

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ  
СІКОРСЬКОГО»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

"На правах рукопису"  
УДК \_\_\_\_\_

«До захисту допущено»

Завідувач кафедри

\_\_\_\_\_ Наталія АУШЕВА

“ ” \_\_\_\_\_ 2025

р.

**Магістерська дисертація**

на здобуття ступеня другого (магістерського) рівня вищої освіти  
за освітньою програмою “Цифрові технології в енергетиці”  
зі спеціальності 122 “Комп’ютерні науки”

на тему: Методи машинного навчання для визначення біологічного віку серця на  
основі ЕКГ

Виконав: студент 2 курсу, групи ТР-42мп

Антонкін Дмитро Олександрович

(прізвище, ім’я, по батькові)

\_\_\_\_\_ (підпис)

Науковий керівник професор каф. Цифрових технологій  
в енергетиці, д.т.н., Володимир СЛІПЧЕНКО

(науковий ступінь, вчене звання, ім’я ПРІЗВИЩЕ)

\_\_\_\_\_ (підпис)

Рецензент доцент каф. Інженерії програмного забезпечення  
в енергетиці, к.т.н., Валерій КУЗЬМІНИХ

(посада, науковий ступінь, вчене звання, ім’я ПРІЗВИЩЕ)

\_\_\_\_\_ (підпис)

Н.контроль доцент каф. Цифрових технологій  
в енергетиці, д.ф., Артем ГУРІН

(посада, ім’я ПРІЗВИЩЕ )

\_\_\_\_\_ (підпис)

Засвідчую, що у цій магістерській дисертації  
немає запозичень з праць інших авторів без  
відповідних посилань.

Студент \_\_\_\_\_

(підпис)

Київ - 2025

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
імені ІГОРЯ СІКОРСЬКОГО”**

**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ**

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти - другий (магістерський)

спеціальність 122 “Комп’ютерні науки”

Освітньо-наукова програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ

Завідувач кафедри ЦТЕ

Наталія АУШЕВА

«\_\_» \_\_\_\_\_ 2025 р.

**ЗАВДАННЯ  
на магістерську дисертацію студенту**

Антонкіну Дмитру Олександровичу

(прізвище, ім’я, по батькові)

1. Тема роботи Методи машинного навчання для визначення біологічного віку серця на основі ЕКГ

керівник роботи Сліпченко Володимир Георгійович, д.т.н, професор

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

затверджені наказом по університету від « 03 листопада 2025 р. № 4779-с

2. Термін подання студентом дисертації 06.12.2025

3. Об’єкт дослідження: методи машинного навчання для аналізу електрокардіографічних сигналів і визначення біологічного віку серця

4. Вихідні дані: мови програмування Python та JavaScript, платформа NodeJS, фреймворк Express, бібліотеки pandas, sklearn, TensorFlow, React, reduxtoolkit, мова розмітки HTML, таблиці стилів CSS.

5. Перелік завдань: 1) провести аналіз існуючих методів визначення біологічного віку та дослідити набори біомедичних даних. 2) розробити та реалізувати алгоритми обробки сигналів ЕКГ. 3) Дослідити та порівняти ефективність підходів машинного навчання: класичних моделей та нейромережових архітектур. 4)

розробити та оптимізувати удосконалену архітектуру глибокої нейронної мережі 5) створити клієнт-серверний програмний прототип системи для демонстрації практичного застосування розробленої моделі в клінічних умовах.

6. Орієнтований перелік ілюстративного матеріалу: типова архітектура CNN, архітектура нейронної мережі Елмана, діаграма варіантів використання, результати навчання класичних моделей, результати навчання нейронних мереж, аналіз тренування нейронних мереж.

7. Орієнтований перелік публікацій : Антонкін Д. О., Полягушко Л. Г., Сліпченко В. Г. Методи машинного навчання для оцінки біологічного віку серця на основі ЕКГ // Інформаційні технології в освіті, техніці та промисловості : матеріали Всеукр. наук.-практ. конф., (Івано-Франківськ, 09 жовт. 2025 р.). – Івано-Франківськ : ІФНТУНГ, 2025. – С. 184–185. – URL: <https://nung.edu.ua/departament/institut-informaciy-nikh-tekh-nologiy/vidannya>. DOI: <https://doi.org/10.5281/zenodo.17295363>.

6. Дата видачі завдання «16» вересня 2024 р.

### КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів підготовки магістерської дисертації             | Термін виконання  | Примітка |
|-------|--------------------------------------------------------------|-------------------|----------|
| 1     | Аналіз вимог, методів дослідження за літературними джерелами | 01.09.25-15.09.25 |          |
| 2     | Вибір та аналіз набору даних                                 | 10.09.25          |          |
| 3     | Програмна реалізація                                         | 26.09.25-17.10.25 |          |
| 4     | Захист програмного продукту                                  | 20.10.25          |          |
| 5     | Передзахист                                                  | 24.11.25          |          |
| 6     | Нормоконтроль                                                | 24.11.25-28.11.25 |          |
| 7     | Подання документів на кафедру                                | 04.12.25          |          |
| 8     | Захист                                                       |                   |          |

Студент

(підпис)

**Дмитро АНТОНКІН**

(ім'я, ПРІЗВИЩЕ)

Керівник роботи

(підпис)

**Володимир СЛІПЧЕНКО**

(ім'я, ПРІЗВИЩЕ)

# РЕФЕРАТ

**Актуальність теми.** Серцево-судинні захворювання (ССЗ) продовжують займати перше місце серед причин смертності у глобальному масштабі, що зумовлює гостру потребу в удосконаленні методів ранньої діагностики. Традиційні маркери ризику, такі як хронологічний вік, часто не відображають реального фізіологічного зносу серцево-судинної системи, оскільки процеси старіння є індивідуальними та гетерогенними. У цьому контексті концепція «біологічного віку серця» (Heart Age) набуває критичного значення як інтегральний показник здоров'я. Існуючі діагностичні методи є або інвазивними, або дороговартісними, або залежать від суб'єктивної оцінки лікаря. Стрімкий розвиток методів штучного інтелекту, зокрема глибокого навчання (Deep Learning), відкриває унікальні можливості для автоматизованого аналізу електрокардіограм (ЕКГ), дозволяючи виявляти приховані нелінійні патерни старіння, недоступні для візуальної інтерпретації.

**Мета роботи** полягає у підвищенні ефективності та точності оцінювання біологічного віку людини шляхом розробки інтелектуальної програмної системи, що базується на передових архітектурах та підходах машинного навчання та глибоких нейронних мереж.

## **Завдання дослідження:**

- Провести системний аналіз існуючих методів визначення біологічного віку та дослідити репрезентативні набори біомедичних даних (зокрема, базу Jena University Hospital).
- Розробити та програмно реалізувати алгоритми попередньої обробки сигналів ЕКГ, включаючи цифрову фільтрацію шумів, адаптивну сегментацію, нормалізацію та аугментацію даних для покращення навчання моделей.
- Дослідити та порівняти ефективність різних підходів машинного навчання: класичних моделей на основі інженерії ознак (Logistic Regression, Random Forest) та нейромережевих архітектур (Autoencoder, RNN, CNN) в умовах дисбалансу класів.

- Розробити та оптимізувати удосконалену архітектуру глибокої нейронної мережі (Advanced CNN), що поєднує залишкові з'єднання (ResNet) та механізм уваги (Attention), для досягнення високої точності класифікації.
- Створити повнофункціональний клієнт-серверний програмний прототип системи для демонстрації практичного застосування розробленої моделі в клінічних умовах.

**Об'єкт дослідження:** методи, моделі та алгоритми глибокого навчання, зокрема згорткові нейронні мережі (CNN), рекурентні мережі (RNN) та механізми уваги, що застосовуються для класифікації біологічного віку за даними багатоканальної ЕКГ.

**Предмет дослідження:** класифікація біологічного віку серця на основі моделей Logistic Regression, Random Forest, Autoencoder, RNN та CNN із використанням даних електрокардіографії.

**Методи дослідження.** У роботі використано методи цифрової обробки сигналів (смугова фільтрація Баттерворта, метод Велча) для очищення та частотного аналізу даних; методи статистичного аналізу для дослідження розподілу ознак; методи машинного навчання та глибокого навчання (градієнтний спуск, зворотне поширення помилки) для тренування прогностичних моделей; методи програмної інженерії та веб-розробки (React, Node.js) для створення архітектури додатку.

**Наукова новизна одержаних результатів.** Вперше обґрунтовано та експериментально підтверджено ефективність застосування архітектури Advanced CNN, посиленої механізмом уваги (Attention) та залишковими зв'язками (ResNet), для вирішення задачі 15-класової класифікації біологічного віку на вкрай незбалансованому наборі даних. Доведено, що запропонований end-to-end підхід дозволяє досягти точності класифікації у 87%, перевершуючи існуючі аналоги, що базуються на ручному розрахунку варіабельності серцевого ритму.

**Практичне значення отриманих результатів.** Розроблено, протестовано та впроваджено у вигляді веб-додатку інтелектуальну систему «BioAge-ECG AI». Система забезпечує автоматизований аналіз ЕКГ, визначаючи біологічний вік серця з наданням оцінки впевненості прогнозу. Створений продукт може бути

використаний як ефективний інструмент підтримки прийняття рішень у кардіологічній практиці, превентивній медицині та системах телемоніторингу пацієнтів.

**Апробація результатів роботи.** Основні теоретичні положення та практичні результати дисертаційної роботи доповідались та обговорювались на Всеукраїнській Інтернет-конференції молодих учених і студентів «Інформаційні технології в освіті, техніці та промисловості» (ІТОТП-2025).

**Структура та обсяг магістерської дисертації.** Робота складається зі вступу, п'яти розділів, висновків, списку використаних джерел та додатків. Повний обсяг роботи становить 114 сторінок.

**Публікації.** За темою дисертації опубліковано 1 наукову працю.

**Ключові слова:** ЕЛЕКТРОКАРДІОГРАМА, БІОЛОГІЧНИЙ ВІК, ШТУЧНИЙ ІНТЕЛЕКТ, ГЛИБОКЕ НАВЧАННЯ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, RESNET, ATTENTION, КЛАСИФІКАЦІЯ, ДИСБАЛАНС ДАНИХ, WEB-ЗАСТОСУНОК.

# ABSTRACT

**Relevance of the topic.** Cardiovascular diseases (CVD) remain the leading cause of death globally, necessitating urgent improvements in early diagnostic methods. Traditional risk markers, such as chronological age, often fail to reflect the true physiological wear and tear of the cardiovascular system, as aging processes are highly individual and heterogeneous. In this context, the concept of "heart biological age" becomes critical as an integral health indicator. Existing diagnostic methods are often invasive, expensive, or reliant on subjective physician assessment. The rapid advancement of artificial intelligence, particularly Deep Learning, opens unique opportunities for automated electrocardiogram (ECG) analysis, enabling the detection of hidden non-linear aging patterns unavailable for visual interpretation.

**The purpose of the work** is to enhance the efficiency and accuracy of human biological age estimation by developing an intelligent software system based on advanced deep convolutional neural network architectures.

## **Research tasks:**

- Conduct a systematic analysis of existing biological age determination methods and investigate representative biomedical datasets (specifically, the Jena University Hospital database).
- Develop and implement algorithms for ECG signal preprocessing, including digital noise filtering, adaptive segmentation, normalization, and data augmentation to improve model training.
- Investigate and compare the efficiency of various machine learning approaches: classical feature engineering-based models (Logistic Regression, Random Forest) and neural network architectures (Autoencoder, RNN, CNN) under class imbalance conditions.
- Develop and optimize an advanced deep neural network architecture (Advanced CNN) combining residual connections (ResNet) and an attention mechanism (Attention) to achieve high classification accuracy.
- Create a fully functional client-server software prototype to demonstrate the

practical application of the developed model in clinical settings.

**Object of research:** the process of automated processing and intelligent analysis of electrocardiographic signals for solving diagnostic tasks.

**Subject of research:** deep learning methods, models, and algorithms, specifically convolutional neural networks (CNN), recurrent neural networks (RNN), and attention mechanisms applied to biological age classification based on multi-channel ECG data.

**Research methods.** The work employs digital signal processing methods (Butterworth bandpass filtering, Welch's method) for data cleaning and frequency analysis; statistical analysis methods for investigating feature distribution; machine learning and deep learning methods (gradient descent, backpropagation) for training predictive models; and software engineering and web development methods (React, Node.js) for creating the application architecture.

**Scientific novelty.** For the first time, the effectiveness of applying an Advanced CNN architecture reinforced with an Attention mechanism and residual connections (ResNet) to solve the 15-class biological age classification problem on a highly imbalanced dataset has been substantiated and experimentally confirmed. It is proven that the proposed end-to-end approach achieves a classification accuracy of 87%, surpassing existing analogues based on manual heart rate variability calculation.

**Practical value of the results obtained.** The "BioAge-ECG AI" intelligent system was developed, tested, and implemented as a web application. The system provides automated ECG analysis, determining the heart's biological age with a confidence score. The created product can be used as an effective decision support tool in cardiological practice, preventive medicine, and patient telemonitoring systems.

**Approbation of results.** The main theoretical provisions and practical results of the thesis were reported and discussed at the All-Ukrainian Internet Conference of Young Scientists and Students "Information Technologies in Education, Technology, and Industry" (ITOTP-2025).

**Structure and volume of the thesis.** The work consists of an introduction, five chapters, conclusions, a list of references, and appendices. The full volume of the work is 114 pages.

**Publications.** 1 scientific papers have been published on the topic of the thesis.

**Keywords:** ELECTROCARDIOGRAM, BIOLOGICAL AGE, ARTIFICIAL INTELLIGENCE, DEEP LEARNING, CONVOLUTIONAL NEURAL NETWORKS, RESNET, ATTENTION, CLASSIFICATION, DATA IMBALANCE, WEB APPLICATION.

# ЗМІСТ

|                                                                                                    |    |
|----------------------------------------------------------------------------------------------------|----|
| ВСТУП.....                                                                                         | 13 |
| 1 ПОСТАНОВКА ЗАДАЧІ ВИЗНАЧЕННЯ БІОЛОГІЧНОГО ВІКУ СЕРЦЯ ЛЮДИНИ.....                                 | 16 |
| 1.1 Актуальність та призначення програмного забезпечення.....                                      | 16 |
| 1.2 Огляд існуючих методів визначення біологічного віку за даними ЕКГ ...                          | 19 |
| 1.3 Аналіз набору даних "Jena University Hospital" .....                                           | 25 |
| 1.4 Формулювання експериментальних задач.....                                                      | 27 |
| 1.5 Висновки до розділу 1.....                                                                     | 29 |
| 2 ОБҐРУНТУВАННЯ МАТЕМАТИЧНИХ МЕТОДІВ ТА ПРОГРАМНИХ ЗАСОБІВ ДОСЛІДЖЕННЯ.....                        | 31 |
| 2.1 Обґрунтування вибору програмних засобів та технологічного стеку.....                           | 33 |
| 2.2 Математичне підґрунтя методів та алгоритмів дослідження.....                                   | 37 |
| 2.3 Обґрунтування архітектури системи .....                                                        | 50 |
| 2.4 Висновки до розділу 2.....                                                                     | 52 |
| 3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ ДЛЯ КЛАСИФІКАЦІЇ БІОЛОГІЧНОГО ВІКУ .....                           | 54 |
| 3.1 Обробка вхідного набору даних для навчання моделей.....                                        | 56 |
| 3.2 Машинне навчання (Logistic Regression та Random Forest) .....                                  | 62 |
| 3.3 Тренування нейронних мереж (1D CNN, RNN, RNN + Autoencoder).....                               | 66 |
| 3.4 Реалізація клієнтського веб-інтерфейсу .....                                                   | 82 |
| 3.5 Серверна частина та інтеграція модуля машинного навчання .....                                 | 86 |
| 3.6 Висновки до розділу 3.....                                                                     | 89 |
| 4 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ТА ДЕМОНСТРАЦІЯ РОБОТИ ПРОГРАМНОЇ СИСТЕМИ.....                          | 91 |
| 4.1 Вимоги до апаратного та програмного забезпечення .....                                         | 91 |
| 4.2 Інсталяція та розгортання програмної системи .....                                             | 92 |
| 4.3 Демонстрація функціоналу та сценарії роботи.....                                               | 93 |
| 4.4 Результати обчислювальних експериментів та переваги розробленої системи відносно аналогів..... | 96 |

|                                                                  |     |
|------------------------------------------------------------------|-----|
| 4.5 Висновки до розділу 4.....                                   | 97  |
| 5 СТАРТАП-ПРОЄКТ.....                                            | 99  |
| 5.1 Опис ідеї стартап-проєкту .....                              | 99  |
| 5.2 Технологічний аудит та аналіз конкурентного середовища ..... | 101 |
| 5.3 Аналіз ринкових можливостей та загроз.....                   | 103 |
| 5.4 Висновки щодо комерціалізації проєкту.....                   | 104 |
| ВИСНОВКИ .....                                                   | 105 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....                                  | 107 |
| ДОДАТОК А .....                                                  | 11  |

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БВ – Біологічний вік.

ХВ – Хронологічний вік.

ЕКГ – Електрокардіограма.

ССЗ – Серцево-судинні захворювання.

ІІІ (AI) – Штучний інтелект (Artificial Intelligence).

МН (ML) – Машинне навчання (Machine Learning).

ГН (DL) – Глибоке навчання (Deep Learning).

НМ – Нейронна мережа.

HRV (Heart Rate Variability) – Варіабельність серцевого ритму.

CNN (Convolutional Neural Network) – Згортова нейронна мережа.

RNN (Recurrent Neural Network) – Рекурентна нейронна мережа.

LSTM (Long Short-Term Memory) – Довга короткострокова пам'ять (тип рекурентної мережі).

Bi-LSTM (Bidirectional LSTM) – Двонаправлена мережа довгої короткострокової пам'яті.

GRU (Gated Recurrent Unit) – Керований рекурентний блок.

AE (Autoencoder) – Автоенкодер (Автокодувальник).

ResNet (Residual Network) – Залишкова нейронна мережа.

PCA (Principal Component Analysis) – Метод головних компонент.

SMOTE (Synthetic Minority Over-sampling Technique) – Техніка синтетичного передискретизування міноритарного класу.

FFT (Fast Fourier Transform) – Швидке перетворення Фур'є.

PSD (Power Spectral Density) – Спектральна густина потужності.

MSE (Mean Squared Error) – Середня квадратична похибка.

ReLU (Rectified Linear Unit) – Випрямлений лінійний вузол (функція активації).

API (Application Programming Interface) – Прикладний програмний інтерфейс.

REST (Representational State Transfer) – Архітектурний стиль обміну даними

(Передача репрезентативного стану).

JSON (JavaScript Object Notation) – Текстовий формат обміну даними.

SaaS (Software as a Service) – Програмне забезпечення як послуга.

## ВСТУП

Серцево-судинні захворювання (ССЗ) залишаються провідною причиною смертності у світі. При цьому, хронологічний вік пацієнта не завжди є точним індикатором ризику, оскільки процеси старіння є гетерогенними. У зв'язку з цим, концепція "біологічного віку серця" (Heart Age) набуває все більшої актуальності як інтегральний біомаркер, що відображає функціональний стан серцево-судинної системи [1]. Електрокардіографія (ЕКГ) є рутинним, доступним та неінвазивним методом діагностики, що містить величезний обсяг інформації про електричну активність міокарда. Сучасні методи машинного навчання (МН) відкривають можливість аналізувати складні, нелінійні патерни в ЕКГ-сигналах, які є недоступними для візуальної інтерпретації, але корелюють з фізіологічними процесами старіння.

Доцільність виконання роботи за освітньо-професійною програмою полягає в комплексному застосуванні передових методів інтелектуального аналізу даних - від класичних моделей до глибоких нейронних мереж (CNN, RNN, Autoencoders) - для вирішення складної біомедичної задачі. Критичний аналіз існуючих рішень [2] показує, що хоча моделі МН успішно застосовуються для детекції аритмій, задача визначення віку є менш дослідженою. Вона вимагає не лише розробки точних моделей, але й створення сучасної програмної інфраструктури (зокрема, клієнт-серверної архітектури на базі React та Node.js) для їх практичного впровадження.

