МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавра
(ступінь вищої освіти (освітній ступінь))

на тему «РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ НЕЧІТКОГО ГОЛОСУ ЛЮДЕЙ НА ОСНОВІ НЕЙРОМЕРЕЖІ»

Виконав: студент 4 курсу,групи КНТ-512сп
спеціальності 123 Комп'ютерна інженерія

Освітня програма (спеціалізація)
Комп'ютерна інженерія
(код і назва спеціальності)

ЯЦЕНКО К.О.
(прізвище та ініціали)
Керівник ІЛЬЯШЕНКО М.Б.
(прізвище та ініціали)
Рецензент КОЗІНА Г.Л. .
(прізвище та ініціали)

2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Запорізька політехніка»

(повне найменування вищого навчального закладу)

Факультет Комп'ютерних наук і технологій

Кафедра «Комп'ютерні системи та мережі»

Ступінь вищої освіти (освітній ступінь) бакалаврський

Спеціальність 123 Комп'ютерна інженерія

(код і назва)

Освітня програма (спеціалізація) Комп'ютерна інженерія

(назва)

ЗАТВЕРДЖУЮ
Зав. кафедри
_____ Кудерметов Р.К.
"14" квітня 2025 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ СТУДЕНТУ

ЯЦЕНКО Клим Олегович

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) «Розробка системи розпізнавання нечіткого голосу людей на основі нейромережі»

керівник проекту (роботи) ІЛЬЯШЕНКО М.Б., к. т. н., доцент

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

затверджені наказом вищого навчального закладу від "08" квітня 2025 року № 151

2. Строк подання студентом проекту (роботи) 01.06. 2025 року

3. Вихідні дані до проекту (роботи) мову програмування C#, Microsoft Visual Studio, нейромережа, карти Кохонена

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

1) Аналіз засобів перевірки користувачів

2) Вибір засобів для розробки системи

3) Розробка системи розпізнавання нечіткого голосу

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Слайди презентації

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-3	ІЛЬЯШЕНКО М.Б., к.т.н.. доцент		
Нормоконтроль	ЩЕРБАК Н.В., ст. викладач		

7. Дата видачі завдання 05.05. 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз засобів перевірки користувачів	05.04.2025	
2	Аналіз біометричних особливостей	10.04.2025	
3	Аналіз розпізнавання	17.04.2025	
4	Вибір засобів для розробки системи	20.04.2025	
5	Вибір мови програмування системи	25.04.2025	
6	Розробка системи розпізнавання	06.05.2025	
7	Розробка програмної частини	15.05.2025	
8	Оформлення отриманих результатів у ПЗ	26.05.2025	
9	Захист роботи	09.06.2025	

Студент _____ **Клим ЯЦЕНКО**
(підпис) (ініціали та прізвище)

Керівник проекту (роботи) _____ **Матвій ІЛЬЯШЕНКО**
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: ПЗ 59 с., 39 рис., 1 табл., 22 джерела.

ІНФОРМАЦІЙНА СИСТЕМА, НЕЙМРОМЕРЕЖА МЕРЕЖА, ГОЛОСОВА ІНДЕТИФІКАЦІЯ

У даній роботі проведено аналіз та реалізовано програмний засіб, призначений для покращення методу розпізнавання нечіткого мовлення осіб із порушеннями вимови окремих звуків, з використанням нейронних мереж.

У процесі виконання дипломної роботи детально досліджено теоретичні основи розпізнавання мовлення за допомогою нейромережевих технологій. Розглянуто методи розпізнавання мови, відповідні параметри та ознаки, що дозволяють нейронним мережам робити висновки щодо мовця. Також було вивчено базові поняття, структуру та класифікацію нейронних мереж.

На основі опрацьованого теоретичного матеріалу реалізовано практичну частину — програмний комплекс для ідентифікації людського голосу. Розробка здійснювалася мовою програмування C# у середовищі Visual Studio.

ABSTRACT

Explanatory note to the master’s work: 59 p., 39 figures, 1 tables, 22 sources.

INFORMATION SYSTEM, NEURAL NETWORK, VOICE IDENTIFICATION.

This work analyzes and implements a software tool designed to improve the method of recognizing fuzzy speech of people with disorders of pronunciation of certain sounds using neural networks.

In the course of the thesis, the theoretical foundations of speech recognition using neural network technologies were studied in detail. Speech recognition methods, relevant parameters, and features that allow neural networks to draw conclusions about the speaker are considered. The basic concepts, structure, and classification of neural networks were also studied.

Based on the theoretical material, the practical part was implemented - a software system for human voice identification. The development was carried out in the C# programming language in the Visual Studio environment.

ЗМІСТ

ВСТУП

Актуальність. Сьогодні спостерігається зростання випадків несанкціонованого доступу в сфері інформаційної безпеки. Зловмисники легко отримують доступ до захищених об’єктів, обходячи системи захисту та зламують паролі. У зв’язку з цим традиційні засоби аутентифікації поступаються місцем біометричним методам. Хоча більшість таких методів є більш надійними, їх використання поки не набуло повсюдного поширення.

Мова є ключовим елементом людської діяльності, що дозволяє пізнавати світ і передавати знання. Усне мовлення реалізується за допомогою голосового апарату. Кожна людина має унікальні голосові характеристики, які залежать від фізіологічних особливостей. Якщо людина може розрізняти голоси інтуїтивно, для комп’ютерних систем це є складним завданням.

Завдання розпізнавання голосу порушувалося ще понад 40 років тому, проте й досі залишаються актуальними дослідження в цій галузі. Попри зростання точності мовного розпізнавання, задача автоматичної ідентифікації диктора в різних умовах залишається відкритою. Тому важливими є як удосконалення існуючих алгоритмів, так і пошук нових підходів.

Розпізнавання особи за голосом зводиться до виділення, класифікації та реакції на мовлення у вхідному звуковому потоці. Виділяють дві підзадачі:

- ідентифікація - встановлення особи шляхом порівняння голосу з наявними шаблонами;

- верифікація - підтвердження особи.

Методами голосової автентифікації займалися такі науковці, як Астаф’єва Н. М., Дремін І. М., Єрмоленко Т. В. та інші.

Мета проєкту - удосконалення методу розпізнавання нечіткого мовлення осіб із вадами вимови окремих звуків на основі нейронних мереж.

Завдання:

– проаналізувати методи біометричної ідентифікації користувачів;

– удосконалити та розробити алгоритм розпізнавання нечіткого голосу осіб із вадами мовлення;

– реалізувати програмний засіб для цього завдання з використанням нейромереж;

– обґрунтувати економічну доцільність розробки.

Об’єкт - процеси ідентифікації користувачів.

Предмет - методи та засоби статистичного і частотного аналізу лексики людини.

Новизна роботи полягає в удосконаленні методу розпізнавання нечіткого голосу осіб із вадами вимови на основі нейронних мереж.

Практична цінність - у створеному програмному засобі, що реалізує удосконалений метод розпізнавання мовлення із використанням нейромереж і карти Кохонена.

1 АНАЛІЗ ЗАСОБІВ ПЕРЕВІРКИ КОРИСТУВАЧІВ

1.1 Аналіз біометричних особливостей при ідентифікації користувача

Біометрія - це фізіологічні або анатомічні характеристики людини, які використовуються для її ідентифікації [1]. На відміну від традиційних паролів, які можна підібрати або викрасти, біометричні системи зламати вкрай складно. Сучасні біометричні параметри включають відбитки пальців, голос, райдужну оболонку ока, почерк, характерні особливості набору тексту на клавіатурі тощо. Усі ці дані зберігаються в базі під час реєстрації користувача, а під час ідентифікації система порівнює нові дані з наявними зразками.

Основна мета біометричної ідентифікації - створити систему, яка забезпечує мінімальну кількість відмов доступу для авторизованих користувачів і повністю унеможливлює несанкціонований доступ до комп’ютерів або приміщень. Біометричні дані кожної людини є унікальними та використовуються для ідентифікації шляхом зіставлення з раніше збереженими зразками [2].

Втім, біометричні технології викликають і занепокоєння. Хоча багато хто вважає їх надійним способом захисту, є критики, які вбачають у біометричних системах загрозу громадянським свободам. Виникає ризик надмірного контролю над особистістю та невідомість подальшого використання біометричної інформації. Такі характеристики можуть бути використані проти самої людини, що створює загрозу її праву на конфіденційність.

Фактори, що впливають на унікальність мовлення. Мовлення людини поділяється на кілька видів [3]:

- нормативне;
- патологічне;
- навмисно змінене;
- емоційно забарвлене тощо.

Одним із ключових чинників, що впливають на унікальність голосу, є тип дихання, який буває:

- ключичний;
- грудний (діафрагмовий).

До цього слід додати індивідуальний об’єм легень, що може коливатися від 1000 см³ у дітей до 6000 см³ у фізично натренованих дорослих. При глибокому діафрагмовому диханні тривалість видиху під час мовлення може досягати 6–8 секунд, що значно перевищує 1,5–2 секунди при спокійному диханні. У таких випадках мовлення стає плавним і невимушеним, без частих вдихів.

У протилежних випадках, коли людина перебуває у стресі, втомлена або має фізіологічні обмеження (наприклад, астма чи інші хронічні захворювання), мовлення стає важким, ритм змінюється, а темп уповільнюється. Ці ознаки мають індивідуальний характер і формують унікальний голосовий профіль.

Також на мовлення впливає артеріальний тиск. Від нього залежить сила, висота та тембр голосу, які формуються ще на етапі проходження звукової хвилі через голосовий апарат.

Нижче наведено таблицю, що ілюструє відмінності між чоловічими та жіночими голосами (табл. 1.1).

Таблиця 1.1 – Відмінність голосів відносно гендерів

Голос	Основні тембральні забарвлення голосів / градації частоти основного тону	Межі зміни частоти основного тону в процесі співу, Гц	Загальна межа зміни частоти основного тону в процесі розмовної мови, Гц	Довжина голосових зв'язок, мм
Чоловічий	Бас/Низький Баритон/Середній Тенор/Високий	80-350 100-400 130 510	90-120	24-25 22-24 18-24
Жіночий	Контральто/Низький Меццо-сопрано/Середній Сопрано/Високий	170-680 220-880 260-1020	160-340	18-21 18-19 14-17

Існують також інші чинники, що впливають на унікальність мовлення конкретної особи: інтонаційна манера (наприклад, вплив іншомовного акценту); спектр голосових імпульсів (визначається їх формою, періодом ТО та шпаруватістю); інтенсивність звуку (може змінюватися в широкому діапазоні); фільтрація голосових імпульсів порожнинами рота і носа.

1.2 Аналіз систем розпізнавання голосу

Під час ідентифікації особи за голосом обробці підлягає записаний мовний сигнал. Аналоговий сигнал кодується як послідовність миттєвих значень амплітуди. Для фіксації та обробки мовного сигналу застосовують частоту дискретизації 8 або 16 кГц [4].

Щоб забезпечити якісне розпізнавання, необхідно уникати таких факторів:

- незадовільна акустика приміщення;
- варіативна відстань між мовцем і мікрофоном;
- невідповідність каналів зв'язку тощо.

Наприклад, при передачі голосу через телефон немає гарантії, що під час реєстрації та ідентифікації використовувався один і той самий мікрофон або канал передачі. Додатково важливо враховувати вплив зовнішніх перешкод. Для досягнення якісного запису канал має складатися з мікрофона, кабелю та аналого-цифрового перетворювача. Попередня обробка включає фільтрацію певних частот і видалення ділянок без мовного сигналу.

Щоб точно визначити початкову точку першого слова, необхідно виконати декілька етапів попередньої обробки.

Вважається, що протягом перших 300 мс запису присутні лише сторонні шуми. Весь сигнал розділяється на 256 сегментів, мовлення представлено як:

$$S = F_p\,(p = 0{,}1),$$
$$F\,p = S_p(t), \tag{1.1}$$

де S - послідовність відліків вхідного сигналу;

t = 0,1,...,255.

Для перших 10 сегментів застосовується швидке перетворення Фур'є (БПФ):

$$X_p(i) = \sum_{i=0}^{255} S_p(t) e^{-j\frac{2\pi}{n} n_i} \tag{1.2}$$

де i = 0,1,...,255;

p = 0,1,...,9.

Далі обчислюється середнє арифметичне:

$$m_i^* = \frac{\sum_{p=0}^{9} \left| X_p(i) \right|}{10}, \tag{1.3}$$

де i = 0,1,...,127 (оскільки сигнал симетричний).

Середньоквадратичне відхилення обчислюється за формулою:

$$\sigma_i^* = \sqrt{\frac{1}{9}\left(\sum_{p=0}^{9} (X_p(i) - m_i^*)^2 \right)}. \tag{1.4}$$

Порогове значення шуму визначається як:

$$П_i = m_i^* + k(\alpha)\sigma_i^*, \tag{1.5}$$

де α=0,95;

$k(\alpha)$=2,33.

У результаті отримуємо 128 порогів шуму. Далі перевіряється кожен сегмент: якщо в одному сегменті перевищено 15 порогів, вважається, що там починається слово. Однак точність визначення становить приблизно 23 мс. Для її покращення сегмент поділяється на 8 частин по 32 відліки, тобто кожна частина триває 3 мс [5].

Всі попередні розрахунки шуму діляться на 80 блоків для визначення середньої модуля амплітуди шуму:

$$S^*_{N_i} = \frac{1}{32}\left(\sum_{j=0}^{31}\left|S_{(32x_i)+j}\right|\right), \tag{1.6}$$

$$m^*_{N_i} = \frac{1}{80}\left(\sum_{i=0}^{79} S^*_{n_i}\right), \tag{1.7}$$

$$\sigma^*_{N_I} = \sqrt{\frac{1}{79}}\left(\sum_{I=0}^{79}(S^*_{N_I} - m^*_N)^2\right), \tag{1.8}$$

$$П_i = m^*_i + k(\alpha)\sigma^*_N, \tag{1.9}$$

де α=0,95;

$k(\alpha)$=2,33.

Фінальним кроком є порівняння середнього модуля кожного блоку S^*_N у сегменті зі значенням порога $П_N$. Якщо два блоки поспіль перевищують цей поріг, робиться висновок, що в одному з них розташований початок слова, вимовленого користувачем під час реєстрації.

1.3 Аналіз реалізованих засобів визначення вад голосу людини

Амплітудно-частотна характеристика слухового апарату людини далека від прямолінійної, а амплітуда не може слугувати точною величиною для вимірювання гучності (див. рис. 1.1). З огляду на це були запроваджені емпіричні одиниці для оцінки гучності звуку [6]. Аналогічно, сприйняття висоти звуку людським слухом також не має лінійної залежності від частоти (див. рис. 1.2).

