чи іншим чином, числом 900 відповідей. Для проведення таких опитувань було використано соціальні мережі Reddit та Discord.

2.3.1 Вимоги до швидкодії

Результати опитувань що до очікувань до швидкодії можна побачити на таблиці 2.1. Таблиця виділяє 3 загальні групи користувачів, згруповані за загальними очікуваннями. До уваги береться саме відповідь про “задовільний” рівень швидкодії, адже ідеальний рівень складає менше 1-ї секунди і буде складним у досягненні за умов обмежень до технічного забезпечення.

Для визначення точки, на яку буде орієнтуватися система, було взято модальне значення, а саме час роботи у 3 секунди. До досягнення цього порогу система має надати коректний результат визначення жесту щоб вважатися успішною. У разі недосягнення позитивного результату у встановлений часовий проміжок, результат роботи системи можна вважати як неточний. Неточним результатом також можна вважати визначення невірного жесту.

Таблиця 2.1

Очікування що до швидкодії

Час роботи, с.	Звичайні користувачі	Користувачі похилого віку	Користувачі з фізичними вадами
1 чи менше	72	2	13
2	80	7	25
3	167	8	81
4	105	28	83
5	64	25	59
6	52	5	32
7	1	6	30
8	0	3	17
9	0	3	15
10	0	2	5
Загалом	541	89	350

2.3.2 Вимоги до точності

Вимоги до точності отриманих результатів можна виміряти наступною відповіддю на питання: чи розпізнала система жест у поставлений проміжок часу? (в розроблюваній системі – 3 секунди). Очікування до точності роботи наведено на таблиці 2.2, відповідь 100% не була включена в проведені опитування, адже така точність є майже недосяжною.

Більшість користувачів назвали точність у 99% оптимальною для зручного користування системою. Це доволі високий показний, отримати який може бути проблематично, проте варто зауважити, що нейронні мережі здатні показувати

таку якість, а тому система розпізнавання жестів буде орієнтуватися саме на таку цифру.

Таблиця 2.2

Очікування що до точності результатів

Точність	Звичайні користувачі	Користувачі похилого віку	Користувачі з фізичними вадами
99%	506	35	167
98%	24	29	158
97%	11	15	22
96%	0	4	3
95%	0	6	0
Загалом	541	89	350

Опитування спиралось на очікування справжніх позитивних результатів, у той час, як помилкові позитивні та помилкові негативні результати розглядаються як помилка.

Справжній позитивний результат у випадку роботи з нейронною мережею направленою на розпізнання жесту можна вважати вірне розпізнання показаного жесту.

У свою чергу помилковий позитивний результат – це розпізнання хибного жесту, а помилковий негативний результат – неможливість побачити чи розпізнати жест, показаний на картинці.

Існує і четвертий тип результату – вірний негативний. У розглянутому випадку, такий результат досягається коли система не бачить жест в ситуації, коли ніякого жесту не показується на екрані. Приклад усіх чотирьох результатів можна розглянути на рисунку 2.1, разом з підписом до кожного з жестів.

Варто зазначити, що під час вимірювання точності системи, загалом, беруться до уваги як справжні позитивні результати так і справжні негативні. У той час, як вимірювання справжніх позитивних результатів можливо за допомогою відсотку точного вимірювання жестів, для розуміння точності функціонування системи загалом буде використано метрику Коефіцієнт Жаккара, що може бути розраховано за формулою 2.1.

$$J = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.1)$$

Де TP – справжній позитивний результат, TN – справжній негативний, FP – хибний позитивний, FN – хибний негативний [19]. Коефіцієнт буде розраховуватись на проміжку в 1 хвилину, з урахуванням усіх результатів наданих нейронною мережею. Під час розрахунку буде використано лише дані з самої нейронної мережі, не беручи до уваги інші фактори.

2.3.3 Загальні вимоги до можливостей у використанні

Опитування мало можливість внести свою власну пропозицію що до того, які функції очікуються у системі жестового керування. На основі пропозицій, що зустрічаються найчастіше, було виділено наступні положення:

- Активація системи не має включати дій, виконання яких не включає жестове керування. Можливість керувати системою дотиком має залишатися як опція упродовж усього процесу роботи з технічним засобом.

- Жести для керування мають бути представлені на екрані певним чином – як жест, що система бачить у поточний момент, так і список жестів, що пояснює можливі дії – або аналогічна система для полегшення роботи з жестовим інтерфейсом.

- Якщо в камері знаходиться більш ніж одна рука, система має надавати перевагу тій, що виконує жест. Якщо помічено декілька жестів, система має пріоритезувати руку, що вже виконувала жести до появи іншої руки.

- Система має розпізнавати жести на певній відстані від терміналу, проте не брати до уваги віддалені жести для пониження кількості факторів, що можуть відволікати систему та понижувати точність її роботи.

Перелічені міри спрямовані на виконання різних завдань – починаючи з потенційного підвищення точності роботи системи і закінчуючи функціоналом необхідним для спрощення процесу роботи користувача як з технічним забезпеченням, так і з самим жестовим інтерфейсом.

Система має ще одну вимогу, яку варто взяти до уваги – вона має бути легкою в налаштуванні до специфічних вимог, які може поставити надавач послуг. Система має видавати вихідні сигнали, що не будуть прив’язані до певного технічного застосунку, проте будуть мати можливість легкого підключення до програмного забезпечення надавача послуг. Це викликано серйозними відмінностями між інтерфейсом програмного забезпечення, що використовують різні компанії і ставить за мету можливість широкої імплементації на різних технічних засобах [20, 21].

2.4 Засоби реалізації

Для створення ефективної мережі необхідно розробити програму, що буде здатна виконувати задачу що до підвищення доступності технологічних засобів для користувачів з обмеженими можливостями та під час пандемії.

Одне з головних питань, що варто визначити – яким чином до системи буде імплементовано нейронну мережу. Ця ціль може бути досягнена у 3 можливі шляхи [22,23]:

- створення CNN мережі “з нуля”, використовуючи функціонал засобів побудови нейронних мереж, тренування такої мережі на основі тренувальних даних та налаштування нейронної мережі для власних потреб. Такий спосіб є

надзвичайно складним та займе велику кількість часу для реалізації, проте має потенціал надати найбільш точні та швидкі результати – за умови, що створена система буде натренована на великій кількості даних (що найменше 2000 зображень).

- використання готової нейронної мережі, підключивши її до проекту. Такий спосіб є найпростішим у реалізації з трьох, проте він має свої недоліки. Головним з них можна назвати проблеми з точністю результатів, адже вже готова нейромережі, зазвичай, націлені на широкий спектр можливостей використання, жертвуючи швидкістю роботи та вірністю результатів.

- використання базової мережі з малою кількістю можливостей, та її подальша модернізація, тренування. Такий спосіб дозволяє зекономити час, у той же час підлаштовуючи нейромережу під власні потреби. Цей спосіб є оптимальним компромісом, адже використовуючи його єдиним недоліком є велика кількість тренувальних даних необхідна для коректної роботи.

2.4.1 React.JS

Для створення базової структури додатку було використано функціонал React.JS. Ця бібліотека є популярною на поточний день і використовується такими гігантами індустрії як Yahoo, Netflix та Airbnb [24].

React (старі назви: React.js, ReactJS) — це відома відкрита JavaScript бібліотека що використовується для створення інтерфейсів користувача. Головна сфера її використання - вирішення проблеми часткового оновлення вмісту вебсторінки, з якими стикаються в розробці односторінкових застосунків.

React дозволяє розробникам створювати великі вебзастосунки, які використовують змінні дані – і при цьому не потребує перезавантаження сторінки. Головна перевага React – його швідкість роботи, простота використання та маштабованість. Використання React у поєднанні з іншими JavaScript бібліотеками або в великих фреймворках можливе завдяки тому, що він обробляє тільки користувацький інтерфейс у застосунках. Такий принцип роботи

називається модель-вид-контролер, MVC. Він також може бути використаний з React на основі надбудов, щоб піклуватися про частини без користувацького інтерфейсу побудови вебзастосунків. Властивості передаються в рендерер компоненту, як властивості html тегу. Можливості компоненту включають здатність до змін властивостей через callback функції – проте можливість внесення змін напряму відсутня. Такий механізм називають «властивості донизу, події нагору»

Розробники використовують React не лише для рендерингу HTML в браузері, а й для виконання інших дій, наприклад динамічного рендерингу графіків в теги <canvas>. Цікавою функцією можна назвати і можливість ізоморфного рендерингу ідентичного HTML коду як на сервері, так і на клієнті, терміналі, тощо [25].

Бібліотека була обрана через свою простоту використання, зручність та можливість створення веб-потоку у браузері, через який нейронні мережі будуть зчитувати зображення. Цей спосіб дозволяє системі отримувати дані з зображення з певним проміжком часу, що можна встановити самостійно – з можливістю подальшого налаштування надавачем послуг.

2.4.2 Tensorflow

TensorFlow є системою машинного навчання Google Brain, випущеною як відкрите програмне забезпечення 9 листопада 2015 року. TensorFlow працює в еталонній формі на одиничних пристроях, проте можливість працювати на декількох центральних та графічних процесорах (включно з додатковими розширеннями що можуть бути використані, наприклад CUDA) також присутня. TensorFlow доступна на таких платформах як Linux, macOS, Windows, та для мобільних обчислювальних платформ, включно з Android та iOS. Обчислення TensorFlow виражаються як станові графи потоків даних [26].

Ця бібліотека здатна задовольнити наші потреби одночасно двома мережами – Handpose та Fingerpose. Ці нейронні мережі є відносно слабкими,

проте вони прості у використанні, модифікації та навчанні. Для успішної реалізації проекту вони будуть модифіковані для покращення результатів що вони здатні показувати [25].

Handpose – Це CNN нейронна мережа здатна розпізнавати руки людини. Цей інструмент здатний розпізнавати 21 точку на руці людини, визначаючи приблизне положення пальців. Ця система здатна розраховувати кут нахилу пальців на зображенні, що є важливим елементів визначення жестів.

Fingerpose – Базова мережа, здатна розпізнавати жести за положеннями пальців. На жаль, у своєму початковому стані вона здатна розпізнавати 2 жести – чого не достатньо для цілей поставлених перед системою підвищення інклюзивності на основі розпізнавання жестів. Для забезпечення вірної роботи системи, варто підвищити кількість жестів, що ця нейронна мережа здатна розпізнавати.

Мережі будуть поєднані у розроблюваній системі і адаптовані до роботи одна з одною. Так як вони розроблені однією компанією та знаходяться в одній базі нейронних мереж, очікується що адаптація буде простою для реалізації та не буде негативно впливати на ефективність роботи системи.

Таблиця 2.3

Розподіл обов’язків нейронних мереж

Handpose	Fingerpose
Розпізнавання положення руки	Визначення жестів
Визначення ключових точок на руці	Підтвердження жесту на основі кутів нахилу пальців
Визначення положення пальців	Перевірка точності роботів попередньої системи
Визначення куту нахилу пальців	Передача отриманих даних у текстовий формат

Варто зазначити, що використання двох нейронних мереж, діючих незалежно одна від одної має свої переваги та недоліки. Головним недоліком можна відзначити можливе накопичення помилок – помилка у розпізнаванні руки призведе до помилки у розпізнаному жесті. Для вирішення цієї проблеми варто додати певне програмне рішення здатне знизити кількість помилок.

Для кращого розуміння обов'язків системи варто буде побудувати use-case діаграму для опису системи, на додачу до стандартних схем. Побудова схем буде виконана у розділі 3.

2.5 Список жестів для розпізнавання

Для реалізації системи керування необхідно визначити, яка кількість жестів є оптимальною для роботи з інтерфейсами терміналів мереж їжі швидкого приготування. Виконання цього завдання потребує і вивчення можливостей пам'яті людини. Частина, що важлива для цього дослідження це коротка пам'ять, адже саме клієнту що відвідують заклад в перший раз (чи користуються жестовим керування в перший раз) найбільш ймовірно зіткнуться з проблемами в використанні системи.

Першим кроком має бути визначення оптимальної кількості жестів для людини. Вивчаються можливості середньостатистичного користувача, що не проходив ніякої спеціальної підготовки та не бажає виконувати складних дій під час походу до закладу.

Хоча аналіз такого питання може виглядати неможливим, беручи до уваги комплексність мозку людини - насправді дослідження цього питання вже були проведені. Згідно дослідження проведеного Георгом Міллером, людина може легко зберігати в своїй короткій пам'яті від 5 до 9 різних об'єктів з якими вона не мала ніякої справи. Що цікаво, таке число є однаковим як для простих об'єктів – наприклад кількості каменів у подвір'ї, так і для комплексних, у тому числі слів що самі складаються з певної кількості літер. "Магічним числом" вважається число 7, що є найбільш оптимальною кількістю елементів для людини [26].

Другим кроком є розгляд можливих виключень до цього правила. Діапазон 5-9 жестів є приблизним і має свої виключення що мають бути прийняті до уваги. Так, наприклад, для людей похилого віку це число зменшується до 4 об’єктів (в середньому), а люди з добре натренованою пам’яттю та високим рівнем концентрації під час роботи можуть запам’ятати аж до 50 різних об’єктів! Проте у випадку системи жестового керування та беручи до уваги поставлену мету – підвищення доступності, на такі високі цифри розраховувати не варто.

Загалом, оптимальна кількість елементів виглядає наступним чином:

- Люди похилого віку, особи з порушенням пам’яті: ~4
- Середньостатистичний користувач: ~7
- Молоді люди, тинейджери: ~9

9 жестів може бути оптимальним числом для системи, доступним для користувача і достатнім для керування терміналом при розбитті можливих дій на категорії. Для створення запасу можливих жестів, орієнтовне число було збільшено до 12.

2.6 Список вимог до системи

Для кращого та наочнішого виду вимог, їх було зібрано в один список. Досягнення цих показників можна вважати як успішну розробку системи, що потребує лише тестування. Ці показники наведено на таблиці 2.4, разом зі списком бажаних можливостей системи.

Варто зауважити, що незначні відмінності від цих показників є допустимимм, адже під час їх визначення брався максимальний реалістичний з бажаних результатів.

Таблиця 2.4

Вимоги до системи жестового керування

Вимога до системи	Показник
Бажана швидкодія	3 с.
Максимальна допустима швидкодія	7 с.
Вимога до системи	Показник
Точність роботи, тільки мережі	95%
Точність роботи, система загалом	99%
Цикл перевірки точності	1 с.
Кількість нейромереж у системі	< 3
Мінімальна кількість розпізнаваних жестів	9
Оптимальна кількість розпізнаваних жестів	12
Бажані можливості	- Передача переваги руці, що показує жест - Присутність жестів, що відповідають за дію над елементом інтерфейсу - Спосіб зрозуміти поточний розпізнаний жест - Адаптивність до різних додатків - Можливість бездотикової активації системи

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Розробка архітектури мережі

Першим кроком у створенні програмного забезпечення є побудова схем та діаграм здатних описати функціонал продукту, логіку його роботи та обов’язки його складових. Виконання цього кроку дозволяє краще розуміти розроблюваний проект, зберігати цілісність початкової ідеї та, у свою чергу, підвищити його якість.