Метою даної роботи є визначення закономірностей між характеристиками ЕКГ-сигналу та віковими групами пацієнтів і визначення найбільш ефективного методу машинного навчання для автоматизованої класифікації біологічного віку серця з подальшою розробкою програмного прототипу.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- провести аналіз предметної області, дослідити репрезентативний набір даних ЕКГ та сформулювати вимоги до програмного комплексу;
- розробити конвеєри підготовки даних: для екстракції ознак класичних моделей та для обробки сигналів для глибоких моделей;

- розробити та порівняти п'ять підходів до моделювання: логістична регресія, випадковий ліс, гібридна модель (AE + RNN), end-to-end RNN та end-to-end CNN;
- провести обчислювальні експерименти, оцінити точність та робастність розроблених моделей на імбалансованих даних, обравши найкращу та зберігши її чекпоінт у форматі .pth;
- розробити клієнт-серверний програмний прототип, що викликає Python-скрипт для демонстрації практичного застосування обраної моделі.

Об'єктом дослідження є методи, моделі та алгоритми глибокого навчання, зокрема згорткові нейронні мережі (CNN), рекурентні мережі (RNN) та механізми уваги, що застосовуються для класифікації біологічного віку за даними багатоканальної ЕКГ.

Предметом дослідження є класифікація біологічного віку серця на основі моделей Logistic Regression, Random Forest, Autoencoder, RNN та CNN із використанням даних електрокардіографії.

Для досягнення мети роботи були використані: методи цифрової обробки сигналів (використання бібліотеки wfdb для читання, фільтрації filtfilt та аналізу gqrs\_detect); методи інтелектуального аналізу даних (PCA, аналіз HRV nn50, rmssd); методи класичного машинного навчання (Логістична регресія, Випадковий ліс); методи глибокого навчання (AE, RNN, CNN); методи боротьби з дисбалансом класів (SMOTE, Focal Loss); методології веб-розробки (Node.js, Express, React) для створення програмного інтерфейсу.

Розроблений програмний комплекс, зокрема клієнт-серверний застосунок, демонструє готовий до впровадження прототип системи підтримки прийняття клінічних рішень. Він може бути використаний лікарями для швидкого скринінгу пацієнтів та ідентифікації осіб з прискореним біологічним старінням серця, що потребують поглибленої діагностики.

Основні результати дослідження були представлені на Всеукраїнській Інтернет-конференція молодих учених і студентів, ІТОТП-2025 [3].

Робота складається зі вступу, п'ятьох розділів, висновків, списку використаних джерел (39 джерел, у тому числі 38 - іноземною мовою) та 1 додатку

на 100 сторінках. Загальний обсяг дисертації - 114 сторінок. Основний зміст викладено на 93 сторінках. Роботу проілюстровано 3 таблицями та 38 рисунками.

# **1 ПОСТАНОВКА ЗАДАЧІ ВИЗНАЧЕННЯ БІОЛОГІЧНОГО ВІКУ СЕРЦЯ ЛЮДИНИ**

У даному розділі проводиться аналіз предметної області, що стосується визначення біологічного віку за фізіологічними сигналами. Виконується детальний огляд та обґрунтування вибору цільового набору даних, що є фундаментом для всіх подальших експериментів. На основі характеристик даних формуються експериментальні задачі, що будуть вирішуватись. Також описується загальна архітектура програмного комплексу, що включає як етапи моделювання, так і етап впровадження у вигляді клієнт-серверного застосунку на базі React та Node.js. Наприкінці розділу обґрунтовується вибір технологічного стеку.

## **1.1 Актуальність та призначення програмного забезпечення**

Людство переживає безпрецедентний демографічний зсув. За даними Всесвітньої організації охорони здоров'я (ВООЗ), частка населення світу віком понад 60 років стрімко зростає. Цей процес має прямий медичний наслідок: старіння є домінуючим, немодифікованим фактором ризику для більшості хронічних неінфекційних захворювань (НІЗ), серед яких абсолютне лідерство за показниками смертності утримують ССЗ [4].

Проблема полягає в тому, що хронологічний вік (кількість прожитих років) є неточним та занадто узагальненим показником для оцінки індивідуального ризику. Існує значна варіабельність у швидкості старіння між індивідуумами [5]. Два пацієнти однакового паспортного віку можуть мати кардинально різний ступінь фізіологічного зносу серцево-судинної системи. Це явище гетерогенності старіння призводить до того, що традиційні моделі оцінки ризику (наприклад, шкала SCORE або Framingham Risk Score) можуть недооцінювати небезпеку для одних пацієнтів і переоцінювати для інших.

Для вирішення цієї проблеми біомаркерів була розроблена концепція

біологічного віку (БВ). На відміну від хронологічного, біологічний вік є інтегральним показником, що відображає реальний стан здоров'я організму на клітинному та системному рівнях. Він є більш точним показником функціонального стану, накопичених пошкоджень та, як наслідок, ризику смертності, ніж хронологічний вік [6]. Оцінка біологічного віку стала однією з ключових задач превентивної та персоналізованої медицини, оскільки дозволяє виявити осіб із прискореним старінням до того, як у них маніфестують клінічні захворювання.

Для оцінки біологічного віку було запропоновано багато біомаркерів:

- геномні - довжина теломер;
- епігенетичні - так звані "епігенетичні годинники", засновані на патернах метилювання ДНК;
- біохімічні - рівні певних білків у крові (протеоміка).

Однак, всі ці методи мають фундаментальні недоліки, що унеможливлюють їх широке клінічне застосування: вони є інвазивними (потребують забору крові або тканин), високовартісними та вимагають складної і тривалої лабораторної обробки. Вони абсолютно не підходять для масового, регулярного та оперативного скринінгу населення.

На цьому тлі електрокардіограма (ЕКГ) виступає як майже ідеальний кандидат для неінвазивної оцінки біологічного віку саме серця. ЕКГ є унікальним поєднанням переваг:

- неінвазивність - процедура абсолютно безпечна і не викликає дискомфорту;
- доступність та дешевизна - вартість процедури мінімальна, а апарати ЕКГ є у кожному медичному закладі, від сільської амбулаторії до високоспеціалізованого кардіоцентру;
- повсюдність - технологія стандартизована і використовується десятиліттями. Більше того, вона активно інтегрується у споживчі пристрої (смарт-годинники, фітнес-трекери), що відкриває шлях до безперервного моніторингу;
- висока інформативність - ЕКГ-сигнал, навіть у одному відведенні, несе

в собі приховані ознаки процесу старіння. З віком відбуваються неминучі структурні та функціональні зміни в міокарді: накопичується фіброз (заміщення м'язової тканини сполучною), змінюється архітектура провідної системи серця, знижується чутливість до вегетативної регуляції. Ці процеси безпосередньо відображаються на ЕКГ:

- змінюється морфологія зубця Р (через фіброз передсердь);
- може подовжуватись інтервал QRS (через уповільнення провідності у шлуночках);
- знижується варіабельність серцевого ритму (HRV - Heart Rate Variability), що є класичною ознакою старіння вегетативної нервової системи [7].

Незважаючи на багатство інформації, традиційний візуальний аналіз ЕКГ лікарем не дозволяє кількісно оцінити ці ледь помітні зміни. Клініцист навчений виявляти гострі, макроскопічні патології (наприклад, інфаркт міокарда з елевацією ST, блокади ніжок пучка Гіса, фібриляцію передсердь). Проте дифузні, нелінійні, розподілені по всьому сигналу мікро-зміни, що сумарно відображають процес старіння, залишаються "невидимими" для людського ока.

Виникає методологічний розрив: ми маємо доступний та інформативний сигнал (ЕКГ), але не маємо інструменту для повноцінного вилучення з нього інформації про біологічний вік.

Цей розрив ефективно долається за допомогою сучасних методів машинного навчання (МН). Моделі глибокого навчання (Deep Learning), зокрема згорткові та рекурентні нейронні мережі, здатні аналізувати "сирий" ЕКГ-сигнал як багатовимірний часовий ряд. Вони автоматично вивчають ієрархію ознак: від низькорівневих (нахил сегмента) до високо-рівневих складних патернів, які ігнорує традиційна діагностика, але які надійно корелюють з віком.

Призначення програмного забезпечення (ПЗ), що розробляється в рамках цієї дисертації, - створити автоматизований, об'єктивний та кількісний інструмент для визначення біологічного віку серця (у вигляді класифікації по вікових групах) виключно на основі ЕКГ-сигналу.

ПЗ розробляється як система підтримки прийняття клінічних рішень (CDSS -

Clinical Decision Support System). Його фундаментальна мета - не замінити лікаря, а надати йому новий, потужний інструмент для стратифікації ризиків.

Ключовим показником, який генерує система, стає "delta-age" ( $\Delta$ -age) - різниця між прогнозованим моделлю віком та реальним хронологічним віком пацієнта. Лікар може використовувати його для скринінгу: якщо ПЗ прогнозує для 40-річного пацієнта вікову групу "60-64 роки", це є потужним тривожним сигналом ("delta-age"  $\approx +20$  років) [8]. Дослідження довели, що "delta-age" є незалежним предиктором розвитку ССЗ та смертності [9]. Такий "червоний прапорець" є безапеляційною підставою для негайного поглибленого обстеження (УЗД серця, стрес-тести, коронарографія) та призначення агресивної превентивної терапії (наприклад, статини, антигіпертензивні препарати, модифікація способу життя).

## **1.2 Огляд існуючих методів визначення біологічного віку за даними ЕКГ**

Зв'язок між фізіологічними змінами в серці, пов'язаними з віком, та їх відображенням на електрокардіограмі є предметом клінічного інтересу протягом десятиліть. Однак методи кількісної оцінки цього зв'язку кардинально еволюціонували. Історично, цей розвиток можна розділити на два великі етапи: (1) підходи, засновані на клінічних знаннях та інженерії ознак, та (2) сучасні "end-to-end" підходи на основі глибокого навчання.

Перші спроби пов'язати ЕКГ з віком базувалися на статистичному аналізі та класичних моделях машинного навчання, що вимагали апріорного (a priori) вибору ознак. Кардіологи та дослідники вручну виділяли з ЕКГ-сигналу набір параметрів, які, як відомо, корелюють з віком.

До цих "ручних" ознак належать:

- Аналіз варіабельності серцевого ритму (HRV): це був один з перших та найбільш досліджених біомаркерів. З віком спостерігається зниження загальної варіабельності серцевого ритму (HRV), що відображає зниження адаптивності

вегетативної нервової системи. Для цього розраховувалися часові (наприклад, SDNN, RMSSD) та частотні (LF, HF) показники HRV [10].

- Морфологічні та інтервальні ознаки:
  - Тривалість інтервалу P-Q (P-R): часто подовжується, що вказує на уповільнення атріовентрикулярної провідності.
  - Тривалість комплексу QRS: може незначно збільшуватися через уповільнення провідності у шлуночках (наприклад, через фіброз).
  - Скоригований інтервал QT (QTc): зміни в тривалості реполяризації шлуночків.
  - Амплітуда та морфологія зубця T: часто спостерігається зниження амплітуди.
  - Електрична вісь серця (QRS axis): з віком часто спостерігається її відхилення вліво.

Ці десятки розрахованих ознак (features) потім подавалися на вхід класичних моделей машинного навчання, таких як:

- Лінійна та Логістична регресія.
- Методи опорних векторів (SVM / SVR).
- Ансамблеві методи (Випадковий ліс, Градієнтний бустинг).

Головним обмеженням цього підходу була його залежність від експертних знань. Процес виділення ознак був трудомістким і, що найважливіше, обмежував модель лише тими патернами, які вже відомі клініцистам. Існувала висока ймовірність втратити тонку, нелінійну або приховану (латентну) інформацію, що міститься в "сирому" сигналі.

Приблизно з 2017-2018 років, завдяки поширенню великих цифрових наборів даних ЕКГ (таких як UK Biobank, PTB-XL та Physionet) та зростанню обчислювальних потужностей (GPU), відбувся парадигмальний зсув. Дослідники перейшли від інженерії ознак до "end-to-end" (наскрізних) моделей глибокого навчання, які навчаються безпосередньо на "сирих" або мінімально оброблених (наприклад, відфільтрованих) сегментах ЕКГ.

1D Згорткові Нейронні Мережі (1D-CNNs) - це де-факто стандарт і найбільш успішна архітектура для цієї задачі (рис. 1.1). Згорткові шари діють як ієрархічні

екстрактори ознак [11]. Перші шари вчаться розпізнавати базові морфології (як крутизна зубця R), тоді як глибші шари комбінують їх у складніші патерни (наприклад, специфічну форму комплексу ST-T), які корелюють з віком, але можуть не мати чіткого клінічного опису. Багато успішних робіт використовують адаптовані архітектури, такі як ResNet (Residual Networks) або Inception, які довели свою ефективність у уникненні затухання градієнта на дуже глибоких мережах.

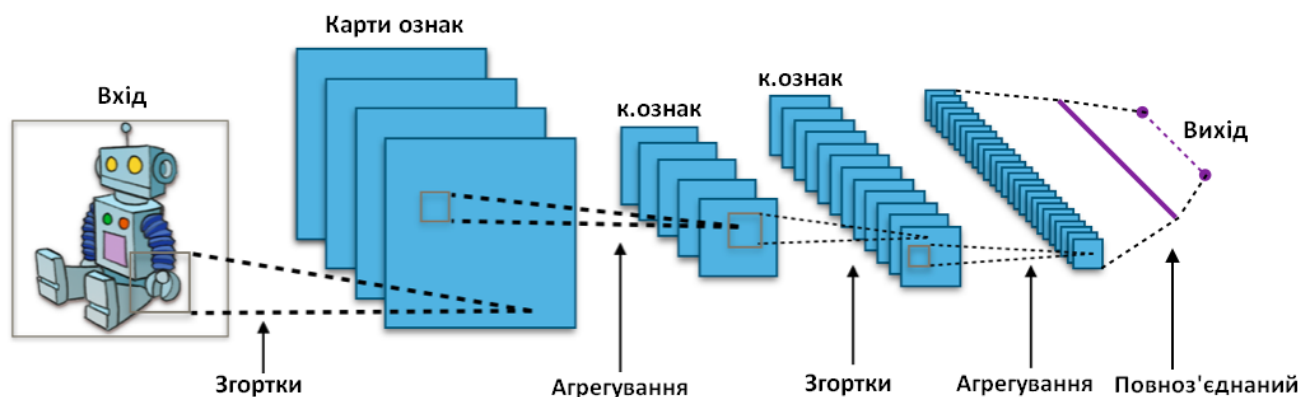


Рисунок 1.1 - Типова архітектура CNN

Архітектури, такі як LSTM (Long Short-Term Memory) та GRU (Gated Recurrent Unit), використовуються для моделювання часових залежностей у сигналі (рис.1.2). Вони здатні аналізувати послідовність серцевих скорочень, що є корисним для ознак типу HRV. Однак у чистих "end-to-end" задачах вони часто поступаються CNN або використовуються в гібридних моделях (наприклад, CNN-LSTM), де CNN виділяє просторові ознаки з кожного удару, а LSTM аналізує їх послідовність [12].

Більш новітні дослідження [13] почали інтегрувати механізми уваги (attention mechanisms), які дозволяють моделі "зважувати" важливість різних частин ЕКГ-сигналу. Наприклад, модель може навчитися приділяти більше уваги Р-хвилі та ST-сегменту, якщо вони є найбільш інформативними для прогнозування віку.

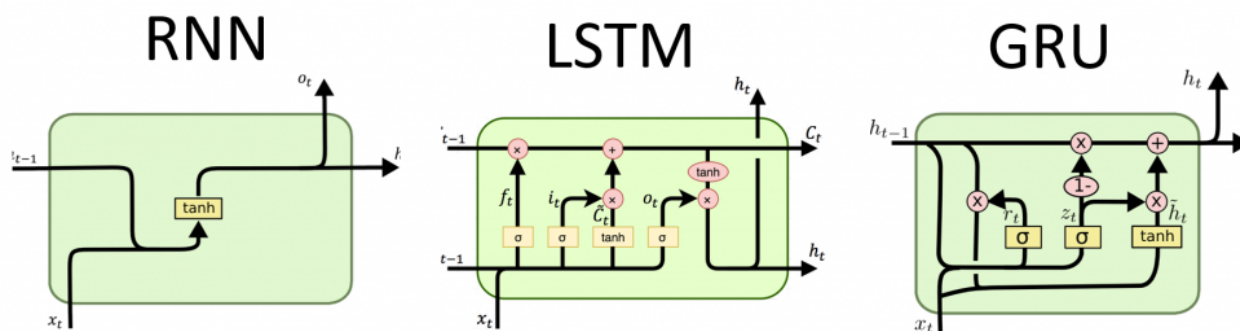


Рисунок 1.2 - Схематичне зображення архітектур RNN та LSTM RNN

Найважливішим результатом, що виник завдяки цим точним моделям глибокого навчання, стала концепція "ECG-age gap" (розрив ЕКГ-віку), або "delta-age" ( $\Delta$ -age).

Першою такою роботою, що отримала широкий розголос, стало дослідження групи під керівництвом Zachy Attia, опубліковане у престижному журналі Nature Communications.

Дослідження проводилось у клініці Мейо (Mayo Clinic, США), одному з провідних світових медичних центрів.

Команда використала величезний набір даних, що включав 499 727 пацієнтів. Для кожного пацієнта була взята як мінімум одна стандартна 12-відвідна ЕКГ у стані спокою (тривалістю 10 секунд).

Була розроблена та навчена згорткова нейронна мережа (CNN) для виконання задачі регресії - прогнозування хронологічного віку пацієнта безпосередньо з "сирих" даних ЕКГ-сигналу.

Модель досягла високої точності, прогнозуючи вік з середньою абсолютною помилкою (MAE) близько 7 років.

Головне відкриття полягало в аналізі "помилки" ( $\Delta$ -age). Було виявлено, що пацієнти з позитивним "розривом" ( $\Delta$ -age > 0, тобто серце "старше" за паспорт) мали значно вищий ризик смертності від усіх причин (all-cause mortality) та серцево-судинної смертності.

Найбільш вражаючим було те, що цей зв'язок залишався сильним навіть у

підгрупі пацієнтів, чий ЕКГ були інтерпретовані досвідченим лікарем-кардіологом як "повністю нормальні". Це довело, що AI виявляв приховані (латентні) патологічні зміни, невидимі для людського ока, але пов'язані з ризиком.

Ці висновки були незалежно підтверджені та розширені [14] у ще масштабнішому дослідженні, проведеному в Geisinger Health System (США) та опублікованому у провідному кардіологічному журналі *Circulation*.

Дослідження проводилось на базі інтегрованої системи охорони здоров'я Geisinger, що дозволило отримати повні історії хвороб пацієнтів.

Ця робота базувалася на ще більшій когорті: 1,77 мільйона 12-відвідних ЕКГ від понад 440 000 пацієнтів.

Вони також розробили власну 1D-CNN (названу "E-AGE") для прогнозування віку за тим самим принципом "end-to-end".

Робота повністю підтвердила висновки Attia et al. щодо зв'язку високого  $\Delta$ -age та підвищеної смертності.

Дослідження зробило критичний крок уперед: воно довело, що  $\Delta$ -age є потужним предиктором розвитку нових (incident) захворювань у майбутньому.

Пацієнти з найбільшим "розривом" віку ( $\Delta$ -age > 10 років) мали значно вищий ризик розвитку фібриляції передсердь (Atrial Fibrillation) та серцевої недостатності (Heart Failure) протягом наступних років спостереження, порівняно з пацієнтами, чий ЕКГ-вік збігався з хронологічним.

Отже, ці два дослідження на масивних даних довели, що пацієнти з високим  $\Delta$ -age (навіть за "нормальної" ЕКГ) складають групу прихованого ризику. Саме ця здатність DL-моделей виявляти приховані ризики, а не просто прогнозувати вік, і становить їхню головну клінічну цінність та актуальність вашого дослідження.

Саме ця здатність DL-моделей виявляти приховані ризики, а не просто прогнозувати вік, і становить їхню головну клінічну цінність та актуальність вашого дослідження.

Аналіз фахової літератури свідчить, що будь-яку задачу прогнозування в машинному навчанні, включно з "ECG-age", можна сформулювати у двох принципово різних парадигмах: регресії або класифікації. Вибір між ними залежить не від самих даних, а від кінцевої мети дослідження.