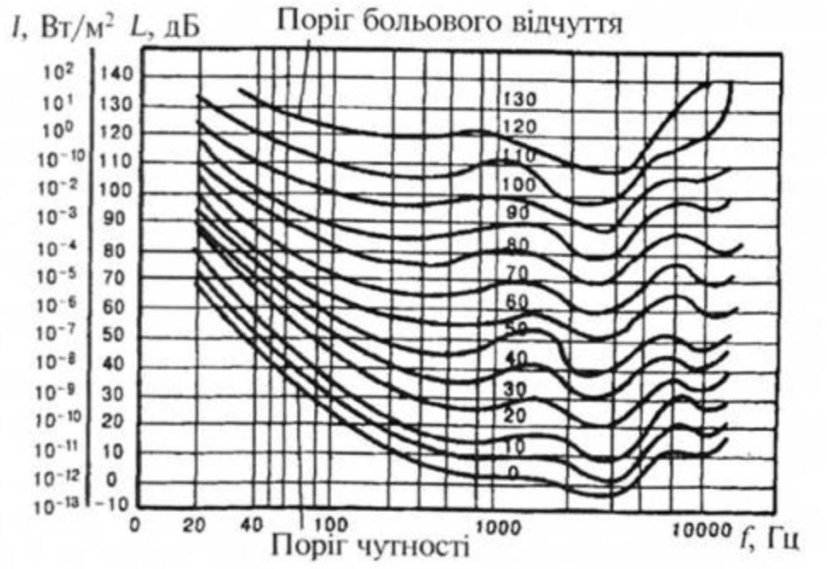


Рисунок 1.1 –АЧХ людського органу слуху

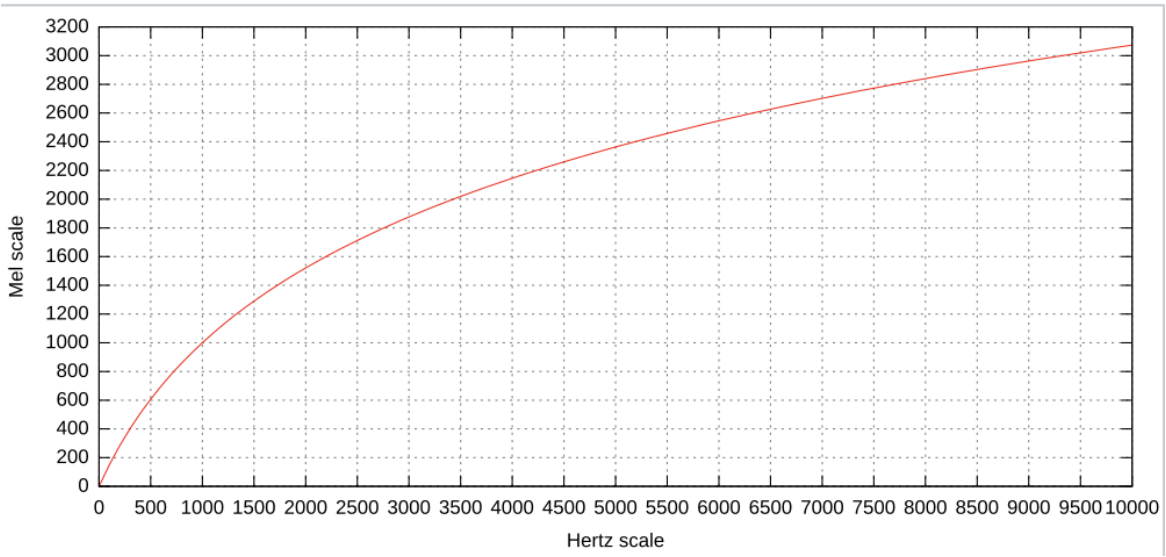


Рисунок 1.2 – Залежність висоти звуку від частоти

Одиниці вимірювання мелів активно застосовуються в системах розпізнавання, оскільки дозволяють моделювати особливості людського слухового сприйняття [7].

Кепстр - термін, утворений перестановкою літер у слові "spectrum", був введений Богертом у 1963 році. Це емпірична характеристика, яка отримується шляхом застосування перетворення Фур'є до логарифма спектра сигналу. Існують такі типи кепстру: енергетичний, комплексний, реальний і фазовий.

Енергетичний кепстр було запропоновано Богертом, Гілі та Тьюкі у 1963 році в роботі "The Quefrency Analysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking". Його можна визначити:

– усно - як спектр Фур'є логарифма квадрата амплітуд спектра сигналу;

– алгоритмічноу вигляді

$$signal \rightarrow FT \rightarrow abs \rightarrow square \rightarrow log \rightarrow FT \rightarrow abs \rightarrow square \rightarrow powercepstrum.$$

Комплексний кепстр, запропонований Оппенгеймом у межах теорії гомоморфних систем, обчислюється за алгоритмом:

$$signal \rightarrow FT \rightarrow abs \rightarrow log \rightarrow phase\ unwrapping \rightarrow FT \rightarrow cepstrum$$

Реальний кепстр використовує логарифм функції, визначеної на дійсних числах. Його зв'язок з іншими типами кепстру виражається наступними співвідношеннями:

– з енергетичним кепстром

$$EK = (4 \cdot PK)^2 \tag{1.10}$$

– з комплексним кепстром

$$PK = 0{,}5 \cdot (KK + KK^*), \tag{1.11}$$

де KK^* - часово обернений комплексний кепстр.

Фазовий кепстр пов’язаний із комплексним за формулою:

$$ФК = (КК - КК^*)^2. \quad (1.12)$$

Комплексний кепстр, на відміну від реального, враховує не лише амплітуду, а й фазу спектра, що дозволяє виконати реконструкцію сигналу.

Первісно кепстр використовувався для вивчення сейсмічних відлунь, але сьогодні його активно застосовують у розпізнаванні мовлення.

Алгоритм методу розпізнавання мовлення на основі кепстрального аналізу включає:

- подача сигналу - відліки сигналу $x_0,...,x_{n-1}$;

- вагова функція - застосовується вікно Хеммінга

$$\omega_n = 0{,}54 - 0{,}46\cos\left(2\pi\frac{n}{N-1}\right), n = 0,..., N-1; \quad (1.13)$$

- дискретне перетворення Фур'є:

$$X_k = \sum_{n=0}^{N-1} x_n \omega_n e^{\frac{-2\pi i}{N}kn}, k = 0,...N-1, \quad \text{або} \quad (1.14)$$
$$f_k = \frac{F_S}{N}k, k = 0,..., N/2,$$

де k – відповідає частотам;

F_S – частота дискретизації;

- мел-фільтрація - застосовується система трикутних фільтрів, межі яких визначаються у шкалі мелів:

$$B(f) = 1127 \cdot \ln\left(1 + \frac{f}{700}\right), B^{-1}(b) = 700\left(e^{b/1127} - 1\right). \quad (1.15)$$

Зазвичай використовують 24 фільтри. Потім обчислюється логарифм енергії:

$$e_m = \ln\left(\sum_{k=0}^{N} |X_k|^2 H_{m,k}\right), m = 0,...N_{FB} - 1, \quad (1.16)$$

де $H_{m,k}$ – отримані вагові коефіцієнти фільтрів.

Дискретне косинусне перетворення (DCT) є отриманням MFCC (мел-частотних кепстральних коефіцієнтів):

$$c_i = \sum_{m=0}^{N_{FB}-1} e_m \cos\left(\frac{\pi i(m+0{,}5)}{N_{FB}}\right), i = 1,..., N_{MFCC} \quad (1.17)$$

де c_0 - коефіцієнт енергії сигналу, зазвичай не використовується (типове число коефіцієнтів дорівнює 12).

Метод ґрунтується на апроксимації поточного значення сигналу лінійною комбінацією попередніх:

$$x_n \approx \sum_{k=1}^{p} a_k \, x_{n-k}. \quad (1.18)$$

Коефіцієнти $a_1,...,a_p$ знаходяться за допомогою алгоритму Дарбіна. Далі з них обчислюються кепстральні коефіцієнти, кількість яких може перевищувати p:

$$c_n = \left\{ a_n + \sum_{k=1}^{n-1} \frac{k}{n} c_k \, a_{n-k}, 1 \le n \le p, \; c_n = \right\} a_n + \sum_{k=n-p}^{n-1} \frac{k}{n} c_k \, a_{n-k}, n \succ p. \quad (1.19)$$

Наприклад, для сигналу з частотою дискретизації 8000 Гц і 12 коефіцієнтами передбачення можна отримати приблизно 18 кепстральних коефіцієнтів.

Для збереження динаміки мовлення застосовують похідні ознак (дельта-коефіцієнти). Існують методи нормалізації, що враховують усі вектори ознак, зокрема найбільш розповсюджений метод CMS (Cepstral Mean Subtraction), що

використовується для зменшення впливу акустичного каналу.

1.4 Варіанти використання нейронних мереж

Для вирішення задачі голосового розпізнавання необхідно навчити нейронну мережу, тобто знайти оптимальні значення ваг зв’язків між нейронами. Існує три основні методи навчання штучних нейронних мереж [8]:

– навчання з учителем (supervised) – коли нейромережі задається бажаний вихід, відповідний вхідним даним;

– навчання без учителя (unsupervised) – коли правильні відповіді для навчальних даних невідомі;

– змішане (гібридне) навчання – включає методи з підкріпленням, стохастичне, пакетне навчання тощо.

Процес навчання починається з подання навчальних даних на вхід моделі. На першому етапі параметри нейронів задаються випадково, після чого обчислюється помилка шляхом порівняння вихідного результату з еталонним. Потім відбувається коригування ваг з метою мінімізації цієї помилки.

Після завершення навчання виконується тестування на окремій вибірці даних, які не використовувались у процесі тренування. Метою є перевірка точності роботи моделі. Якщо точність не задовольняє вимог, структуру мережі змінюють і повторюють навчання.

Побудова нейронної мережі включає кілька етапів:

– вибір вхідних даних - відбираються лише релевантні ознаки, нерелевантні виключаються;

– нормалізація - дані приводяться до єдиного масштабу, наприклад, в інтервалі [0,1], що покращує роботу моделі;

– проєктування архітектури - визначається кількість шарів, нейронів, типи активаційних функцій, відповідно до поставленої задачі;

– налаштування ваг - здійснюється за допомогою прогону еталонних прикладів через мережу;

– перевірка моделі - тестування проводиться на незалежних даних.

Універсального алгоритму, який підходив би для всіх типів нейромереж, не існує.

Нейронні мережі поділяють на синхронні та асинхронні. У синхронних мережах свій стан змінює один нейрон за раз, а в асинхронних – група нейронів, найчастіше – цілий шар.

Існують дві базові архітектури нейромереж: шаруваті та пов'язані. У шаруватих мережах нейрони організовані в шари. Кожен шар складається з одного або кількох нейронів, які отримують однаковий вхідний сигнал [9]. Інформація в таких мережах обробляється пошарово: нейрони *i*-го шару передають оброблений сигнал нейронам *(i+1)*-го шару. Останній шар формує вихідний результат. Кількість нейронів у шарах може відрізнятися і обирається довільно.

Всередині одного шару обробка відбувається паралельно, між шарами – послідовно. У деяких мережах, як-от когнітрони, нейрони поточного шару отримують сигнали лише від сусідніх нейронів попереднього шару.

Шаруваті мережі поділяються на одношарові та багатошарові. У багатошарових мережах перший шар є вхідним, останній – вихідним, проміжні – прихованими (рис.1.3). Вхідний шар зв’язує мережу з вхідними даними, вихідний – із результатами. Відповідно, нейрони бувають вхідними та прихованими.

Вхідний шар складається з вхідних нейронів (input neuron), які приймають зовнішні дані та передають їх на нейрони прихованого шару. Приховані нейрони (hidden neuron) розміщені у прихованому шарі нейромережі. Вихідні нейрони (output neuron) формують вихідний шар, який видає результати обробки мережі.

У повнозв'язних нейронних мережах кожен нейрон передає свій вихідний сигнал усім іншим нейронам, включно з собою. Усі вхідні сигнали розподіляються на всі нейрони. Вихідними сигналами нейромережі можуть бути всі або частина сигналів нейронів після декількох тактів її роботи.

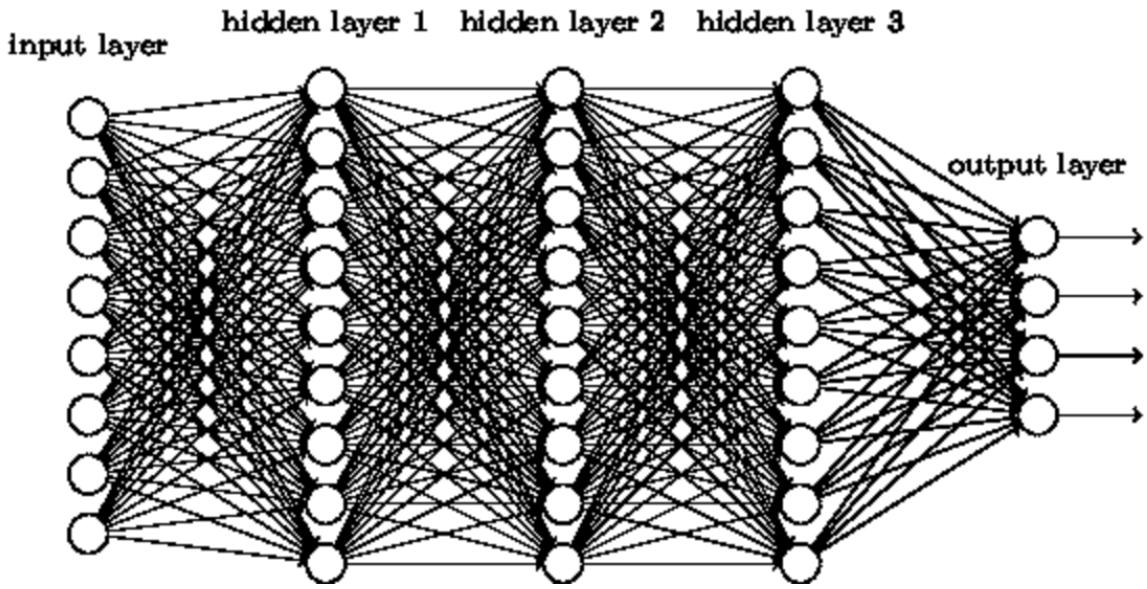


Рисунок 1.3 – Приклад багатошарової мережі

Однією з найпростіших моделей нейронної мережі є одношаровий персептрон**.** Персептрон (від лат. *perceptio* – сприйняття; англ. *perceptron*) – це математична або комп'ютерна модель сприйняття інформації мозком, розроблена Френком Розенблаттом у 1957 році й реалізована у вигляді електронної машини «Марк-1» у 1960 році. Персептрон став першою відомою нейромережею, а «Марк-1» – першим нейрокомп’ютером [10].

Одношаровий персептрон (персептрон Розенблатта) є простою нейромережею з одним шаром нейронів, які використовують жорстку граничну функцію активації. Він має простий алгоритм навчання та здатен розв’язувати лише базові задачі. У 1960-х роках ця модель викликала значний інтерес і стала поштовхом до подальшого розвитку нейромережевих технологій.

Класичний приклад – одношаровий тринейронний персептрон, зображений на рис. 1.4. У нього є *n* входів, сигнали з яких через синапси надходять до трьох нейронів єдиного шару. Ці нейрони формують вихідні сигнали мережі.

Багатошаровий персептрон (MLP**)** – це нейромережа прямого поширення сигналу (без зворотних зв’язків), у якій вхідна інформація проходить послідовно крізь кілька шарів і на виході перетворюється в результат.