Першою діаграмою, що варто побудувати – це діаграма обов’язків кожного з елементів. Для цього можна використати діаграму use-case, де кожен з елементів системи буде показано як актора. Діаграму можна побачити на рисунку 3.1

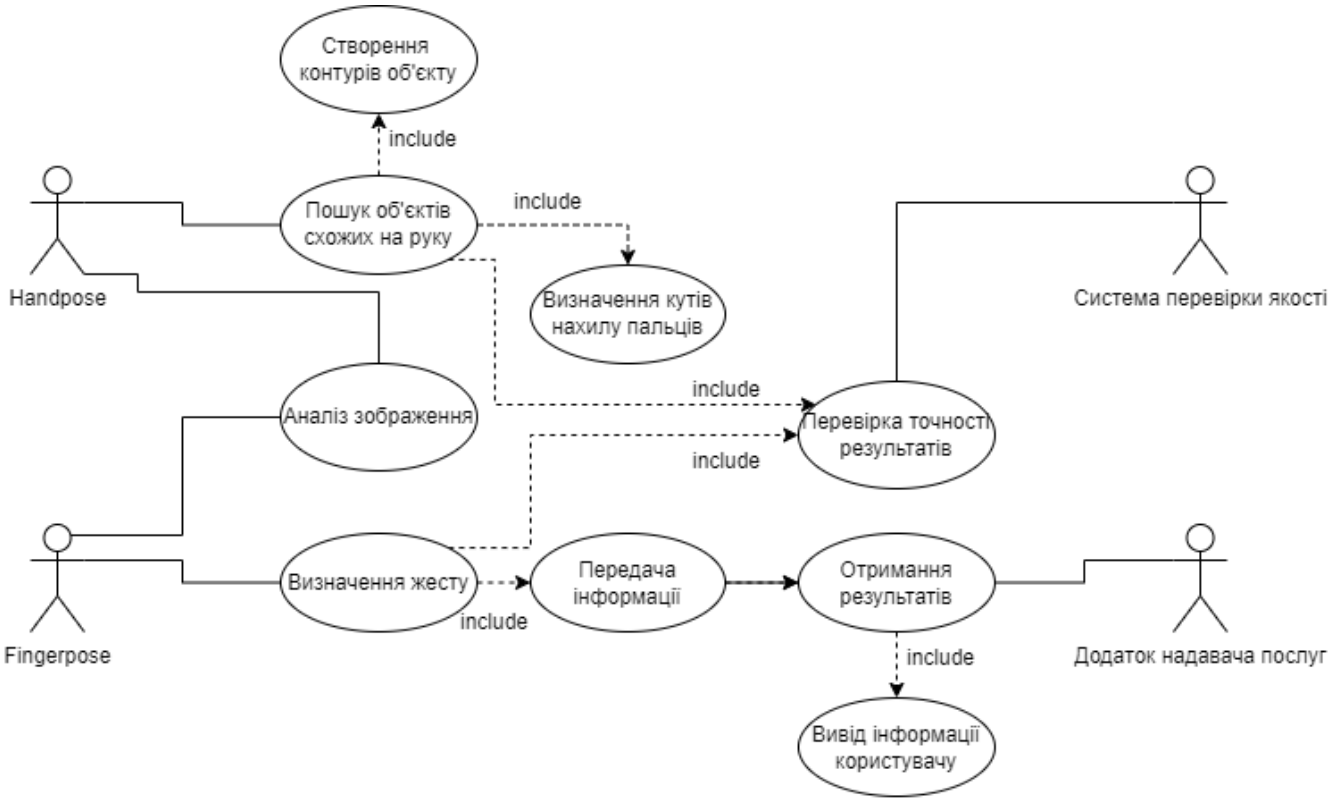


Рис. 3.1 Обов’язки елементів системи

Як можна побачити з діаграми, Додаток надавача послуг, не залежно від своєї побудови, виконує лише одну функцію, що взаємодіє з системою – отримання інформації про жест від нейронної мережі Fingerpose. За необхідності, ця взаємодія буде включати в себе продовження дії – вивід інформації користувачу, в залежності від отриманого жесту. Така низька кількість взаємодій має підвищити адаптивність системи до різних видів додатків.

Система Handpose відповідає за:

- аналіз зображення
- пошук руки людини

У разі успішного виконання цих дій, система також може:

- створити контури визначеного об’єкту (руки)
- визначити кути нахилу пальців
- перевірити точність отриманого результату

Система Fingerpose відповідає за:

- аналіз зображень
- визначення жесту

У разі успішного виконання цих дій, система також може:

- перевірити точність отриманого результату
- передавати результат далі по ланцюгу до додатку користувача.

Система перевірки точності результатів нейромереж виконує лише одну функцію – власне, кінцеву перевірку точності результату, що складає з себе отримання одного і того ж результату впродовж секунди.

Як можна побачити, одразу 3 системи відповідають за перевірку точності результатів – це пов’язано з використанням одразу двох нейронних мереж та з бажаним рівнем точності результатів у 99%.

На діаграмі діяльності процес можна зобразити шляхом показаним на рисунку 3.2, надаючи можливість побачити порядок дій усередині системи та взаємодії між елементами самої системи.

Для простоти розуміння, функції нейронних мереж Handpose та Fingerpose зібрано в одну дію, що виконується системою.

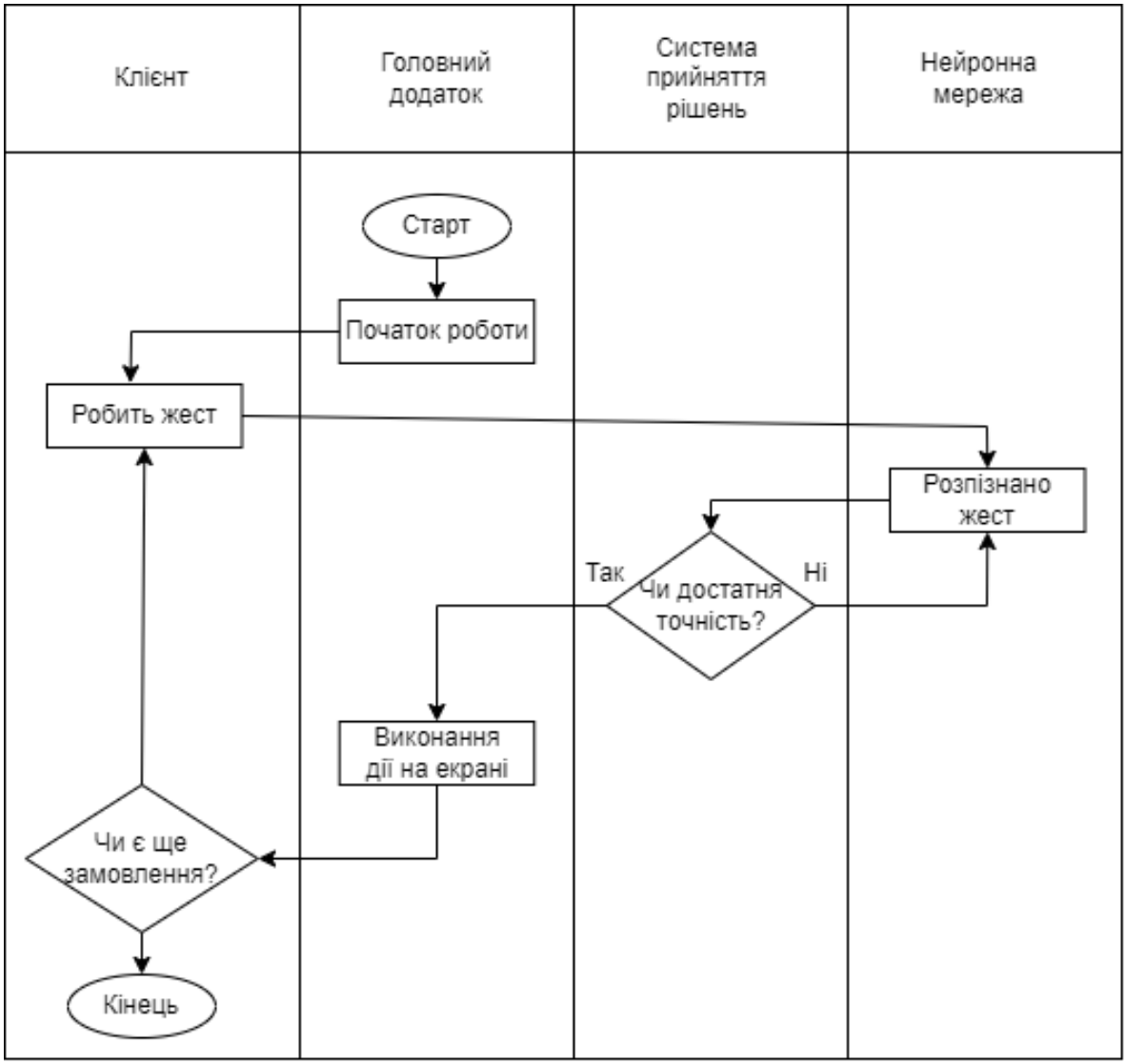


Рис. 3.2 Діаграма діяльності системи

Як можна побачити з діаграми, робота системи починається саме з головного додатку, адже термінали що використовуються у індустрії швидкого харчування працюють упродовж усього робочого дня. Під час відсутності

користувача, система знаходиться у стані очікування, чекаючи активацію та аналізуючи зображення з камери очікуючи коректний жест для початку роботи.

Після того, як користувач виконує жест – система запускає цикл аналізу, що починається з розпізнавання цього жесту. Під час початку роботи він один, проте після цього таких жестів буде набагато більше.

При отриманні позитивного результату у розпізнаванні жесту, система посилає результат до модулю прийняття рішень про точність роботи самої системи. У випадку, коли 10 останніх результатів є однаковими, сигнал буде відправлено далі по ланцюгу, до головного додатку.

Відповідно до розпізнаного жесту вказаного у отриманому сигналі, головний додаток виконує відповідну дію (наприклад додає товар “х” до замовлення, переходить до іншої категорії товарів або завершує оформлення замовлення). В залежності від виконаної дії, система може продовжувати свою роботу, або завершити оформлення замовлення.

Варто зауважити, що робота системи активується саме жестом користувача, обходячи необхідність дотику до екрану задля вибору необхідного типу керування – адже така необхідність повністю нівелює переваги системи у її гігієнічності. Клієнт все ще може перемкнутися на керування дотиком як на початку роботи, так і у ході процесу, в тому разі якщо він/вона надають перевагу такому методу або не розуміють роботи жестового керування/мають певні проблеми з використанням цього методу керування.

Після виконання повного циклу, система знову повертається у стан очікування, чекаючи на наступного користувача.

Такий показ розподілення зобов’язань має допомогти у знаходженні помилок в середині системи під час їх появи. Така ситуація є дуже вірогідною, адже кожна компанія надавачів послуг має свій власний додаток для оформлення замовлень, хоч вони і мають схожі між собою елементи інтерфейсу. У разі некоректної роботи одного з елементів не буде необхідності розглядати усю систему, необхідно буде лише виправити проблему у невірно функціонуючому елементі.

У свою чергу, сам процес розпізнавання жесту можна побачити на рисунку 3.3, зображений за допомогою блок-схеми.

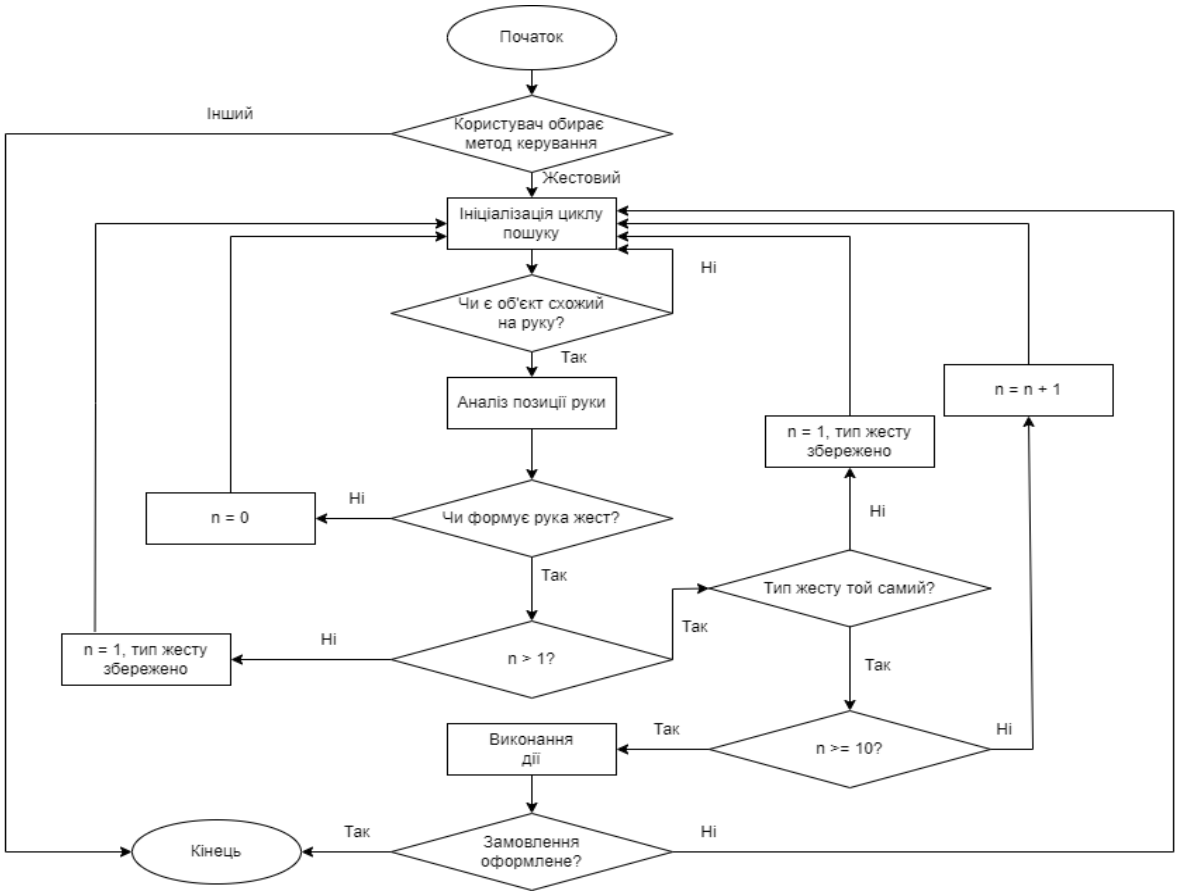


Рис. 3.3 Процес розпізнавання жесту

На схемі можна побачити логіку роботи перевірки кількості однакових результатів отриманих в якості вихідних даних з системи.