Задачі регресії використовуються, коли цільова змінна (те, що ми прогнозуємо) є неперервною числовою величиною. Модель навчається знаходити функціональну залежність, яка дозволяє передбачити точне значення на числовій осі. Класичним загальним прикладом є прогнозування ціни на нерухомість: на основі ознак (площа, кількість кімнат, район) модель видає точну ринкову вартість, наприклад, 121 500 доларів, а не просто "дорога". У контексті "ECG-age" це означає, що модель навчається прогнозувати точний хронологічний вік пацієнта, наприклад, 47.5, 51.2 або 63.0 років. Цей підхід є фундаментальним і, по суті, єдиним можливим для розрахунку клінічно значущого показника "delta-age" ( $\Delta$ -age), оскільки він вимагає віднімання двох неперервних величин. Ефективність регресійних моделей у цьому випадку традиційно оцінюється за тим, наскільки близькі її прогнози до реальних значень, за допомогою метрик "похибки", таких як середня абсолютна похибка (MAE) або коефіцієнт детермінації ( $R^2$ ).

На противагу цьому, задачі класифікації використовуються, коли цільова змінна є дискретною категорією (класом). Модель навчається призначати кожному вхідному об'єкту один із заздалегідь визначених, фіксованих "ярликів". Загальновідомий приклад - фільтрація спаму в електронній пошті: на основі тексту листа модель приймає чітке рішення про належність до однієї з двох категорій: "Спам" або "Не спам". Для застосування цього підходу до "ECG-age", неперервна шкала віку має бути штучно "дискретизована" - розбита на вікові групи (біни), наприклад '18-25', '26-30' і т.д. Модель у цьому випадку навчається прогнозувати належність пацієнта до однієї з цих груп. Такий підхід часто вважається більш стійким (робастним) до "шуму" у фізіологічних даних, оскільки невеликі коливання сигналу, що не змінюють приналежність до групи, ігноруються. З практичної точки зору, класифікація є дуже ефективним інструментом для стратифікації ризиків, дозволяючи швидко віднести пацієнта до відповідної категорії (наприклад, "групи підвищеного ризику 60+"). Для оцінки якості тут використовуються метрики "правильності", такі як Точність (Accuracy), Повнота (Recall), Точність (Precision) та F1-Score.

Таким чином, вибір між регресією та класифікацією є стратегічним методологічним рішенням, яке визначає як архітектуру моделі, так і спосіб

інтерпретації її результатів.

Отже, огляд літератури демонструє чітку еволюцію: від аналізу кількох десятків експертних ознак до використання глибоких 1D-CNN, що аналізують тисячі точок даних сигналу. Сучасні DL-моделі є SOTA (state-of-the-art) за точністю та здатні виявити клінічно значущий "ECG-age gap". Водночас класичні моделі на базі інженерії ознак залишаються цінним "бейзлайном" (baseline) для порівняння та інтерпретації. Це обґрунтовує обрану у дисертації методологію порівняльного аналізу, яка включає як класичні (Логістична регресія, Випадковий ліс), так і глибокі (RNN, CNN) підходи для вирішення задачі класифікації вікових груп.

### **1.3 Аналіз набору даних "Jena University Hospital"**

Вибір репрезентативного набору даних є фундаментальним етапом, що визначає валідність та успіх усього подальшого моделювання. Для даного дисертаційного дослідження було обрано публічний, анонімізований набір даних "Autonomic function regulating blood pressure and cardiac rhythm in aging and disease", зібраний в Університетській лікарні Єни (Jena University Hospital) та доступний на авторитетному репозиторії PhysioNet [15].

Вибір цього датасету обумовлений трьома ключовими перевагами:

1) "здорова" популяція: набір даних складається з 1121 здорового волонтера. Це є критично важливим, оскільки дозволяє моделювати процеси фізіологічного (нормального) старіння, а не патологічного, де зміни в ЕКГ можуть бути викликані супутніми захворюваннями (наприклад, ішемічною хворобою серця);

2) висока якість сигналу: записи ЕКГ надані з високою частотою дискретизації 1000 Гц. Це забезпечує надзвичайно високу часову роздільну здатність (1000 точок на секунду), що дозволяє моделям глибокого навчання (зокрема CNN) виявляти найдрібніші морфологічні зміни в зубцях (P, QRS, T), які були б втрачені при нижчих частотах (наприклад, 250 Гц) [16];

3) тривалість записів: на відміну від стандартних 10-секундних клінічних

ЕКГ, даний датасет містить довготривалі записи (в середньому 19 хвилин). Це надає величезний обсяг даних на одного пацієнта, що дозволяє не тільки надійно виділити чисті, репрезентативні серцеві цикли, але й відкриває можливості для глибокого аналізу варіабельності серцевого ритму (HRV) у часовій та частотній областях.

Для цілей даної роботи з усього набору сигналів використовується одне відведення ЕКГ (Lead II), оскільки воно є найбільш інформативним для аналізу ритму та морфології зубця Р.

Оригінальні дані містили 15 вікових груп. У цьому проєкті, для створення більш стандартизованих та логічних вікових "бінів" (категорій), ці групи були перемашлені у нову 15-класову структуру (з індексами від 0 до 14). Ця структура, яка використовується як цільова змінна (target variable) для всіх експериментів з класифікації, виглядає наступним чином:

- 1) 0: '18-25 years'
- 2) 1: '26-30 years'
- 3) 2: '31-35 years'
- 4) ... (і так далі, з кроком у 5 років) ...
- 5) 13: '86-90 years'
- 6) 14: '91+ years'

Таким чином, задача зводиться до багатокласової класифікації з 15 можливими виходами.

Найважливішою характеристикою цього (та будь-якого іншого реального) набору даних є вкрай нерівномірний розподіл пацієнтів за цими 15 групами. Як видно з результатів розвідувального аналізу даних (зображених на рисунках 3.16 та 3.20), кількість зразків у мажоритарних класах (наприклад, середній вік) у десятки, а то й сотні разів перевищує кількість зразків у міноритарних класах (зокрема, наймолодші '18-25' та найстарші '91+' групи).

Цей демографічний перекис створює фундаментальну методологічну проблему для машинного навчання. "Наївна" модель, навчена на таких даних, швидко оптимізує свою точність (Ассигасу), навчившись завжди прогнозувати мажоритарний клас. Це призведе до створення моделі, яка має високу загальну

точність, але є абсолютно непридатною на практиці, оскільки вона матиме нульову або близьку до нульової чутливість (Recall) до клінічно найцікавіших - найстарших - вікових груп.

Ця проблема сильного дисбалансу класів зумовлює неможливість використання стандартних підходів до навчання та вимагає застосування спеціалізованих методів, які будуть розглянуті далі. Зокрема, для боротьби з цією проблемою в експериментальних задачах будуть застосовані:

- методи на рівні даних - синтетичне передискретизація міноритарних класів (SMOTE - Synthetic Minority Over-sampling Technique) [17];
- методи на рівні функції втрат - модифікація функції втрат для збільшення "уваги" моделі до складних, рідкісних прикладів (Focal Loss) [18].

## **1.4 Формулювання експериментальних задач**

Виходячи з проведеного аналізу предметної області та специфіки набору даних, метою даного дослідження є порівняльний аналіз двох фундаментальних підходів до визначення віку за ЕКГ, ідентифікованих в огляді літератури:

- Класичний підхід: навчання моделей на основі "ручної" інженерії ознак.
- Глибокий підхід: навчання "end-to-end" моделей безпосередньо на "сирих" сегментах ЕКГ-сигналу.

Крім того, ключовою проблемою даних було визначено сильний дисбаланс класів у цільовій 15-класовій задачі.

Для систематичного порівняння цих підходів та вирішення поставлених проблем, було сформульовано п'ять послідовних експериментальних задач.

Перша експериментальна задача полягає у розробці базової моделі (Baseline), що є критично важливим для встановлення початкового рівня якості (benchmark) для складної 15-класової класифікації. Методологічно, ця задача повністю відповідає класичному підходу. На вхід моделі подається набір фізіологічних та

статистичних ознак, який був попередньо згенерований (feature engineering) з "сирих" ЕКГ-сигналів, включаючи показники HRV та морфологічні параметри. В якості класифікатора обрана Логістична регресія через її високий рівень інтерпретованості та відносну простоту. Ключовим викликом тут є екстремальний дисбаланс класів; для його подолання на етапі навчання тренувальна вибірка буде збалансована за допомогою техніки синтетичної генерації зразків SMOTE (Synthetic Minority Over-sampling Technique). Отримані метрики якості, зокрема F1-score [19] та повнота (Recall) для кожного класу, слугуватимуть точкою відліку для порівняння.

Друга задача є логічним розвитком першої і спрямована на дослідження ефективності класичного підходу на укрупненій задачі. Мета цього експерименту - оцінити максимальну ("стельову") якість, яку здатні продемонструвати "ручні" ознаки на спрощеній проблемі, де вплив екстремального дисбалансу нівельовано. Для цього початкові 15 вікових груп агрегуються у 4 більш збалансовані, логічні категорії (наприклад, 'Юність', 'Зрілість', 'Середній вік', 'Похилий вік'). Використовуючи той самий набір згенерованих ознак, що й у Задачі 1, буде застосовано більш потужний ансамблевий класифікатор - Випадковий ліс (Random Forest). Ця модель здатна фіксувати нелінійні залежності та взаємодії між ознаками, що є недоступним для Логістичної регресії [20].

Наступні три експерименти формують єдиний блок порівняльного аналізу "end-to-end" моделей глибокого навчання. Їхньою спільною метою є визначення найбільш ефективної архітектури для вирішення початкової, найскладнішої 15-класової задачі, працюючи безпосередньо з "сирими" сегментами ЕКГ-сигналу. Це дозволить перевірити гіпотезу, що DL-моделі здатні автоматично вилучати більш інформативні ознаки, ніж "ручна" інженерія. Важливим методологічним рішенням є зміна стратегії боротьби з дисбалансом: замість модифікації даних (SMOTE), у цих задачах буде використана модифікація функції втрат - Focal Loss. Ця функція дозволяє моделі під час навчання прицільно фокусуватися на складних, рідкісних прикладах з міноритарних класів (наймолодші та найстарші пацієнти). У рамках цього блоку будуть реалізовані та порівняні наступні архітектури:

- задача 3 використовує гібридну модель Автокодувальник + RNN

(AE+RNN), де автокодувальник слугує екстрактором латентних ознак;

- задача 4 застосовує рекурентну архітектуру двонаправленої LSTM (Bidirectional LSTM) для аналізу часових послідовностей;
- задача 5, відповідно до висновків огляду літератури, реалізує 1D згорткову нейронну мережу (1D-CNN) як де-факто стандарт для ієрархічного виявлення морфологічних ознак в ЕКГ.

Успішне виконання цих п'яти задач дозволить надати обґрунтовану відповідь на головне дослідницьке питання: який методологічний підхід (класичний з інженерією ознак чи "end-to-end" глибоке навчання) та яка конкретна архітектура моделі є найбільш ефективною для вирішення складної, незбалансованої задачі визначення біологічного віку серця за ЕКГ.

## Висновки до розділу 1

У першому розділі було проведено комплексний аналіз предметної області та обґрунтовано високу актуальність задачі визначення біологічного віку серця. Встановлено, що, на відміну від дорогих інвазивних біомаркерів (епігенетика, протеоміка), ЕКГ у поєднанні з методами машинного навчання є перспективним, доступним та неінвазивним інструментом для масового скринінгу. Клінічна значущість такого підходу полягає у виявленні "delta-age" (розриву ЕКГ-віку), який, згідно з проаналізованими дослідженнями (Attia et al., Gajjala et al.), є потужним незалежним предиктором серцево-судинних ризиків та смертності.

Проведений детальний огляд існуючих наукових методів дозволив ідентифікувати два фундаментальні підходи до вирішення задачі:

- 1) класичний (на базі інженерії ознак - HRV та інтервалів);
- 2) глибокий "end-to-end" (на базі 1D-CNN та RNN), що працює з "сирими" даними і демонструє найвищу точність.

Також було проаналізовано дві постановки задачі: регресію (для розрахунку "delta-age") та класифікацію (для стратифікації ризиків).

Виконано детальний аналіз та обґрунтування вибору набору даних "Jena

University Hospital". Його ключовими перевагами визначено високу частоту дискретизації (1000 Гц), наявність довготривалих записів та популяцію "здорових" волонтерів, що дозволяє моделювати саме фізіологічне старіння. На основі цих даних було сформовано цільову 15-класову структуру (від '18-25' до '91+'). Головним практичним викликом цього набору даних визначено сильний дисбаланс класів, що унеможлиблює "наївне" навчання моделей.

На основі аналізу (двох підходів та проблеми дисбалансу) сформульовано п'ять експериментальних задач. Вони спрямовані на системний порівняльний аналіз класичних моделей (логістична регресія, випадковий ліс) із застосуванням методів балансування даних (SMOTE) та глибоких "end-to-end" архітектур (AE+RNN, Bi-LSTM, 1D-CNN) із застосуванням модифікованої функції втрат (Focal Loss) [21].

## 2 ОБҐРУНТУВАННЯ МАТЕМАТИЧНИХ МЕТОДІВ ТА ПРОГРАМНИХ ЗАСОБІВ ДОСЛІДЖЕННЯ

Проведений аналіз предметної області та існуючих наукових підходів до задачі визначення біологічного віку за фізіологічними сигналами дозволив сформулювати чіткі експериментальні задачі. Для їхнього успішного вирішення необхідним є вибір та обґрунтування відповідного математичного апарату, алгоритмів машинного навчання та технологічного стеку. Цей розділ присвячений детальному теоретичному огляду та академічному обґрунтуванню кожного з обраних методів та програмних засобів, що складають основу даного дисертаційного дослідження.

Дослідження структуроване навколо порівняльного аналізу двох фундаментально різних методологічних підходів. Перший підхід є класичним і базується на "ручній" інженерії ознак (feature engineering). Він передбачає глибоку експертну обробку ЕКГ-сигналу, виділення статистичних, часових та частотних характеристик (зокрема, показників варіабельності серцевого ритму, HRV) та подальше їх подання на вхід традиційних моделей машинного навчання. Другий підхід - це "end-to-end" глибоке навчання, де моделі, зокрема згорткові та рекурентні нейронні мережі, навчаються безпосередньо на "сирих" або мінімально оброблених сегментах сигналу, самостійно вивчаючи необхідні ієрархічні представлення даних.

Ключовою методологічною проблемою, виявленою під час аналізу даних, є сильний дисбаланс класів у цільовій 15-класовій структурі. "Наївне" навчання на таких даних призводить до деградації моделі та її нездатності розпізнавати рідкісні (міноритарні) вікові групи. Тому невід'ємною частиною дослідження є теоретичний огляд та застосування спеціалізованих методів боротьби з дисбалансом. Для класичного конвеєра детально розглядається математичний апарат методу передискретизації SMOTE (Synthetic Minority Over-sampling Technique). Для конвеєра глибокого навчання, як більш сучасна альтернатива, обґрунтовується використання модифікованої функції втрат - Focal Loss, яка

динамічно зважає внесок "складних" та "легких" прикладів.

Для реалізації обох дослідницьких конвеєрів, проведення розвідувального аналізу даних (РАД) [22], попередньої обробки сигналів та навчання моделей було обрано мову програмування Python. Цей вибір обумовлений її домінуванням у сфері Data Science та наявністю найпотужнішої наукової екосистеми. Фундаментом для числових операцій є бібліотека NumPy, для роботи з табличними даними - Pandas, а для специфічних завдань цифрової обробки сигналів (фільтрації) - SciPy. Для реалізації класичного конвеєра (Логістичної регресії, Випадкового лісу) обрано "золотий стандарт" - бібліотеку Scikit-learn. Для реалізації "end-to-end" моделей глибокого навчання (Автокодуювальників, RNN, 1D-CNN) обрано фреймворк PyTorch [23]. Його перевага полягає у гнучкості та "Pythonic" підході, зокрема у використанні динамічного обчислювального графу, що значно спрощує процес дослідження та налагодження складних, кастомних архітектур, таких як використані ResNet та Attention блоки.

Для практичного впровадження найкращої розробленої моделі у вигляді програмного прототипу було обрано декаплену (роз'єднану) клієнт-серверну архітектуру. Такий підхід є сучасним стандартом, що дозволяє чітко розділити логіку відображення (клієнт) та "важкі" обчислення (сервер), забезпечуючи незалежну розробку, масштабування та підтримку.

Для реалізації серверної частини (Backend) було обрано платформу Node.js та фреймворк Express.js. Цей вибір є архітектурно виправданим, оскільки задачею сервера є не виконання обчислень, а асинхронна диспетчеризація I/O-залежних операцій (прийом файлів, очікування відповіді). Асинхронна, неблокуюча модель "Event Loop" в Node.js [24] ідеально вирішує цю задачу. Ключовим архітектурним рішенням є те, що сервер Node.js не виконує прогнозування сам, а натомість породжує (spawn) дочірній Python-процес, який завантажує навчену PyTorch модель (.pth файл), виконує обчислення та повертає результат через stdout.

Для розробки клієнтської частини (Frontend) обрано бібліотеку React. Її компонентний підхід дозволяє створювати складні, інтерактивні інтерфейси, розбиваючи їх на малі, незалежні та перевикористовувані модулі. Замість чистого JavaScript, розробка велася на TypeScript - сучасній надмножині JS, що додає

статичну типізацію, яка кардинально підвищує надійність коду та спрощує обробку даних, що надходять з сервера.

## **2.1 Обґрунтування вибору програмних засобів та технологічного стеку**

Вибір програмних засобів є фундаментальним етапом, що визначає методологічні рамки та інструментальні можливості будь-якого наукового дослідження та розробки програмного продукту. Сучасні задачі в галузі машинного навчання, особливо ті, що пов'язані з аналізом складних біомедичних сигналів та впровадженням моделей у практичне використання, вимагають застосування гібридного, багатокомпонентного технологічного стеку. Такий підхід, хоч і ускладнює розробку, є єдиним можливим для одночасного вирішення завдань: по-перше, глибокого наукового дослідження та, по-друге, створення надійного, масштабованого та інтерактивного програмного продукту. Цей підрозділ надає детальне теоретичне обґрунтування та академічний огляд кожної з ключових технологій, обраних для формування дослідницького та продуктового конвеєрів.

Python як наукова *lingua franca*. Для проведення всього дослідницького циклу - від розвідувального аналізу даних (EDA) до навчання та валідації моделей - було обрано мову програмування Python. За останнє десятиліття вона пройшла шлях від скриптової мови до де-факто *lingua franca* у сферах науки про дані, штучного інтелекту та наукових обчислень, витіснивши в цих галузях такі мови, як R, MATLAB та FORTRAN. Її домінування обумовлене не лише синтаксичною простотою та читабельністю, що відповідає філософії "executable pseudocode" (виконуваний псевдокод), але й наявністю найпотужнішої та найзрілішої екосистеми наукових бібліотек. Ця екосистема дозволяє реалізувати весь конвеєр дослідження - від завантаження та очищення даних до навчання та валідації складних архітектур глибокого навчання - в єдиному, узгодженому середовищі.

Фундамент наукових обчислень: SciPy Stack. В основі цієї екосистеми

лежить так званий SciPy Stack - набір фундаментальних бібліотек, що є обов'язковим для будь-якого серйозного аналізу даних. NumPy (Numerical Python) є базовою бібліотекою для високоефективних числових обчислень. Вона надає фундаментальний об'єкт `ndarray` - оптимізований N-вимірний масив. На відміну від стандартних списків Python, які є наборами покажчиків на об'єкти, `ndarray` є суцільним (contiguous) блоком пам'яті, що дозволяє виконувати векторизовані операції. Це, у свою чергу, дозволяє уникнути повільних `for`-циклів Python і перекласти обчислення на оптимізований, низькорівневий код, написаний на C або FORTRAN. Усі подальші наукові бібліотеки, включно з PyTorch та Scikit-learn, базуються на NumPy для своїх внутрішніх операцій з даними.

Pandas [25] є наступним рівнем абстракції, що надає високорівневі, оптимізовані структури даних, зокрема `DataFrame` (таблиця) та `Series` (стовпець). Створена початково для аналізу фінансових часових рядів, ця бібліотека є галузевим стандартом для роботи зі структурованими (табличними) даними. Її потужність полягає у можливостях індексації на основі міток (а не лише цілих чисел), інтуїтивних інструментах для очищення, агрегації, групування та обробки пропущених значень. У контексті даної роботи вона є ключовою для організації та аналізу метаданих пацієнтів.