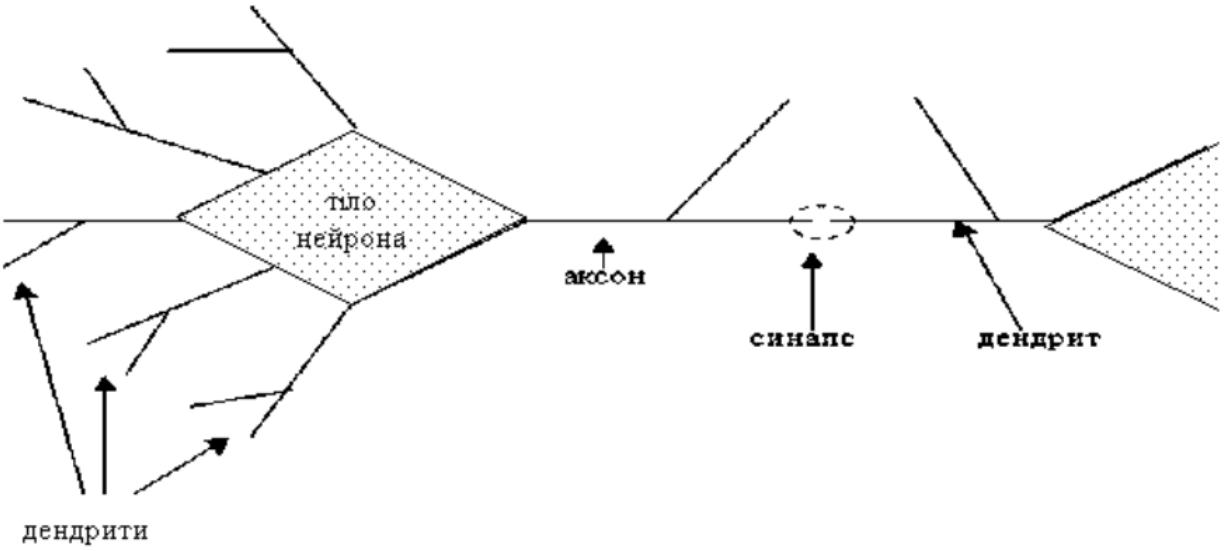


Рисунок 1.4 – Одношаровий тринейронний персептрон

Перший шар називають вхідним, останній - вихідним. Ці шари містять вироджені нейрони і іноді не враховуються у загальній кількості шарів. Окрім вхідного та вихідного, у багатошаровому персептроні є один або кілька проміжних, або прихованих, шарів. У цій моделі персептрона має бути принаймні один прихований шар. Наявність кількох прихованих шарів обґрунтовується лише при застосуванні нелінійних функцій активації. Приклад двошарового персептрона наведено на рис. 1.5. Мережа має n входів, сигнали з яких через синапси надходять на три нейрони першого шару. Вихідні сигнали цього шару передаються двом нейронам другого шару, які формують два вихідні сигнали.

Мережа радіально-базових функцій (рис. 1.6) - штучна нейронна мережа, запропонована Брумхедом і Лоу у 1988 році, що використовує радіальні базисні функції як функції активації. Вихід мережі є лінійною комбінацією радіальних базисних функцій входів і параметрів нейрона.

Зазначимо дві переваги мереж радіально-базисних функцій (RBF) порівняно з іншими нейронними мережами:

- активаційна функція прихованих нейронів набуває великих значень лише тоді, коли вхідний сигнал знаходиться поблизу опорної точки нейрона. Поза областю, «покритою» навчальними зразками, мережа формує низькі вихідні значення, тоді як мережі з сигмоїдальними функціями активації (наприклад,

багатошарові персептрони) можуть давати непередбачувані виходи;

- проста структура RBF-мережі, що містить лише один прихований шар, дозволяє напряму розраховувати ваги, що є значною перевагою порівняно з іншими нейронними мережами, які здебільшого потребують рекурентних і трудомістких алгоритмів навчання. Крім того, існує можливість ініціалізації ваг шляхом прямого розрахунку з подальшим донавчанням за допомогою алгоритмів зворотного поширення помилки.

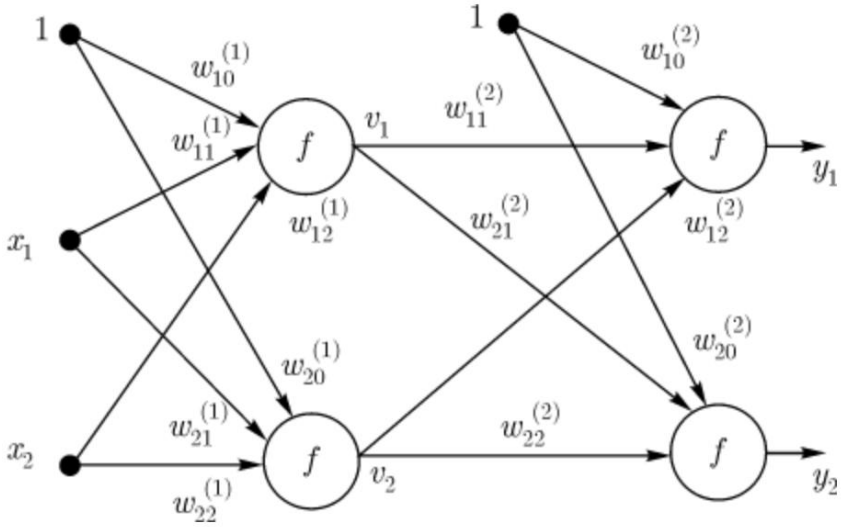


Рисунок 1.5 – Двошаровий персептрон

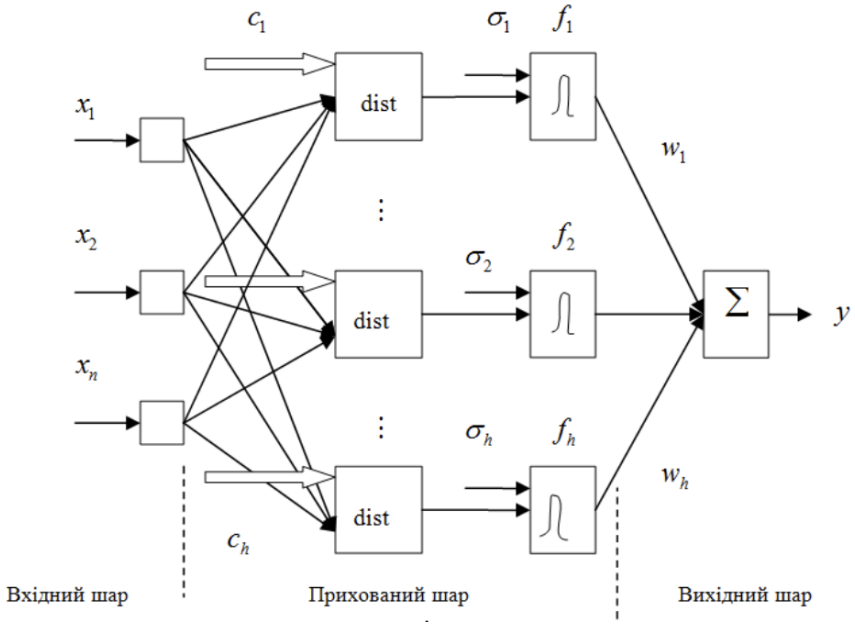


Рисунок 1.6 – Мережа радіально-базисних функцій

Мережі радіально-базисних функцій широко застосовуються для апроксимації функцій, прогнозування часових рядів, класифікації та побудови систем керування.

1.5 Висновки розділу

У цьому розділі проведено теоретичний огляд галузі розробки, здійснено аналіз особливостей голосової ідентифікації користувачів, зокрема систем обробки голосу з вадами мови, розглянуто методи визначення голосових характеристик та використання нейронних мереж для біометричної ідентифікації.

На основі проведеного аналізу теоретичного матеріалу, мети та актуальності теми, визначено наступні завдання подальшої роботи:

- удосконалити та розробити алгоритм програмного засобу для розпізнавання нечіткого голосу людей із вадами вимови на основі нейромережі;
- реалізувати програмний додаток для розпізнавання такого голосу;
- обґрунтувати економічну доцільність розробки.

Виконання цих завдань сприятиме досягненню основної мети роботи - розробці та впровадженню методу розпізнавання нечіткого голосу на основі нейронної мережі.

2 ВИБІР ЗАСОБІВ ДЛЯ РОЗРОБКИСИСТЕМИ

2.1 Обрання методу для розпізнавання голосу

Прототипом нейрона є біологічна нервова клітина, яка складається з тіла клітини (соми) та двох типів відгалужень - аксона і дендритів. Тіло клітини містить ядро з інформацією про спадкові властивості та плазму, що забезпечує

виробництво і передачу необхідних матеріалів нейрону . Нейрон приймає сигнали (імпульси) через дендрити від інших нейронів і передає сформовані сигнали тілом клітини вздовж аксона, що на кінці розгалужується у волокна з синапсами [11]. Математична модель нейрона описується співвідношенням:

$$y = f(s), s = \sum_{i=1}^{n} x_i \omega_1 + b, \tag{2.1}$$

де ω_i - вага синапсу,

b – зсув;

s - вхідний сигнал;

y - вихідний сигнал;

n - кількість входів нейрона;

f - функція активації.

Технічна модель нейрона наведена на рисунку 2.1.

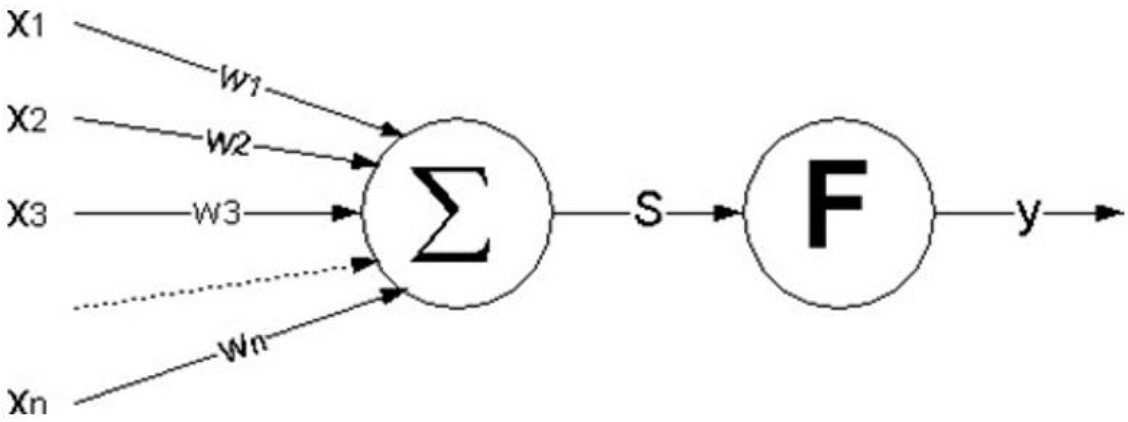


Рисунок 2.1 – Технічна модель нейрона

Нейроподібний елемент виконує прості операції зваженого підсумовування з подальшим нелінійним пороговим перетворенням. Особливість нейромереж полягає в тому, що складна організація зв’язків між простими однорідними елементами дозволяє розв’язувати складні задачі. Саме структура зв’язків визначає функціональні властивості мережі в цілому.

Функціонування мережі залежить від величин синаптичних зв'язків. Після визначення структури мережі на етапі проектування, знаходять оптимальні значення вагових коефіцієнтів $w_1,w_2,...,w_n$ та зсувів нейронів - цей процес називають навчанням. Розв'язання задачі розпізнавання мови полягає у пошуку конфігурації мережі, яка забезпечить оптимальну роботу на основі навчальної вибірки з k пар вхідних та вихідних векторів $(X_i,Y_i),i=1,...,k$. Існують два підходи до навчання: з учителем та без учителя (самонавчання). При навчанні з учителем на вхід подається вектор Xi , вихід мережі Y порівнюється з еталонним Y_i і за потреби коригуються ваги та усуваються нейрони. При навчанні без учителя ваги підлаштовуються на основі стану нейронів і поточних ваг за визначеним правилом до стабілізації вихідних значень мережі [12].

Нейрони можуть об'єднуватись у різні мережеві структури, що визначає особливості нейромережі. Для задач ідентифікації та управління найбільш ефективними є багатошарові нейронні мережі (МНС) прямої дії або багатошарові персептрони (рис.2.2). При проектуванні МНС нейрони групують у шари, кожен з яких обробляє вхідні сигнали від попереднього шару. Мінімальною реалізацією є двошарова мережа, що містить вхідний (розподільний), проміжний (прихований) і вихідний шари.

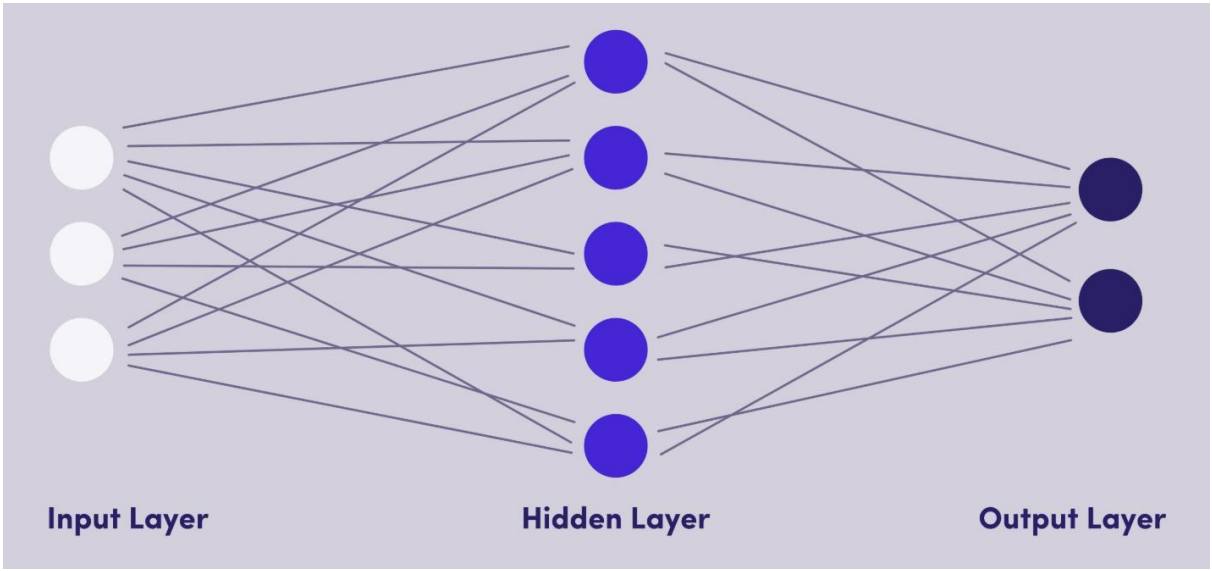


Рисунок 2.2 – Структурна схема двошарової нейронної мережі

Реалізація моделі двошарової нейронної мережі прямої дії описується формулою:

$$y = (\theta) F_i \left(\sum_{j=1}^{n_h} W_{ij} \; f_j \left(\sum_{j-1}^{n_h} w_{ij} \, \varphi_j + w_{j0} \right) + W_{j0} \right), \tag{2.2}$$

де $n\varphi$ - розмірність вхідного вектора;

nh - кількість нейронів прихованого шару,

θ - вектор налаштовуваних параметрів (wji,Wijw);

$f_j(x)$ - функція активації нейронів прихованого шару;

$F_i(x)$ - функція активації нейронів вихідного шару.

Розглянемо класичний варіант багатошарової мережі із довільною кількістю шарів, де синаптичні ваги - будь-які дійсні числа, а вихід нейрона - значення від 0 до 1. В якості активаційної функції використовується сигмо їда [13].

Крок 1. Ініціалізуємо M матриць ваг W розміром $N \times N$, де M - кількість шарів, N - кількість нейронів у шарі. Wi,j,k - вага jj-го входу kk-го нейрона ii-го шару, заповнена малими випадковими значеннями.

Крок.2. Подамо на вхід мережі вектор X із відомими правильними виходами Y^*.

Крок 3. Обчислюємо вихід мережі YY для поточних ваг. Для кожного i - шару:

$$O_i = F(XW_i), \text{ якщо } i \text{ – не перший шар,} \tag{2.3}$$

$$O_i = F(O_i\text{-}1W_i), \text{ якщо } i \text{ – перший шар,} \tag{2.4}$$

де O_i - вихід ii-го шару;

F - функція активації.

Крок 4. Обчислюємо помилку $\Delta Y = Y - Y^*$.