Як можна побачити зі схеми, для зарахування відповіді мережі, системі необхідно отримати однаковий результат визначення жесту 10 разів підряд. Враховуючи швидкість кожної обробки в 0.1 секунду це встановлює мінімальний час обробки жесту в 1 секунду.

Використання такого методу підвищення точності результатів можливо лише при базовій точності роботи системи рівній або вищій 90%.

3.2 Процес тренування нейронної мережі

Тип системи нейронної мережі, що використовується у цій роботі було визначено як CNN мережу (згорткова нейронна мережа). Для виконання тренування такого типу мережі знадобиться набір зображень для аналізу, в нашому випадку зображень що включають у себе руку людини яка показує жести. Зображення мають розмір 200x200 пікселів, такий низький розмір використовується для пришвидшення процесу тренування. Під час навчання нейронної мережі використовуються методи стохастичного градієнтного спуску та метод сегментації зображень (метод пулінгу розглянутий раніше). Вихідне зображення є меншим у розмірах за рахунок логіки роботи методу згортки. До отриманого зображення використовується функція soft-max для розрахунків значень та крос-ентропії.

$$E = \sum_{x\in\Omega} w(x)\log\left(p_{l(x)}(x)\right) \tag{3.1}$$

За допомогою морфологічних операцій розраховується границя розділення та значення карти вагових коефіцієнтів.

$$w(x) = w_c(x) + w_0 * \exp\left(-\frac{(d_1(x)+ d_2(x))^2}{2\sigma^2}\right) \tag{3.2}$$

Де w_c - карта вагів для балансування частот класів, $d1$ - відстань до границі 1-ї клітики, $d2$ - відстань до границі 2-ї клітинки.

Для проведення кожної з ітерацій навчання розглянутої системи потрібно брати до уваги, яким чином через них можна отримати градієнти функції помилки для ваг. З самого початку, помилка проходить без змін через функцію отримання максимуму. Під час проходження цього етапу, дизкретизаційний шар не здійснює навчання.

В момент проходу через граф, розраховані градієнти знаходяться в розрідженому стані, адже з усіх елементів вікна субдискретизації $z_{i,j}^{l}$, приватна похідна $\frac{\partial E}{\partial x_{i,j}^{l+1}}$ відноситься тільки до одного, максимального. Градієнти, що не є максимальними отримають статус нульового градієнту. Після виконання цього процесу, навчання нейронної мережі можна вважати завершеним.

Отримані елементи проходять через похідну нелінійності, формулу 3.3.

$$\frac{\partial E}{\partial y_{i,j}^{l}} = \frac{\partial E}{\partial z_{i,j}^{l}} * \frac{\partial z_{i,j}^{l}}{\partial y_{i,j}^{l}} = \frac{\partial E}{\partial z_{i,j}^{l}} h'(y_{i,j}^{l}) \tag{3.3}$$

Розглянемо дані, що є відомими на цьому кроці. Ми маємо $\frac{\partial E}{\partial x_{i,j}^{l}}$, проте значення $h(y_{i,j}^{l})$ все ще є невідомим. Його значення можна отримати шляхом проведення подальших розрахунків. Це етап, на якому з'являються ваги згорткового рівня, навчання яких буде необхідним. Цей етап є доволі вимогливим до ресурсів системи, адже усі отримані ваги діляться та приймають участь у процесі пошуку.

$$\frac{\partial E}{\partial w_{a,b}^{l}} = \sum_i \sum_j \frac{\partial E}{\partial y_{i,j}^{l}} * \frac{\partial y_{i,j}^{l}}{\partial w_{a,b}^{l}} = \sum_i \sum_j \frac{\partial E}{\partial z_{i+a,j+b}^{l-1}} \tag{3.4}$$

Елементи i та j проходять через всі складові зображення на розглядаємому проміжному шарі $y_{i,j}^{l}$ (після процесу згортки – до завершення процесу убдискретизації). Після завершення цього кроку необхідно лише запустити градієнти через попередній нейронний шар мережі.

$$\frac{\partial E}{\partial x_{i,j}^{l}} = \sum_a \sum_b \frac{\partial E}{\partial y_{i-a,j-a}^{l}} * \frac{\partial y_{i-a,j-b}^{l}}{\partial x_{i,j}^{l}} = \sum_i \sum_j \frac{\partial E}{\partial y_{i-a,j-b}^{l}} * w_{a,b} \tag{3.5}$$

Розглянувши формулу зворотнього проходу згортки та згортку з тими ж вагами $w_{a,b}$ можна побачити, що овни є доволі схожими. Головна відмінність полягає у заміні i + a та j + b на i - a та j − b. Це призводить до наступної ситуації: розмірності зберігаються коли зображення доповнюються нулями, а тому зворотний обхід можна аналогічним прямому проходу - тільки розвернутими осями [27].

Для навчання системи використано набір тренувальних даних Hand Gesture Recognition Database, що містить набір інфрачервоних зображень руки, а також власні створені фотографії, переведені в чорно-білий формат. Зображення такого формати є набагато простішими у процесі аналізу, адже системі не потрібно брати колір до уваги, а чітка розбіжність між розпізнаваним жестом та непотрібною частиною зображення пришвидшує процес тренування нейронної мережі. Приклад навчальних даних можна побачити на рисунку 3.4. Цей безкоштовний набір було завантажено з мережі Kaggle [28].

Набір навчальних даних включає в себе 2000 зображень для кожного з 10 жестів, доводячи кількість жестів до 20000. Для ефективного навчання мережам, зазвичай, необхідно щонайменш 10000 зображень, а тому набір навчальних даних визнано достатнім для виконання навчання мережі. Вибір забагато високої кількості зображень також є небажаним, адже може призвести до перенавчання мережі – що у свою чергу тільки підвищить кількість помилок [29].

В процесі навчання моделі, варто приділяти увагу запобіганню домінації одного з класів. Загалом, цю проблему можна описати наступним чином: один з класів отримує пріоритет зі сторони нейронної мережі і завжди вважається правильним, навіть якщо це призводить до виводу невірних вихідних даних Ця ситуація може виникнути за наступних умов:

- Недостатня кількість даних у наборі тренувальних даних
- Надмірна однорідність даних у тренувальних наборах
- Сеплінг не надав очікуваних результатів

Розглянувши метод навчання CNN мереж можна побачити, яким чином він дозволяє навчити нейронну мережу розпізнаванню жестів. Необхідно лише

визначити жести, що мають бути включені до списку, що мережа має розпізнавати. Цей список має включати бажані жести для керування, проте у той же час він має мати обмежені розміри – це дозволить полегшити сприйняття та використання набору жестів. До цього списку можна додати жести, що система вже вміє розпізнавати: палець вгору та перемога. Для підтвердження ефективності проведеного навчання, варто провести замірювання точності розпізнавання усіх жестів після проходження навчання. Таке замірювання варто проводити не за результатами точності розпізнавання жестів, а за більш комплексним коефіцієнтом Жаккара [30].

Рис. 3.4 Приклад зображень для навчання нейронної мережі

3.3 Список бажаних жестів

Вимоги до жесту є доволі простими і можуть бути описані наступними трьома пунктами:

- Жест має бути несхожим на інші, щоб нейронній мережі було його легко розпізнати.

- Жест має бути простим у використанні, щоб користувачу не доводилось напружуватися при його використанні.

- Жест має бути культурно допустимим, щоб не образити користувачів закладу.

Згідно цих вимог та базуючись на навчальних даних до нейронної мережі, було сформовано наступний список бажаних жестів:

- Великий палець вгору
- Великий палець вниз
- Долоня
- Кулак
- Окей
- Хапаюча рука
- Стиснуті пальці
- Один
- Два
- Три
- Чотири
- П’ять

Для того, щоб навчена система розуміла жести та надавала текстову інформацію що до розпізнаного жесту, необхідно також імплементувати таблицю кутів напрямку пальців. Це необхідно, що система не помилялась у розумінні перевернути жестів, наприклад Великий палець вгору та Великий палець вниз.

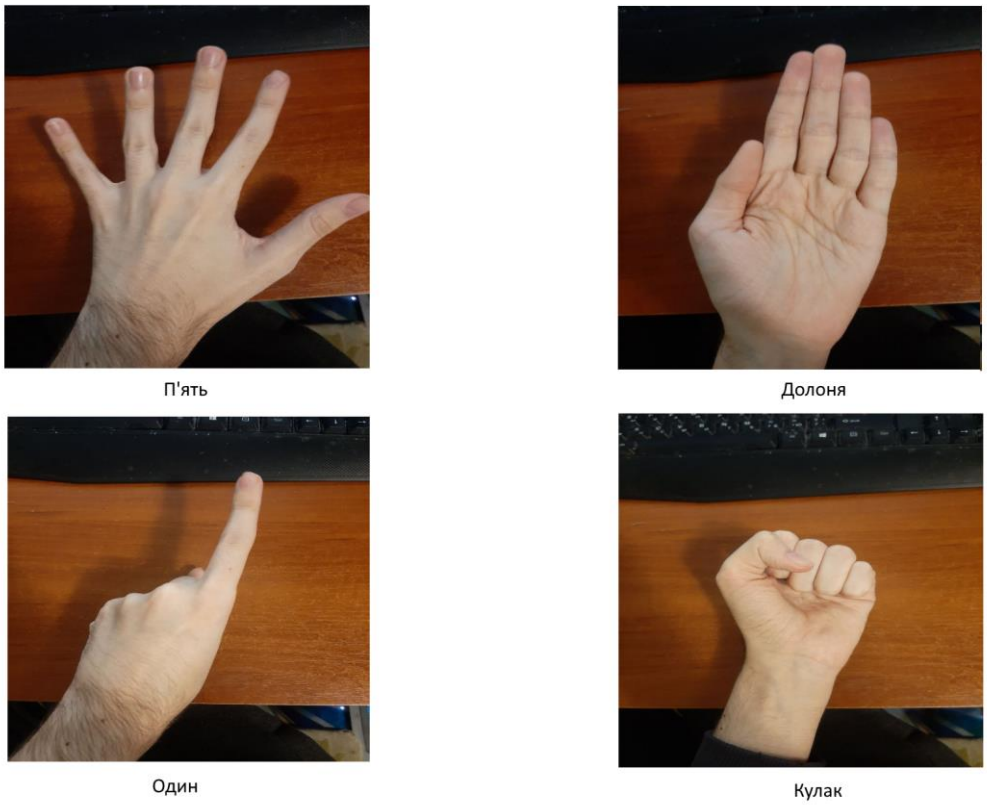


Рис. 3.5 Жести п‘ять, долоня, один та кулак

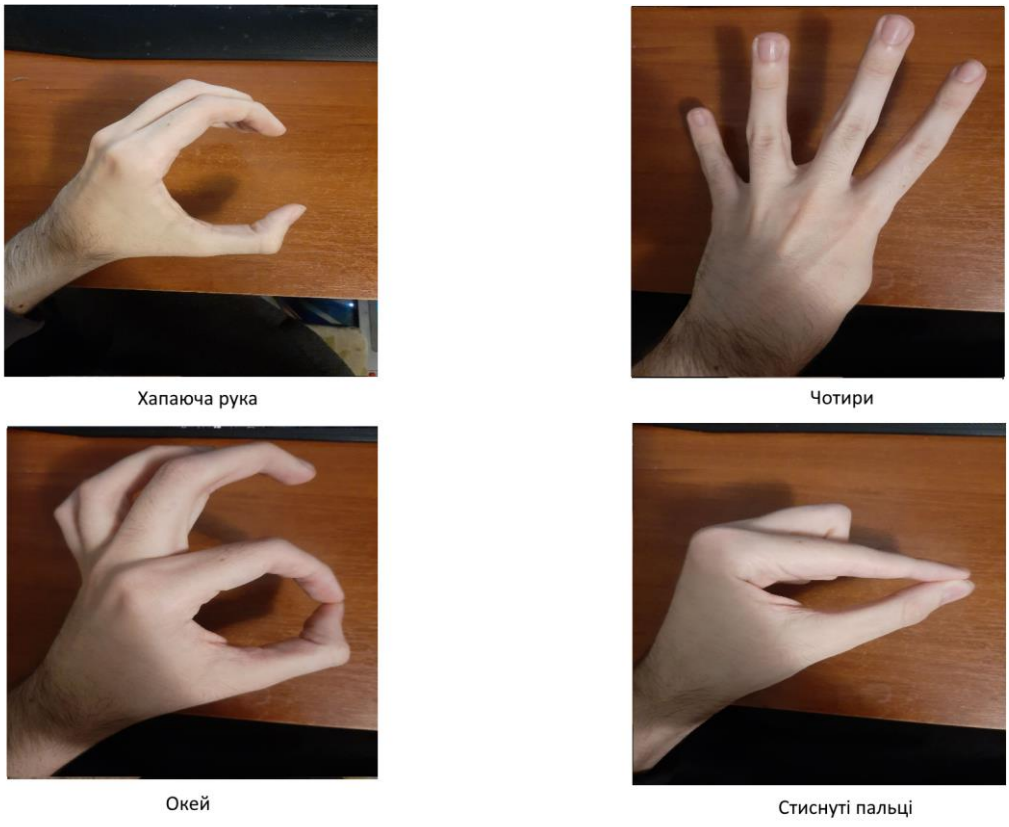


Рис. 3.6 Жести хапаюча рука, чотири, окей та стиснуті пальці

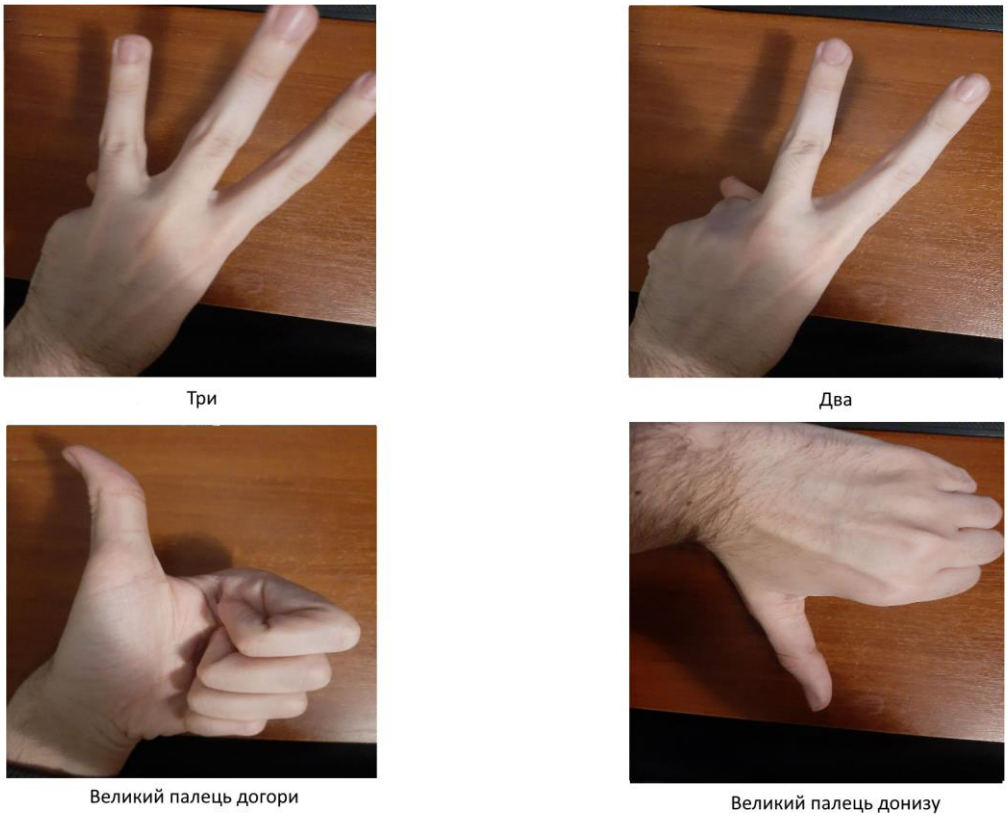


Рис. 3.7 Жести три, два, великий палець догори та великий палець донизу

Ці жести відповідають поставленим вимогам у повній мірі. Їх можна розділити на наступні групи:

- Один, два, три, чотири та п’ять є інтуїтивно зрозумілими для людини і часто використовуються у повсякденному житті. Вони можуть відповідати номеру елементу в обраній групі замовлень, або номеру групи замовлень.