SciPy (Scientific Python), у свою чергу, розширює NumPy, надаючи широкий набір інструментів для наукових та інженерних обчислень, включаючи модулі для оптимізації, лінійної алгебри та інтегрування. Для даного дослідження її роль є критичною у класичному конвеєрі, оскільки вона надає необхідний інструментарій для цифрової обробки сигналів (ЦОС). Модуль `scipy.signal` надає оптимізовані реалізації цифрових фільтрів, зокрема фільтрів Баттерворта, що використовуються для видалення шумів та дрейфу ізолінії з ЕКГ-сигналу. Що найважливіше, він надає реалізацію фільтрації без фазового зсуву ("zero-phase filtering"), яка є абсолютно необхідною для коректної обробки фізіологічних сигналів. Звичайні ІІР-фільтри вносять фазовий зсув, що спотворює часові співвідношення та морфологію сигналу; метод "вперед-назад" (`filtfilt`) вирішує цю проблему, дозволяючи зберегти точність вимірювання інтервалів ЕКГ.

Спеціалізовані бібліотеки для обробки даних. Для забезпечення

відтворюваності та стандартизації у біомедичних дослідженнях, багато авторитетних баз даних (зокрема, PhysioNet, на якій розміщено датасет "Jena University Hospital") використовують спеціалізований бінарний формат WFDB. Стандартні інструменти не здатні зчитувати цей складний формат. Тому вибір бібліотеки WFDB for Python [26] не є питанням уподобань, а технічною необхідністю. Ця бібліотека є єдиним офіційним, стандартизованим програмним інтерфейсом (API) для роботи з такими даними. Вона надає критично важливі функції для зчитування як метаданих (з файлів .hea), так і самих "сирих" сигналів (з файлів .dat) у NumPy-масиви, що робить можливим їх подальший аналіз у Python-середовищі. Крім того, оскільки ключовою проблемою даних є дисбаланс класів, обрано бібліотеку Imbalanced-learn. Вона є сумісним розширенням Scikit-learn і надає необхідні реалізації алгоритмів передискретизації, зокрема SMOTE.

Фреймворки машинного навчання. Для класичного машинного навчання обрано Scikit-learn (sklearn). Це "золотий стандарт" галузі, цінний завдяки своєму уніфікованому та узгодженому API. Scikit-learn реалізує єдиний дизайн-патерн для всіх моделей: методи `.fit()` для навчання, `.predict()` для прогнозування та `.transform()` для перетворення даних. Це дозволяє швидко та надійно реалізувати експериментальні "бейзлайни", використовуючи надійні реалізації алгоритмів, як-от Логістична регресія та Випадковий ліс.

Для глибокого навчання обрано фреймворк PyTorch. Цей вибір є ключовим для всього дослідження. У порівнянні з альтернативними фреймворками (напр., TensorFlow), PyTorch має низку фундаментальних переваг саме для дослідницької роботи. Головна перевага - "eager execution" (динамічний обчислювальний граф). На відміну від статичного графу (де модель спершу компілюється, а потім виконується), динамічний граф PyTorch виконує операції "на льоту". Це робить фреймворк глибоко інтегрованим у Python ("Pythonic"). На практиці це означає, що процес налагодження (debug) складних, кастомних архітектур (як-от моделі з Residual чи Attention блоками) стає тривіальним - дослідник може використовувати стандартні інструменти Python (напр., `print()`) для перевірки розмірностей тензорів на будь-якому етапі роботи моделі. PyTorch надає гнучкий об'єктно-орієнтований інструментарій, зокрема базовий клас `torch.nn.Module` для побудови будь-яких

нейронних мереж, та `torch.utils.data.Dataset` як абстракцію для створення ефективних конвеєрів завантаження даних. Важливою перевагою є легкість визначення кастомних функцій втрат, що є критичним для реалізації таких методів, як Focal Loss.

Для практичного використання навчених моделей необхідна їх інтеграція у програмний продукт. Сучасним стандартом для цього є декаплена (роз'єднана) клієнт-серверна архітектура. Цей підхід чітко розділяє логіку відображення (Frontend), яка виконується у браузері користувача, та обчислювальну логіку (Backend), що виконується на сервері.

Для серверної частини (Backend) обрано Node.js та фреймворк Express.js. Це обґрунтовано тим, що сервер виконує I/O-залежну (I/O-bound) роботу, виступаючи в ролі "диспетчера". Node.js працює на асинхронній, неблокуючій моделі "Event Loop". Це робить його теоретично та практично найкращим інструментом для обробки тисяч одночасних HTTP-запитів, не блокуючись. На відміну від "важких" CPU-залежних обчислень (якими є робота нейронної мережі), Node.js не виконує їх сам, а делегує їх іншим процесам через Inter-Process Communication (IPC). Тобто, легкий Node.js-сервер приймає запит, породжує дочірній Python-процес для виконання прогнозу, і асинхронно очікує на результат, залишаючись вільним для обробки запитів від інших користувачів.

Для клієнтської частини (Frontend) обрано бібліотеку React [27]. Її фундаментальною парадигмою є компонентний підхід, що дозволяє будувати складні, інтерактивні інтерфейси з малих, ізольованих компонентів. Використання Virtual DOM забезпечує високу продуктивність. Критичним вибором для забезпечення надійності коду є TypeScript. На відміну від чистого JavaScript, TypeScript є мовою зі статичною типізацією. Це дозволяє виявляти цілий клас помилок (напр., неправильну структуру об'єкта) на етапі компіляції, а не під час виконання програми у браузері користувача, що кардинально підвищує надійність додатку.

## 2.2 Математичне підґрунтя методів та алгоритмів дослідження

Для вирішення поставленої задачі визначення біологічного віку за ЕКГ, необхідно застосувати комплекс математичних методів, що охоплюють як класичні підходи до аналізу сигналів та машинного навчання, так і сучасні архітектури глибокого навчання. Цей підрозділ надає теоретичний опис та обґрунтування обраних алгоритмів.

Перший, класичний, підхід базується на парадигмі "інженерії ознак" (feature engineering). Ця парадигма передбачає, що "сирий" ЕКГ-сигнал спершу має бути перетворений на структурований, інформативний вектор ознак за допомогою експертних знань та математичних методів, і лише потім цей вектор подається на вхід відносно простих моделей машинного навчання.

Цей процес починається з цифрової обробки сигналів (ЦОС). Необроблений ЕКГ-сигнал забруднений шумами: високочастотним (електроміографічний шум) та низькочастотним (дрейф ізолінії через дихання). Для їх видалення застосовуються смугові цифрові фільтри, зокрема фільтр Баттерворта, який обирається за його максимально плоску амплітудно-частотну характеристику у смузі пропускання, що не створює "дзвону" (ringing). Фільтр Баттерворта [28] є одним з фундаментальних типів аналогових та цифрових фільтрів, що характеризується максимально плоскою амплітудно-частотною характеристикою (АЧХ) у смузі пропускання. Це означає, що фільтр забезпечує практично однакове підсилення для всіх частотних компонентів у межах своєї робочої смуги, мінімізуючи спотворення форми сигналу, що пропускається. Дана властивість також відома як "відсутність пульсацій" (ripples) у смузі пропускання.

Характерною особливістю фільтра Баттерворта є компроміс між плоскою АЧХ у смузі пропускання та відносно повільним спадом АЧХ у перехідній зоні, що відділяє смугу пропускання від смуги загородження. На відміну від фільтрів Чебишева або еліптичних (фільтрів Кауера), які досягають крутішого спаду за рахунок допущення пульсацій в одній або обох смугах, фільтр Баттерворта забезпечує монотонний спад АЧХ за межами частоти зрізу.

Крутизна цього спаду (roll-off) безпосередньо залежить від порядку фільтра

(N). Зі збільшенням порядку фільтра його АЧХ у перехідній зоні стає крутішою, наближаючись до характеристики ідеального фільтра, однак це призводить до ускладнення схемної реалізації та збільшення фазового зсуву. За будь-якого порядку АЧХ фільтра Баттерворта залишається монотонною як у смузі пропускання, так і в смузі загородження.

Як і інші лінійні фільтри, фільтр Баттерворта може бути реалізований у стандартних конфігураціях: фільтр нижніх частот (ФНЧ), фільтр верхніх частот (ФВЧ), смуговий фільтр та режекторний (загороджувальний) фільтр. Завдяки своїй високій лінійності АЧХ у смузі пропускання, він широко застосовується в аудіотехніці, зокрема в кросоверах акустичних систем, а також у вимірювальних приладах, системах зв'язку та цифровій обробці сигналів, де пріоритетом є збереження форми сигналу без амплітудних спотворень.

Фундаментальною проблемою ІІР-фільтрів є внесення фазового зсуву, що спотворює часові співвідношення у сигналі. Для фізіологічних сигналів, де вимірювання інтервалів (QRS, QT) є критичним, це неприпустимо. Тому застосовується фільтрація без фазового зсуву (zero-phase filtering), що досягається шляхом застосування фільтра до сигналу спочатку у прямому, а потім у зворотному напрямку, в результаті чого фазові спотворення взаємно компенсуються.

Після очищення сигналу, з нього вилучаються ознаки. Ключовою групою ознак є показники варіабельності серцевого ритму (HRV). Вони базуються на аналізі часового ряду RR-інтервалів. У часовій області (Time-Domain) розраховуються статистичні показники, що відображають активність вегетативної нервової системи. Одним з найважливіших є RMSSD (Root Mean Square of Successive Differences), що обчислюється за формулою:

$$RMSSD = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N-1} (x_{i+1} - x_i)^2}$$

Він відображає короткострокову, високочастотну варіабельність і є надійним індикатором парасимпатичної (вагусної) активності. У частотній області (Frequency-Domain) аналізується спектральна густина потужності (PSD) ряду RR-інтервалів. Для її надійної оцінки застосовується метод Велча: сигнал розбивається

на сегменти, що перекриваються, кожен сегмент "віконується" (для зменшення спектрального витоку), піддається Швидкому Перетворенню Фур'є (FFT), і результати усереднюються. Метод Велча (Welch's method) є одним із ключових методів у цифровій обробці сигналів, що використовується для оцінки спектральної густини потужності (PSD) випадкового процесу. Цей метод, запропонований Пітером Велчем у 1967 році, являє собою вдосконалений підхід, що базується на методі Бартлетта, і призначений для зменшення дисперсії (variance) спектральної оцінки.

Основна проблема стандартної періодограми (обчисленої як квадрат модуля перетворення Фур'є від усього сигналу) полягає в її високій дисперсії; оцінка виходить "зашумленою" і не стає стабільнішою зі збільшенням довжини сигналу. Метод Велча вирішує цю проблему шляхом усереднення.

Процедура методу Велча включає три основні кроки. По-перше, вхідний сигнал розділяється на декілька сегментів, які, на відміну від методу Бартлетта, можуть частково перекриватися (зазвичай на 50%). По-друге, до кожного окремого сегмента застосовується вагова функція (вікно), така як вікно Хеннінга, Хеммінга або Блекмена. Це віконне згладжування необхідне для зменшення ефектів спектрального витоку (spectral leakage), що виникають через скінченну довжину сегментів. По-третє, для кожного обробленого вікном сегмента обчислюється своя періодограма зазвичай за допомогою Швидкого перетворення Фур'є (ШПФ), що у свою чергу є фундаментальним алгоритмом для ефективного обчислення Дискретного перетворення Фур'є (ДПФ) та оберненого ДПФ. Важливо зазначити, що ШПФ не є самостійним математичним перетворенням, а радше сімейством обчислювальних методів, що значно оптимізують процедуру ДПФ. Призначення даного алгоритму полягає у перетворенні дискретного сигналу з часової (або просторової) області у частотний спектр, що дозволяє аналізувати його частотні компоненти, їх амплітуди та фази.

Ключова перевага ШПФ [29], що зумовила його революційний вплив на обробку сигналів, полягає у зниженні обчислювальної складності. Прямий розрахунок ДПФ для  $N$  точок даних має квадратичну складність, що вимагає квадратичного порядку арифметичних операцій. Натомість, алгоритми ШПФ,

найвідомішим з яких є алгоритм Кулі-Тьюкі, використовують стратегію "розділяй та володарюй" (divide and conquer) для рекурсивного розбиття задачі. Це дозволяє досягти логарифмічної обчислювальної складності порядку що є експоненційним прискоренням, особливо для великих масивів даних  $N$ .

Методологічно, алгоритм рекурсивно розбиває  $N$ -точкове ДПФ на менші ДПФ (наприклад, на парні та непарні індекси), аж до досягнення тривіальних випадків, з подальшим об'єднанням результатів. Найвища ефективність таких алгоритмів зазвичай досягається, коли  $N$  є степенем двійки, хоча існують варіації для інших розмірів вхідних даних.

Завдяки своїй обчислювальній ефективності, ШПФ стало наріжним каменем сучасної цифрової обробки сигналів. Воно є невід'ємною частиною спектрального аналізу, цифрової фільтрації, систем зв'язку (зокрема, в технологіях OFDM, що лежать в основі стандартів Wi-Fi та 4G/5G), стиснення та обробки зображень, а також широкого спектру наукових та інженерних застосувань, включаючи сейсмологію, радіоастрономію та медичну візуалізацію.

Кінцева оцінка спектральної густини потужності за методом Велча отримується як усереднення всіх цих індивідуальних періодограм. Використання перекриття сегментів дозволяє частково компенсувати втрату інформації на краях сегментів, спричинену застосуванням віконних функцій, що робить оцінку більш надійною.

Результатом застосування методу Велча є значно плавніша та статистично стабільніша (з меншою дисперсією) оцінка спектральної густини потужності. Однак, платою за це покращення є зниження частотної роздільної здатності порівняно з періодограмою, розрахованою для всього сигналу одразу. Розмір сегмента та ступінь перекриття є компромісними параметрами: більші сегменти дають кращу роздільну здатність, але гірше зменшують дисперсію, і навпаки.

Це дозволяє кількісно оцінити потужність у стандартних діапазонах (HF, LF), що корелюють з парасимпатичною та симпатичною активністю відповідно.

Після формування вектора ознак, він подається на класичні моделі машинного навчання. Як базовий метод (baseline) використовується Логістична регресія. Це узагальнена лінійна модель, яка, незважаючи на назву, є методом

класифікації. Вона моделює логіт (log-odds) ймовірності належності до класу як лінійну комбінацію вхідних ознак:

$$\log\left(\frac{P(y = 1)}{1 - P(y = 1)}\right) = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$$

Для багатокласової задачі використовується її узагальнення - Softmax-регресія.

Як більш потужна, нелінійна модель, використовується Випадковий ліс (Random Forest). Це ансамблевий метод, що вирішує проблему високої дисперсії (нестабільності) окремих дерев рішень. Він базується на двох принципах: "беггінгу" (побудова багатьох дерев на випадкових підвбірках даних) та випадковості ознак (при кожному розбитті вузла розглядається лише випадкова підмножина ознак). Фінальне рішення приймається голосуванням більшості дерев, що забезпечує високу узагальнюючу здатність та стійкість до перенавчання.

Для боротьби з дисбалансом класів у цьому конвеєрі застосовується метод SMOTE (Synthetic Minority Over-sampling Technique). Це алгоритм передискретизації, що працює у просторі ознак. Замість простого дублювання (що призводить до перенавчання), SMOTE генерує синтетичні зразки. Для зразка міноритарного класу він знаходить його найближчих сусідів, випадково обирає одного з них та генерує новий зразок шляхом лінійної інтерполяції:

$$x_{new} = x_i + \delta \cdot (x_j - x_i),$$

де  $\delta$  - випадкове число в діапазоні  $[0, 1]$ .

Другий, "end-to-end", підхід відкидає "ручну" інженерію ознак і покладається на здатність глибоких нейронних мереж самостійно вивчати необхідні представлення (features) безпосередньо з "сирого" сигналу.

Основним інструментом для аналізу ЕКГ у цій парадигмі є 1D Згорткові нейронні мережі (1D-CNN). Вони ідеально підходять для часових рядів, оскільки є інваріантними до зсуву (патерн QRS-комплексу буде розпізнано незалежно від його позиції у вікні).

Їхній математичний апарат базується на операції 1D згортки: ядро (фільтр), що є невеликим вектором ваг, ковзає (конволюціонує) вздовж вхідного сигналу:

$$(x \cdot k)[n] = \sum_m^x [m] \cdot k[n - m]$$

Ця операція, що є лінійною, додатково визначається гіперпараметрами, такими як крок (stride), який диктує, на скільки елементів зсувається ядро, та доповнення (padding) - додавання нульових значень на межах сигналу для контролю розмірності вихідної карти ознак.

Однак стандартний архітектурний блок CNN складається не лише зі згортки. Після неї кожен елемент отриманої карти ознак пропускається через нелінійну функцію активації. Найчастіше це ReLU (Rectified Linear Unit) [30], що описується як:

$$f(z) = \max(0, z)$$

Ця порогова функція є обчислювально ефективною та вирішує проблему згасання градієнта, притаманну старим функціям. Опціонально, за цим слідує шар підвибірки (pooling), зазвичай Max Pooling. Його математична модель полягає у ковзному вікні, що вибирає максимальне значення з околу, чим зменшує часову розмірність сигналу та забезпечує локальну інваріантність (наприклад, до мікрозсувів R-зубця).

Ключова перевага CNN (рис. 2.1), що складається з послідовності таких блоків (Convolution - ReLU - Pooling), - це ієрархічне вивчення ознак: перші шари мережі навчаються розпізнавати прості мотиви (нахили, піки), тоді як глибші шари комбінують їх у складні, абстрактні патерни (QRS-комплекси, специфічну морфологію ST-сегмента) [31].

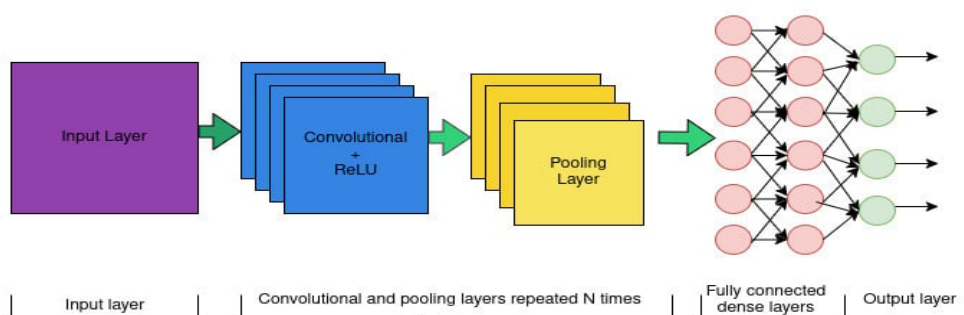


Рисунок 2.1 - Типова архітектура CNN

$$y = f(W \cdot x + b),$$

де  $W$  - матриця ваг.

Для задач класифікації (наприклад, діагностики аритмій) останній шар зазвичай використовує функцію Softmax:

$$P_j = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

яка перетворює вихідні логіти у ймовірнісний розподіл по класах.

Фундаментальним алгоритмом навчання всієї цієї архітектури є зворотне поширення помилки (Backpropagation). Математично, це алгоритм, що базується на ланцюговому правилі диференціювання для обчислення градієнта функції втрат (наприклад, перехресної ентропії) відносно кожної ваги в мережі. Отримані градієнти використовуються в оптимізаційних алгоритмах, таких як стохастичний градієнтний спуск (SGD) або його адаптивні варіанти (наприклад, Adam), для ітеративного оновлення ваг за формулою:

$$\theta_{new} = \theta_{old} - \eta \cdot \nabla L(\theta),$$

де  $\eta$  - швидкість навчання.

Саме в контексті цього алгоритму навчання при побудові дуже глибоких CNN використовується архітектура ResNet (Residual Networks). Вона вводить залишкові з'єднання (skip connections), які дозволяють градієнту "перестрибувати" через шари. Це математично вирішує проблему згасання градієнта при його поширенні назад через десятки шарів і дозволяє тренувати значно глибші та потужніші моделі.

Для моделювання часових залежностей також використовуються рекурентні нейронні мережі (RNN). На відміну від CNN, RNN мають "пам'ять" - їхній прихований стан у момент часу залежить не лише від входу, але й від стану з попереднього кроку. Математична модель простої RNN, відомої як мережа Елмана (рис. 2.2), описує цю рекурентність формулою:

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t + b_h),$$

де  $f$  - нелінійна функція активації (зазвичай тангенс або ReLU),

$W$  та  $b$  - це матриці ваг та вектори зсуву, які є спільними (shared) для всіх часових кроків.