Крок 5. Якщо ΔY менше заданої похибки, переходимо до кроку 9.

Крок 6. Для останнього шару M виконуємо наступні дії:

а) для кожного нейрона *k* та кожної ваги *j*:

1) обчислюємо $M=X(1-X)\,\Delta Y$;

2) розраховуємо корекцію ваги $\Delta WM, j, k = \eta \delta M, kO_i - 1, j,$

де η - швидкість навчання;

3) оновлюємо ваги $\Delta WM, j, k$.

Крок 7. Для шарів від $M-1$ до *1* послідовно виконуємо:

а) для кожного нейрона *k* і ваги *j*:

1) обчислюємо $\delta = O_i + 1(1 - O_i + 1)\left[\sum \delta_i + 1, jW_i + 1, j, k\right].$

2) розраховуємо $\Delta WM, j, k = \eta \delta_i, kO_i - 1, j,$

де η – коефіцієнт швидкості навчання, від 0,01 до 1,0;

3) оновлюємо ваги $Wi,j,k=Wi,j,k+\Delta Wi,j,kW.$

Крок 8. Повертаємося до кроку 3.

Крок 9. Навчання завершено.

Алгоритм повторюється до досягнення заданої точності вихідних значень для різних вхідних векторів.

Для покращення обробки великого обсягу даних автор пропонує використовувати самоорганізовані карти Кохонена, які виділяють нейрони з максимальною активністю, що дозволить скоротити час розпізнавання на 30–80% порівняно з існуючими рішеннями.

Самоорганізовані карти Кохонена (рис.2.3) - це ефективний механізм самонавчання та кластеризації, що відображає результати у вигляді компактних, зручних для інтерпретації двовимірних карт [14].

Основні переваги алгоритму: стійкість до зашумлених даних, швидке та некероване навчання, а також можливість візуалізації багатовимірних вхідних даних.

Самоорганізовані карти Кохонена використовують для аналізу даних, прогнозування поведінки користувачів, виявлення аномалій тощо. Вони здатні розпізнавати кластери та визначати близькість класів, що допомагає краще розуміти структуру даних і уточнювати модель.

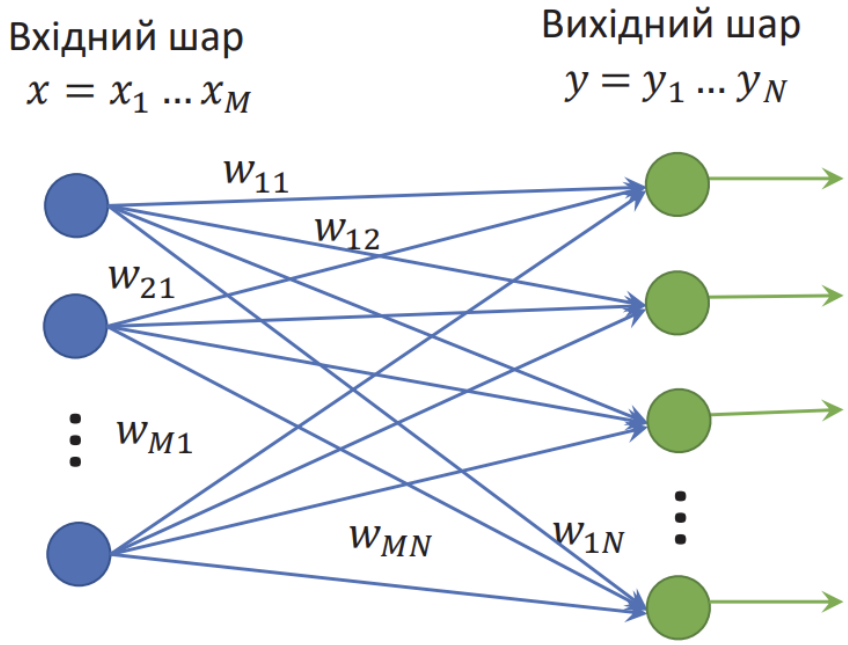


Рисунок 2.3 – Схематичне подання структури нейронної мережі Кохонена

Виявлені класи можна позначити, що дозволить мережі виконувати завдання класифікації. Наприклад, побудувавши карту Кохонена з кластерами клієнтів за рівнем лояльності, можна прогнозувати їх поведінку та застосовувати відповідну маркетингову стратегію.

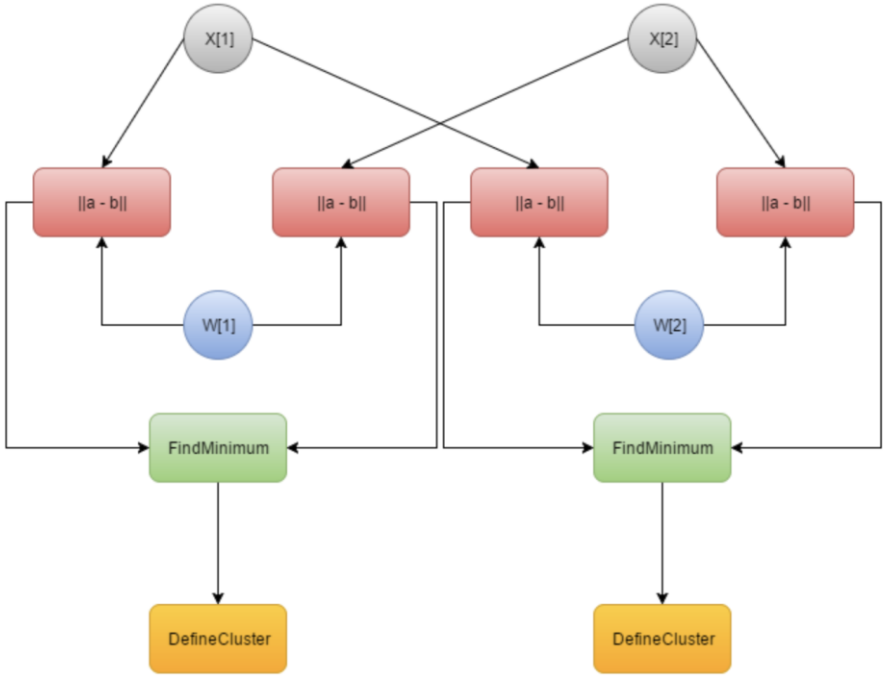


Рисунок 2.4 – Схематичний граф алгоритм роботи карти Кохонена

Для виявлення аномалій карта Кохонена класифікує навчальні дані у відповідні кластери [15]. Якщо зустрічається набір даних, який не схожий на жоден з відомих кластерів, карта не зможе його класифікувати, що вказує на аномальність цього набору (рис.2.4).

Метою методу є пошук прихованих закономірностей шляхом зниження розмірності вхідного простору до меншого (зазвичай двовимірного для зручності візуалізації) із збереженням топології. Результатом навчання є ґрати з навчених нейронів, які утворюють "карту" вихідного простору.Навчання мережі (рис.2.5) полягає у визначенні нейрона-переможця для кожного вхідного вектора, після чого коригуються ваги цього нейрона та його сусідів. Процедура повторюється для всіх вхідних векторів навчальної вибірки.

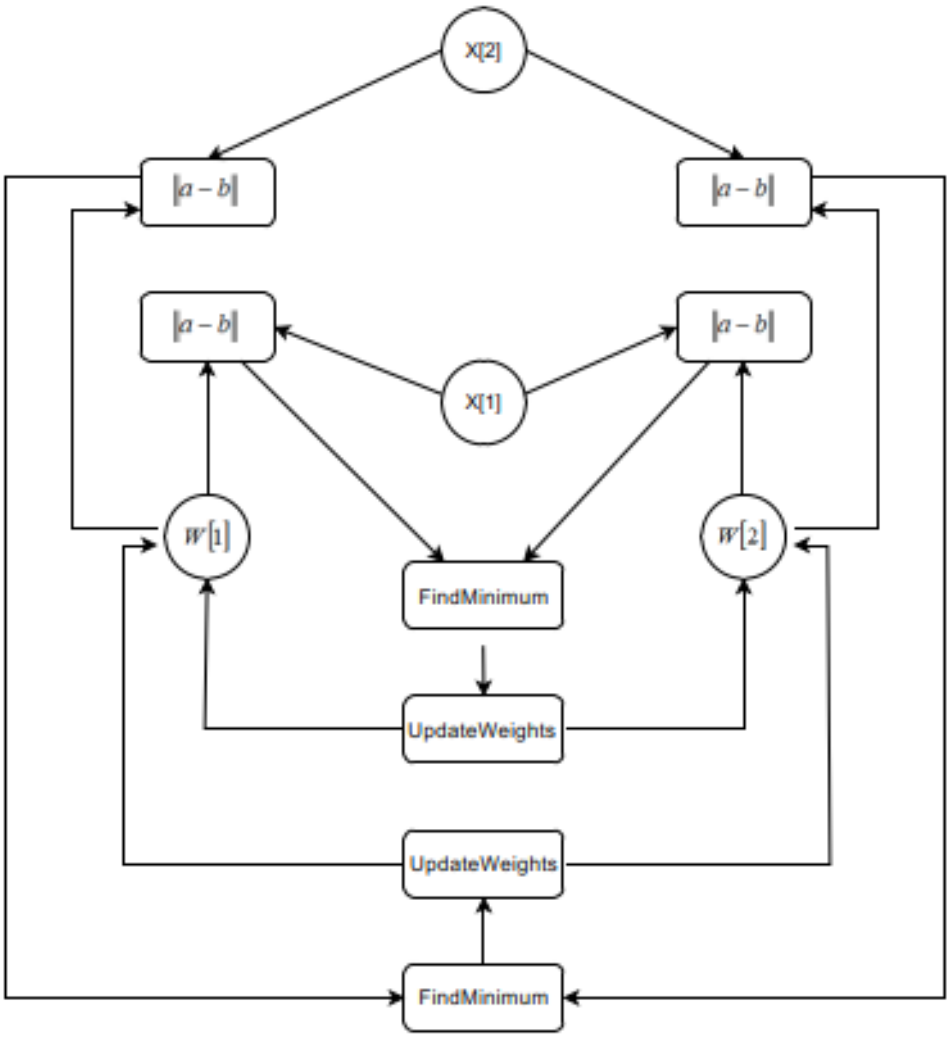


Рисунок 2.5 – Схема графу навчання карти Кохонена

Мережа з запропонованою архітектурою навчається на навчальній вибірці шляхом коригування ваг нейронів шару Кохонена. Після навчання мережа класифікує кожен нейрон тестової вибірки, відносячи його до відповідного кластера.

2.2 Вибір мови програмування для розробки системи

Для практичної реалізації мети роботи та розробки програмної частини заплановано використати об’єктно-орієнтовану мову програмування C# у середовищі Microsoft Visual Studio на основі Windows Forms [16].

C# - сучасна, проста в освоєнні об’єктно-орієнтована мова, що належить до сімейства C і буде знайомою користувачам C, C++, Java чи JavaScript. Вона підтримує як об’єктно-, так і компонентно-орієнтоване програмування, що передбачає створення автономних компонентів із властивостями, методами та подіями. Кожен компонент має атрибути з вбудованою документацією, що забезпечує зручність розробки та використання.

Для надійності додатків C# забезпечує автоматичне звільнення пам’яті, обробку виключень і сувору типізацію, яка виключає доступ до неініціалізованих змінних, вихід за межі масивів і неконтрольоване приведення типів. Всі типи успадковують від кореневого типу object, що уніфікує роботу з ними, а також підтримуються як посилальні, так і типи значень.Особлива увага в C# приділена управлінню версіями, що підвищує сумісність додатків і бібліотек [17]. Це реалізовано через модифікатори virtual/override, правила перевантаження методів та явне оголошення інтерфейсів.

Переваги мови: широка підтримка продуктів Microsoft; безкоштовність інструментів для малих компаній і індивідуальних розробників (Visual Studio, Azure тощо); великий набір синтаксичного цукру для зручності кодування; низький поріг входження завдяки схожості з іншими мовами; можливість

кросплатформної розробки (iOS, Android, MacOS, Linux) через Xamarin; активна спільнота досвідчених розробників.

Недоліки: орієнтація переважно на Windows; безкоштовність лише для малих користувачів, великі компанії мають витрати на ліцензії; збереження оператора безумовного переходу.

Таким чином, C# є ефективним інструментом для розробки програмних додатків з урахуванням сучасних вимог.

2.3 Вибір засобів програмування для розробки системи

Visual Studio є найбільш популярним інтегрованим середовищем розробки (IDE) серед програмістів, утримуючи лідерство на ринку понад двадцять років.

Microsoft Developer Network (MSDN) - підрозділ Microsoft, що підтримує розробників, допомога якого доступна не лише користувачам Visual Studio, а й іншим програмістам із відповідною підпискою. Завдяки тісній взаємодії співробітників і спільноти користувачів, Visual Studio є одним із найпідтримуваніших продуктів, хоча MSDN підтримує лише мови й інструменти Microsoft [18].

IDE - це набір програмних засобів, що оптимізує процес розробки програмного забезпечення. Незважаючи на широкий функціонал багатьох текстових редакторів коду, лише програми з певним набором інструментів можна назвати IDE. До них належать: редактор коду з підсвічуванням синтаксису та помилок; компілятор або інтерпретатор для відповідних мов (Visual Studio підтримує обидва); автоматизація збірки; дебагер для виявлення помилок, необхідний при розробці великих додатків.

Якщо всі ці елементи присутні, програма є IDE. Visual Studio додатково містить систему управління версіями, оглядач класів та інші інструменти.

Основні переваги Visual Studio:

– IntelliSense - інструмент автодоповнення та виправлення коду, який спрощує написання і генерує фрагменти коду, використовуючи вбудовану документацію;

– CodeLens - вбудований коректор, що допомагає знаходити помилки;

– хвиляста лінія - класичний індикатор помилок із підказками для їх усунення.

Visual Studio популярна завдяки:

– вбудованому серверу ASP.NET, що дозволяє запускати веб-додатки прямо у середовищі без потреби зовнішніх серверів;

– широкій підтримці численних мов програмування, що особливо зручно при розробці веб-додатків з використанням різних технологій в одному проекті;

– інтелектуальній системі, яка зменшує обсяг коду та покращує його читаність;

– автоматичному форматуванню коду, що підвищує його зрозумілість і зручність редагування.

Окрім MSDN, Visual Studio підтримує платформу Team System для спільної роботи, тестування та налагодження проектів. Користувачі можуть звертатися з пропозиціями щодо розвитку продукту, які розробники активно розглядають. Тому для створення надійних і ефективних додатків варто обирати саме цю платформу.

У процесі проєктування та реалізації програмного забезпечення одним із ключових етапів є обґрунтований вибір мови програмування. Цей вибір має стратегічне значення, оскільки безпосередньо впливає на архітектуру системи, її технічні характеристики, підтримку, безпеку, продуктивність, швидкість розробки та можливість майбутнього масштабування. У випадку створення десктопного додатку для операційної системи Windows, особливу увагу слід приділити інтеграції з Windows API, можливостям побудови графічного інтерфейсу користувача, роботі з файловою системою, мережею та базами даних, а також безпеці та надійності роботи застосунку.

Сьогодні розробник має доступ до широкого спектру мов програмування та середовищ розробки, кожна з яких має свої переваги та обмеження. У рамках даного проєкту було розглянуто декілька найбільш релевантних мов програмування для реалізації настільного додатку: C#, C++, Java, Python та менш поширені, але потенційно придатні варіанти, такі як Delphi, Go або Rust. Вибір серед них здійснювався на основі декількох основних критеріїв:

– інтеграція з Windows - наскільки мова дозволяє працювати з системними функціями, API, реєстром, файлами тощо;

– підтримка створення графічного інтерфейсу - наявність зручних бібліотек або фреймворків для побудови GUI;

– продуктивність та споживання ресурсів - наскільки ефективно створений застосунок використовує ресурси ПК;

– швидкість розробки - наскільки просто і швидко можна реалізовувати функціонал у цій мові;

– безпека та стабільність - рівень захисту від помилок, витоків пам’яті, підтримка оновлень;

– підтримка інструментів розробки - наявність IDE, дебагерів, аналізаторів коду;

– спільнота та документація - наскільки легко знайти підтримку або готові рішення.