- Великий палець догори та вниз також є інтуїтивно зрозумілими для більшості користувачів і часто асоціюються з відповіддю так/ні.

- Кулак, окей, долоня, хапаюча рука та стиснуті пальці є зручними у використанні та достатньо незвичними для запобігання плутанини як самої системи так і користувачів.

Варто зауважити, що хоч не всі ці жести є частиною тренувальних даних, усі вони є схожими на такі жести, а тому вони можуть бути використані за допомогою налаштування кутів пальців та їх напрямків.

3.4 Створення програмної бази

3.4.1 Розпізнавання руки користувача

Наступним кроком є створення камери для отримання відеопотоку. Завдяки інструментам, що надає нам React JS, ця задача легко виконується. Отриманий результат – локальний сервер за адресою localhost.3000, що включає в себе сірий фон та вікно з відеопотоком. Для отримання потоку було використано стару вебкамеру для симуляції реальних умов використання системи жестового керування.

Рис. 3.10 Відеопотік на локальному сервері

Отриманий потік надає можливість проводити аналіз, проте на цьому етапі нейронні мережі ще не підключені. Так як тренування мереж вже було проведено, їх можна підключати до системи.

Для кращого розуміння результатів було створено візуальне зображення головних точок руки, які бачить система, а також ліній що бачать нейронні мережі.

Таке візуальне зображення результатів є корисним для аналізу результатів роботи мережі, кращого розуміння процесу її роботи, а також для пошуку можливих помилок.

Результат роботи системи можна побачити на рисунку 3.11, 3.12 та 3.13. Як можна побачити, система здатна бачити руку користувача навіть під кутом, вверх ногами чи у положенні “ребром”. У зв’язку з тим, що для розпізнавання жестів необхідне лише зап’ястя та пальці користувача, інші частини руки не розглядаються.

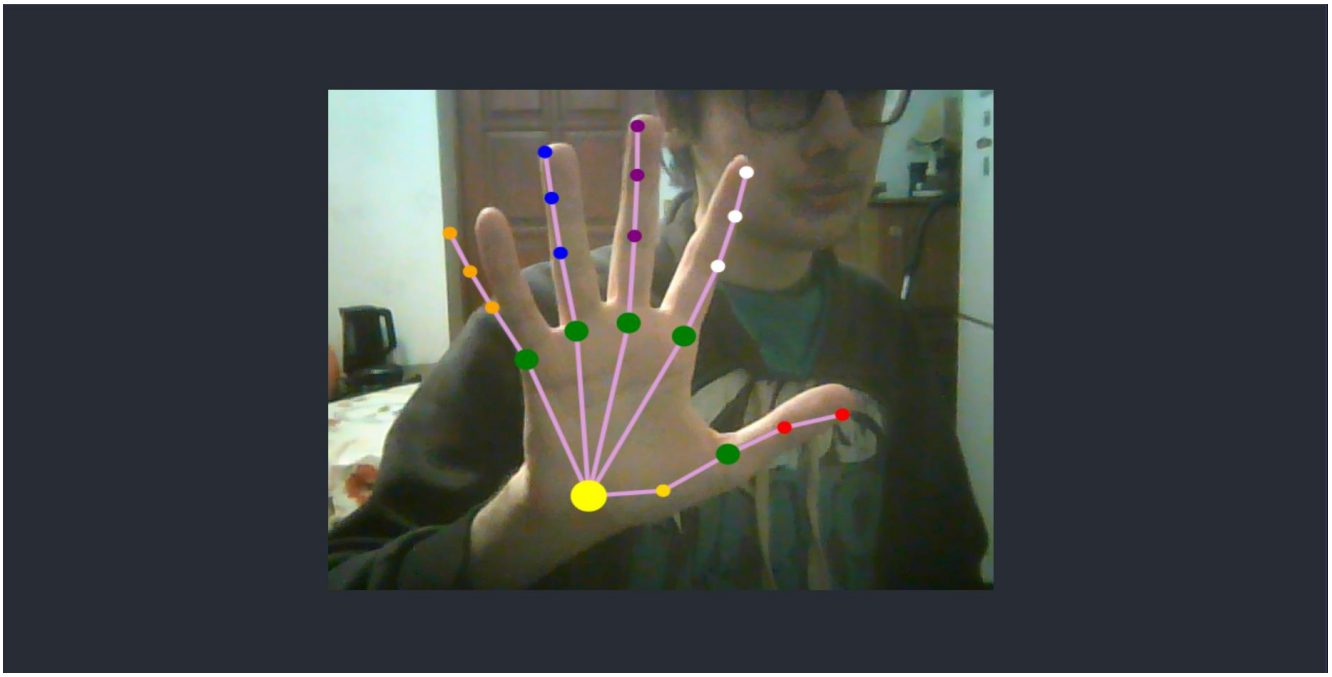


Рис. 3.11 Визначення руки користувача

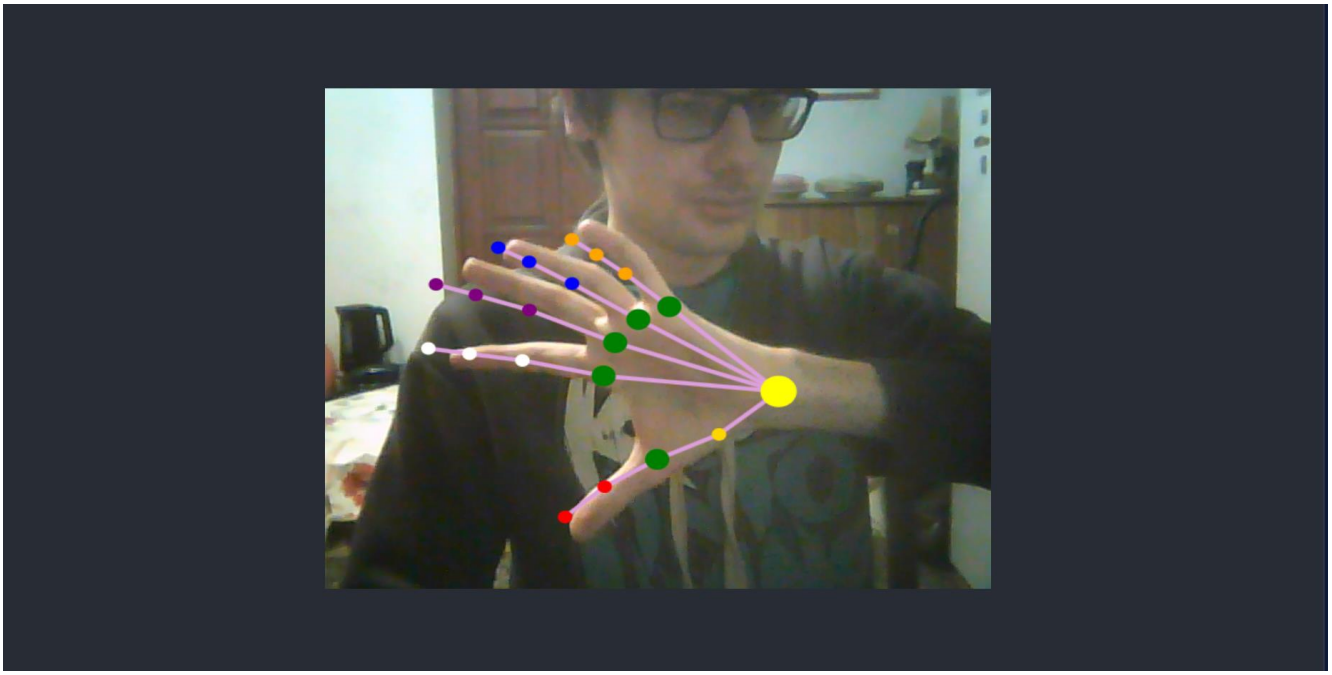


Рис. 3.12 Визначення руки під кутом

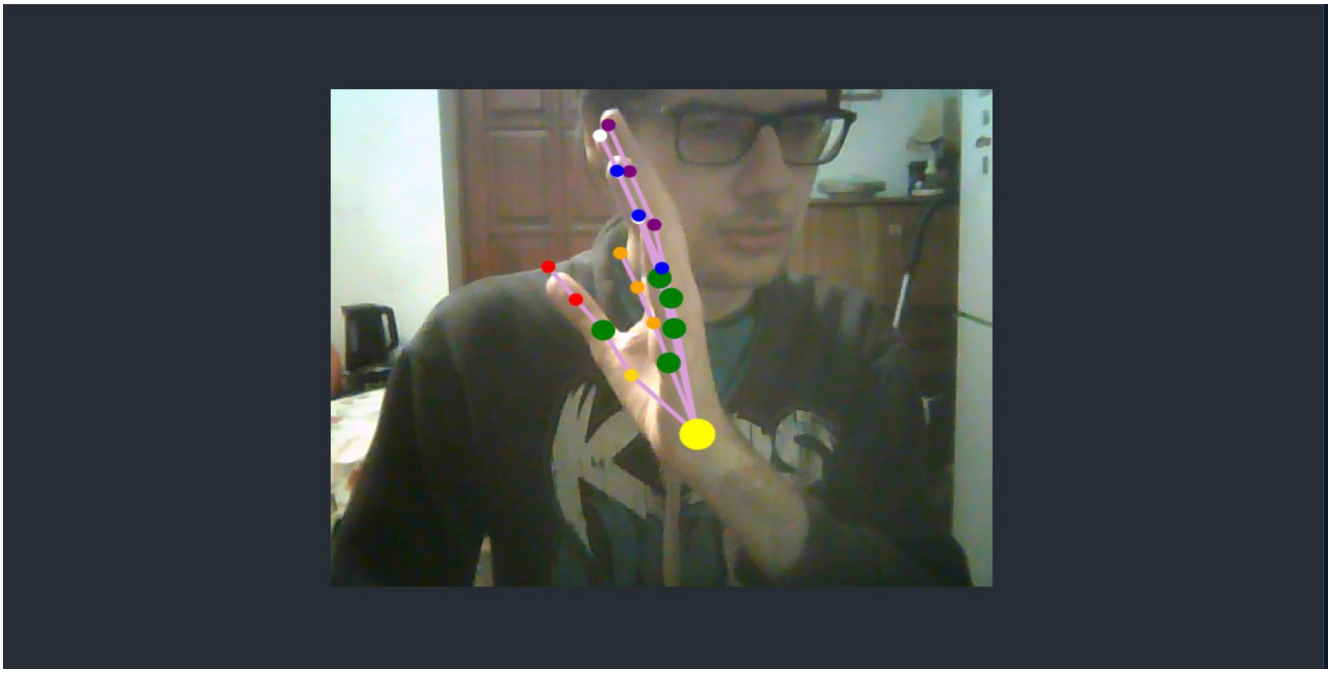


Рис. 3.13 Визначення руки що стоїть «ребром»

Як можна побачити, система розділяє зображення руки на 21 точку. Для кращого розуміння зображення їх було розділено на 6 кольорових зон.

- Жовта зона включає в себе 2 точки які формують зап’ястя та є базою руки. Ці точки не будуть приймати прямої участі у визначенні жесту, проте є важливими для визначення початку руки.

- Зелена зона з 5 точок включає в себе 5 кісток що формують початок пальцю. Ці точки добре видно для системи завдяки формі руки. Вони слугують початком визначення пальцю.

- Червона зона, сформована з 2 точок, показує великий палець. Вона складається з двох точок через те, що великий палець може буде зігнуто лише в 2 місцях – на його початку та по середині. В системі вона визначається як “Thumb”

- Біла зона, сформована з 3 точок, показує вказівний палець. Кількість точок відповідає кількості точок можливого згину. В системі ця зона визначена як “Index”

- Фіолетова зона, сформована з 3 точок, показує середній палець. Кількість точок відповідає кількості точок можливого згину. В системі ця зона визначена як “Middle”

- Синя зона, сформована з 3 точок, показує безіменний палець. Кількість точок відповідає кількості точок можливого згину. В системі ця зона визначена як “Ring”

- Помаранчева зона, сформована з 3 точок, показує мізинець. В системі ця зона визначена як “Pinky”

Визначення зон руки є однаковим для обох рук, так само як і аналіз потенційних жестів - проте лише один результат може бути виведений на екран одночасно. Це викликано через обмеження можливостей нейронної мережі, проте не має викликати особливих проблем. Створення системи, здатної розпізнавати жести з двох рук, може підвищити кількість жестів яку система може використовувати без втрат точності. Проте, варто зазначити що це також підвищить складність розуміння системи та зробить її непридатною для використання особами що можуть використовувати лише одну руку – що йде всупереч меті підвищення інклюзивності.

Для забезпечення безперервної роботи системи було імплементовано функцію зміни фокусу системи на руку, що вказує певний символ. Перемикання здійснюється в автоматично, як тільки минула рука перестає показувати жест, в той час як інша показує його. Загалом правила для цього методу можна описати як:

- Одна рука вказує жест, інша ні: виводиться результат аналізу руки, що вказує певний жест.

- Обидві руки показують жест: аналізується рука, жест якої було розпізнано першим.

- Жодна рука не показує жест: аналізується рука, що була розпізнана першою.