Вихід мережі, у момент обчислюється з прихованого стану:

$$y_t = g(W_{hy}h_t + b_y)$$

Основним алгоритмом навчання для таких архітектур є зворотне поширення помилки в часі (Backpropagation Through Time, BPTT). Цей алгоритм, по суті, "розгортає" рекурентну мережу у глибоку мережу прямого поширення, де кожен шар відповідає одному часовому кроку. Саме через цю значну "глибину в часі" прості RNN страждають від проблеми згасання (або вибуху) градієнта і, як наслідок, мають коротку пам'ять, будучи нездатними захопити довготривалі залежності.

Цю проблему вирішує LSTM (Long Short-Term Memory) - досконала архітектура RNN, що вводить окремий "стан комірки" (cell state), який діє як магістраль для транспортування інформації, та три "вентилі" (gates) - забування, входу та виходу. Ці вентилі, що є керованими нейронними мережами (з сигмоїдною функцією активації  $\sigma$ ), дозволяють комірці вибірково записувати, стирати та зчитувати інформацію.

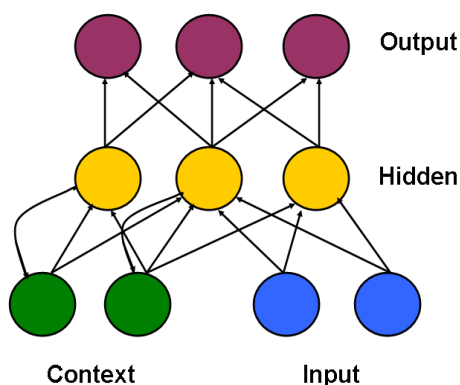


Рисунок 2.2 - Архітектура нейронної мережі Елмана

Математично, вентиль забування вирішує, що стерти зі стану:

$$f_t = \sigma(W_t \cdot [h_{t-1}, x_t] + b_f)$$

Вентиль входу та кандидатний стан визначають, яку нову інформацію

додати:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tan(\sigma(W_C \cdot [h_{t-1}, x_t] + b_C))$$

Новий стан комірки оновлюється через комбінацію старого стану (помноженого на вентиль забування) та нового кандидата (помноженого на вентиль входу):

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

Нарешті, вентиль виходу визначає, яка частина стану комірки стане новим прихованим станом:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \cdot \tan(C_t)$$

Ця складна архітектура дозволяє градієнтам протікати безперешкодно через стан, моделюючи дуже довгі часові залежності у сигналі.

Існує також популярний і дещо спрощений варіант LSTM, відомий як GRU (Gated Recurrent Unit). GRU [33] об'єднує стан комірки та прихований стан і використовує лише два вентиля: вентиль оновлення та вентиль скидання. Математична модель GRU дозволяє вентилю оновлення одночасно контролювати, скільки старої інформації зберегти і скільки нової кандидатської інформації додати:

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \bar{h}_t$$

Розширюючи опис GRU (Gated Recurrent Unit), ця архітектура, хоч і об'єднує стан комірки та прихований стан, математично реалізує це через два складних вентиля, які керують потоком інформації. Вентиль оновлення та вентиль скидання обчислюються на основі входу та попереднього прихованого стану. Їхні математичні моделі:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r),$$

де  $\sigma$  - сигмоїдна функція,

$W$  та  $b$  - треновані ваги та зсуви,

$[h_{t-1}, x_t]$  - позначає конкатенацію векторів.

Вентиль скидання визначає, яку частину попереднього стану слід "забути"

при обчисленні нового кандидатського прихованого стану. Цей кандидатський стан розраховується з урахуванням лише "відфільтрованої" частини попереднього стану:

$$\bar{h}_t = \tan(W_h \cdot [r_t \cdot h_{t-1}, x_t] + b_h)$$

Нарешті, вентиль оновлення виконує роль, яку в LSTM відіграють вентиля забуття та входу: він лінійно інтерполює між старим станом та новим кандидатським станом. Це дозволяє одночасно контролювати, скільки старої інформації зберегти і скільки нової додати, що і описується фінальною формулою:

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \bar{h}_t$$

Завдяки меншій кількості параметрів GRU є обчислювально ефективнішим за LSTM, зберігаючи при цьому здатність моделювати довготривалі залежності.

Як метод неконтрольованого навчання для виділення ознак використовуються автокодувальники (АЕ) [34]. Це мережа з архітектурою "пісочного годинника", що складається з енкодера та декодера. Енкодер стискає вхідний сигнал у латентний вектор малої розмірності, а Декодер намагається відновити оригінал. Мережа навчається мінімізувати помилку реконструкції. Після навчання, енкодер стає потужним екстрактором ознак, здатним перетворити "сирий" сигнал у компактне, інформативне представлення.

Щодо двонаправлених RNN (Bidirectional RNN, BRNN), то це не окремий тип комірки, а архітектурний підхід, який можна застосувати до будь-якого типу рекурентних комірок простих RNN, LSTM або GRU (рис.2.3). Фундаментальна ідея полягає в тому, що для багатьох задач (наприклад, аналізу ЕКГ, де діагноз залежить від контексту навколо аномалії) інформація з "майбутнього" послідовності так само важлива, як і з "минулого". Архітектурно це реалізується шляхом створення двох незалежних рекурентних шарів. Перший шар (прямий, "forward") обробляє послідовність у стандартному порядку, від 1 до T, генеруючи послідовність "прямих" прихованих станів. Другий шар (зворотний, "backward"), який має власний, окремий набір ваг, обробляє ту саму послідовність у зворотному порядку, від T до 1, генеруючи послідовність "зворотних" прихованих станів.

Математично, вихід мережі у будь-який момент часу залежить від обох прихованих станів у цей момент. Найчастіше кінцевий вихідний вектор отримують

шляхом конкатенації (з'єднання) цих двох векторів:

$$y_t = g(W_y \cdot [h_t \rightarrow, \leftarrow h_t] + b_y)$$

Це дозволяє мережі приймати рішення в момент часу, маючи повну інформацію про всю вхідну послідовність, що її оточує, значно підвищуючи точність моделювання.

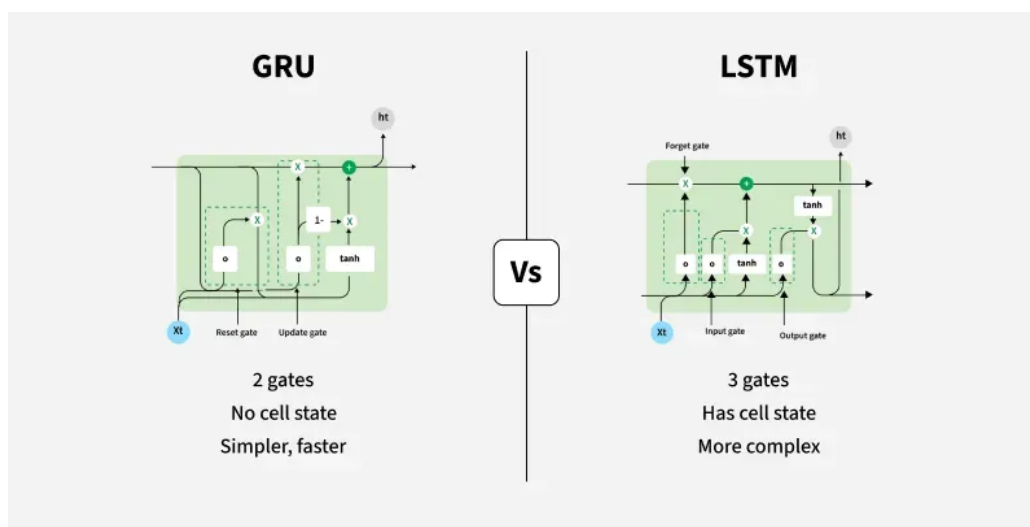


Рисунок 2.3 - Схематичне порівняння архітектури GRU та LSTM RNN

Для підвищення ефективності моделей застосовуються механізми уваги (Attention). Їхня суть полягає у тому, щоб навчити модель не просто усереднювати весь сигнал, а динамічно зважувати важливість різних його частин. Модель вчиться призначати високі "ваги уваги" найбільш інформативним сегментам (напр., Р-хвилі або Т-хвилі) та ігнорувати (призначати низькі ваги) неінформативні ділянки (ізолінію).

Існує декілька ключових реалізацій цього підходу. Історично першими та фундаментальними для рекурентних архітектур стали адитивна увага (Bahdanau Attention) та мультиплікативна увага (Luwong Attention). У моделі Бахданау [35] оцінка відповідності (alignment score), яка визначає важливість кожного елемента послідовності для кожного стану, обчислюється за допомогою невеликої повнозв'язної мережі:

$$e_{ij} = v_a^T \tan(W_a q_i + U_a k_j),$$

де  $q_i$  - це "запит" (наприклад, прихований стан декодера),  
 $k_j$  - "ключ" (прихований стан кожного елемента вхідної послідовності),  
 $U_a, W_a, v_a$  - треновані матриці ваг.

Модель Луонга, натомість, використовує обчислювально ефективнішу мультиплікативну (скалярний добуток) модель:

$$e_{ij} = q_i^T W_a k_j$$

В обох випадках ці оцінки потім нормалізуються за допомогою функції Softmax:

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})}$$

Перетворюючись на ті самі "ваги уваги". Фінальний контекстний вектор, що репрезентує зважену інформацію, обчислюється як:

$$c_i = \sum_{j_{ij}^a} k_j$$

Однак найпотужнішим і найпопулярнішим сучасним алгоритмом є механізм само-уваги (Self-Attention), який лежить в основі архітектури Трансформер (Transformer) (рис.2.4).

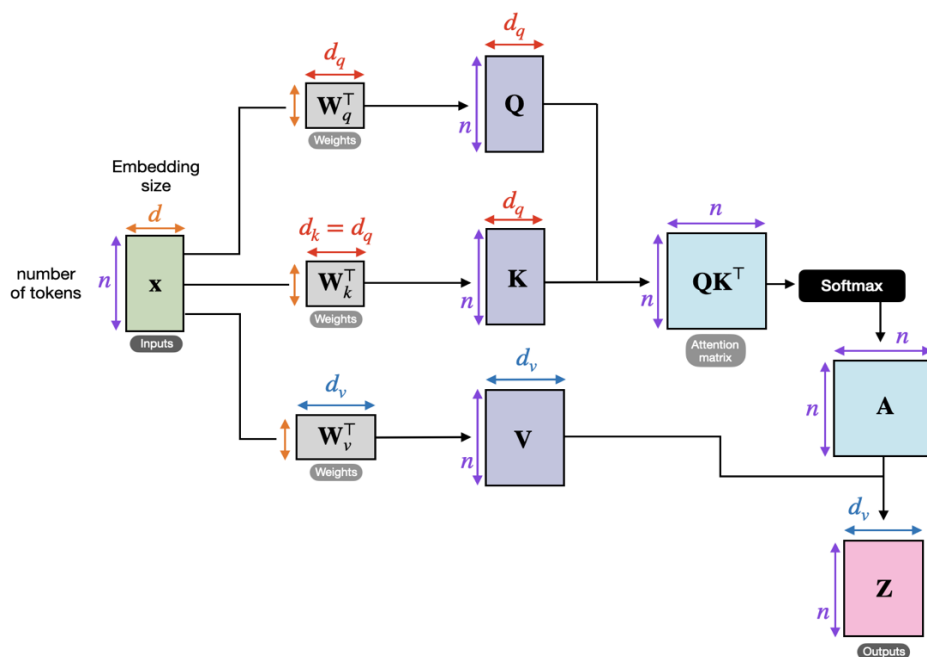


Рисунок 2.4 - Типова архітектура Self-Attention Трансформеру

На відміну від попередніх, він дозволяє моделі зважувати важливість кожного елемента послідовності відносно кожного іншого елемента в тій самій послідовності.

Математично, для кожного елемента обчислюються три вектори: Запит (Query), Ключ (Key) та Значення (Value) шляхом множення на відповідні матриці ваг.

Увага обчислюється за формулою масштабованого скалярного добутку:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V,$$

де  $QK^T$  - обчислює матрицю відповідності "всіх до всіх",

$\sqrt{d_k}$  є стабілізуючим коефіцієнтом (де  $d_k$  - розмірність ключа),

*Softmax* генерує ваги, які потім застосовуються до матриці  $V$ .

Цей підхід часто реалізується у вигляді багатоголової уваги (Multi-Head Attention), де цей процес виконується паралельно в кількох "головах", що дозволяє моделі одночасно фокусуватися на різних аспектах взаємозв'язків у сигналі.

Для боротьби з дисбалансом класів у глибоких мережах, де SMOTE не є ефективним, застосовується Focal Loss. Це модифікація стандартної функції втрат крос-ентропії. Вона вводить модулюючий фактор:

$$(1 - p_t)^\gamma,$$

де  $p_t$  - це ймовірність правильного класу, яку видає модель.

Якщо зразок легкий (мажоритарний, модель в ньому впевнена,  $p_t \rightarrow 1$ ), фактор  $(1 - p_t)^\gamma \rightarrow 0$ , і внесок цього зразка у загальну помилку майже обнуляється. Якщо зразок складний (міноритарний, модель помиляється,  $p_t \rightarrow 0$ ), фактор фактор  $(1 - p_t)^\gamma \rightarrow 1$ , і його внесок залишається високим. Таким чином, функція втрат змушує модель фокусуватися на навчанні складних, рідкісних класів, ігноруючи "баласт" легких мажоритарних зразків.

## 2.3 Обґрунтування архітектури системи

У якості загального шаблону програмного забезпечення було обрано клієнт-

серверну архітектуру. Це дозволяє логічно та фізично розділити логіку користувацького інтерфейсу, серверну обробку бізнес-логіки та ресурсоемні обчислення нейронних мереж.

Однак, з огляду на специфіку задачі, яка поєднує сучасний веб-інтерфейс та складні обчислення на Python, була реалізована багаторівнева (n-tier) архітектура (рис. 2.5), що складається з трьох основних компонентів.

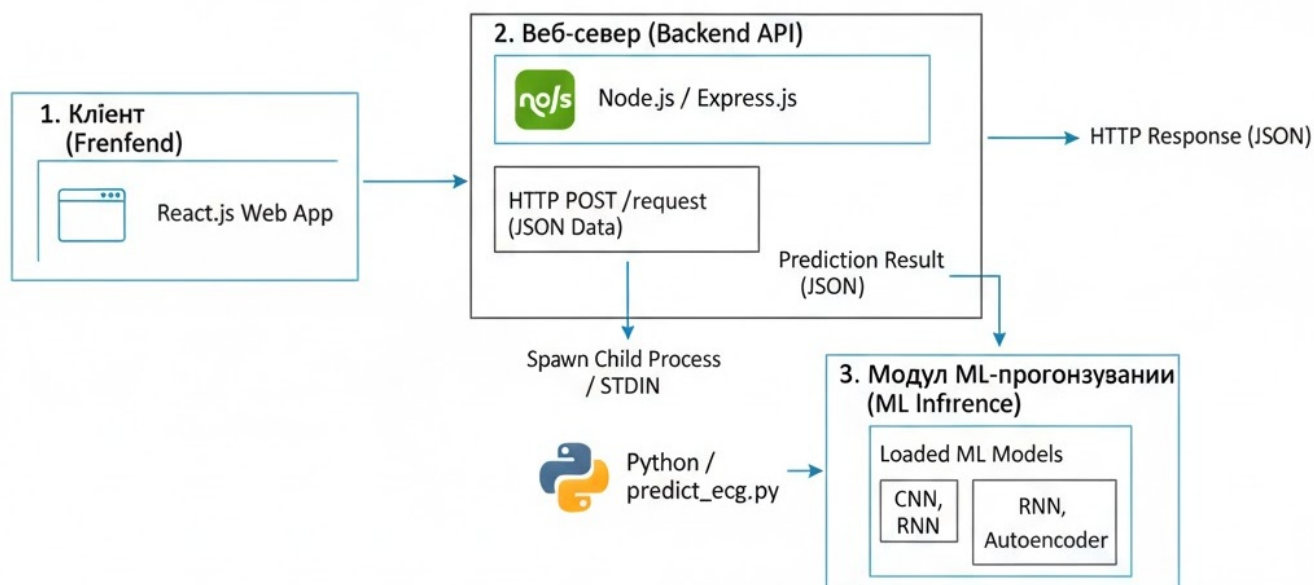


Рисунок 2.5 - Схематичне зображення реалізованої архітектури системи

Перший рівень - клієнт (Frontend). Це веб-додаток, реалізований з використанням бібліотеки React (каталог client/). Він повністю виконується у браузері користувача і відповідає за користувацький інтерфейс (UI), візуалізацію даних ЕКГ, формування запитів на сервер та відображення отриманих результатів діагностики.

Другий рівень - веб-сервер (Backend API). Це сервер-посередник, реалізований на платформі Node.js з використанням фреймворку Express (файл server/server.js). Його основна відповідальність - обробка HTTP-запитів, що надходять від клієнта React.

Третій рівень - модуль висновку (ML Inference). Це обчислювальний модуль, реалізований у вигляді скрипту Python (`server/predict_ecg.py`), що інкапсулює логіку завантаження попередньо навчених моделей та виконання прогнозу.

Важливим аспектом цієї архітектури є передача сирих даних ЕКГ від клієнта до сервера. Коли користувач завантажує або записує сигнал, клієнтський додаток React серіалізує цей часовий ряд (що по суті є масивом числових значень) у формат JSON (JavaScript Object Notation). Потім ці дані відправляються на веб-сервер Node.js за допомогою асинхронного HTTP POST-запиту. Обраний формат JSON є текстовим, легко інтерпретується як JavaScript-об'єктами на стороні Node.js, так і структурами даних Python, що робить його ідеальним для такої гетерогенної системи.

Ключовим архітектурним рішенням є подальша взаємодія другого та третього рівнів. Сервер Node.js слугує "клеєм" (glue) архітектури. Отримавши HTTP-запит з JSON-даними, він не виконує ML-обчислень самостійно, а натомість викликає скрипт `predict_ecg.py` як окремий дочірній процес (child process), передаючи йому серіалізовані дані (найчастіше через стандартний потік введення, `stdin`).

Таке роз'єднання обґрунтоване кількома причинами:

- використання сильних сторін технологій - React є галузевим стандартом для побудови динамічних інтерфейсів; Node.js чудово підходить для створення API-шлюзів та асинхронної обробки I/O; Python є *de facto* стандартом для машинного навчання;
- ізоляція та стабільність - виконання ML-прогнозу є ресурсоемною, процесорною задачею (CPU-bound). Винесення його в окремий Python-процес гарантує, що інтенсивні обчислення не заблокують основний потік Node.js, дозволяючи серверу продовжувати приймати інші запити;
- незалежне обслуговування - моделі машинного навчання (\*.py, \*.h5) можна оновлювати, перенавчати та модифікувати незалежно від кодової бази веб-сервера (Node.js) та клієнта (React).

## Висновки до розділу 2

У даному розділі було представлено комплексне теоретичне та методологічне обґрунтування математичного апарату, алгоритмів машинного навчання та програмно-технологічного стеку, що складають фундамент дисертаційного дослідження. Вибір кожного інструменту та методу диктувався необхідністю вирішення двох ключових, але взаємопов'язаних завдань: по-перше, проведення глибокого порівняльного наукового експерименту та, по-друге, розробки надійного програмного прототипу для практичного застосування.

Обґрунтування програмних засобів продемонструвало, що дослідницька частина роботи базується на екосистемі Python, обраній як *lingua franca* сучасної науки про дані. Фундамент для числових операцій та обробки сигналів закладено синергією бібліотек NumPy, Pandas та SciPy, остання з яких є критичною для коректної нуль-фазової фільтрації ЕКГ. Для реалізації двох експериментальних конвеєрів було обґрунтовано вибір двох провідних фреймворків: Scikit-learn обрано для швидкого та надійного створення "класичних" моделей завдяки його уніфікованому API, тоді як PyTorch обрано для "end-to-end" конвеєра. Перевагою PyTorch визначено його "Pythonic" природу та динамічний обчислювальний граф, що є незамінним для дослідницької роботи, налагодження та реалізації складних, кастомних архітектур на кшталт ResNet та Attention.

Математичне підґрунтя детально декомпонувало обидва дослідницькі підходи. Для класичного конвеєра було математично обґрунтовано весь ланцюг обробки: від цифрової фільтрації (фільтр Баттерворта) та спектрального аналізу (метод Велча на основі ШПФ) до нелінійних ансамблевих моделей (Випадковий ліс). Для "end-to-end" конвеєра було надано математичний опис фундаментальних блоків глибокого навчання, зокрема 1D-CNN (інваріантність до зсуву), рекурентних мереж (LSTM/GRU для моделювання часових залежностей) та механізмів уваги (Self-Attention) для динамічного зважування ознак.