Проведений аналіз показав, що оптимальним вибором для реалізації системи є мова програмування C# у поєднанні з платформою .NET Framework / .NET Core та середовищем розробки Visual Studio. Це рішення базується на ряді факторів, детально розглянутих нижче.

C# - це сучасна, об’єктно-орієнтована мова програмування, розроблена компанією Microsoft спеціально для побудови програмного забезпечення під Windows. Вона тісно інтегрується з платформою .NET, яка надає широкий спектр функціональних можливостей: від роботи з базами даних (через Entity Framework), до побудови потужних GUI (через Windows Forms або WPF), роботи з API, файлами, мережею тощо. Серед переваг C#:

– повна підтримка Windows - розробка, орієнтована на роботу з Windows API, інтеграцію з сервісами Windows, доступ до системних ресурсів і служб;

– широкий вибір бібліотек та фреймворків - WPF, WinForms, ASP.NET, MAUI (для кросплатформеності) — усе це дозволяє реалізовувати як прості, так і складні системи;

– високий рівень безпеки - C# працює в рамках керованого середовища виконання CLR (Common Language Runtime), що забезпечує захист від небезпечних операцій з пам’яттю, перевірку типів і збір сміття;

– інтенсивна підтримка спільноти та Microsoft - C# активно розвивається, має чудову документацію, велику кількість прикладів і навчальних матеріалів;

– зручне налагодження - Visual Studio пропонує потужні інструменти для відлагодження, профілювання та тестування коду, включно з UI-дебагером.

Для порівняння, мова C++, хоча і забезпечує максимально високу продуктивність та контроль над ресурсами, значно ускладнює процес розробки через складний синтаксис, необхідність ручного управління пам’яттю, а також потребу в використанні сторонніх бібліотек (наприклад, Qt, Boost) для побудови GUI. У контексті створення прикладного програмного забезпечення, де час розробки, зручність підтримки та стабільність важливіші за мікрооптимізації - C++ поступається C#.

Мова Java, попри свою потужність і кросплатформеність, має дещо інші пріоритети. Вона частіше використовується у веб- і мобільній розробці. Створення нативного додатку під Windows з використанням Java (через Swing або JavaFX) не забезпечує бажаної інтеграції з Windows API, має нижчу продуктивність та потребує встановлення JVM на кожному ПК користувача.

Python, незважаючи на свою популярність, більше підходить для швидкого прототипування, написання скриптів або аналізу даних. Для створення стабільного GUI-додатку у Windows Python вимагає сторонніх бібліотек (Tkinter, PyQt, PySide), має обмежену продуктивність і часто не забезпечує достатню

безпеку для кінцевих користувачів. Крім того, Python-програми складно компілювати у виконувані файли, що важливо при розгортанні під Windows.

Серед менш популярних, але цікавих рішень - Go та Rust. Вони мають високий рівень безпеки та продуктивності, однак не мають достатньої кількості бібліотек для розробки повноцінних GUI-додатків під Windows, що робить їх менш придатними в даному контексті.

З урахуванням усіх факторів, мова C# з використанням Windows Forms або WPF та .NET платформи була обрана як найкраще рішення для розробки настільного додатку під Windows. Вона забезпечує оптимальний баланс між продуктивністю, простотою розробки, масштабованістю та безпекою. Такий вибір дозволяє ефективно реалізувати функціональні вимоги системи, спростити обслуговування й оновлення програмного забезпечення, а також гарантувати стабільну роботу користувацького застосунку в середовищі Windows.

Таким чином, правильний вибір мови програмування не лише оптимізує технічну реалізацію системи, але й забезпечує її довгострокову підтримку, гнучкість адаптації до нових вимог та стабільність у роботі кінцевого продукту.

Отже, застосування мови C# у середовищі Microsoft Visual Studio на базі Windows Forms забезпечує зручну й ефективну реалізацію запланованого програмного додатку для досягнення поставленої мети.

2.4 Висновки до розділу

У цьому розділі пропонується обрання методу для розпізнавання голосу за рахунок застосування нейромережі та карт Кохонена. Детально описано удосконалення обраного методу, обґрунтовано вибір мови програмування та середовища розробки. Для практичної реалізації роботи та розробки програмної частини заплановано використати об’єктно-орієнтовану мову програмування C# у середовищі Microsoft Visual Studio на основі Windows Forms. що забезпечить

зручну й ефективну реалізацію запланованого програмного додатку для досягнення поставленої мети.

3 РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ НЕЧІТКОГО ГОЛОСУ

3.1 Розробка алгоритму для системи ідентифікації

Для розробки програмного засобу з удосконалення методу розпізнавання нечіткого голосу користувачів з вадами вимови, на основі нейромережі, сформовано алгоритм роботи з урахуванням математичних моделей та голосового ідентифікатора. Покроково опишемо цей алгоритм.

Крок 1. Запуск програмного засобу.

Крок 2. Авторизація:

- якщо користувач зареєстрований - перехід до кроку 5;

- якщо користувач не зареєстрований - перехід до кроку 3.

Крок 3. Реєстрація користувача: введення електронної пошти та запис голосового зразка.

Крок 4. Перевірка коректності введених даних:

- при коректних даних - перехід до кроку 5;

- при некоректних - повернення до кроку 3.

Крок 5. Введення даних для авторизації: логін (електронна пошта) та голосовий зразок через мікрофон.

Крок 6. Перевірка достовірності авторизаційних даних:

- якщо дані невірні - відображення повідомлення і повернення до кроку 5;

- якщо дані вірні - відкривається головне вікно програми, доступна подальша робота.

Крок 7. Робота авторизованого користувача в системі.

Крок 8. Завершення роботи, вихід із системи через кнопку «Вихід».

Алгоритм представлено на рисунку 3.1.

Рисунок 3.1 – Блок – схема алгоритму роботи

Враховуючи наведений алгоритм, виділяють чотири етапи роботи системи (рис.3.2):

- етап реєстрації користувача;
- інформаційний етап;
- етап візуалізації;
- етап ідентифікації користувача.

Отже, запропонований алгоритм роботи додатку спрямований на підвищення достовірності ідентифікації шляхом біометричного розпізнавання користувача на основі голосової ідентифікації.

Система може застосовуватися для контролю доступу, наприклад, при вході у web-додатки, запобігаючи несанкціонованому використанню.

Для зменшення кількості помилок необхідно використати нейромережу, зокрема нейромережу Кохонена [19], що дозволить скоротити час розпізнавання на 30–80% порівняно з існуючими рішеннями.

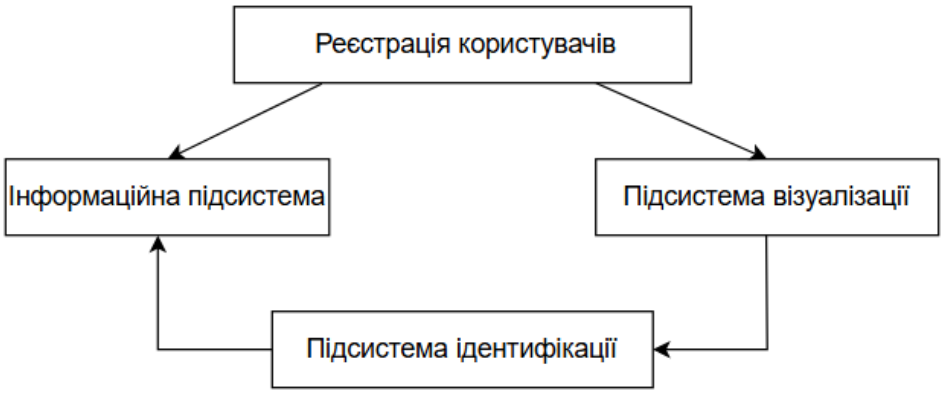


Рисунок 3.2 – Основні етапи роботи в системі

3.2 Проєктування візуального інтерфейсу користувача

Інтерфейс користувача є засобом забезпечення зручної взаємодії між користувачем та інформаційною системою. Це сукупність інструментів для

обробки й відображення інформації, максимально адаптованих до потреб користувача. У графічних інформаційних системах інтерфейс реалізується через багатовіконний режим, зміну кольору, розміру та видимості вікон (включаючи прозорість і невидимість), їх розміщення, сортування елементів, а також можливість гнучкого налаштування як самих вікон, так і їхніх окремих компонентів - таких як файли, папки, ярлики, шрифти тощо. Крім того, підтримуються багатокористувацькі параметри, що дозволяє адаптувати систему під індивідуальні потреби різних користувачів [20].

У зв’язку з цим наступним етапом є розробка інтерфейсу користувача для чіткого уявлення про зовнішній вигляд програмного забезпечення.

На рисунку 3.3 представлено приклад головного діалогового вікна додатку. У цьому вікні розміщено інформаційні текстові поля, графічні елементи, а також функціональні кнопки: «Головна», «Про систему» та «Вхід до системи».

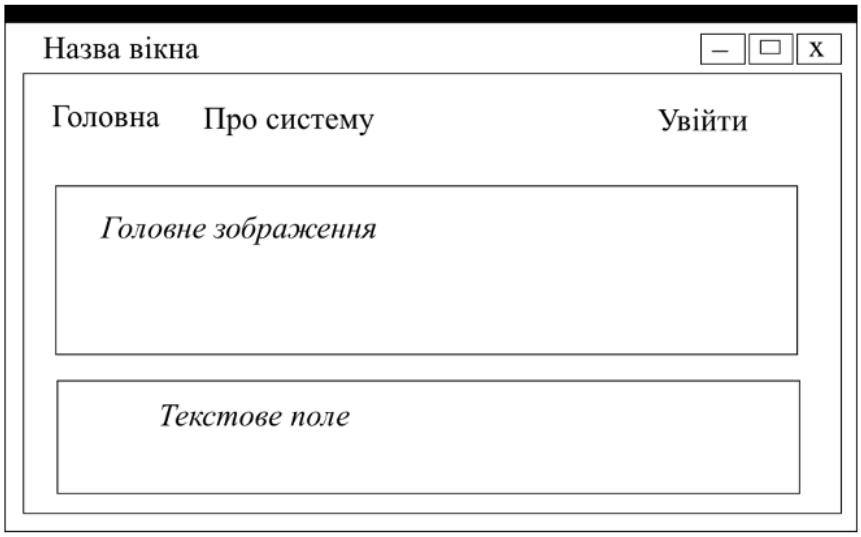


Рисунок 3.3 – Проектування головного вікна додатку

На рисунку 3.4 представлено проєкт діалогового вікна, що використовується під час авторизації користувача в системі.

У цьому вікні передбачено введення користувацьких даних, зокрема логіна, а також подання голосового зразка для здійснення біометричної ідентифікації.

У нижній частині розміщено кнопку «Увійти», яка запускає процес

перевірки та підтвердження особи. Крім того, передбачена можливість обрати функцію «Реєстрації» нового користувача.

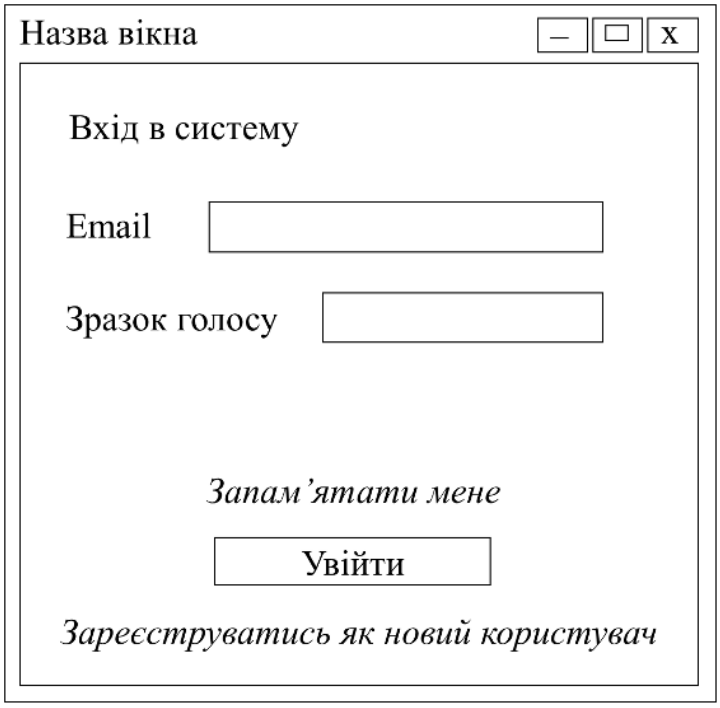


Рисунок 3.4 – Проектування вікна «Авторизація користувача»

На рисунку 3.5 представлено проєкт діалогового вікна, призначеного для реєстрації користувача в системі.

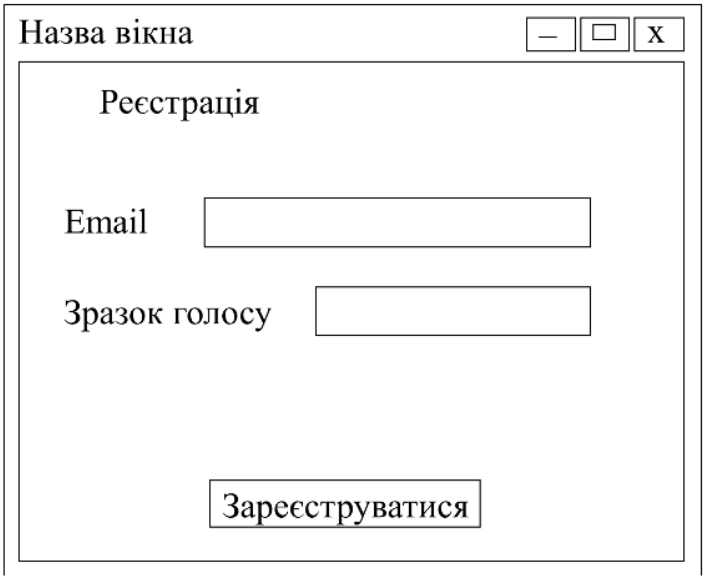


Рисунок 3.5 – Проєктування вікна «Реєстрація користувача»

У цьому вікні користувачеві, як і під час авторизації, потрібно ввести власні дані, зокрема логін, а також надати голосовий зразок для подальшої біометричної ідентифікації.

У нижній частині розташовано кнопку «Зареєструватися», яка підтверджує введену інформацію та запускає процес реєстрації.

Отже, на основі спроєктованих діалогових вікон формується основна структура майбутнього програмного засобу. Ключовою вимогою до інтерфейсу є забезпечення його зручності та інтуїтивної зрозумілості для користувача.

3.3 Програмна частина додатку

У даному підрозділі розглядаються розроблені та практично застосовані ключові кодові фрагменти, які стали основою реалізації програмного додатку для біометричної ідентифікації користувачів шляхом зчитування унікальних голосових параметрів.

Під час програмної реалізації було використано такі бібліотеки [21]:

– System.Speech.Recognition – для розпізнавання мови;

– System.Speech.Synthesis – для синтезу мовлення (озвучування повідомлень);

– System.IO – для роботи з файлами та потоками даних;

– System.Windows.Forms – для створення графічного інтерфейсу користувача;

– System.Drawing – для графічного відображення елементів інтерфейсу;

– Accord.Audio та Accord.Neuro (з пакету Accord.NET) – для обробки аудіоданих, реалізації нейронної мережі, а також побудови та тренування карти Кохонена;

– NAudio – для захоплення, збереження та обробки аудіосигналу в режимі реального часу.