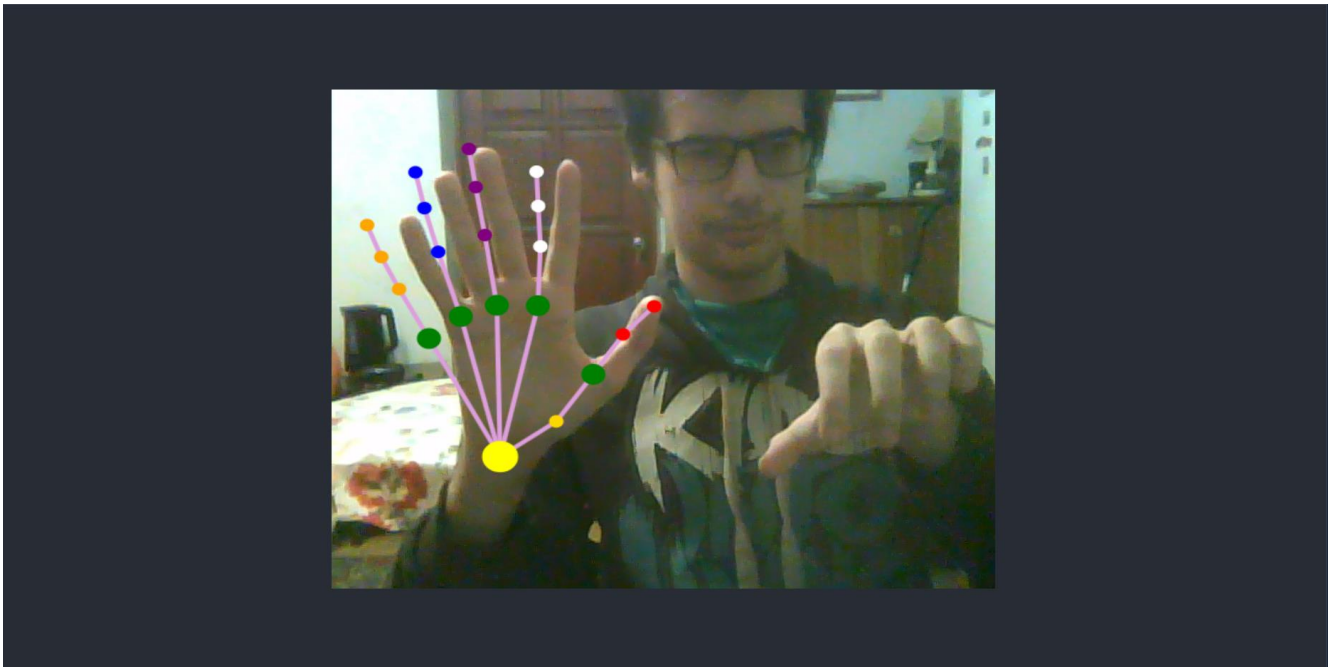


Рис. 3.14 Виведення результатів для активної руки

Рис. 3.15 Обидві руки активні, система аналізує першу

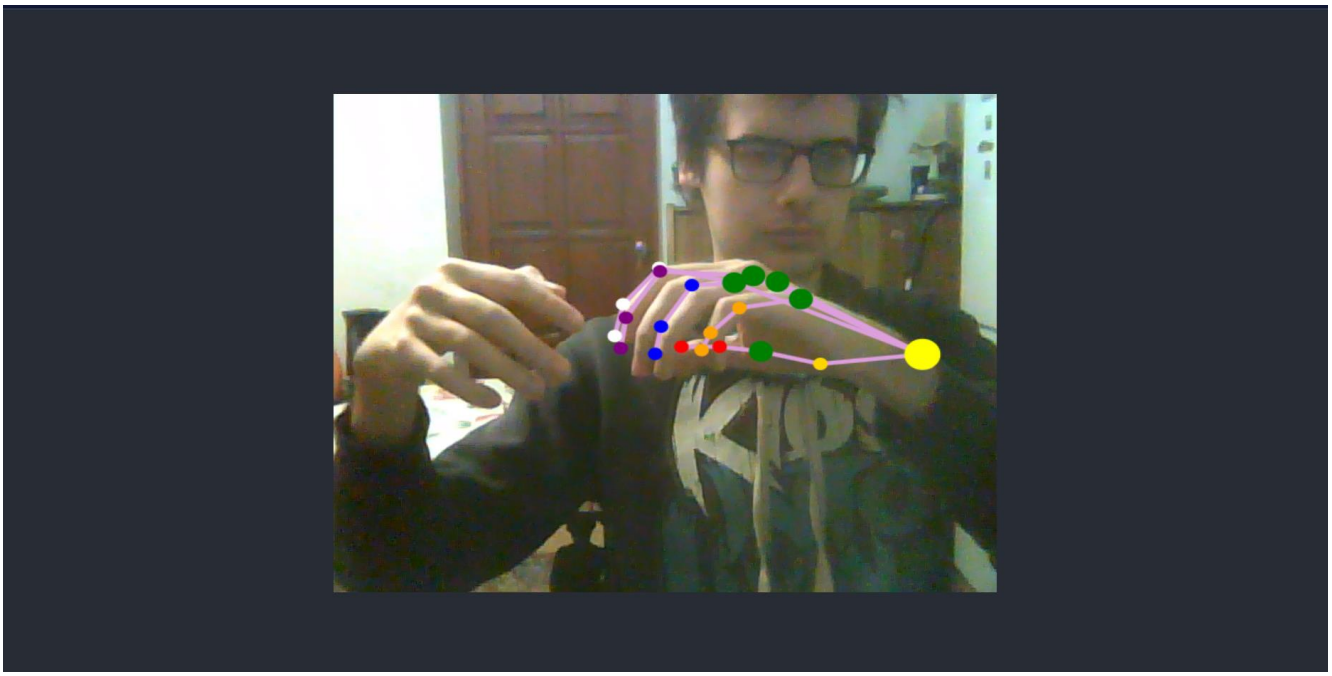


Рис. 3.16 Жодна з рук не вказує жест, система розглядає останню активну руку

Імплементація функціоналу перемикання руки є важливим кроком для запобігання ситуацій, коли система банально “застряє” на аналізі руки що не показує ніяких жестів, у той час, як користувач намагається використовувати іншу руку.

3.4.2 Розпізнавання жестів

Наступним кроком є реалізація визначення самих жестів. Для реалізації цього функціоналу системи буде використано нейронну мережу Fingerpose з виводом результатів у консоль. Для кращого аналізу у результатах буде показано усі результати, які бачить система. При роботі з кінцевим додатком ця інформація не буде потрібною, проте на стадії розробки системи вона є важливою для кращого розуміння логіки роботи.

Параметри системи на цьому кроці:

- Частота аналізу: 0.1 секунда
- Мінімальна впевненість у результаті: 50%
- Результати аналізу: інформація про жест та положення руки

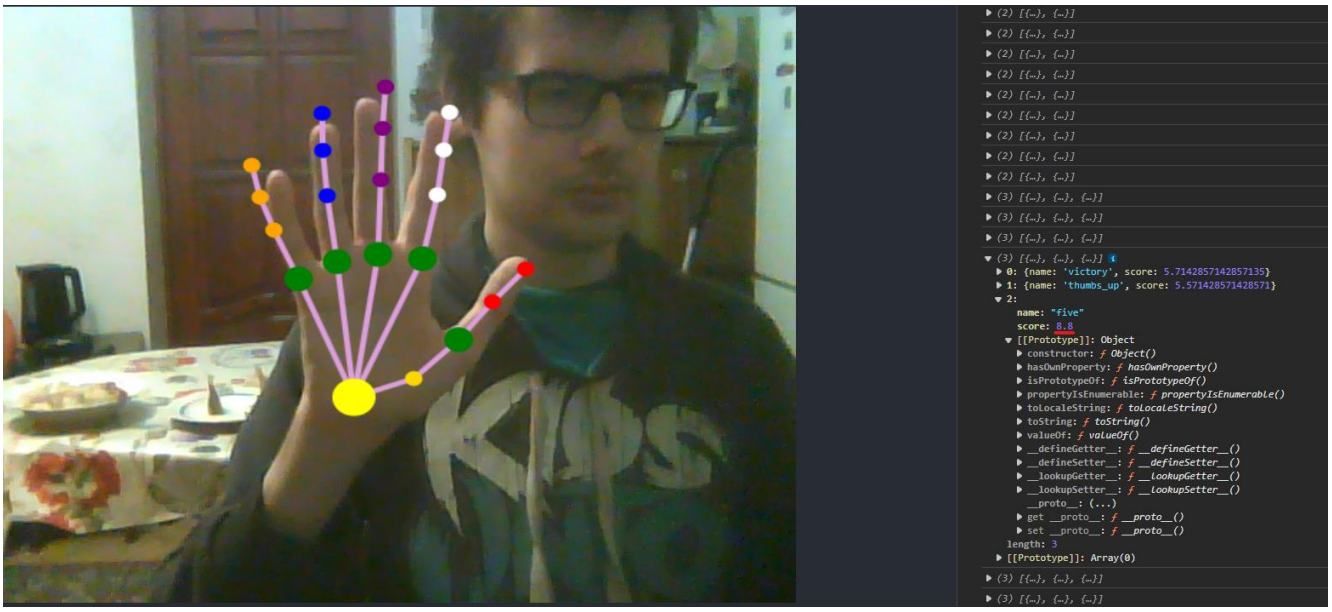


Рис. 3.17 Аналіз зображення з виводом результату у консоль

Як можна побачити, аналіз жесту є детальним, включаючи в себе одразу декілька можливих жестів. В залежності від вказаного мінімуму точності, вона може виводити одразу декілька потенційних результатів. З встановленим мінімумом у 50%, вона розпізнає 3 потенційні жести: 2 “пальці”, “великий палець догори” та “п’ять”. Так як жест “п’ять” має найбільшу точність, саме він буде використаний як кінцевий результат.

Більш детальний розгляд результатів можна побачити на рисунку 3.18. Кожен з отриманих результатів аналізу представляється як звіт, що в своєму закритому стані вказує лише кількість розпізнаних жестів. У відкритому стані, звіт включає наступні елементи:

- Список жестів
- Впевненість системи у результаті
- Складові елементу
- Складові обраного жесту

Для отримання достовірного результату варто обирати результати з вірогідністю 75%+. Це є простим правилом для отримання точних результатів, проте існують ситуації коли одразу декілька відповідей попадають у список можливих.

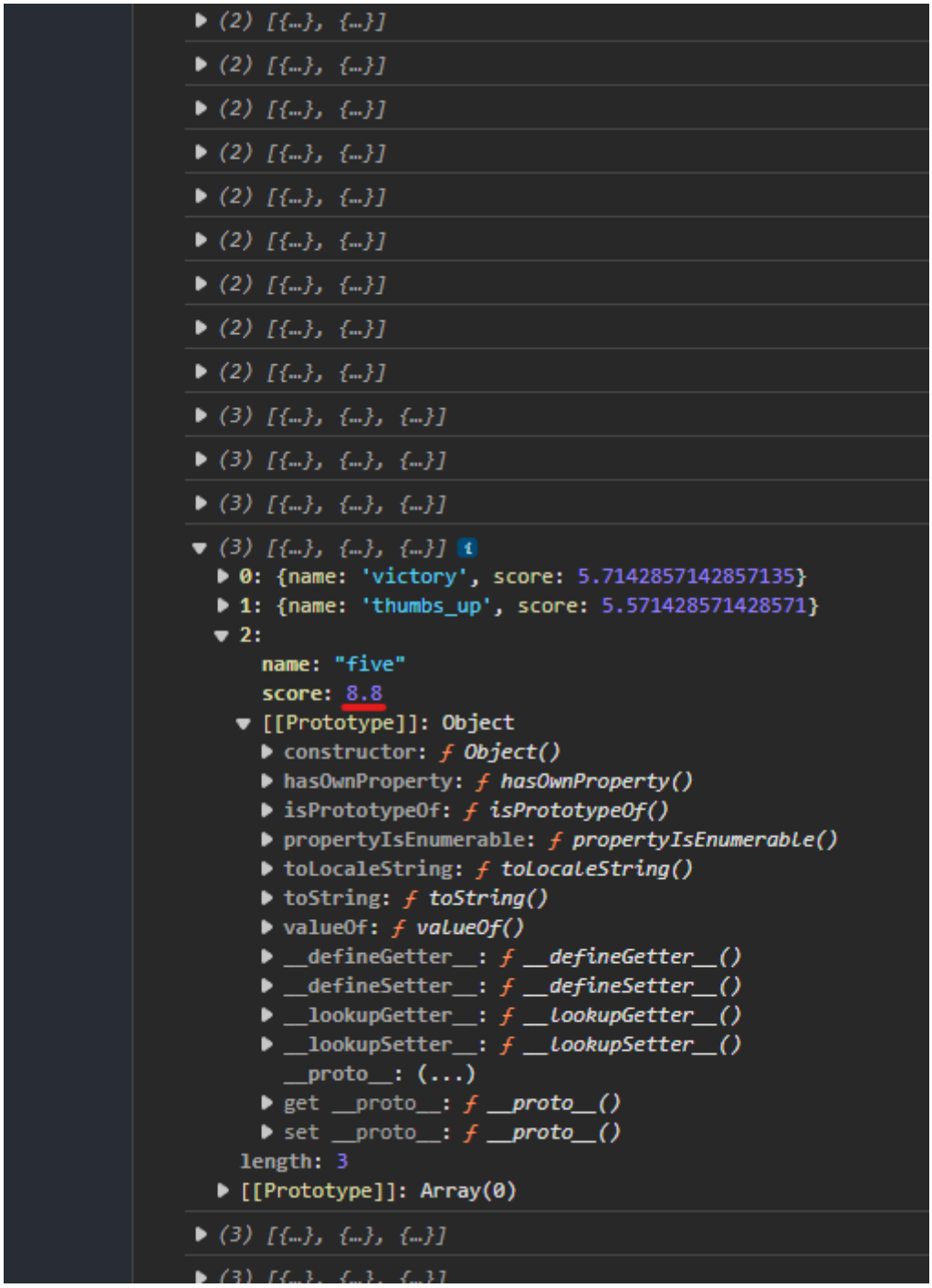


Рис. 3.18 Результат аналізу жесту “п’ять”

У продемонстрованому на рисунку 3.19 прикладі можна побачити ситуацію де одразу 2 жести мають ймовірність 75%+. Хоча це не призводить до невірного результату, адже жест “кулак” має точність у 98% і тому обирається системою як правильний, падіння точності вірного жесту з будь-яких причин може призвести до отримання невірного результату.