Критичною проблемою, виявленою в даних, є сильний дисбаланс класів. Даний розділ обґрунтував двовекторний підхід до її вирішення: SMOTE для маніпуляцій у просторі ознак класичного конвеєра та Focal Loss як модифікована

функція втрат, що змушує глибокі моделі фокусуватися на "складних" міноритарних зразках.

Нарешті, обґрунтування архітектури системи представило чітку стратегію впровадження результатів дослідження у практику. Обрана багаторівнева (n-tier) архітектура є сучасним стандартом, що забезпечує гнучкість та масштабованість. Ключове архітектурне рішення - повне роз'єднання (decoupling) компонентів:

- React (Frontend), посилений TypeScript, відповідає за надійний та інтерактивний користувацький інтерфейс.
- Node.js (Backend) виступає не як обчислювач, а як високоефективний I/O-диспетчер, що асинхронно обробляє HTTP-запити.
- Python (ML Inference) виконує GPU-залежні обчислення в ізольованому дочірньому процесі, що гарантує стабільність всієї системи.

Таким чином, у розділі було сформовано повний, логічно узгоджений та всебічно обґрунтований інструментарій. Кожен обраний програмний засіб та математичний метод має чітке теоретичне підґрунтя та доведену ефективність у вирішенні специфічних підзадач дослідження - від попередньої обробки біосигналів до розгортання моделі у вигляді веб-сервісу.

## 3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ ДЛЯ КЛАСИФІКАЦІЇ БІОЛОГІЧНОГО ВІКУ

Попередній розділ надав всебічне теоретичне обґрунтування математичних методів, алгоритмів машинного навчання та обраного технологічного стеку. Цей розділ присвячено безпосередньо практичній реалізації та програмній розробці системи для оцінки біологічного віку на основі аналізу електрокардіограм, детально описуючи перехід від теоретичних моделей до функціонального програмного прототипу.

Оскільки, у якості архітектури програмного забезпечення (рис. 3.1) було обрано багаторівневий клієнт-серверний шаблон, фінальна частина розділу присвячена опису розробки програмного прототипу. Цей прототип складається з двох основних частин: клієнтського веб-додатку, реалізованого з використанням бібліотеки React та TypeScript, який реалізує користувацький інтерфейс, та серверної системи на Node.js (Express), яка реалізує API, обробку запитів, виклик Python-скриптів для виконання прогнозу та повернення результату.

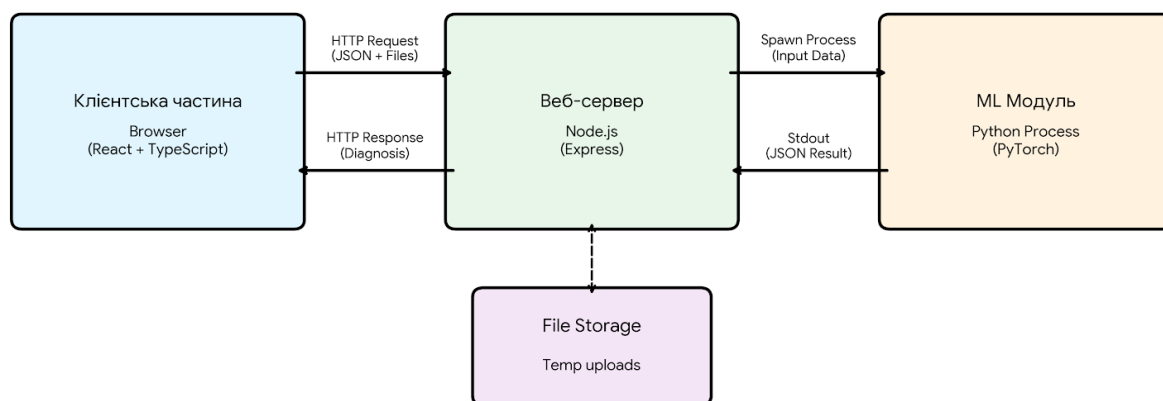


Рисунок 3.1 - Загальна архітектура розробленого програмного продукту

Розробка структурована відповідно до двох методологічних підходів, обґрунтованих раніше: класичного машинного навчання з інженерією ознак та

"end-to-end" глибокого навчання. Послідовно буде описано процес підготовки та обробки вхідного набору даних, включно з методами очищення та фільтрації. Далі, буде детально розглянуто програмну реалізацію класичного конвеєра з використанням бібліотеки Scikit-learn [36] для навчання моделей Логістичної регресії та Випадкового лісу. Після цього буде представлено розробку та тренування "end-to-end" архітектур глибокого навчання, зокрема 1D-CNN, RNN (LSTM) та гібридних моделей, з використанням фреймворку PyTorch.

Розроблений додаток дозволяє користувачу завантажити або записати ЕКГ-сигнал та, натиснувши кнопку, отримати оцінку біологічного віку, розраховану за допомогою найкращої з розроблених моделей. Більш детальні можливості взаємодії з програмним продуктом представлено на діаграмі варіантів використання (рис. 3.2).

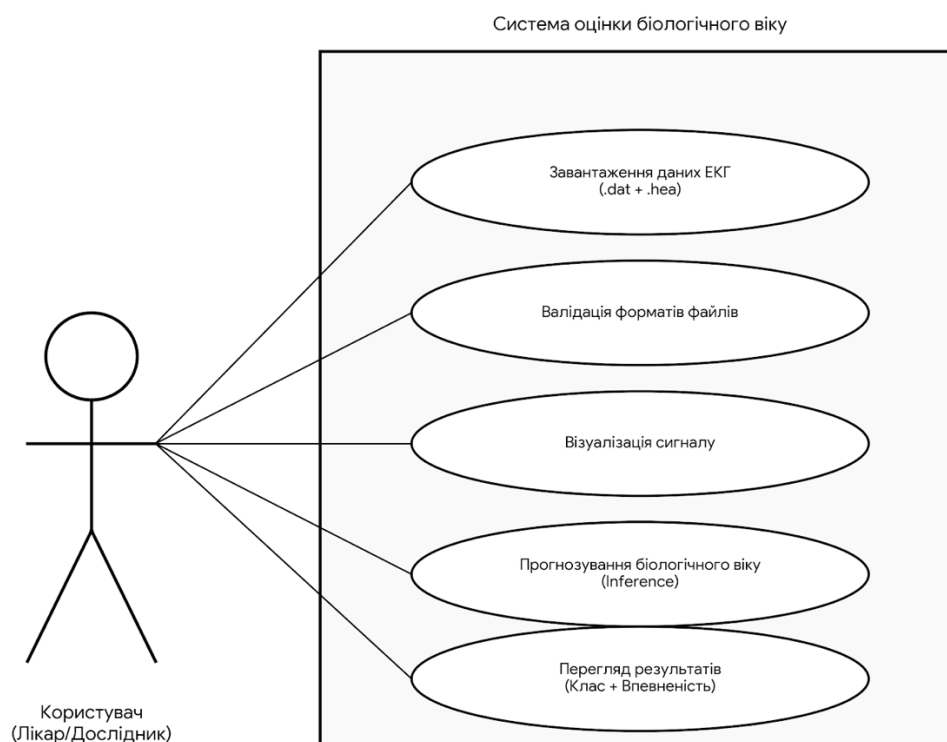


Рисунок 3.2 - Діаграма варіантів використання (Use Case) програмного продукту

Кожен з етапів розробки, від обробки даних до реалізації кінцевих точок API,

буде проілюстрований ключовими фрагментами коду та описом застосованих алгоритмів та програмних рішень.

### 3.1 Обробка вхідного набору даних для навчання моделей

Першим і фундаментальним етапом розробки є підготовка та трансформація вхідних даних. "Сири" біомедичні сигнали з бази даних не можуть бути безпосередньо подані на вхід класичних моделей машинного навчання; вони вимагають ретельної обробки, очищення та процесу інженерії ознак (feature engineering) для перетворення часових рядів у структурований табличний вигляд. Програмна реалізація цього конвеєра була виконана на Python з використанням фундаментальних наукових бібліотек, зокрема wfdb, Pandas, NumPy та SciPy.

Процес починається з інтеграції даних. Скрипт завантажує таблицю (рис. 3.3) з метаданими пацієнтів (subject-info.csv) за допомогою Pandas, що містить демографічну інформацію, зокрема цільову змінну - вікову групу (Age\_group). Паралельно, за допомогою бібліотеки wfdb, відбувається ітеративне зчитування всіх записів \*.hea та \*.dat з основного каталогу даних. Кожен запис, що містить багатоканальний "сирий" сигнал ЕКГ (p\_signal) та частоту дискретизації (fs), зіставляється з відповідним пацієнтом з таблиці метаданих.

Loading: [/Users/dmytro/Diploma/ecg\\_ml\\_analysis/v2/input/autonomic-aging-a-dataset-to-quantify-changes-of](#)  
 Shape: (1121, 6)  
 Data loaded successfully

|   | ID | Age_group | Sex | BMI  | Length | Device |
|---|----|-----------|-----|------|--------|--------|
| 0 | 1  | 2.0       | 1.0 | 23.0 | 15     | 0      |
| 1 | 2  | 2.0       | 1.0 | 24.0 | 32     | 0      |
| 2 | 3  | 7.0       | 1.0 | 28.0 | 30     | 0      |
| 3 | 4  | 4.0       | 0.0 | 24.0 | 15     | 0      |
| 4 | 5  | 2.0       | 1.0 | 20.0 | 15     | 0      |
| 5 | 6  | 4.0       | 1.0 | 21.0 | 15     | 1      |
| 6 | 7  | 2.0       | 1.0 | 30.0 | 20     | 0      |
| 7 | 8  | 3.0       | 1.0 | 25.0 | 20     | 0      |
| 8 | 9  | 3.0       | 1.0 | 22.0 | 15     | 1      |
| 9 | 10 | 2.0       | 0.0 | 26.0 | 10     | 0      |

Рисунок 3.3 - Приклад записів у метаданих subject-info.csv

Наступним кроком є санітизація та попередня обробка сигналу, реалізована у функції `sanitize_signal`. Ця процедура є критичною для видалення артефактів та шумів. По-перше, для боротьби з екстремальними, нефізіологічними викидами (артефактами), сигнал "відсікається" (clipping) до наперед визначеного діапазону (ECG\_CLIP\_RANGE). По-друге, як було теоретично обґрунтовано, застосовується цифрова фільтрація. Програмно це реалізовано за допомогою `scipy.signal` шляхом застосування смугового фільтра Баттерворта 3-го порядку. Це дозволяє одночасно видалити низькочастотний дрейф ізолінії та високочастотний міографічний шум. Принципово важливим є використання функції `filtfilt`, яка реалізує безфазову фільтрацію "вперед-назад", що запобігає спотворенню часових характеристик та морфології QRS-комплексів. Наостанок, для видалення будь-якого залишкового лінійного тренду застосовується функція `detrend`.

Після очищення сигналу починається етап інженерії ознак, який програмно розділений на два потоки: вилучення поканальних ознак (`extract_signal_features`) та вилучення міжканальних ознак (`extract_multi_lead_features`).

Для поканальних ознак, скрипт ітерує по кожному відведенню ЕКГ і розраховує набір дескрипторів (рис.3.4). По-перше, це базові статистичні моменти (`mean`, `std`, `min`, `max`, `median`) та ознаки форми сигналу (`skew`, `kurtosis`). По-друге, що є ключовим, вилучаються ознаки варіабельності серцевого ритму (HRV). Для цього спершу за допомогою алгоритму `gqrs_detect` з `wfdb.processing` детектуються R-піки. На основі отриманого ряду RR-інтервалів, який проходить фізіологічну фільтрацію (видалення значень  $< 0.3\text{c}$  та  $> 2.0\text{c}$ ), розраховуються часові (Time-Domain) показники (`rmssd`, `nn50`, `pnn50`) та частотні (Frequency-Domain) показники. Для останніх, як було обґрунтовано раніше, програмно реалізовано метод Велча (`scipy.signal.welch`) для отримання спектральної густини потужності (PSD) RR-інтервалів, з подальшим інтегруванням потужності у стандартних діапазонах (VLF, LF, HF) та розрахунком їх співвідношення LF/HF.

```

def extract_signal_features(signal, fs):
    features = {}
    signal = sanitize_signal(signal, fs=fs, bandpass=True)

    # Time-domain features
    features['mean'] = np.mean(signal)
    features['std'] = np.std(signal)
    features['min'] = np.min(signal)
    features['max'] = np.max(signal)
    features['median'] = np.median(signal)
    features['skew'] = safe_skew(signal)
    features['kurtosis'] = safe_kurtosis(signal)

    # --- R-peaks and HRV using WFDB GQRS ---
    try:
        peaks = wp.gQRS_detect(sig=signal, fs=fs)
        if len(peaks) < 3: # Need at least 3 peaks for meaningful HRV
            raise ValueError("Insufficient R-peaks detected")

        rr_intervals = np.diff(peaks) / fs
        # Filter out physiologically impossible RR intervals
        rr_intervals = rr_intervals[(rr_intervals > 0.3) & (rr_intervals < 2.0)]

        if len(rr_intervals) < 2:
            raise ValueError("Insufficient valid RR intervals")

    except Exception as e:
        print(f"R-peak detection failed: {e}")
        rr_intervals = np.array([])

    if len(rr_intervals) > 1:
        features['hr_mean'] = 60 / np.mean(rr_intervals)
        features['hr_std'] = 60 * np.std(rr_intervals)
        features['rmssd'] = np.sqrt(np.mean(np.diff(rr_intervals)**2))
        features['nn50'] = np.sum(np.abs(np.diff(rr_intervals)) > 0.05)
        features['pnn50'] = features['nn50'] / len(rr_intervals)

```

Рисунок 3.4 - Алгоритм парсингу поканальних ознак ЕКГ сигналу

Для міжканальних ознак аналізується взаємозв'язок між відведеннями. За допомогою NumPy розраховуються матриці кореляції (`np.corrcoef`) та коваріації (`np.cov`), з яких вилучаються унікальні попарні значення між усіма каналами (рис. 3.5). Окрім того, як метод зниження розмірності та виділення головних напрямків варіації, застосовується Метод Головних Компонент (PCA) [37] з Scikit-learn, з якого в якості ознак беруться перші три показники поясненої дисперсії (`explained_variance_ratio_`).

```
def extract_multi_lead_features(signals):
    n_channels = signals.shape[1]
    features = {}
    signals_clean = np.clip(np.nan_to_num(signals, nan=0.0), ECG_CLIP_RANGE[0], ECG_CLIP_RANGE[1])

    # Correlation and covariance
    try:
        with np.errstate(divide='ignore', invalid='ignore'):
            corr_matrix = np.corrcoef(signals_clean.T)
            cov_matrix = np.cov(signals_clean.T)
            upper_idx = np.triu_indices(n_channels, k=1)
            for r, c in zip(*upper_idx):
                if not (np.isnan(corr_matrix[r, c]) or np.isnan(cov_matrix[r, c])):
                    features[f'corr_ch{r}_ch{c}'] = corr_matrix[r, c]
                    features[f'cov_ch{r}_ch{c}'] = cov_matrix[r, c]
                else:
                    features[f'corr_ch{r}_ch{c}'] = 0.0
                    features[f'cov_ch{r}_ch{c}'] = 0.0
    except Exception:
        print("Error computing correlation/covariance, using zeros")
        for r in range(n_channels):
            for c in range(r+1, n_channels):
                features[f'corr_ch{r}_ch{c}'] = 0.0
                features[f'cov_ch{r}_ch{c}'] = 0.0
```

Рисунок 3.5 - Алгоритм парсингу міжканальних ознак

На фінальному етапі, всі поканальні ознаки усереднюються по всіх відведеннях для отримання єдиного вектора ознак для запису. Цей агрегований вектор об'єднується з міжканальними ознаками та метаданими (вік, стать). У результаті формується єдиний, структурований DataFrame, який зберігається у файл `ecg_features.csv` (рис. 3.6).

|   | mean          | std      | min       | max       | median    | skew        | kurtosis     | hr_mean   | hr_s    |
|---|---------------|----------|-----------|-----------|-----------|-------------|--------------|-----------|---------|
| 0 | 5.000000e+00  | 0.103834 | 4.736477  | 5.667927  | 4.978795  | 3.534223    | 16.788716    | 39.433108 | 1.27781 |
| 1 | 1.156490e-17  | 0.143917 | -0.620451 | 1.043088  | -0.046955 | -116.966317 | 30872.227666 | 23.574785 | 2.81191 |
| 2 | 3.333333e+00  | 0.090217 | 2.985992  | 4.039514  | 3.310691  | 2.828664    | 19.366225    | 29.191409 | 4.20338 |
| 3 | -5.000000e+00 | 0.044118 | -5.131624 | -4.643481 | -5.010116 | 4.892410    | 28.992977    | 30.258378 | 2.76351 |
| 4 | -5.000000e+00 | 0.092562 | -5.752685 | -4.260004 | -5.030029 | 4.252412    | 21.278652    | 32.180474 | 2.49024 |
| 5 | 3.333333e+00  | 0.116059 | 2.931765  | 4.098617  | 3.301733  | 2.871457    | 13.663969    | 46.186859 | 1.69761 |
| 6 | 3.333333e+00  | 0.485164 | -3.981842 | 5.904980  | 3.333901  | -0.737600   | 29.248471    | 81.059433 | 5.03203 |
| 7 | 5.000000e+00  | 0.240395 | 4.817800  | 6.374306  | 4.929390  | 3.996398    | 15.992554    | 47.781628 | 0.83581 |
| 8 | 3.333333e+00  | 0.128013 | 2.849605  | 4.076728  | 3.289854  | 1.709589    | 9.077898     | 43.856281 | 2.78031 |
| 9 | 3.333333e+00  | 0.083633 | 2.891642  | 3.926245  | 3.310332  | 3.310508    | 14.396552    | 39.719118 | 2.78161 |

Рисунок 3.6 - Приклад записів збережених ознак для кожної ЕКГ

Цей табличний набір даних, що містить 46 ознак, стає безпосереднім вхідним матеріалом для навчання класичних моделей машинного навчання, що буде

описано у наступному підрозділі.

Другий конвеєр, реалізований у `wfdb_parser.py`, кардинально відрізняється. Він не створює статичного файлу ознак, а натомість реалізує динамічний конвеєр завантаження даних (data pipeline), що є ідіоматичним для фреймворку PyTorch. Його мета - подавати на вхід нейронних мереж (CNN/RNN) "сирі" сегменти сигналу, дозволяючи моделі самостійно вивчати ознаки.

Програмно це реалізовано у вигляді класу `WFDBECGDataset` (рис. 3.7), який успадковує базовий клас `torch.utils.data.Dataset`. При ініціалізації (`__init__`) клас сканує файли `*.hea`, завантажує метадані пацієнтів та готує `LabelEncoder` з `Scikit-learn` для перетворення категоріальних міток вікових груп (цільова змінна) у цілочисельні індекси, необхідні для функції втрат.

```
class WFDBECGDataset(Dataset):
    def __init__(self, data_path, subject_info_csv, window_size_sec=10,
                 window_step_sec=5, augment=False, preload=False):
        self.data_path = data_path
        self.window_size_sec = window_size_sec
        self.window_step_sec = window_step_sec
        self.augment = augment
        self.preload = preload

        # Load subject information
        self.subject_info = pd.read_csv(subject_info_csv)
        self.subject_info = self.subject_info.set_index('ID')

        # Find all .hea files
        self.hea_files = glob.glob(os.path.join(data_path, "*.hea"))
        self.hea_files.sort()

        # Process files and build dataset
        self.samples = []
        self.classes_set = set()
        self.max_channels = 0

        self._build_dataset()

        # Setup label encoder
        self.classes_ = sorted([c for c in self.classes_set if not np.isnan(c)])
        self.le = LabelEncoder()
        self.le.fit(self.classes_)

        # Preload data if requested
        if self.preload:
            self._preload_data()
```

Рисунок 3.7 - Конструктор `WFDBECGDataset`

Ключова логіка полягає у сегментації (віконному аналізі), що відбувається у методі `_process_record`. Замість агрегації, довгі сигнали нарізаються на короткі, перекривані вікна (напр., 10 секунд з кроком 5 секунд), що задаються параметрами `window_size_sec` та `window_step_sec`. Це не лише готує дані до формату, придатного

для CNN/RNN, але й слугує ефективним методом аугментації даних, значно збільшуючи кількість тренувальних зразків.