Ці бібліотеки забезпечують комплексну роботу всіх модулів додатку: від інтерфейсної частини до функціоналу голосового розпізнавання, обробки сигналу та нейромережевої ідентифікації.

На рисунках 3.6 – 3.8 надані приклади використання вищенаведених бібліотек:

```
using System;
using System.Threading.Tasks; using System.IO;
using System.Threading; using NAudio.Wave;
```

Рисунок 3.6 – Бібліотеки для програмної реалізації

```
public AccountController()
{
}
public AccountController(ApplicationUser ManageruserManager, ApplicationSignInManager signInManager,
VoiceRecognizeContext context)
{
UserManager = userManager; SignInManager = signInManager;
_context = context;
}
public ApplicationSignInManager SignInManager
{
get=>_signInManager??
HttpContext.GetOwinContext().Get<ApplicationSignInManager>(); private set => _signInManager = value;
}
public ApplicationUserManager UserManager
{
get=>_userManager??
HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>(); private set => _userManager = value;
}
```

Рисунок 3.7 - Доступ користувачів до програми:

```
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
if (!ModelState.IsValid)
{
return View(model);
}
varpath=
$"{Path.Combine(Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory),"Temp", "temp.wav")}";
SignInStatus result = SignInStatus.Success ; var verificationResult = "";
if (string.IsNullOrEmpty(model.SampleString) || model.SampleString.Length < 5 ||
!System.IO.File.Exists(path))
result = SignInStatus.Failure;
```

Рисунок 3.8 - Введення, зчитування логіну програмно:

Якщо дані голосової ідентифікації є невірними, у програмному додатку виконується обробка помилки з виведенням відповідного повідомлення користувачу. Зокрема, реалізується перевірка результатів біометричного аналізу, і у випадку невідповідності збереженим еталонним даним система генерує повідомлення про помилку авторизації (рис.3.9).

```
public async Task<ActionResult> VerifyCode(string provider, string returnUrl, bool rememberMe)
{
if (!await SignInManager.HasBeenVerifiedAsync())
return View("Error");
return View(new VerifyCodeViewModel { Provider = provider, ReturnUrl = returnUrl, RememberMe = rememberMe });
}
```

Рисунок 3.9 – Помилка авторизації

На наступних рисунках відображено процес реєстрації нового користувача (рис.3.10) та запису голосу користувача (рис.3.11).

```
public class Program
{
static async Task Main(string[] args)
{
//todo login
var recorder = new NAudioRecorder(); var client = new SpidHttpClient(); string profileId = null;
//todo all from db and set profile Id
// profileId = "4ab2f1d0-9e94-4380-a4bb-5613a91e3c6d";

//todo registration
```

Рисунок 3.10 - Процес реєстрації нового користувача

```
Console.WriteLine("Press any key to start verification test"); while (true)onsole.ReadLine();
Console.WriteLine("Please say anything to verify your voice, press any key to
recorder.StartRec(); Console.ReadLine(); recorder.StopRec(); Console.WriteLine("Verification");

var verificationResult  =   client.Varify(profileId, NAudioRecorder.CurentAudioStreem());
Console.WriteLine(verificationResult);

Console.WriteLine("Do you want to try again? Press any key to start");
}
```

Рисунок 3.11 - Запис зразку голосу нового користувача

Надалі на рисунках 3.12 - 3.21 будуть наведені скришити програмних кодів щодо реалізації завдання проєкту.

```
public class NAudioRecorder
{
public WaveInEvent waveSource = null; public WaveFileWriter waveFile = null; public void StartRec()
{
waveSource = new WaveInEvent { WaveFormat = new WaveFormat(44100, 1) }; waveSource.DataAvailable+=new
EventHandler<WaveInEventArgs>(waveSource_DataAvailable);
waveSource.RecordingStopped+=new EventHandler<StoppedEventArgs>(waveSource_RecordingStopped);

varpath=$"{Path.Combine(Directory.GetCurrentDirectory(),"Temp", "Test0001.wav")}";
waveFile = new WaveFileWriter(path, waveSource.WaveFormat); waveSource.StartRecording();
}
```

Рисунок 3.12 - Звернення до мікрофону для запису звуку

```
void waveSource_DataAvailable(object sender, WaveInEventArgs e)
{
if (waveFile != null)
{
waveFile.Write(e.Buffer, 0, e.BytesRecorded); waveFile.Flush();
}
}
```

Рисунок 3.13 - Запис звуку до бази даних

```
void waveSource_RecordingStopped(object sender, StoppedEventArgs e)
{
if (waveSource != null)
{
waveSource.Dispose(); waveSource = null;
}
if (waveFile != null)
{
waveFile.Dispose(); waveFile = null;
}
```

Рисунок 3.14 - Обробка зразку голосу, що було записано

```
public static MemoryStream CurentAudioStreem()
{
varpath=$"{Path.Combine(Directory.GetCurrentDirectory(),"Temp", "Test0001.wav")}";
Thread.Sleep(2000);
MemoryStream ms = new MemoryStream();
using (FileStream file = new FileStream(path, FileMode.Open, FileAccess.Read)) file.CopyTo(ms);
return ms;
}
```

Рисунок 3.15 - Збереження аудіо файлу зразку:

```
private async void recoTimer_TickAsync(object sender, object e)
{
recoTimer.Stop();
if (!inkAnalyzer.IsAnalyzing)
{
InkAnalysisResult result = await inkAnalyzer.AnalyzeAsync(); if (result.Status == InkAnalysisStatus.Updated)
{
var inkwordNodes = inkAnalyzer.AnalysisRoot.FindNodes(
InkAnalysisNodeKind.InkWord);
foreach (InkAnalysisInkWord node in inkwordNodes)
{
string recognizedText = node.RecognizedText; TheTextBlock.Text = recognizedText;
foreach (var strokeId in node.GetStrokeIds())
{
var stroke = TheInkCanvas.InkPresenter.StrokeContainer.GetStrokeById(strokeId);
stroke.Selected = true;
}
inkAnalyzer.RemoveDataForStrokes(node.GetStrokeIds());
}
TheInkCanvas.InkPresenter.StrokeContainer.DeleteSelected();
}
}
else
{
// Ink analyzer is busy. Wait a while and try again. recoTimer.Start();
}
}
```

Рисунок 3.16 - Перевірка зразку голосу при авторизації користувача

```
public class Neuron {
[XmlAttribute("weight")] public string data; [XmlIgnore]
public int[,] weight; // веса нейронов

[XmlIgnore]
public int minimum = 50; // порог

[XmlIgnore]
public int row = 64,column = 64;
//  Конструктор нейрона, public Neuron()
{
weight = new int[row,column]; randomizeWeights();
}
```

Рисунок 3.17 - Створення нейронної мережі

```
public int transferHard(int[,] input)
{
int power = 0;
for(int r = 0; r < row;r++)
{
for(int c = 0; c < column;c++)
{
power += weight[r,c]*input[r,c];
}
}
//Debug.Log("Power: " + power); return power >= minimum ? 1 : 0;
}
```

Рисунок 3.18 - Відповіді від нейронної мережі

```
// Вхід
public class Input
{
// Зв'язки з нейронами public Link[] OutgoingLinks;
}
// Связь входа с нейроном public class Link
{
// Нейрон
public Neuron Neuron;
// Вес связи
public double Weight;
}
public class Neuron
{
//Всі входи нейронаpublic Link[] IncomingLinks;
// Накопленный нейроном заряд public double Power { get; set; }
}
```

Рисунок 3.19 - Класи для представлення карти Кохонена:

```
public int Handle(int[] input)
{
for (var i = 0; i < _inputs.Length; i++)
{
var inputNeuron = _inputs[i];
foreach (var outgoingLink in inputNeuron.OutgoingLinks)
{
outgoingLink.Neuron.Power += outgoingLink.Weight * input[i];
}
}
var maxIndex = 0;
for (var i = 1; i < _neurons.Length; i++)
{
if (_neurons[i].Power > _neurons[maxIndex].Power) maxIndex = i;
}
//снять импульс со всех нейронов: foreach (var outputNeuron in _neurons)
{
outputNeuron.Power = 0;
}
return maxIndex;
}
```

Рисунок 3.20 - Обробка на основі нейронної мережі

```
public void Study(int[] input, int correctAnswer)
{
var neuron = _neurons[correctAnswer];
for (var i = 0; i < neuron.IncomingLinks.Length; i++)
{
var incomingLink = neuron.IncomingLinks[i];
incomingLink.Weight = incomingLink.Weight + 0.5 * (input[i] - incomingLink.Weight);
}
}
```

Рисунок 3.21 - Метод для навчання мережі розпізнавання

Таким чином, на основі створеного програмного коду була практично реалізована задача, що полягає в удосконаленні методу розпізнавання нечіткого мовлення осіб із порушеннями вимови окремих звуків шляхом використання нейромережевих технологій. Застосовані засоби програмування повністю забезпечили виконання вимог, визначених для практичної частини розробки.

3.4 Розробка інтерфейсу користувача

Під час створення інтерфейсу користувача були використані попередньо спроєктовані зразки діалогових вікон, подані на початку розділу. Далі розглянемо послідовність відображення діалогових вікон під час взаємодії з програмним засобом.

Після запуску додатку користувачу відкривається головна сторінка інтерфейсу. Як показано на рисунку 3.22, ця сторінка містить кілька основних кнопок: «Головна сторінка» (для швидкого повернення до початкового вікна), «Про нас» (для виведення інформації про систему та розробника), «Зв’язки» (для надання контактних даних), а також функціонально важливі кнопки «Увійти» та «Реєстрація».

Рисунок 3.22 – Загальний вигляд основного вікна

У вікні «Реєстрація» (рис. 3.23) користувачу надається можливість створити новий обліковий запис. Для цього потрібно ввести адресу електронної пошти та записати зразок голосу. Завершити процес реєстрації можна натисненням кнопки «Зареєструватися».

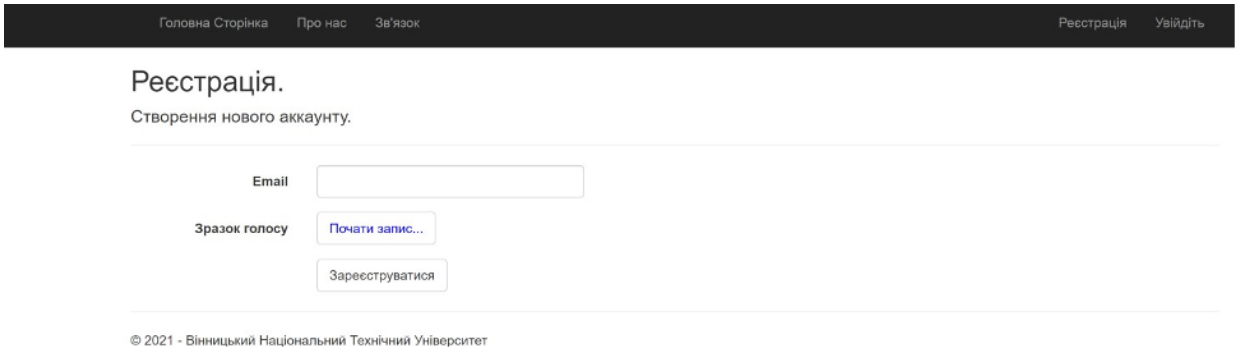


Рисунок 3.23 – Вигляд сторінки реєстрації користувача

У разі правильного введення реєстраційних даних (унікальний логін та коректно записаний голосовий зразок) відбувається успішна реєстрація користувача (рис.3.24). Важливо, щоб логін і голосовий зразок належали одному й тому самому користувачеві. Після цього виводиться повідомлення про успішне завершення реєстрації.

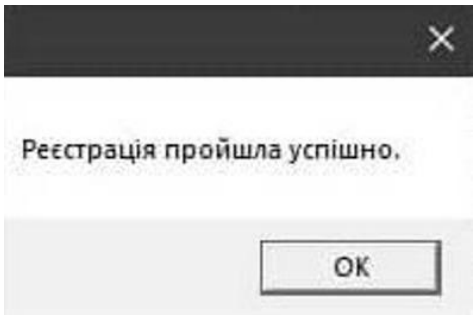


Рисунок 3.24 – Вигляд вікна з повідомленням про успішну реєстрацію

У вікні «Авторизація» користувач вводить адресу електронної пошти та записує голосовий зразок для входу в систему (рис.3.25). Підтвердження дій здійснюється кнопкою «Увійдить» .

Далі система перевіряє наявність користувача в базі даних: якщо логін та

голосовий зразок співпадають із зареєстрованими, вхід дозволяється автоматично. У разі некоректних даних, спроби підробки або невідповідності голосу, користувач отримує повідомлення про відсутність запису з такими параметрами (рис. 3.26), а доступ до системи залишається забороненим.

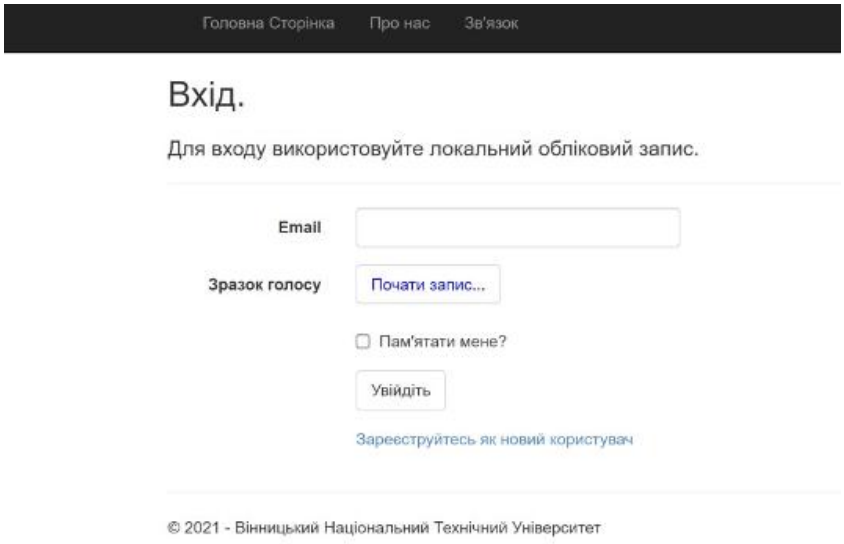


Рисунок 3.25 – Вигляд сторінки реєстрації користувача

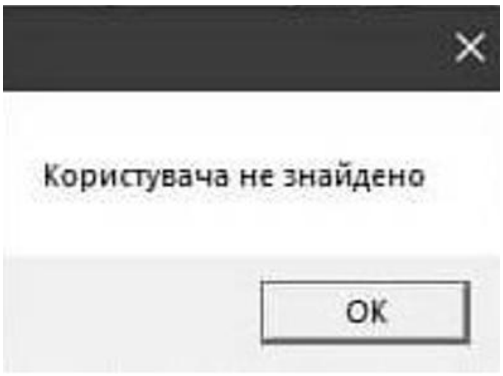


Рисунок 3.26 – Вікно з повідомленням про невдало авторизованого користувача

Якщо введені користувачем дані (логін, зразок голосу) співпадають із збереженими в базі даних та є достовірними, користувач отримує повідомлення про успішну авторизацію і йому надається доступ до захищеної системи (рис. 3.27).