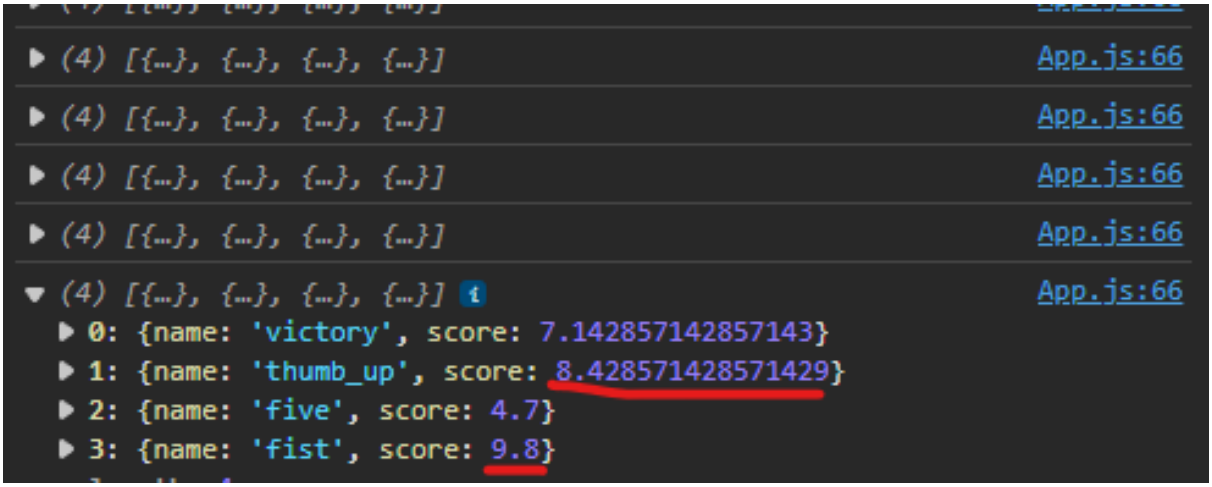


Рис. 3.19 Висока ймовірність одразу для двох жестів

Для запобігання такої ситуації варто імплементувати додаткову підсистему, що буде перевіряти результати роботи головної системи. Для цього було прийнято рішення використовувати лише результати роботи, які впродовж 10 циклів аналізу вказують на однаковий жест. Така перевірка займе лише секунду часу, а отже не перевищить поставлену мету у 3 секунди.

Так як отриманий результат бажано виводи користувачу для підтвердження роботи системи, його було винесено з консолі на екран. Такий результат можна зобразити у формі тексту або зображення, яке показує поточний жест який бачить система.

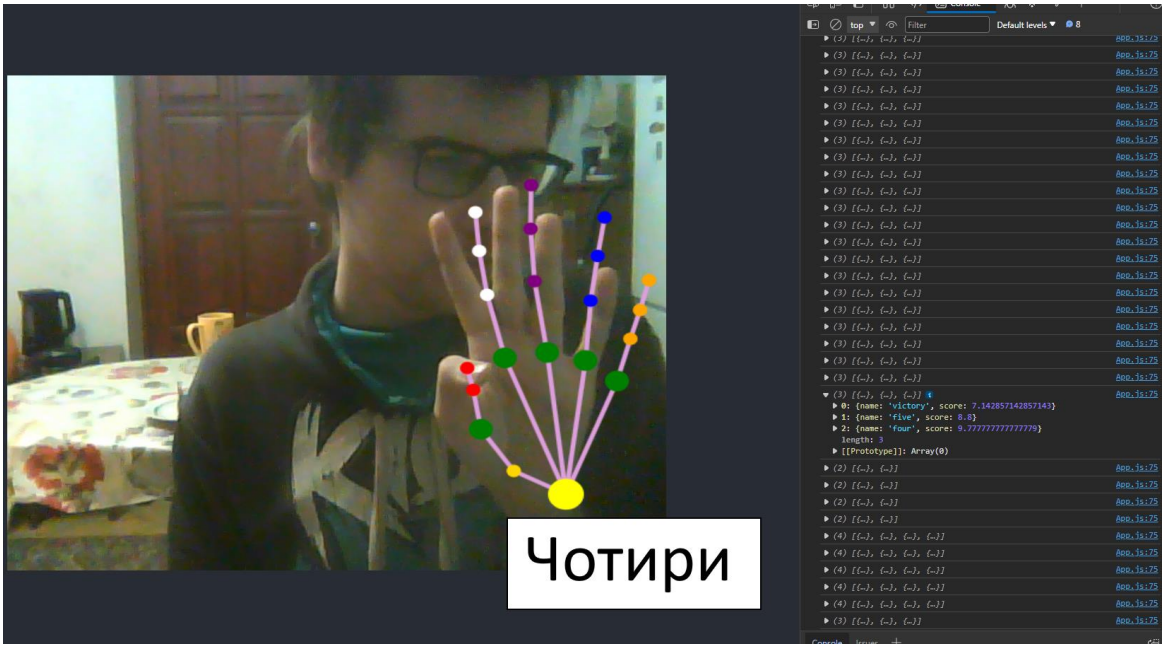


Рис. 3.20 Вивід результату користувачу

3.5 Тестування навчених жестів

З метою тестування результатів роботи мережі було виконано серію тестів , де мережі показувався жест і розглядалась точність її роботи за коефіцієнтом Жаккару.

Умови тестування були наступними:

- Кількість тестів: 100 для кожного жесту, де 90 намагаються отримати справжній позитивний результат, а 10 – справжній негативний. В такій ситуації справжні позитивні результати допоможуть оцінити здатність системи коректно розпізнавати жести, а справжні негативні – здатність не розпізнавати схожі на жест форми.

- Успішність тесту – отримання правильного результату впродовж 3 секунд після початку тестування жесту.

- Підрахунок як Справжніх позитивних, так і справжніх негативних, помилкових позитивних та помилкових негативних.

Результати тестування можна побачити на таблиці 3.1

Таблиця 3.1

Коефіцієнти точності розпізнавання жестів

Жест	Коефіцієнт Жаккару	Чи є схожий жест?
Великий палець вгору	0.98	Ні
Великий палець вниз	0.97	Ні
Долоня	0.64	Так, “П’ять”
Кулак	0.95	Ні
Окей	0.92	Так, “Стиснуті пальці’
Хапаюча рука	0.93	Ні
Стиснуті пальці	0.96	Так, “Окей”
Один	0.89	Ні
Два	0.91	Ні
Три	0.88	Ні
Чотири	0.94	Ні
П’ять	0.81	Так, “Долоня”

На прикладі жесту “Кулак” розглянемо принцип розрахунків використовуючи $J = \frac{TP+TN}{TP+TN+FP+FN}$. Для жесту кулак було отримано TP = 86, TN – 9, FP – 4, FN – 1.

В такому випадку J = (86+9)/(86+9+4+1) = 0.95.

Точність роботи 95% є достатньою для використання у системі розпізнавання жестів, адже отримання 10 жестів підряд для виконання дії є дуже ймовірним.

Такі ж розрахунки були проведені і для інших жестів. Загалом, система показала задовільні результати, з деякими виключеннями. До таких результатів можна віднести жести “П’ять” та “Долоня”.

Як можна побачити, до результатів тестування були включені помилкові жести, що найчастіше зустрічалися під час тестування. У випадку з “Окей” та “Стиснуті пальці”, система давала правильному жесту більшу точність, що призводило до вірної роботи системи. Проте в ситуації з жестом “Долоня”. Система часто плутала його з жестом “П’ять” – і, в меншій мірі, навпаки.

Причиною є відносна схожість цих жестів за своєю структурою, що призвело до низької точності. Це може бути виправлено за допомогою:

- Збільшення набору навчальних даних по цим жестам
- Заміна жесту на менш схожий
- Покращення опису жестів за кутом нахилу пальців
- Використання одного з жестів боком

Для вирішення проблеми було обрано саме останній варіант, адже він вимагає найменшої кількості часу для виправлення ситуації. Як результат, точність визначення обох жестів піднялась до допустимої межі у 90%.

Додаткове тестування жестів окей та стиснутих пальців показало, що вони не впливають один на одного достатньою мірою для необхідності внесення змін в систему, а тому вони були залишені у своєму початковому стані.

3.6 Опис розроблених класів

У цьому розділі розглянуто імпортовані функції, побудовані класи та методи що використовуються у системі.

import React, { useRef, useState, useEffect } from "react" – імпорт методів React js

import * as tf from "@tensorflow/tfjs" – імпорт бібліотеки Tensorflow

import Webcam from "react-webcam" – імпорт методу Webcam

import "./App.css" – імпорт додаткової інформації, побудованої під час створення React js додатку

import { drawHand } from "./utilities" – імпорт класу створення графічного зображення руки

import {handFive} from "./HandFive" – імпорт ознак жесту “п’ять”

import {handFist} from "./HandFist" – імпорт ознак жесту “кулак”

import { thumbsUpDescription } from "./ThumbsUp"; – імпорт ознак жесту “великий палець догори”

import * as fp from "fingerpose" – імпорт моделі Fingerpose

function App() – ця функція включає в себе увесь головний функціонал системи жестового керування інтерфейсом. Розглянуті надалі елементи є частиною цієї функції.

const webcamRef = useRef(null); - створення константи для камери

const canvasRef = useRef(null); - створення константи для вебсайту

const runHandpose = async () => {} -клас, що включає в себе запуск моделі Handpose, повідомлення про її запуск, а також встановлює частоту роботи моделі.

const net = await handpose.load(); - завантаження нейронної мережі Handpose

setInterval(() => {detect(net);}, 10); - елемент, що встановлює частоту роботи.

const detect = async (net) => {} – клас, що встановлює налаштування сайту, вебкамери та проводить оцінку точності результатів розпізнавання жестів.

```
if (
    typeof webcamRef.current !== "undefined" &&
    webcamRef.current !== null &&
```

webcamRef.current.video.readyState === 4

) {} – if блок, що працює за умови роботи вебкамери

Елементи блоку:

const video = webcamRef.current.video; - відеопотік

const videoWidth = webcamRef.current.video.videoWidth; - отримання ширини відеопотоку

const videoHeight = webcamRef.current.video.videoHeight; - отримання висоти відеопотоку

webcamRef.current.video.width = videoWidth; - встановлення ширини відеопотоку

webcamRef.current.video.height = videoHeight; - встановлення висоти відеопотоку

canvasRef.current.width = videoWidth; - встановлення ширини відеопотоку

canvasRef.current.height = videoHeight; - встановлення висоти відеопотоку

const hand = await net.estimateHands(video); - активація пошуку руки

console.log(hand); - вивід результатів у консоль (без жестів)

if (hand.length > 0) {} – блок, що активується якщо у відеопотоці є рука

Елементи блоку:

const GE = new fp.GestureEstimator([fp.Gestures.VictoryGesture, thumbsUpDescription, handFive, handFist]); - створення матриці, що включає можливі жести

const gesture = await GE.estimate(hand[0].landmarks, 4); - очікування елементів жесту, активація методу визначення жесту

if (gesture.gestures !== undefined && gesture.gestures.length > 0) {} – if блок, що активується якщо жест визначено

Елементи блоку:

const confidence = gesture.gestures.map((prediction) => prediction.confidence); - розрахунок впевненості у визначеному жесті

const maxConfidence = confidence.indexOf(Math.max.apply(null, confidence)); - визначення жесту з максимальною точністю

console.log(gesture.gestures); - вивід результату визначення жестів у консоль

console.log(gesture.gestures[maxConfidence].name); - вивід жесту з найбільшою впевненістю у консоль

const ctx = canvasRef.current.getContext("2d");

drawHand(hand, ctx); - створення графічного зображення вихідних даних нейромережі та виведення їх на екран.

Кінець App

```
const fingerJoints = {
  thumb: [0, 1, 2, 3, 4],
  indexFinger: [0, 5, 6, 7, 8],
  middleFinger: [0, 9, 10, 11, 12],
  ringFinger: [0, 13, 14, 15, 16],
  pinky: [0, 17, 18, 19, 20],
};
```

- призначення точок до пальців

```
const style = {
  0: { color: "yellow", size: 15 }, 1: { color: "gold", size: 6 }, 2: { color: "green", size: 10 }, 3: { color: "red", size: 6 }, 4: { color: "red", size: 6 }, 5: { color: "green", size: 10 }, 6: { color: "white", size: 6 }, 7: { color: "white", size: 6 }, 8: { color: "white", size: 6 }, 9: { color: "green", size: 10 }, 10: { color: "purple", size: 6 }, 11: { color: "purple", size: 6 }, 12: { color: "purple", size: 6 }, 13: { color: "green", size: 10 }, 14: { color: "blue", size: 6 }, 15: { color: "blue", size: 6 }, 16: { color: "blue", size: 6 }, 17: { color: "green", size: 10 }, 18: { color: "orange", size: 6 }, 19: { color: "orange", size: 6 }, 20: { color: "orange", size: 6 },
};
```

- призначення кольорів кожній з точок

export const drawHand = (predictions, ctx) => {} – функція побудови графічного зображення руки.

if (predictions.length > 0) {

predictions.forEach((prediction) => { - цикл роботи для кожного з обрахувань точності

const landmarks = prediction.landmarks; - вибір точок

for (let j = 0; j < Object.keys(fingerJoints).length; j++) { - цикл опрацювання пальців

3.7 Статистика використання системи

Після закінчення роботи над системою жестового керування було виконано серію тестів з участю користувачів різних груп. Метою тестування було визначити доступність системи для людей з різними фізичними та психологічними можливостями. Через неможливість повноцінного тестування у мережі інтернет, вибірку тестувальників системи було зменшено до 80 осіб, проте цього було достатньо для розгляду наступних груп:

- Користувачі похилого віку
- Користувачі з порушенням моторики рук
- Користувачі, що вже мали досвід роботи з системами жестового керування
- Випадкова вибірка, користувачі з якої не мають певних особливостей у роботі з системою.

Показники, отримані від результатів тестування мають показати можливості користувачів у користуванні системою для різних груп самих користувачів – як для людей що мають переваги (в цьому випадку – досвід роботи), так і для людей які можуть мати проблеми з розумінням принципу роботи системи або складності з виконанням жестів.

Для аналізу результатів роботи системи, тестування проходило за наступними категоріями:

- Час, необхідний для розуміння принципу роботи системи
- Час, необхідний системі для розпізнавання жесту
- Процент вірно розпізнаних жестів

Такі показники було обрано так як час для розпізнавання жесту та процент вірно розпізнаних жестів дозволяють зрозуміти технічні можливості системи при використанні користувачами, а час необхідний для розуміння принципу роботи системи описує її доступність для користувачів – що є важливою складовою інклюзивного дизайну.

Таблиця 3.2

Результати тестування системи

Тип користувача	Час, необхідний для розуміння принципу роботи системи	Час, необхідний системі для розпізнавання коректного жесту	Вірно розпізнані жести
Випадкова вибірка	< 10с.	~ 2с.	98%
Користувачі, що мають досвід роботи з жестовим керуванням	< 5с.	~ 2с.	99%
Користувачі похилого віку	~ 1хв.	~ 4с.	96%

Продовження таблиці 3.2

Тип користувача	Час, необхідний для розуміння принципу роботи системи	Час, необхідний системі для розпізнавання коректного жесту	Вірно розпізнані жести
Користувачі з порушенням моторики руху	< 10с.	~ 5с.	95%

Результати тестування можна побачити на таблиці 3.2. Як можна побачити з даних, новому користувачу, в середньому, необхідно 10 секунд для того щоб ознайомитись з можливостями системи та почати працювати з нею. Користувачі що вже використовували інші системи жестового керування здатні зрозуміти системи значно швидше – їм необхідно лише приблизно п’ять секунд для того, щоб почати з нею працювати. Цей час, здебільшого, витрачається на вивчення жестів. Ситуація є іншою для користувачів похилого віку – назвати одну цифру доволі складно, адже такі користувачі можуть зрозуміти принцип користування системою як за 10 секунд, так і за декілька хвилин. Середнім числом було обрано 1 хвилину. Такий час може стати проблематичним у ситуації реального використання та призвести до черг. Ця проблема може бути вирішена розміщенням графічних інструкцій що до принципу керування системою.