Кожен такий сегмент (вікно) проходить попередню обробку "на льоту" у методі `_preprocess_window`. Ця обробка ідентична першому конвеєру: `np.nan_to_num`, `np.clip` (до -10.0, 10.0), та безфазова фільтрація Баттерворта (0.5-40 Гц) за допомогою `filtfilt`. Однак, критично важливим доповненням для глибокого навчання є Z-score нормалізація кожного каналу  $((\text{signal} - \text{np.mean}(\text{signal})) / \text{std})$ , яка центрує дані та приводить їх до єдиного масштабу, що є необхідною умовою для стабільного навчання нейронних мереж.

Метод `__getitem__`, який викликається PyTorch під час тренування, відповідає за завантаження та фінальну підготовку одного зразка. Він або дістає попередньо завантажені дані (якщо `preload=True`), або зчитує їх з диска, застосовує описану обробку, а також, опціонально, додаткову аугментацію (`_augment_signal`), таку як додавання шуму, випадкове масштабування чи часові зсуви (рис. 3.8).

```
def _augment_signal(self, signal):
    """Apply data augmentation to ECG signal."""
    if not self.augment:
        return signal

    # Random noise
    if np.random.random() < 0.3:
        noise = np.random.normal(0, 0.01, signal.shape)
        signal = signal + noise

    # Random scaling
    if np.random.random() < 0.3:
        scale = np.random.uniform(0.8, 1.2)
        signal = signal * scale

    # Random time shift (circular shift)
    if np.random.random() < 0.3:
        shift = np.random.randint(-len(signal)//10, len(signal)//10)
        signal = np.roll(signal, shift, axis=0)

    return signal
```

Рисунок 3.8 - Алгоритм аугментації даних

Наостанок, він вирівнює кількість каналів (шляхом доповнення нулями) та

транспонує тензор до формату (кількість каналів, довжина сигналу), який є стандартним для 1D-згорток у PyTorch. Таким чином, цей конвеєр забезпечує моделі глибокого навчання потоком готових до обробки, нормалізованих та аугментованих сегментів "сирого" сигналу.

### 3.2 Машинне навчання (Logistic Regression та Random Forest)

Цей підрозділ описує програмну реалізацію класичного конвеєра машинного навчання, який застосовується до структурованого набору даних `ecg_features.csv`, що був згенерований на етапі інженерії ознак. Реалізація спирається на "золотий стандарт" класичного ML - бібліотеку Scikit-learn - та інструменти для роботи з даними з екосистеми Pandas та NumPy.

Програмна реалізація для обох моделей (Логістична регресія та Випадковий ліс) має спільну базову підготовку даних. Процес починається із завантаження `ecg_features.csv` у Pandas DataFrame. Негайно після цього проводиться очищення: видаляються службові та метадані, що не несуть прогностичної цінності (наприклад, `record`, `gender`, `device`), а також будь-які записи з пропущеними значеннями (`dropna()`) для забезпечення цілісності даних.

Ключовим кроком є розділення даних на матрицю ознак `X` (що містить 46 числових дескрипторів) та цільовий вектор `y` (`age_group`). Оскільки цільова змінна є категоріальною (15 вікових груп), для її коректної обробки моделями застосовується LabelEncoder з Scikit-learn, який перетворює текстові/числові мітки класів у цілочисельні індекси (від 0 до 14).

Після цього набір даних розділяється на тренувальний та тестовий (`train_test_split`) у стандартній пропорції, що є фундаментальним для валідації моделі та уникнення "погляду в майбутнє".

Критичною проблемою, що вирішується програмно, є сильний дисбаланс класів, виявлений у даних. Як було теоретично обґрунтовано, для її вирішення застосовується метод SMOTE (Synthetic Minority Over-sampling Technique) з бібліотеки `imbalanced-learn`. Для коректного застосування та уникнення витоку

даних (data leakage), SMOTE не застосовується до всього набору даних одразу. Натомість, він програмно інтегрується як перший крок у конвеєр Scikit-learn (Pipeline). Такий підхід гарантує, що операція передискретизації (генерація синтетичних зразків) виконується лише на тренувальних даних під час виклику методу `.fit()`, тоді як тестовий набір даних (при оцінці) залишається незмінним.

Реалізація Логістичної регресії (нотатка `04_linear_training.ipynb`) слугує базовою (baseline) лінійною моделлю. Її конвеєр (Pipeline) складається з трьох послідовних кроків:

- SMOTE: описана вище обробка дисбалансу.
- StandardScaler: Z-score нормалізація ознак. Цей крок є обов'язковим для лінійних моделей, що використовують регуляризацию (L1/L2), оскільки він приводить всі ознаки до єдиного масштабу, забезпечуючи коректну та стабільну збіжність алгоритму оптимізації.
- LogisticRegression: власне класифікатор. Програмно він налаштований на багатокласову задачу (`multi_class='ovr'` або `'multinomial'`) з використанням стандартного розв'язувача (напр., `solver='lbfgs'`).

Реалізація Випадкового лісу (нотатка `05_random_forest_training.ipynb`) представляє нелінійний ансамблевий підхід. Його конвеєр (Pipeline) складається з двох кроків:

- SMOTE: обробка дисбалансу, аналогічно до лінійної моделі.
- RandomForestClassifier: ансамблевий класифікатор. Важливо, що в цьому конвеєрі відсутній крок StandardScaler, оскільки архітектури, засновані на деревах рішень, є нечутливими до масштабу ознак, що робить цей крок обчислювально непотрібним.

Модель ініціалізується з набором гіперпараметрів, зокрема `n_estimators`, що визначає кількість дерев в ансамблі.

Обидві навчені моделі (конвеєри) оцінюються на тестовому наборі. Програмно це реалізовано шляхом виклику функцій `accuracy_score`, `classification_report` (що надає деталізовані метрики `precision`, `recall`, `f1-score` для кожного класу) та `confusion_matrix` (для візуального аналізу помилок) з `sklearn.metrics`.

Аналіз результатів логістичної регресії (рис. 3.9) показує низьку ефективність лінійного підходу. Незважаючи на застосування SMOTE, модель демонструє загальну точність (accuracy) на рівні ~18%. Матриця помилок [38] підтверджує, що модель здатна розпізнавати лише домінантні, молодші вікові групи, і практично повністю ігнорує міноритарні (старші) класи, що свідчить про її нездатність знайти складні нелінійні залежності у даних.

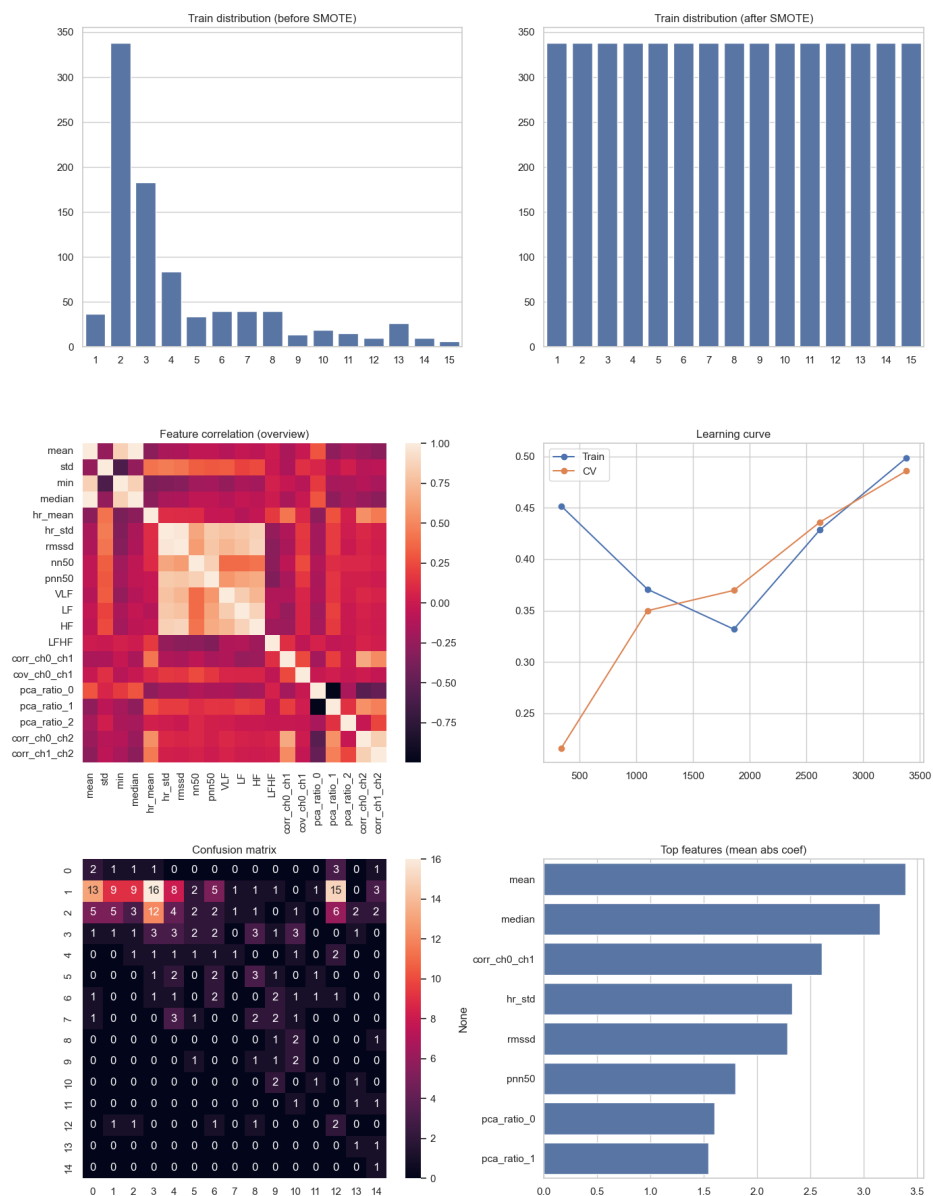


Рисунок 3.9 - Результат навчання Логістичної регресії

Натомість, випадковий ліс (рис. 3.10) демонструє значно вищу продуктивність, досягаючи загальної точності ~49%. Звіт про класифікацію та

матриця помилок показують, що ця нелінійна ансамблева модель здатна успішно ідентифікувати представників усіх вікових груп, хоча і зберігається певна плутанина між сусідніми класами, що є очікуваним для даної задачі. Аналіз важливості ознак (feature importance) також підтвердив, що найбільший внесок у прогнозування роблять показники HRV (rmssd, pnn50, hf) та статистичні характеристики сигналу (std).

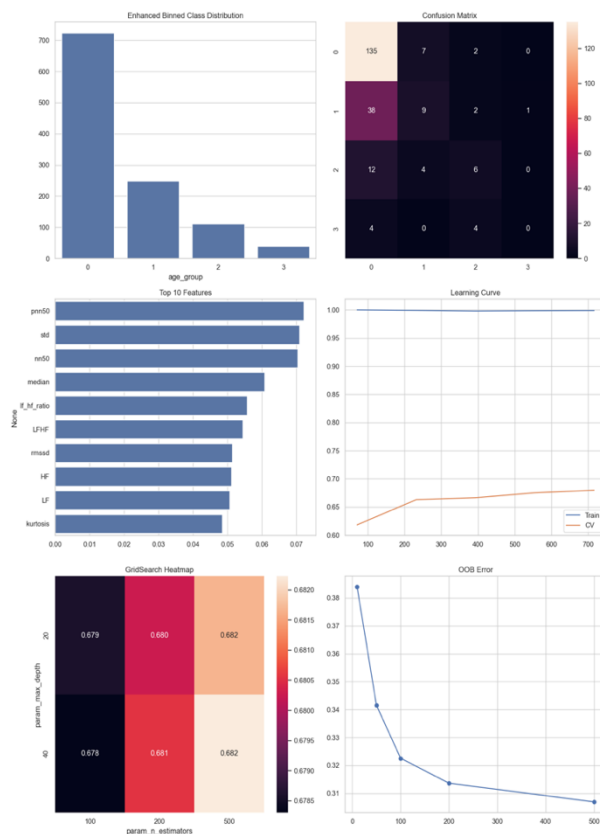


Рисунок 3.10 - Результати навчання Випадкового лісу

Після завершення експериментів, весь навчений об'єкт Pipeline (що інкапсулює кроки SMOTE, нормалізації та сам класифікатор) для кожної з моделей серіалізується та зберігається у файл (чекпоінт) за допомогою бібліотеки joblib або pickle. Це дозволяє в майбутньому миттєво завантажити готову до використання модель у серверній частині додатку для виконання прогнозів.

### 3.3 Тренування нейронних мереж (1D CNN, RNN, RNN + Autoencoder)

У рамках дисертаційного дослідження було реалізовано та експериментально перевірено "end-to-end" підхід, де моделі глибокого навчання самостійно вивчають ієрархічні ознаки з "сирого" ЕКГ-сигналу. Програмна реалізація виконувалася з використанням фреймворку PyTorch, що дозволило гнучко конструювати складні архітектури та кастомні цикли навчання. Для забезпечення відтворюваності експериментів усі моделі використовували єдиний конвеєр подачі даних WFDBECGDataset, описаний у попередніх підрозділах, який забезпечував сегментацію, нормалізацію та аугментацію сигналів у реальному часі.

Першим етапом у серії експериментів з глибоким навчанням стала розробка та дослідження моделі неконтрольованого навчання - одновимірного згорткового автоенкодера (1D Convolutional Autoencoder). Основною метою цього експерименту була не безпосередня класифікація віку, а перевірка гіпотези щодо можливості автоматичної екстракції інформативних, спресованих ознак (feature extraction) з "сирого" ЕКГ-сигналу без використання розмічених даних. Передбачалося, що модель навчиться кодувати високорозмірний вхідний сигнал у компактний латентний вектор, зберігаючи при цьому найбільш суттєву морфологічну інформацію та відкидаючи шуми.

Програмна реалізація архітектури, представлена у файлі 01\_autoencoder\_training.ipynb, базується на симетричній структурі типу "пісочний годинник", що складається з двох функціональних блоків: кодера (Encoder) та декодера (Decoder). Кодер спроектовано як послідовність згорткових блоків, кожен з яких виконує операцію одновимірної згортки (`nn.Conv1d`) для виявлення локальних патернів сигналу, після чого застосовується функція активації ReLU. Критичним елементом архітектури є використання шарів максимальної підвибірки (`nn.MaxPool1d`) після кожної згортки. Це дозволяє прогресивно зменшувати часову розмірність сигналу, одночасно збільшуючи кількість каналів ознак ("глибину" тензора). У найвужчому місці мережі формується так зване "пляшкове горлечко"

(bottleneck), де вхідний 10-секундний сигнал представляється у вигляді стисненого латентного вектора фіксованої розмірності (64 ознаки). Декодер має дзеркальну архітектуру відносно кодера, використовуючи шари транспонованої згортки (`nn.ConvTranspose1d`) для виконання операції підвищення дискретизації (`upsampling`) та поступового відновлення початкової часової розмірності сигналу.

Процес навчання моделі здійснювався в неконтрольованому режимі, де цільовим значенням виступав сам вхідний сигнал. Функцією втрат було обрано середньоквадратичну помилку (`nn.MSELoss`), мінімізація якої змушує мережу генерувати на виході сигнал, максимально наближений до вхідного. Аналіз динаміки навчання, наведений на графіках (рис. 3.11), демонструє стабільну збіжність алгоритму. Як видно з кривих втрат протягом 50 епох, і тренувальна, і валідаційна помилки стрімко знижуються на початку навчання і стабілізуються на низькому рівні ( $MSE < 0.05$ ). Синхронність цих кривих та відсутність їх розходження свідчить про відсутність перенавчання та здатність моделі до узагальнення морфології сигналів на нових даних.

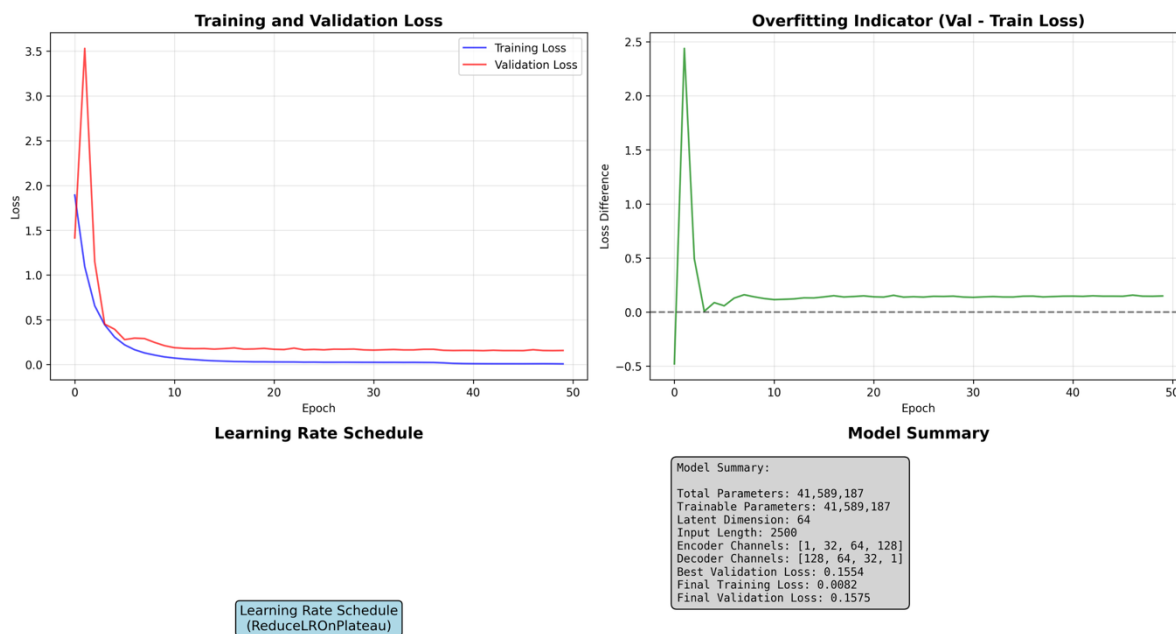


Рисунок 3.11 - Аналіз тренування Autoencoder

Якісна оцінка роботи моделі, проілюстрована на графіках (рис. 3.12), підтверджує, що автоенкодер успішно навчився відновлювати основні компоненти ЕКГ-циклу. При накладанні оригінального сигналу (синя лінія) та його реконструкції (помаранчева лінія) видно чіткий збіг у положенні та формі QRS-комплексів та зубців Т. Важливим побічним ефектом, що спостерігається на реконструкціях, є ефект згладжування: відновлений сигнал виглядає "чистішим" за оригінал. Це свідчить про те, що модель, відновлюючи сигнал з латентного коду, ефективно відфільтровує високочастотний шум та незначні артефакти, діючи як інтелектуальний нелінійний фільтр.