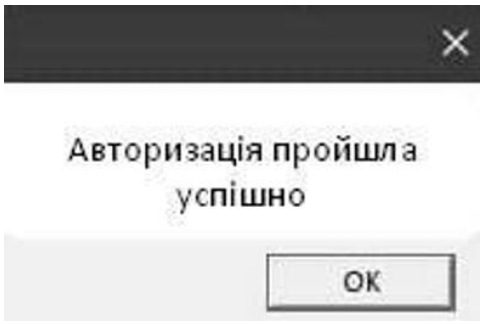


Рисунок 3.27 – Вигляд вікна про вдало авторизованого користувача

Отже, на основі розробленого інтерфейсу та реалізованого програмного коду в цьому розділі описано графічне оформлення додатку для користувача та особливості його використання.

3.5 Розробка інтерфейсу користувача

Для оцінки ефективності та доцільності застосування розробленого програмного засобу проведено тестування. Тестування здійснювалося на вибірці з 20 осіб. Звуковий сигнал записувався вбудованим мікрофоном з частотою дискретизації 16 кГц та 16-бітною розрядністю АЦП у моно режимі. Тривалість основного висловлювання становила близько 45 секунд, тестового – 15 секунд. Перевіряли працездатність при кількості компонентів моделі гаусових сумішей: 1, 2, 3, 4, 5, 10, 15, 20, 25, 30. На рисунку 3.28 наведено залежність кількості коректно ідентифікованих дикторів від числа компонентів карт.

Отже, оптимальна робота системи досягається при числі компонент, рівному 5, що забезпечує точність ідентифікації до 95%. Це свідчить про можливість успішного застосування реалізованого алгоритму для поставленої задачі.

Окрім оцінки точності, проведено вимір часу навчання нейромережі. При класичній нейромережі для ініціалізації параметрів моделі з 5 компонентами навчання тривало 7 хвилин 35 секунд, тоді як з використанням нейромережі на

основі карт Кохонена — 5 хвилин 3 секунди, що підтверджує перевагу останньої.

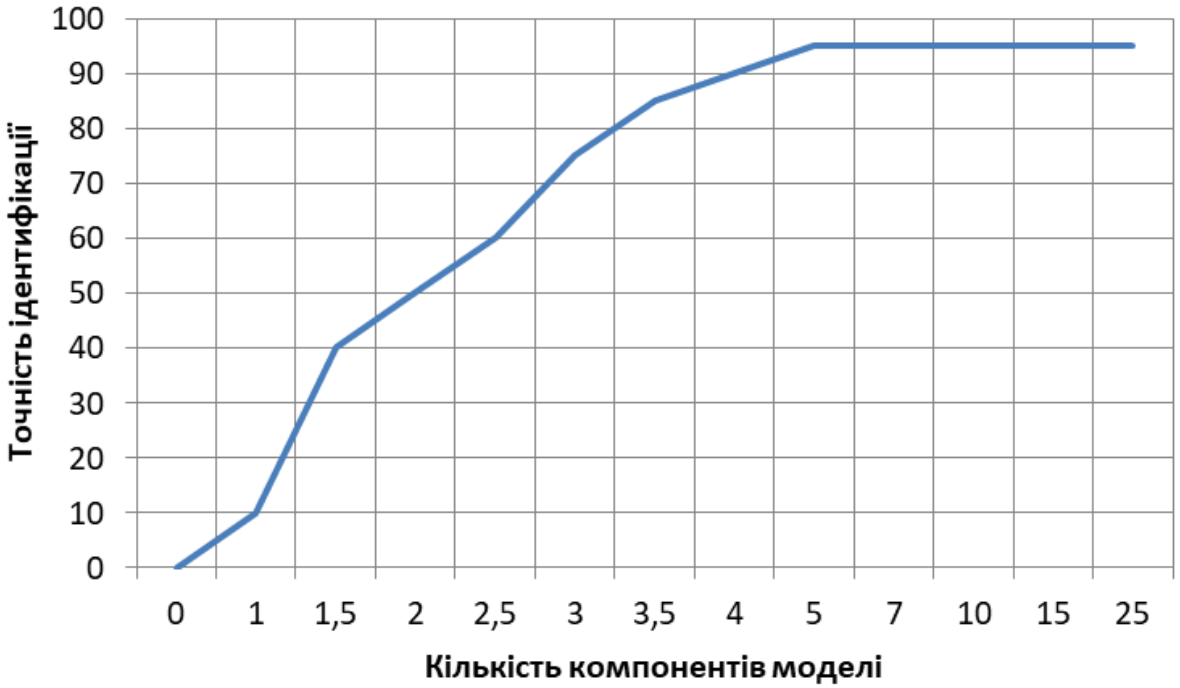


Рисунок 3.28 –Тестування системи

Таким чином, результати тестування свідчать про успішність і доцільність удосконаленого методу розпізнавання нечіткого голосу людей з вадами вимови на основі нейромережі карт Кохонена для практичного застосування.

3.6 Висновки до розділу

Отже, у цьому розділі детально описано практичну реалізацію програмного засобу, спрямованого на удосконалення методу розпізнавання нечіткого голосу у людей, які мають вади вимови окремих звуків, із застосуванням нейромережевих технологій. Розглянуто етапи проектування користувацького інтерфейсу, зокрема структуру основних сторінок додатку, а також кроки програмної реалізації, що включають розробку та інтеграцію ключових алгоритмів. Надано детальний опис інтерфейсу користувача, який орієнтований на зручність та зрозумілість, що є

важливим для практичного застосування системи.

Для створення програмного продукту була обрана мова програмування C#, що забезпечила можливість повної реалізації необхідного функціоналу і відповідає вимогам сучасної розробки. Особлива увага приділялась сумісності, продуктивності та можливостям інтеграції з іншими системами.

У розділі також наведено результати тестування розробленого програмного засобу, яке було проведено на вибірці користувачів з різними характеристиками голосу. Тестування підтвердило високу ефективність і точність системи, а також її здатність успішно розпізнавати голос користувачів із вадами вимови. Це свідчить про практичну доцільність і надійність запропонованого рішення.

Отже, підсумовуючи, розроблена інформаційна система є успішним прикладом застосування нейромережевих методів у біометричній ідентифікації, що відкриває перспективи для подальшого вдосконалення систем розпізнавання голосу з урахуванням особливостей користувачів.

ВИСНОВКИ

У дипломній роботі удосконалено метод розпізнавання нечіткого голосу людей із вадами вимови окремих звуків на основі нейромережі. Особливе значення у задачі розпізнавання мови мають методи, засновані на нейромережевих технологіях, де результат є продуктом роботи мережі певної топології. Нейронні мережі складаються з великої кількості взаємопов’язаних простих обчислювальних елементів (нейроноподібних), кожен з яких виконує базові функції.

Оскільки ідентифікація користувачів за унікальними характеристиками голосу доповнюється розпізнаванням нечіткого голосу, основним завданням роботи стало підвищення достовірності ідентифікації, щоб вади вимови не призводили до помилок реєстрації або авторизації.

Відповідно до поставлених завдань було розроблено програмний засіб, робота над яким описана у чотирьох розділах.

У першому розділі проведено теоретичний огляд галузі, здійснено аналіз методів і засобів ідентифікації та автентифікації користувачів, зокрема біометричної ідентифікації на основі голосу та нейронних мереж. Враховуючи опрацьований матеріал, запропоновано підвищення достовірності голосової ідентифікації за допомогою нейромережі та карт Кохонена.

Другий розділ присвячено удосконаленню обраного методу, розробці алгоритму роботи програмного засобу, а також обґрунтуванню вибору мови та середовища програмування. Для зменшення помилок застосовано нейромережу Кохонена, яка дозволяє скоротити час розпізнавання на 30–80% порівняно з існуючими рішеннями. У цьому розділі описано практичну реалізацію програмного засобу для розпізнавання нечіткого голосу з урахуванням вад вимови.

У третьому розділі висвітлено проектування інтерфейсу користувача, етапи програмної реалізації та тестування розробленої системи. Для створення

програмного продукту обрана мова програмування C#.

Виявлено, що оптимальна робота системи досягається при числі компонент, що дорівнює 5, з точністю ідентифікації до 95%, що підтверджує ефективність реалізованого алгоритму. Крім того, проведено оцінку часу навчання нейромережі: класична нейромережа навчалась 7 хв. 35 сек., тоді як нейромережа на основі карт Кохонена — 5 хв. 3 сек., що підтверджує перевагу застосованого підходу. Тестування показало, що розроблений програмний засіб є успішним і придатним до використання.

Отже, аналіз отриманих результатів свідчить, що в ході дослідження вдосконалено метод розпізнавання нечіткого голосу людей з вадами вимови на основі нейромережі та створено програмний продукт для реалізації цього методу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Джигирей В. С. Біометричні технології: принципи та засоби реалізації / В. С. Джигирей. – Київ : Академвидав, 2021. – 280 с.

2. Біометрія. URL: https://uk.wikipedia. org/wiki/Біометрія#: ~:text=Біоме́трія%20 (дата звернення 28.04.2025).

3. Furui S. Digital Speech Processing: Synthesis, and Recognition / S. Furui. – 3rd ed. – Boca Raton : CRC Press, 2021. – 352 p.

4. Rabiner L. R., Schafer R. W. Theory and Applications of Digital Speech Processing /L.R. Rabiner, R.W. Schafer. – Upper Saddle River : Pearson, 2019. – 912 p.

5. Deller J. R., Proakis J. G., Hansen J. H. Discrete-Time Processing of Speech Signals / J. R. Deller, J. G. Proakis, J. H. Hansen. – Hoboken : Wiley-IEEE Press, 2022. – 752 p.

6. Амплітудно-частотна характеристика. URL: https://uk.wikipedia. org/wiki/ Амплітудно-частотна _характеристика (дата звернення 10.05.2025).

7. Huang X., Acero A., Hon H. Spoken Language Processing: A Guide to Theory, Algorithm and System Development / X. Huang, A. Acero, H. Hon. – Upper Saddle River : Prentice Hall, 2020. – 960 p.

8. Шевчук О. В. Нейронні мережі: теорія і застосування : навч. посіб. / О. В. Шевчук. – Київ : НаУКМА, 2022. – 340 с.

9. Goodfellow I., Bengio Y., Courville A. Deep Learning / I. Goodfellow, Y. Bengio, A. Courville. – Cambridge : MIT Press, 2016. – 800 p.

10. Haykin S. Neural Networks and Learning Machines / S. Haykin. – 3rd ed. – New York : Pearson, 2021. – 936 p.

11. Kohonen T. Self-Organizing Maps / T. Kohonen. – 3rd ed. – Berlin : Springer, 2020. – 512 p.

12. Stevens S. S., Volkmann J., Newman E. A scale for the measurement of the psychological magnitude of pitch / S. S. Stevens, J. Volkmann, E. Newman // Journal of the Acoustical Society of America. – 1937. – Vol. 8, No. 3. – P. 185–190.

13. Jurafsky D., Martin J. H. Speech and Language Processing / D. Jurafsky, J. H. Martin. – 3rd ed. – Boston : Pearson, 2023. – 960 p.

14. Bishop C. M. Pattern Recognition and Machine Learning / C. M. Bishop. – New York : Springer, 2022. – 738 p.

15. Troelsen A., Japikse P. Pro C# 11 with .NET 7: Foundational Principles and Practices in Programming / A. Troelsen, P. Japikse. – New York : Apress, 2023. – 1340 p.

16. Liberty J. Programming C# 10.0 / J. Liberty. – 9th ed. – Sebastopol : O'Reilly Media, 2022. – 800 p.

17. Freeman A., MacDonald M. Pro .NET Windows Forms / A. Freeman, M. MacDonald. – New York : Apress, 2023. – 750 p.

18. Гнатюк С. С. Visual Studio для студентів: посібник / С. С. Гнатюк. – Львів : Видавництво Львівської політехніки, 2021. – 160 с.

19. Introduction to Self-Organizing Maps (SOM. URL: https://towards datascience.com/introduction-to-self-organizing-maps-4ab3fa4e5f17 (Laste of access: 03.05.2025).

20. Могильний Олег, Ставицький Андрій. Основи проєктування інтерфейсів користувача. - Львів : ЛНУ ім. І. Франка, 2020. - 138 с.

22. Nielsen Norman Group. User Interface Design URL: https://www.nngroup.com/articles/user-interface-design-basics/ (Laste of access: 20.05.2025).

ДОДАТОК А
ЛІСТИНГИ ДЛЯ СИСТЕМИИРОЗПІЗНАВАННЯ НЕЧІТКОГО ГОЛОСУ

Лістинг А.1 - Program

```
using System;
using System.Threading.Tasks;

namespace HelperRecognize
{
  public class Program
  {
    static async Task Main(string[] args)
    {
      //todo login
      var recorder = new

      NAudioRecorder(); var

      client = new

      SpidHttpClient(); string

      profileId = null;

      //todo all from db and set profile Id
     //  profileId = "4ab2f1d0-9e94-4380-a4bb-5613a91e3c6d";

      //todo
      registratio
      n if
      (profileId
      == null)
      {
        profileId = client.CreateProfile();

        Console.WriteLine("Your profileId:

        " + profileId);


        Console.WriteLine("Press any key for
        start record"); Console.ReadLine();

        recorder.StartRec();

        Console.WriteLine("Recording has started.  Press any
        key to stop.."); Console.ReadLine();
```

```
            recorder.StopRec();
            Console.WriteLine("Encoding
            the voice.................");

            var       encollmentResult   =
                      client.EncrollProfile(profileId,
NAudioRecorder.CurentAudioStreem());


            Console.WriteLine(encollmen
            tResult);
            Console.WriteLine("Enrollme
            nt complete");

          }
```

Лістинг А2 - Recorder

```
using
System;
using
System.IO;
using
System.Threadin
g; using
NAudio.Wave;

namespace HelperRecognize
{
  public class NAudioRecorder
  {
    public WaveInEvent
    waveSource = null; public
    WaveFileWriter waveFile =
    null;

    public void StartRec()
    {
      waveSource = new WaveInEvent { WaveFormat = new
      WaveFormat(44100, 1) };

      waveSource.DataAvailable            +=
                                          new
EventHandler<WaveInEventArgs>(waveSource_DataAvailable);
      waveSource.RecordingStopped          +=
                                           new
EventHandler<StoppedEventArgs>(waveSource_RecordingStopped);
```

```
  var  path  =
       $"{Path.Combine(Directory.GetCurrentDirectory(),
       "Temp", "Test0001.wav")}";

  waveFile = new WaveFileWriter(path,
  waveSource.WaveFormat);

  waveSource.StartRecording();
}

public void StopRec()
{
  waveSource.StopRecording();

}

void waveSource_DataAvailable(object sender, WaveInEventArgs
e)
{
  if (waveFile != null)
  {
    waveFile.Write(e.Buffer, 0,
    e.BytesRecorded); waveFile.Flush();
  }
}

void waveSource_RecordingStopped(object sender,
StoppedEventArgs e)
{

  if (waveSource != null)
  {
    waveSource.Disp
    ose();
    waveSource =
    null;
  }

  if (waveFile != null)
  {
    waveFile.Dispos
    e(); waveFile =
    null;
  }
}
public static MemoryStream CurentAudioStreem()
{
  var  path  =
       $"{Path.Combine(Directory.GetCurrentDirectory(),
       "Temp", "Test0001.wav")}";

  Thread.Sleep(2000);
  MemoryStream ms = new MemoryStream();
```

```
        using (FileStream file = new FileStream(path,
          FileMode.Open, FileAccess.Read)) file.CopyTo(ms);

        return ms;
      }

    }
  }
```

Національний університет «Запорізька політехніка»
Кафедра «Комп'ютерні системи та мережі»

1

ДИПЛОМНИЙ ПРОЄКТ

на тему: «РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ НЕЧІТКОГО ГОЛОСУ ЛЮДЕЙ НА ОСНОВІ НЕЙРОМЕРЕЖІ»

Ст. групи КНТ- 512сп Клим ЯЦЕНКО
Керівник: М.Б. ІЛЬЯШЕНКО

МЕТА РОБОТИ:

удосконалення методу розпізнавання нечіткого мовлення осіб із вадами вимови окремих звуків на основі нейронних мереж.