Час, необхідний системі для розпізнавання жесту в багатьох випадках залежав від моторики рук користувача, що призвело до повільнішого розпізнавання жестів користувачів похилого віку та людей з порушеннями моторики рук. Для таких користувачів максимальний час розпізнавання жесту було піднято до 7 секунд.

Кількість вірно розпізнаних жестів у процесі тестування відрізняється від очікуваних показників, проте все ще знаходиться в допустимому діапазоні значень для звичайних користувачів. Варто відзначити, що результати для користувачів з

порушенням моторики є відносно низькими і становлять 95%, що може призвести до періодичного виконання невірної команди.

Загалом, результати тестування можна визнати задовільними. Система підвищення інклюзивності за рахунок жестового інтерфейсу змогла досягнути високих показників точності та швидкодії, зберігаючи свою доступність та зрозумілість для користувачів.

ВИСНОВКИ

В результаті виконання магістерської роботи було розроблено метод та програмні засоби підтримки інклюзивної взаємодії на основі жестового інтерфейсу з використанням методів машинного навчання.

При виконанні роботи було виконано наступні задачі:

1. Проведено огляд існуючих нейронних мереж, їх видів та особливостей. Розглянуто можливості таких мереж у підвищенні інклюзивності користувачів шляхом зчитування жестів та використання цієї інформації для керування інтерфейсами.
2. Визначено найбільш вагомі вимоги користувачів та надавачів послуг до жестового інтерфейсу на основі нейронних мереж. Проведено аналіз мір, які можуть допомогти досягнути поставлених цілей у швидкодії, вірності результатів та зрозумілості використання для звичайного користувача. Сформовано список вимог до системи на основі побажань користувачів. Побажання було сформовано на основі 980 відгуків отриманих з опитувань в соціальних мережах.
3. Створено графічний опис системи на основі визначених вимог, проаналізовано можливі засоби для створення такої системи.
4. Побудована модель розпізнавання жестів на основі бібліотеки нейронних мереж TensorFlow за допомогою JavaScript, HTML та методів машинного навчання.
5. Розроблено метод підтримки інклюзивної взаємодії на основі жестового інтерфейсу з використанням елементів машинного навчання.
6. Проведено тестування створеної системи розпізнавання жестів на прикладі інтерфейсу терміналів мереж фаст-фудів. Зібрано статистику щодо роботи мережі у тестових ситуаціях з різними групами користувачів, вибірка становила 88 осіб та була отримана на основі відгуків користувачів з мережі інтернет. Результати експериментального долідження підтверджують високу ефективність системи і показали, що для користувачів з порушенням

моторики руху правильність розпізнавання жестів становить 95%, при цьому час розпізнавання становить близько 5 с. Відсоток вірно розпізнаних жестів для користувачів похилого віку становить 96% при середньому часі розпізнавання близько 4 секунд. Найкращій результат система демонструє для користувачів, що вже мають певний досвід роботи з жестовим керуванням - 99% вірно розпізнаних жестів при середньому часі розпізначання близько 2 с.

ПЕРЕЛІК ПОСИЛАНЬ

1. Chandrahas Mishra, D. L. Gupta. Deep Machine Learning and Neural Networks: An Overview. IAES International Journal of Artificial Intelligence (IJ-AI). 2017. Vol. 6.

2. Sampath Kumar Gajawada. The Math behind Artificial Neural Networks. Medium. URL: https://towardsdatascience.com/the-heart-of-artificial-neural-networks-26627e8c03ba.

3. Krishnan S. How do determine the number of layers and neurons in the hidden layer?. Medium. URL: https://medium.com/geekculture/introduction-to-neural-network-2f8b8221fbd3.

4. Abdel-Nasser Sharkawy. Principle of Neural Network and Its Main Types: Review. Journal of Advances in Applied & Computational Mathematics. 2020. Vol. 7, no. 1.

5. Saha S. A Comprehensive Guide to Convolutional Neural Networks–the ELI5 way. Medium. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

6. Abien Fred Agarap. Deep Learning using Rectified Linear Units (ReLU). 2018

7. Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do Kaori Togashi. Convolutional neural networks: an overview and application in radiology. Insights into Imaging. 2018. Vol. 9.

8. Designing for Inclusivity: AI's Role in Accessibility. https://wolfpack-digital.com. URL: https://www.wolfpack-digital.com/blogposts/designing-for-inclusivity-ai-s-role-in-accessibility

9. Understanding Artificial Intelligence – and how it affects the disability community. - European Disability Forum. European Disability Forum. URL: https://www.edf-feph.org/understanding-artificial-intelligence-and-how-it-affects-the-disability-community/

10. AI for inclusive innovation. Board of Innovation. URL: https://www.boardofinnovation.com/blog/ai-for-inclusive-innovation/

11. AI and human enchancement: Americans’ openness is tempered by a range of comcerns. Pew Research Center, 2022

12. Clarkson J. Inclusive Design: Design for the Whole Population : монографія. Springer Science & Business Media, 2003. 608 p.

13. Lindberg O. Inclusive Design: 12 Ways to Design for Everyone. Shopify. URL: https://www.shopify.com/partners/blog/inclusive-design (date of access: 19.11.2023).

14. Nikhil S. Gesture Recognition Market. Market Research Reports, Business Consulting | In-depth Insights | TMR. URL: https://www.transparencymarketresearch.com/gesture-recognition-market.html.

15. Yuejiao Wang, Zhanjun Hao. UltrasonicGS: A Highly Robust Gesture and Sign Language Recognition Method Based on Ultrasonic Signals. Sensors (Basel). 2023. Vol. 23, no. 4.

16. Fast Food Chains and the Kiosk Revolution. Synergy Restaurant Consultants. URL: https://www.synergyconsultants.com/fast-food-chains-and-the-kiosk-revolution-enhancing-customer-experience-and-efficiency/

17. HOW TRAINING DATA AFFECT THE ACCURACY AND ROBUSTNESS OF NEURAL NETWORKS FOR IMAGE CLASSIFICATION. ICLR. 2019.

18. Design for service inclusion: creating inclusive service systems by 2050 / Raymond P. Fisk et al. Journal of Service Management. 2018. Vol. 29, no. 3.

19. Учасники проектів Вікімедіа. Коефіцієнт Жаккара – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Коефіцієнт_Жаккара.

20. Wajeeha Ahmed, Areeshia Chaudhary, Gulfraz Naqvi. Role of Artificial Neural Networks in AI. 2023.

21. Joyce A. Inclusive Design. Nielsen Norman Group. URL: https://www.nngroup.com/articles/inclusive-design/ (date of access: 19.11.2023).

22. Mohammad Mustafa Taye. Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions. Data Science and Artificial Intelligence, Philadelphia University. 2023. Vol. 11, no. 3.

23. AI for inclusive innovation. Board of Innovation. URL: https://www.boardofinnovation.com/blog/ai-for-inclusive-innovation/

24. React – JavaScript-бібліотека для створення користувацьких інтерфейсів. React – JavaScript-бібліотека для створення користувацьких інтерфейсів. URL: https://uk.legacy.reactjs.org/

25. Overview of React.js. Patterns.dev. URL: https://www.patterns.dev/react/

26. TensorFlow. TensorFlow. URL: https://www.tensorflow.org/?hl=en.

25. How Does TensorFlow Work and Why is it Vital for AI? - Spiceworks. Spiceworks. URL: https://www.spiceworks.com/tech/devops/articles/what-is-tensorflow/#:~:text=TensorFlow%20enables%20developers%20to%20design,a%20multi-layered%20data%20array.

26. Contributors to Wikimedia projects. The Magical Number Seven, Plus or Minus Two - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/The_Magical_Number_Seven,_Plus_or_Minus_Two

27. Hugo Larochelle, Y. Bengio, Jérôme Louradour Pascal Lamblin. Exploring Strategies for Training Deep Neural Networks. Journal of Machine Learning Research. 2009

28. Hand Gesture Recognition Database. Kaggle: Your Machine Learning and Data Science Community. URL: https://www.kaggle.com/datasets/gti-upm/leapgestrecog/.

29. Jason Brownlee. How Much Training Data is Required for Machine Learning? - MachineLearningMastery.com. MachineLearningMastery.com. URL: https://machinelearningmastery.com/much-training-data-required-machine-learning/.

30. Н.І Недашківська. МЕТОДИ І МОДЕЛІ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ. Електронне мережне навчальне видання. 2020. с. 53-55.

Додаток А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

МАГІСТЕРСЬКА РОБОТА

«РОЗРОБКА МЕТОДУ ТА ПРОГРАМНИХ ЗАСОБІВ ПІДТРИМКИ ІНКЛЮЗИВНОЇ ВЗАЄМОДІЇ НА ОСНОВІ ЖЕСТОВОГО ІНТЕРФЕЙСУ З ВИКОРИСТАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ»

Виконав: студент групи ПДМ-62, Панібратов Андрій Іванович

Керівник: к.т.н., доц., доцент кафедри ІПЗ Золотухіна Оксана Анатоліївна

Київ - 2023

1

МЕТА, ОБ’ЄКТ, ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: підвищення рівня інклюзивності взаємодії «людина-комп’ютер» на основі жестового інтерфейсу за рахунок використання методів машинного навчання

Об’єкт дослідження: інклюзивна взаємодія «людина-комп’ютер» на основі жестового інтерфейсу

Предмет дослідження: методи та засоби забезпечення інклюзивної взаємодії «людина-комп’ютер» на основі жестового інтерфейсу

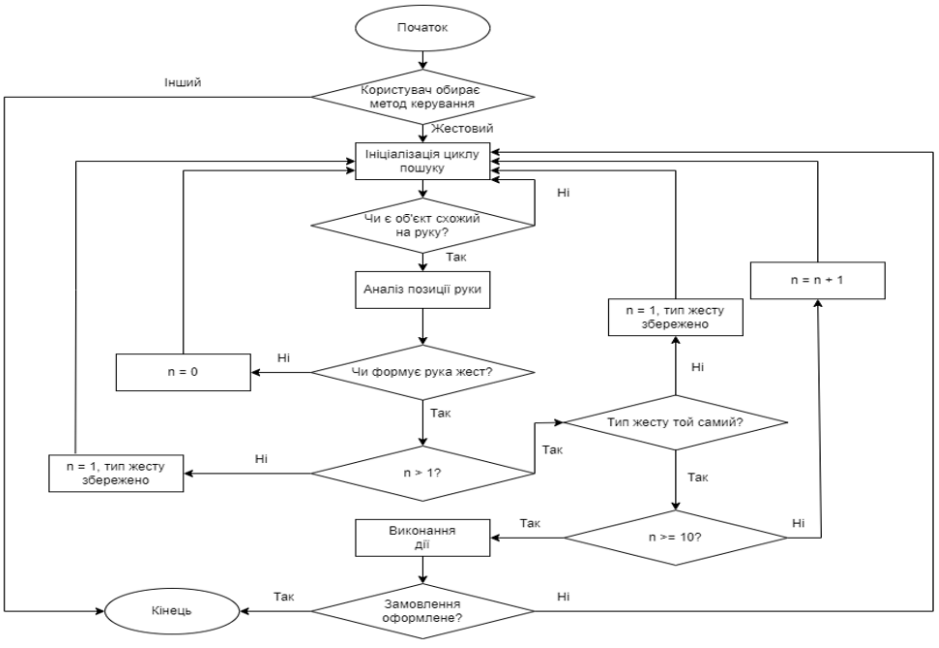
2

ОБҐРУНТУВАННЯ ВИБОРУ НЕЙРОННОЇ МЕРЕЖІ

Можливі підходи	Переваги	Недоліки
Розробка та тренування власної нейронної мережі з нуля	+ Висока точність результатів + Низькі технічні вимоги кінцевого продукту	- Надмірні фінансові витрати - Необхідність у потужному технічному обладнанні - Великий об’єм даних для навчання
Використання повністю готової та навченої нейронної мережі	+ Швидко у налаштуванні + Низькі витрати під час розробки	- Низька адаптивність - Можливі додаткові витрати під час експлуатації - Висока вразливість до хакерських атак
Завантаження базової нейронної мережі та її подальше навчання	+ Низькі витрати під час розробки + Висока точність результатів + Можливість швидкої адаптації до нових вимог	- Великий об’єм даних для навчання

3

АЛГОРИТМ ОФОРМЛЕННЯ ЗАМОВЛЕННЯ З ВИКОРИСТАННЯМ ЖЕСТОВОГО ІНТЕРФЕЙСУ



4

НАВЧАННЯ НЕЙРОННОЇ МОДЕЛІ

Функціонування штучного нейрону: $y = \psi\ (\varphi(w, x))$

Функції вхідних сигналів:

Зважена сума – $\varphi(w,x) = \sum_{j=1}^{N} w_j x_j + w_0$

Лінійна – $\psi(x) = cx$

Функції активації:

Порогова – $\psi(x) = \begin{cases} 1, x \geq 0; \\ 0, x < 0; \end{cases}$

Сигмоїдна логістична – $\psi(x) = \frac{1}{1+e^{-cx}}$

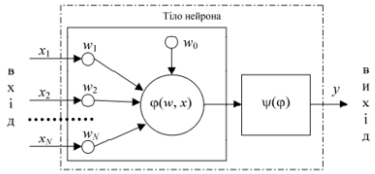


Рис. 1: Схема роботи штучного нейрону

5

МАТЕМАТИЧНА МОДЕЛЬ ВИЗНАЧЕННЯ ПОЛОЖЕННЯ РУКИ

Пошук руки

Формула обчислення положень всіх пікселів руки: $L = (\vec{x} \mid S(r(\vec{x}), g(\vec{x}), b(\vec{x})) = 1)$

Центроїди

- руки: $\bar{c}_p = \frac{1}{|L|} \sum_{\bar{x} \in L} \bar{x}$

- зап'ястя: $\bar{c}_з = \frac{1}{|L_з|} \sum_{\bar{x} \in L_з} \bar{x}$

Пошук зап'ястя

Вектор визначення центроїдів: $\bar{c}_{dif} = (x_{dif}, y_{dif}) = \bar{c}_p - \bar{c}_з$