Завдання:

проаналізувати методи біометричної ідентифікації користувачів;

удосконалити та розробити алгоритм розпізнавання нечіткого голосу осіб із вадами мовлення;

реалізувати програмний засіб для цього завдання з використанням нейромереж;

2

АНАЛІЗ ЗАСОБІВ ПЕРЕВІРКИ КОРИСТУВАЧІВ

- **Аналіз біометричних особливостей при ідентифікації користувача,**
- **Аналіз систем розпізнавання голосу**
- **Аналіз реалізованих засобів визначення вад голосу людини**
- **Варіанти використання нейронних мереж:**
  - навчання з учителем (supervised) ;
  - навчання без учителя (unsupervised);
  - змішане (гібридне) навчання

3

ЗАСОБИ ДЛЯ РОЗРОБКИ СИСТЕМИ

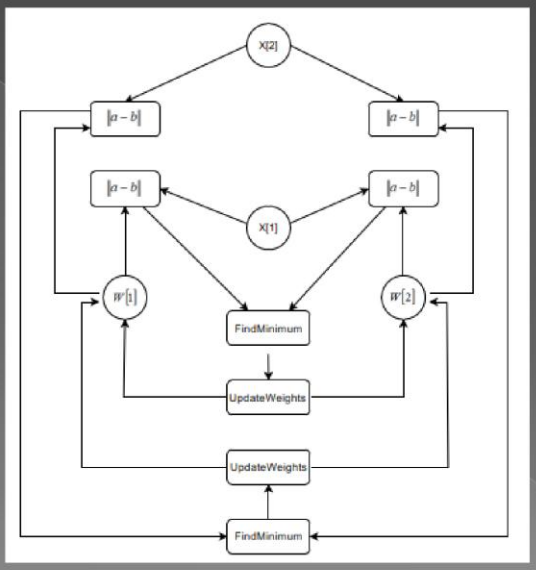
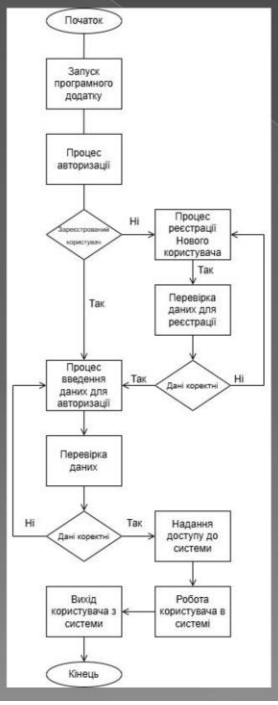


Схема графу навчання карти Кохонена

4

РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ НЕЧІТКОГО ГОЛОСУ



Крок 1. Запуск програмного засобу.

Крок 2. Авторизація:

- якщо користувач зареєстрований - перехід до кроку 5;
- якщо користувач не зареєстрований - перехід до кроку 3.

Крок 3. Реєстрація користувача: введення електронної пошти та запис голосового зразка.

Крок 4. Перевірка коректності введених даних:

- при коректних даних - перехід до кроку 5;
- при некоректних - повернення до кроку 3.

Крок 5. Введення даних для авторизації: логін (електронна пошта) та голосовий зразок через мікрофон.

Крок 6. Перевірка достовірності авторизаційних даних:

- якщо дані невірні - відображення повідомлення і повернення до кроку 5;
- якщо дані вірні - відкривається головне вікно програми, доступна подальша робота.

Крок 7. Робота авторизованого користувача в системі.

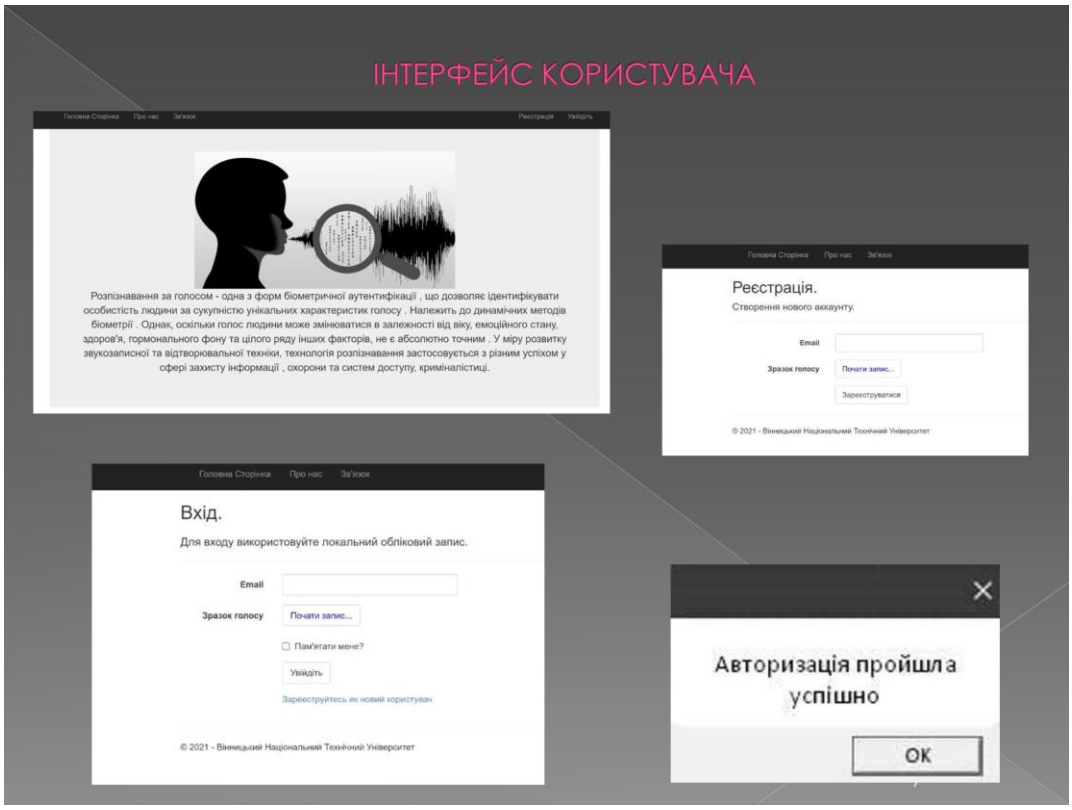
Крок 8. Завершення роботи, вихід із системи через кнопку «Вихід».

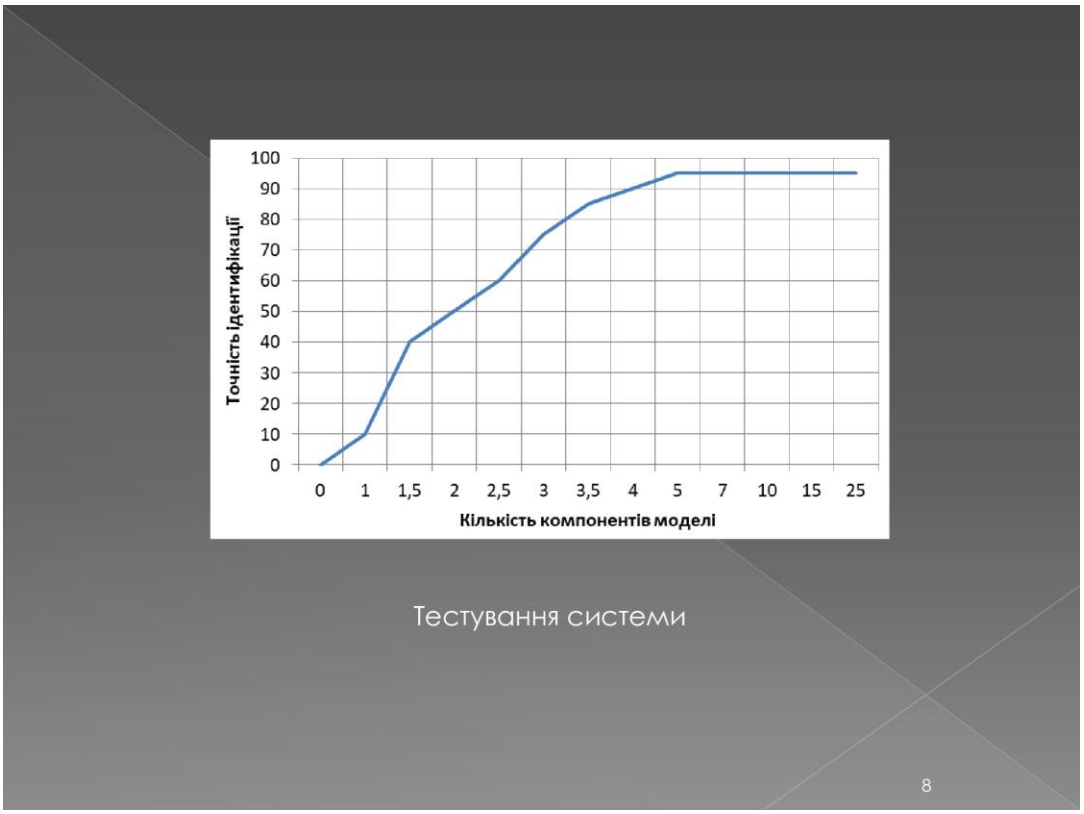
5

Бібліотеки :

- ‘System.Speech.Recognition – для розпізнавання мови;
- ‘System.Speech.Synthesis – для синтезу мовлення (озвучування повідомлень);
- ‘System.IO – для роботи з файлами та потоками даних;
- System.Windows.Forms – для створення графічного інтерфейсу користувача;
- ‘System.Drawing – для графічного відображення елементів інтерфейсу;
- Accord.Audio та Accord.Neuro (з пакету Accord.NET) – для обробки аудіоданих, реалізації нейронної мережі, а також побудови та тренування карти Кохонена;
- ‘NAudio – для захоплення, збереження та обробки аудіосигналу в режимі реального часу.

6


ІНТЕРФЕЙС КОРИСТУВАЧА
Головна Сторінка Про нас Зв'язок Реєстрація Увійдіть
Розпізнавання за голосом - одна з форм біометричної аутентифікації , що дозволяє ідентифікувати особистість людини за сукупністю унікальних характеристик голосу . Належить до динамічних методів біометрії . Однак, оскільки голос людини може змінюватися в залежності від віку, емоційного стану, здоров'я, гормонального фону та цілого ряду інших факторів, не є абсолютно точним . У міру розвитку звукозаписної та відтворювальної техніки, технологія розпізнавання застосовується з різним успіхом у сфері захисту інформації , охорони та систем доступу, криміналістиці.
Головна Сторінка Про нас Зв'язок
Реєстрація.
Створення нового аккаунту.
Email
Зразок голосу
Почати запис...
Зареєструватися
© 2021 - Вінницький Національний Технічний Університет
Головна Сторінка Про нас Зв'язок
Вхід.
Для входу використовуйте локальний обліковий запис.
Email
Зразок голосу
Почати запис...
Пам'ятати мене?
Увійдіть
Зареєструйтесь як новий користувач
© 2021 - Вінницький Національний Технічний Університет
Авторизація пройшла успішно
OK


Точність ідентифікації
100
90
80
70
60
50
40
30
20
10
0
0 1 1,5 2 2,5 3 3,5 4 5 7 10 15 25
Кількість компонентів моделі
Тестування системи
8

Лістинг А.1 – Файл Program

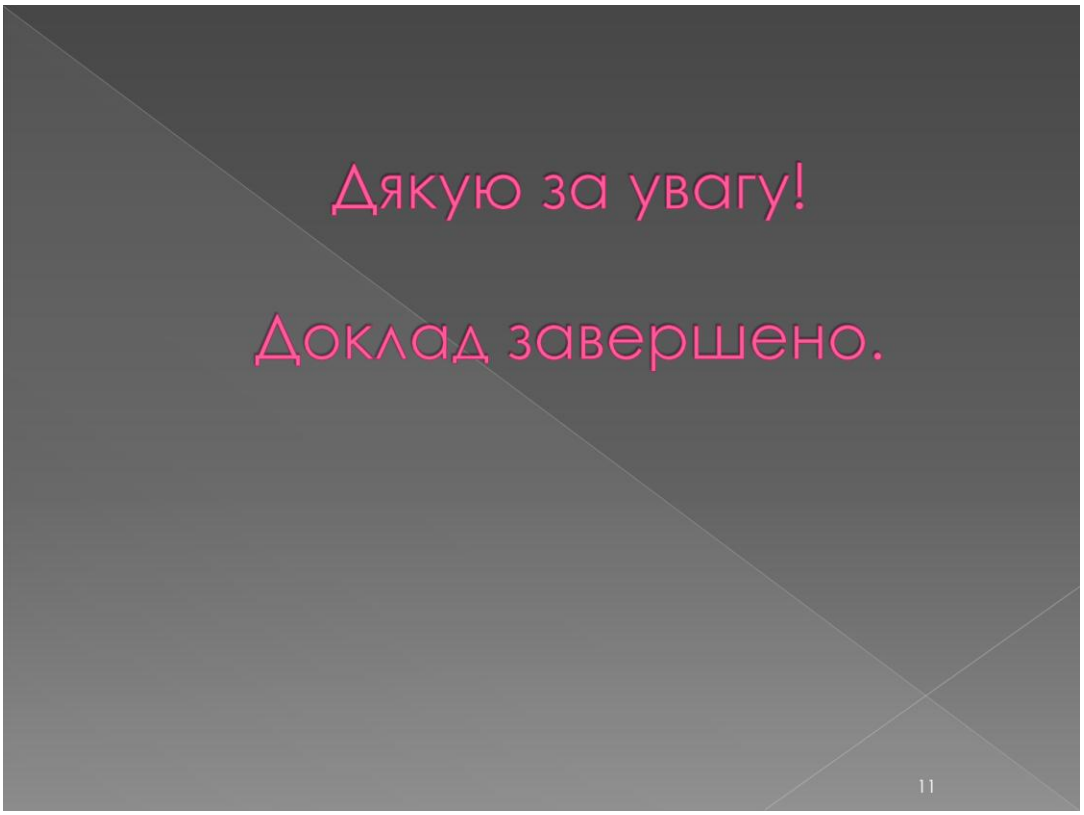
```
using System;
using System.Threading.Tasks;
namespace HelperRecognize
{
public class Program
{
static async Task Main(string[] args)
{
//todo login
var recorder = new NAudioRecorder(); var client = new SpidHttpClient();
string profileId = null;
//todo all from db and set profile Id
// profileId = "4ab2f1d0-9e94-4380-a4bb-5613a91e3c6d";
//todo registration if (profileId == null)
{
profileId = client.CreateProfile(); Console.WriteLine("Your profileId:
" + profileId);

Console.WriteLine("Press any key for start record");
Console.ReadLine();
recorder.StartRec();
```

9

ВИСНОВКИ:

- здійснено аналіз методів і засобів ідентифікації та автентифікації користувачів, зокрема біометричної ідентифікації на основі голосу та нейронних мереж за допомогою нейромережі та карт Кохонена;
- удосконалено метод розпізнавання нечіткого голосу людей із вадами вимови окремих звуків на основі нейромережі;
- було розроблено програмний продукт мовою програмування C#;
- спроєктувано інтерфейс користувача;
- тестування розробленої системи .

10


Дякую за увагу!
Доклад завершено.
11