Кут повороту руки: $\theta_p = \tan^{-1}\left(\frac{y_{dif}}{x_{dif}}\right)$

Виділення пальців

Поточний кут напрямку пальця: $\theta = \boldsymbol{arctan}(v_{maj} / u_{maj})$

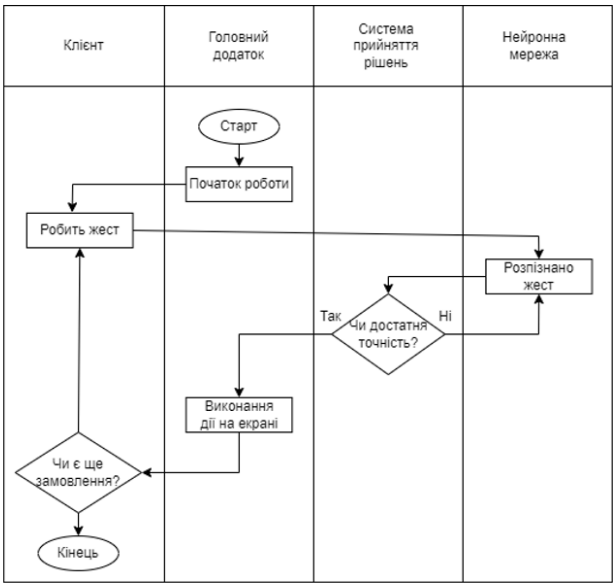
Обертання пальцю: $R = \begin{bmatrix} \boldsymbol{cos}(\Delta\theta) & -\boldsymbol{sin}(\Delta\theta) \\ \boldsymbol{sin}(\Delta\theta) & \boldsymbol{cos}(\Delta\theta) \end{bmatrix}$

Визначення точності роботи

Коефіцієнт Жаккара: $J = \frac{TP+TN}{TP+TN+FP+FN}$

6

СХЕМА ІМПЛЕМЕНТАЦІЇ СИСТЕМИ РОЗПІЗНАВАННЯ ЖЕСТІВ У ВЖЕ ІСНУЮЧІ ДОДАТКИ



7

ТЕХНІЧНІ ВИМОГИ ДО СИСТЕМИ ІНКЛЮЗИВНОЇ ВЗАЄМОДІЇ НА ОСНОВІ ЖЕСТОВОГО ІНТЕРФЕЙСУ

Користувач:

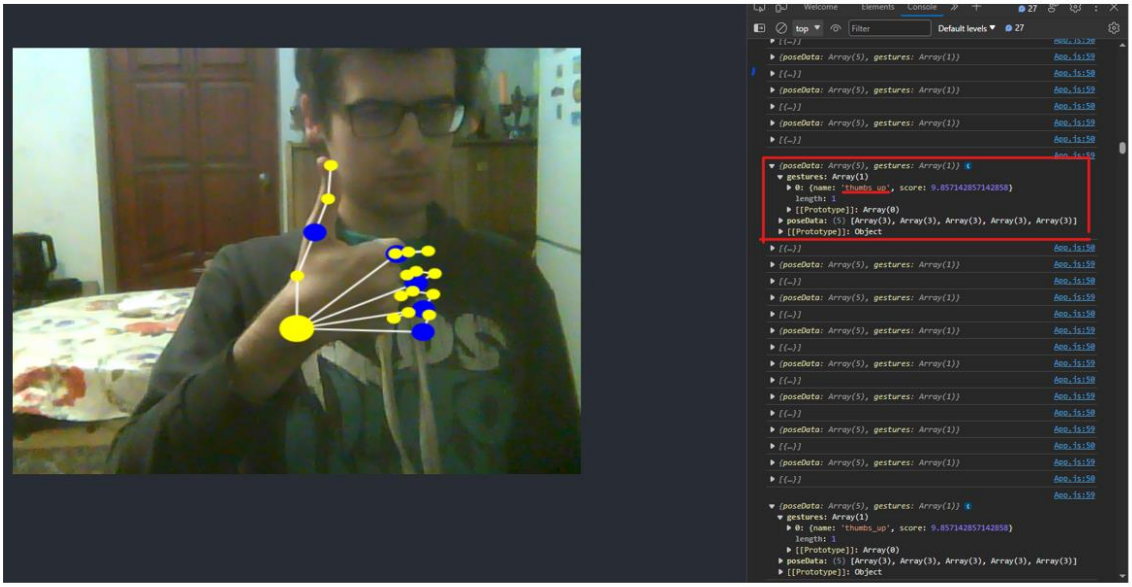
- Швидкість розпізнавання жесту: 3-10 секунд
- Похибка розпізнавання жесту: <= 1 %
- Відстань розпізнавання жесту: 0-2 метри
- Можливість обрати іншу систему не-жестовим інтерфейсом

Виробник:

- Можливість використання на технічно слабких терміналах
- Можливість швидкої та дешевої імплементації до вже існуючих систем
- Можливість адаптації системи до нових потреб

8

ВИЗНАЧЕННЯ ЖЕСТУ



9

РЕЗУЛЬТАТИ ТЕСТУВАННЯ

Тип користувача	Час, необхідний для розуміння принципу роботи системи	Час, необхідний системі для розпізнавання коректного жесту	Вірно розпізнані жести
Випадкова вибірка	< 10 секунд	до 2 секунд	98%
Користувачі, що мають досвід роботи з жестовим керуванням	< 5 секунд	до 2 секунд	99%
Користувачі похилого віку	~ 1 хвилина	до 4 секунд	96%
Користувачі з порушенням моторики руху	< 10 секунд	до 5 секунд	95%

10

ВИСНОВКИ

1. Проведено огляд існуючих нейронних мереж, їх видів та особливостей. Розглянуто можливості таких мереж у підвищенні інклюзивності користувачів шляхом зчитування жестів та використання цієї інформації для керування інтерфейсами.
2. Визначено найбільш вагомі вимоги користувачів та надавачів послуг до жестового інтерфейсу на основі нейронних мереж. Проведено аналіз мір, які можуть допомогти досягнути поставлених цілей у швидкодії, вірності результатів та зрозумілості використання для звичайного користувача. Сформовано список вимог до системи на основі побажань користувачів. Побажання було сформовано на основі 980 відгуків отриманих з опитувань в соціальних мережах.
3. Створено графічний опис системи на основі визначених вимог, проаналізовано можливі засоби для створення такої системи.
4. Побудована модель розпізнавання жестів на основі бібліотеки нейронних мереж TensorFlow за допомогою JavaScript, HTML та методів машинного навчання.
5. Розроблено метод підтримки інклюзивної взаємодії на основі жестового інтерфейсу з використанням елементів машинного навчання.
6. Проведено тестування створеної системи розпізнавання жестів на прикладі інтерфейсу терміналів мереж фаст-фудів. Зібрано статистику щодо роботи мережі у тестових ситуаціях з різними групами користувачів, вибірка становила 88 осіб та була отримана на основі відгуків користувачів з мережі інтернет. Результати експериментального долідження підтверджують високу ефективність системи і показали, що для користувачів з порушенням моторики руху правильність розпізнавання жестів становить 95%, при цьому час розпізнавання становить близько 5 с. Відсоток вірно розпізнаних жестів для користувачів похилого віку становить 96% при середньому часі розпізнавання близько 4 секунд. Найкращій результат система демонструє для користувачів, що вже мають певний досвід роботи з жестовим керуванням - 99% вірно розпізнаних жестів при середньому часі розпізначання близько 2 с.

11

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей:

1. Золотухіна О.А, Панібратов А.І. Використання штучного інтелекту у дизайні програмного забезпечення з метою підвищення інклюзивності - Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і Світу - Київ: ДУТ, 2023. , С – 295.
2. Золотухіна О.А, Панібратов А.І. Огляд та аналіз методів для підвищення інклюзивності у сфері надання послуг - Науковий простір: аналіз, сучасний стан, тренди та перспективи– Івано-Франківськ: Молодіжна Наукова Ліга, 2023., С – 370.

12

ДЯКУЮ ЗА УВАГУ!

Додаток Б

ФРАГМЕНТИ ОСНОВНИХ ПРОГРАМНИХ МОДУЛІВ

Приклад фрагменту опису пальцю

```
import { Finger, FingerCurl, FingerDirection, GestureDescription } from 'fingerpose';
export const handFive = new GestureDescription('five');
//Thumb
handFive.addCurl(Finger.Thumb, FingerCurl.NoCurl, 1.0);
handFive.addDirection(Finger.Thumb, FingerDirection.VerticalUp, 1.0);
handFive.addDirection(Finger.Thumb, FingerDirection.DiagonalUpLeft, 0.9);
handFive.addDirection(Finger.Thumb, FingerDirection.DiagonalUpRight, 0.9);
//Index
handFive.addCurl(Finger.Index, FingerCurl.NoCurl, 1.0);
handFive.addDirection(Finger.Index, FingerDirection.VerticalUp, 1.0);
handFive.addDirection(Finger.Index, FingerDirection.DiagonalUpLeft, 0.9);
handFive.addDirection(Finger.Index, FingerDirection.DiagonalUpRight, 0.9);
//Middle
handFive.addCurl(Finger.Middle, FingerCurl.NoCurl, 1.0);
handFive.addDirection(Finger.Middle, FingerDirection.VerticalUp, 1.0);
handFive.addDirection(Finger.Middle, FingerDirection.DiagonalUpLeft, 0.9);
handFive.addDirection(Finger.Middle, FingerDirection.DiagonalUpRight, 0.9);
//Ring
handFive.addCurl(Finger.Ring, FingerCurl.NoCurl, 1.0);
handFive.addDirection(Finger.Ring, FingerDirection.VerticalUp, 1.0);
handFive.addDirection(Finger.Ring, FingerDirection.DiagonalUpLeft, 0.9);
handFive.addDirection(Finger.Ring, FingerDirection.DiagonalUpRight, 0.9);
//Pinky
handFive.addCurl(Finger.Pinky, FingerCurl.NoCurl, 1.0);
handFive.addDirection(Finger.Pinky, FingerDirection.VerticalUp, 1.0);
handFive.addDirection(Finger.Pinky, FingerDirection.DiagonalUpLeft, 0.9);
handFive.addDirection(Finger.Pinky, FingerDirection.DiagonalUpRight, 0.9);
```

Код APP

```
function App() {
  const webcamRef = useRef(null);
```

```
  const canvasRef = useRef(null);
  const runHandpose = async () => {
    const net = await handpose.load();
    console.log("Handpose model loaded.");
    setInterval(() => {
      detect(net);
    }, 10);
  };
  const detect = async (net) => {
    if (
      typeof webcamRef.current !== "undefined" &&
      webcamRef.current !== null &&
      webcamRef.current.video.readyState === 4
    ) {
      const video = webcamRef.current.video;
      const videoWidth = webcamRef.current.video.videoWidth;
      const videoHeight = webcamRef.current.video.videoHeight;
      webcamRef.current.video.width = videoWidth;
      webcamRef.current.video.height = videoHeight;
      canvasRef.current.width = videoWidth;
      canvasRef.current.height = videoHeight;
      const hand = await net.estimateHands(video);
      console.log(hand);
      if (hand.length > 0) {
        const GE = new fp.GestureEstimator([
          thumbsUpDescription,
          thumbsDownDescription,
          handPalmSide,
          handFive,
          handFour,
          handThree,
          fp.Gestures.VictoryGesture,
          handOne,
          handOkay,
          handPinched,
          handGrab,
          handFist
        ]);
        const gesture = await GE.estimate(hand[0].landmarks, 4);
        if (gesture.gestures !== undefined && gesture.gestures.length > 0) {
          console.log(gesture.gestures);
          const confidence = gesture.gestures.map(
            (prediction) => prediction.confidence
          );
          const maxConfidence = confidence.indexOf(
            Math.max.apply(null, confidence)
          );
          //console.log(gesture.gestures[maxConfidence].name);
        }
      }
      const ctx = canvasRef.current.getContext("2d");
      drawHand(hand, ctx);
    }
  };

return (
    <div className="App">
```

```
    <header className="App-header">
     <Webcam
      ref={webcamRef}
      style={{
       position: "absolute",
       marginLeft: "auto",
       marginRight: "auto",
       left: 0,
       right: 0,
       textAlign: "center",
       zindex: 9,
       width: 1080,
       height: 700,
      }}
     />

     <canvas
      ref={canvasRef}
      style={{
       position: "absolute",
       marginLeft: "auto",
       marginRight: "auto",
       left: 0,
       right: 0,
       textAlign: "center",
       zindex: 9,
       width: 1080,
       height: 700,
      }}
     />
     {emoji !== null ? (
      <img
       src={images[emoji]}
       style={{
        position: "absolute",
        marginLeft: "auto",
        marginRight: "auto",
        left: 400,
        bottom: 500,
        right: 0,
        textAlign: "center",
        height: 100,
       }}
      />
     ) : (
      ""
     )}
    </header>
   </div>
  );
```

Зміст файлу utilities

```
const fingerJoints = {
  thumb: [0, 1, 2, 3, 4],
  indexFinger: [0, 5, 6, 7, 8],
  middleFinger: [0, 9, 10, 11, 12],
```

```
  ringFinger: [0, 13, 14, 15, 16],
  pinky: [0, 17, 18, 19, 20],
};
const style = {
  0: { color: "yellow", size: 15 },
  1: { color: "gold", size: 6 },
  2: { color: "green", size: 10 },
  3: { color: "red", size: 6 },
  4: { color: "red", size: 6 },
  5: { color: "green", size: 10 },
  6: { color: "white", size: 6 },
  7: { color: "white", size: 6 },
  8: { color: "white", size: 6 },
  9: { color: "green", size: 10 },
  10: { color: "purple", size: 6 },
  11: { color: "purple", size: 6 },
  12: { color: "purple", size: 6 },
  13: { color: "green", size: 10 },
  14: { color: "blue", size: 6 },
  15: { color: "blue", size: 6 },
  16: { color: "blue", size: 6 },
  17: { color: "green", size: 10 },
  18: { color: "orange", size: 6 },
  19: { color: "orange", size: 6 },
  20: { color: "orange", size: 6 },
};
export const drawHand = (predictions, ctx) => {
  if (predictions.length > 0) {
    predictions.forEach((prediction) => {
      const landmarks = prediction.landmarks;
      for (let j = 0; j < Object.keys(fingerJoints).length; j++) {
        let finger = Object.keys(fingerJoints)[j];
        for (let k = 0; k < fingerJoints[finger].length - 1; k++) {
          const firstJointIndex = fingerJoints[finger][k];
          const secondJointIndex = fingerJoints[finger][k + 1];
          ctx.beginPath();
          ctx.moveTo(
            landmarks[firstJointIndex][0],
            landmarks[firstJointIndex][1]
          );
          ctx.lineTo(
            landmarks[secondJointIndex][0],
            landmarks[secondJointIndex][1]
          );
          ctx.strokeStyle = "plum";
          ctx.lineWidth = 4;
          ctx.stroke();
        }
      }

      for (let i = 0; i < landmarks.length; i++) {
        const x = landmarks[i][0];
        const y = landmarks[i][1];
        ctx.beginPath();
        ctx.arc(x, y, style[i]["size"], 0, 3 * Math.PI);
        ctx.fillStyle = style[i]["color"];
        ctx.fill();
```

```
    }
   });
 }
```

Налашування App js

```
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}
```