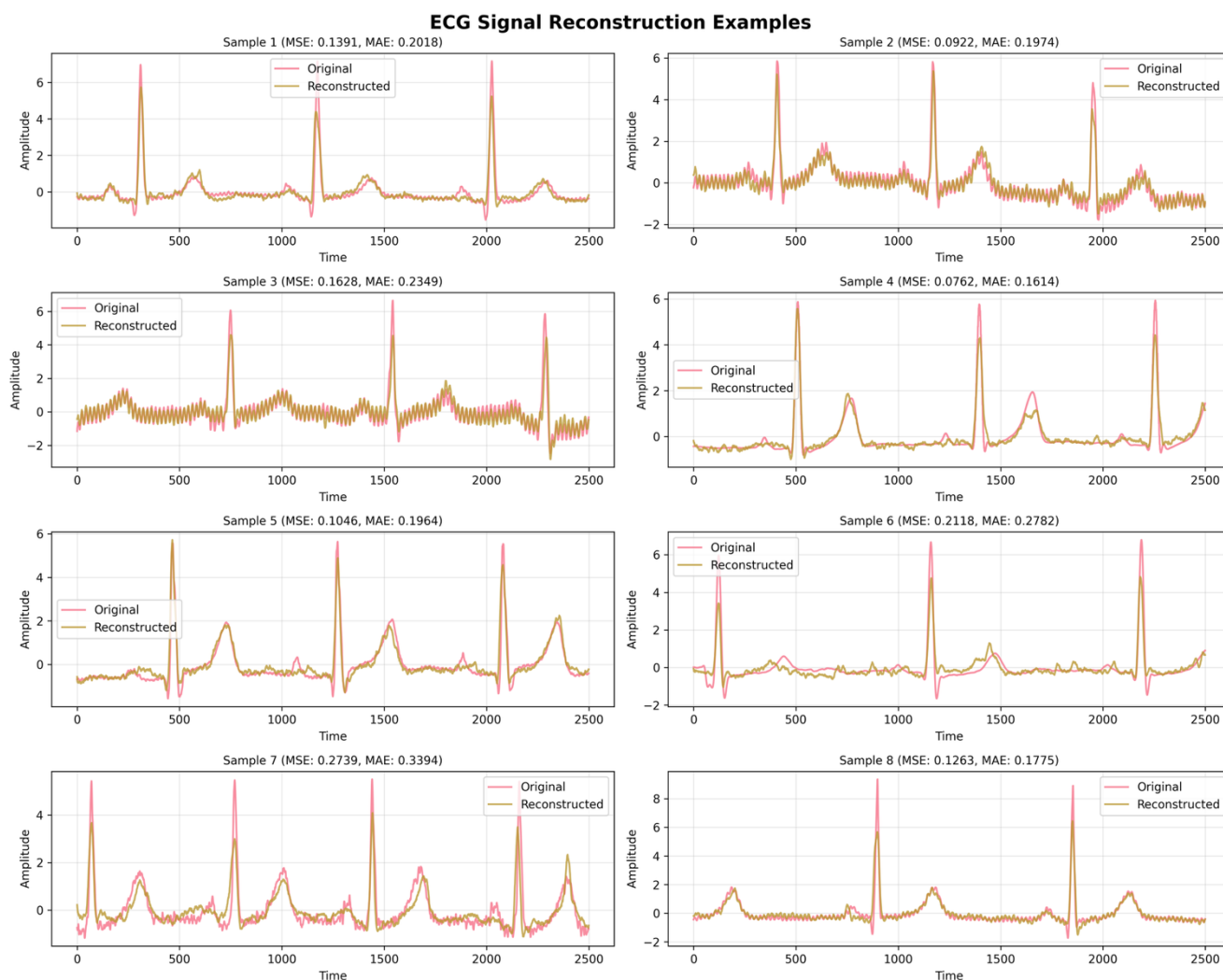


Рисунок 3.12 - Графіки реконструкції ознак Autoencoder

Для оцінки корисності виділених латентних ознак для задачі класифікації

було проведено аналіз латентного простору за допомогою алгоритму t-SNE, а також навчання класичних класифікаторів (Random Forest) на отриманих стиснених векторах. Графік Latent Space [39] (рис. 3.13), де точки забарвлені відповідно до вікових груп, показує наявність певної загальної структури, проте чіткого кластерного розділення за віком не спостерігається - точки різних класів значно перемішані. Це пояснюється відсутністю інформації про класи під час навчання автоенкодера (unsupervised learning).

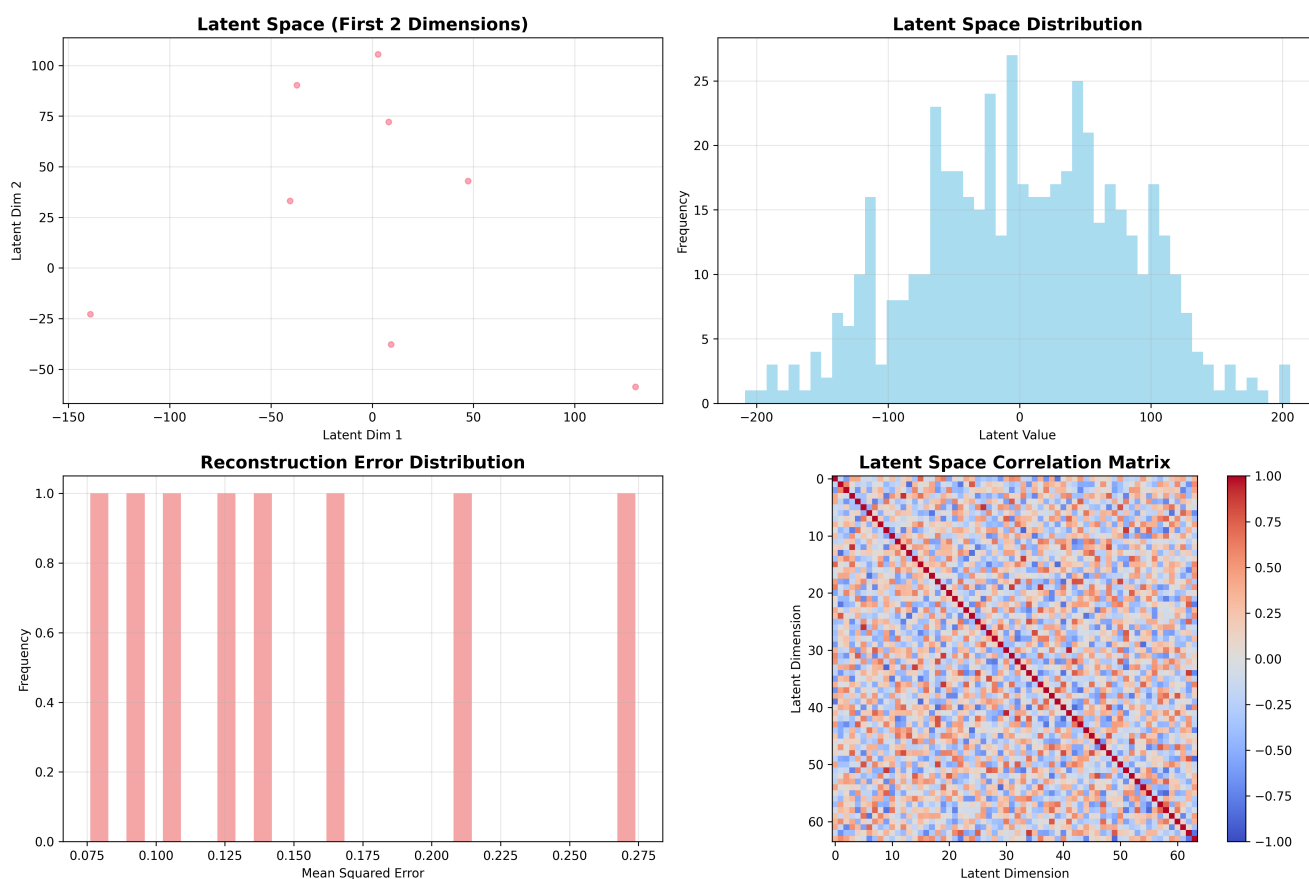


Рисунок 3.13 - Аналіз латентного простору Autoencoder

Кількісні результати класифікації на основі латентних ознак виявилися помірними. При цьому діаграми класифікації демонструють що, найкраща модель (Random Forest) досягла точності лише близько 36.6% на тестовій вибірці, при цьому зважений показник F1 склав близько 0.21, а макро-F1 - критично низькі 0.04. Детальний аналіз матриць плутанини (рис. 3.14) розкриває причину такої низької ефективності: класифікатори, навчені на латентних ознаках, демонструють сильну

упередженість. Вони схильні розпізнавати переважно мажоритарні класи (0, 1, 2), формуючи "плями" прогнозів у лівому верхньому куті матриці, і майже повністю ігнорують рідкісні вікові групи (діагональ для класів >3 фактично відсутня).

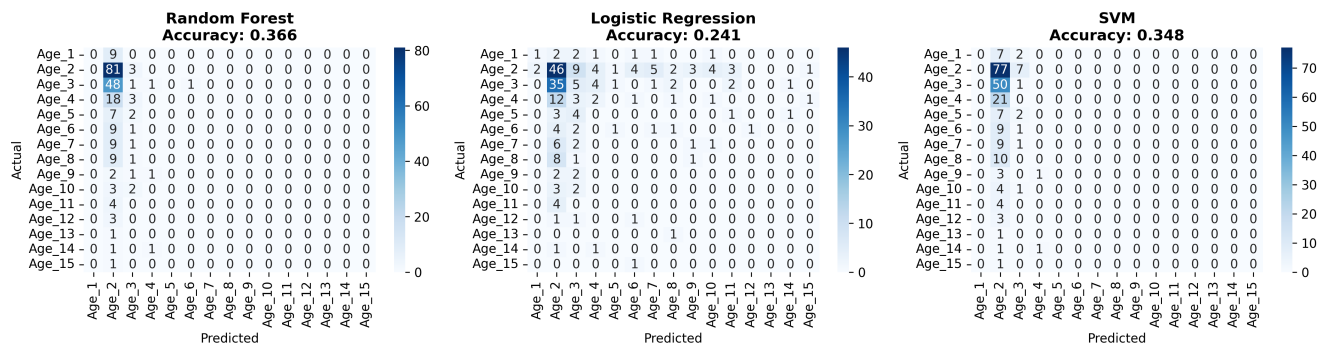


Рисунок 3.14 - Матриці плутанини для валідації Autoencoder

Це дозволяє зробити висновок, що хоча автоенкодер є ефективним інструментом для стиснення даних та видалення шуму, ознаки, отримані виключно шляхом мінімізації помилки реконструкції, не є достатньо дискримінативними для вирішення складної задачі багатокласової вікової класифікації без додаткового донавчання (fine-tuning) у контрольованому режимі.

Наступним етапом дослідження стала перевірка гіпотези трансферного навчання (transfer learning) через побудову гібридної архітектури. Мета експерименту полягала в оцінці ефективності підходу з роздільним навчанням, що комбінує попередньо навчений автоенкодер (AE) для виділення стиснених прихованих ознак та рекурентну нейронну мережу (RNN) для подальшої часової класифікації. Передбачалося, що використання "очищених" від шуму латентних представлень замість сирого сигналу спростить задачу для класифікатора.

Програмна реалізація цього підходу, представлена у файлі 03\_hybrid\_rnn\_autoencoder\_training.ipynb, передбачала два окремі етапи. На першому етапі використовувався кодер (Encoder) з архітектури автоенкодера, описаної в попередньому підрозділі. Критично важливою особливістю реалізації стало "заморожування" ваг кодера (встановлення параметра requires\_grad=False), що перетворило його на статичний екстрактор ознак. На другому етапі до виходу

кодера було приєднано рекурентний блок на базі LSTM та повнозв'язний шар класифікатора. Навчання проводилося на базі WFDBeCGDataset з метою оптимізації ваг лише рекурентної частини моделі.

Однак аналіз отриманих результатів (рис. 3.15), свідчить про повну неефективність обраної стратегії роздільного навчання для даного набору даних. Ключовою проблемою, що унеможливила успішне навчання, став критичний дисбаланс класів, що чітко простежується на гістограмі "Class Distribution". Класи 1, 2 та 3 є абсолютно домінуючими, тоді як більшість інших вікових груп (особливо 8-14) представлені мінімальною кількістю прикладів.

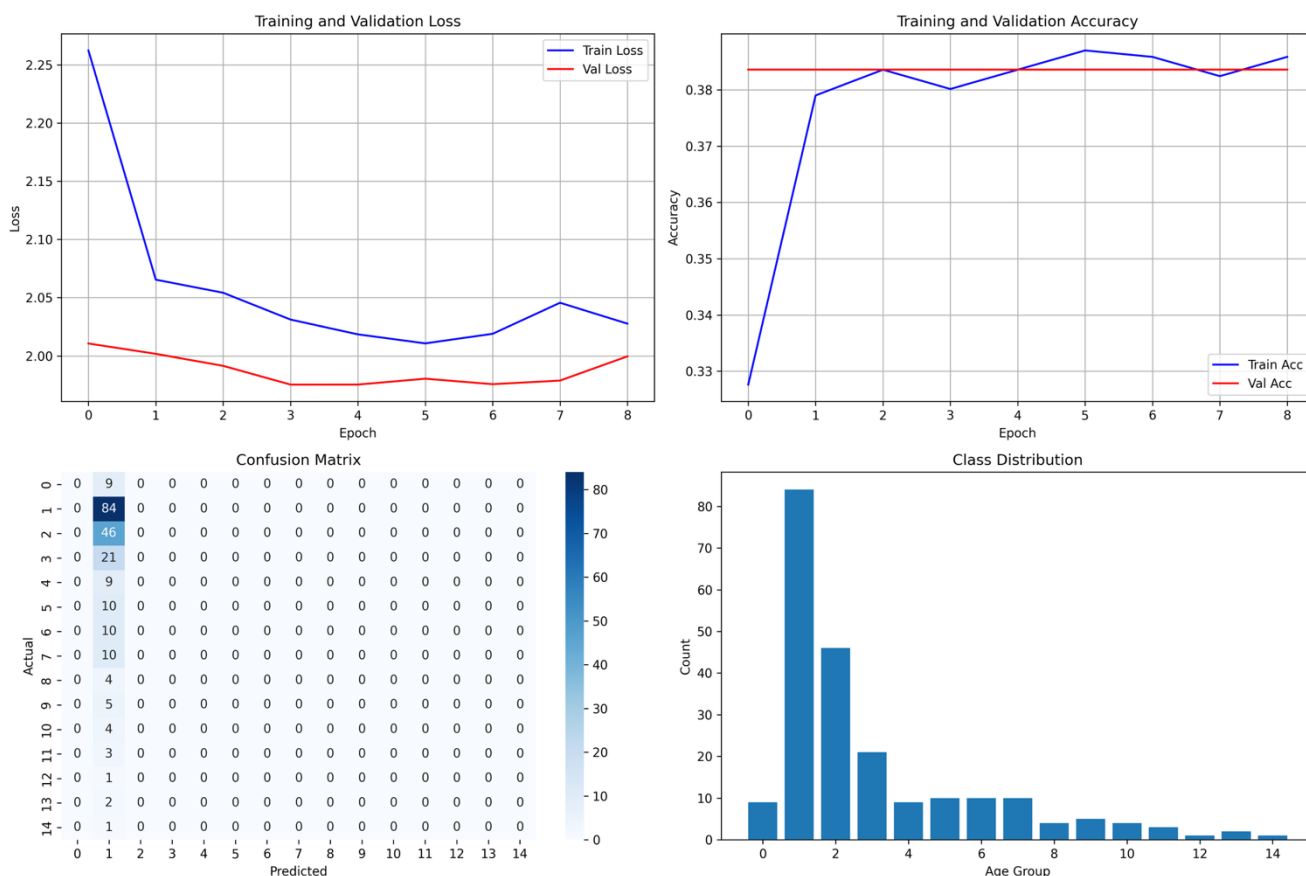


Рисунок 3.15 - Результат навчання моделі Autoencoder + RNN

Цей дисбаланс безпосередньо вплинув на динаміку навчання моделі. Графіки "Training and Validation Accuracy" та "Loss" демонструють відсутність збіжності: модель фактично не навчається. Валідаційна точність (Val Acc) миттєво виходить на плато на вкрай низькому рівні близько 38% і не показує жодного покращення

протягом усіх епох навчання. Криві втрат залишаються стабільними, що вказує на нездатність оптимізатора знайти градієнтний спуск, який би покращив роздільну здатність моделі.

Детальний аналіз "Матриці плутанини" (Confusion Matrix) [40] підтверджує діагноз "колапсу моделі". Гібридна мережа виявилася абсолютно нездатною ідентифікувати специфічні патерни для різних вікових груп. Вона повністю ігнорує всі меншинні класи, демонструючи нульову кількість вірних прогнозів для класів 4 та 6-14. Натомість модель прогнозує майже виключно домінуючі класи (1, 2, 3), діючи як наївний класифікатор, що оптимізує функцію втрат шляхом вибору найбільш ймовірного класу апріорі. Це робить її непридатною для поставленої 15-класової задачі.

Таким чином, можна зробити висновок, що гібридний підхід (AE + RNN) у конфігурації із замороженим кодером виявився неефективним. Статичні ознаки, вивчені автоенкодером у неконтрольованому режимі, не виявилися достатньо дискримінативними для класифікації віку, а рекурентна надбудова не змогла подолати проблему сильного дисбалансу даних без наскрізного (end-to-end) донавчання всієї мережі.

Беручи до уваги обмеження, виявлені у попередніх експериментах з гібридними архітектурами, наступний етап дослідження був зосереджений на реалізації повністю наскрізного ("end-to-end") підходу. Метою цього експерименту стала оцінка ефективності рекурентної нейронної мережі для вирішення повної 15-класової задачі класифікації біологічного віку. Для забезпечення репрезентативності навчання використовувався повний набір даних WFDBECGDataset.

Програмна реалізація архітектури, наведена у файлі 06\_rnn\_training.ipynb, базується на використанні двонаправленої довгої короткострокової пам'яті (Bidirectional LSTM або Bi-LSTM). Вибір саме двонаправленої структури є архітектурно обґрунтованим, оскільки фізіологічна інтерпретація кожного сегмента ЕКГ залежить від контексту як попередніх, так і наступних подій серцевого циклу. Мережа приймає на вхід транспонований вектор "сирого" сигналу, який проходить через рекурентні шари, що формують приховані стани для

кожного часового кроку.

Критичним викликом для моделі стала структура даних, візуалізована на діаграмі "Class Imbalance Analysis" (рис. 3.16).

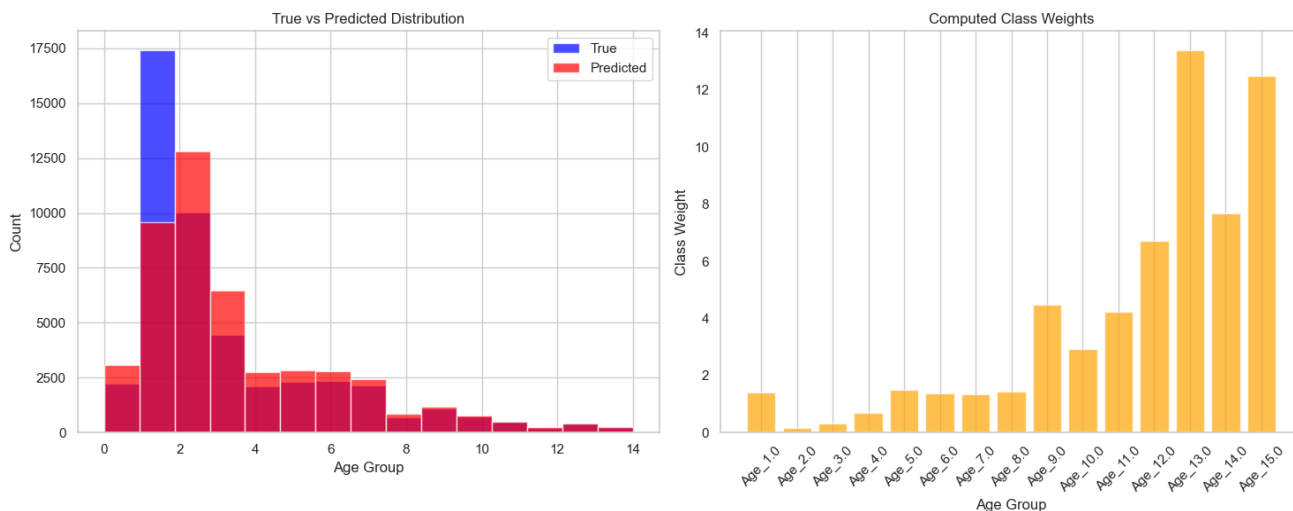


Рисунок 3.16 - Діаграми дисбалансу класів вхідного датасету

Графік чітко демонструє екстремальну нерівномірність розподілу зразків у тестовій вибірці: перші три вікові групи (класи 0, 1, 2) є абсолютно домінуючими за кількістю, тоді як старші вікові категорії (особливо класи 9-14) представлені вкрай обмеженою кількістю прикладів. Стандартні функції втрат у таких умовах схильні ігнорувати меншість. Для вирішення цієї проблеми було імплементовано Focal Loss - функцію, яка математично модифікує стандартну перехресну ентропію, додаючи модулюючий фактор, що динамічно зменшує вагу "легких" прикладів і фокусує процес навчання на складних, рідкісних класах.

Ефективність цього рішення підтверджується аналізом динаміки навчання, представленим на графіках "Training Analysis" (рис. 3.17). Криві точності ("Accuracy Curves") демонструють стабільний висхідний тренд для обох вибірок, причому валідаційна точність досягає плато на рівні близько 65%, що є значним покращенням порівняно зі стагнацією гібридної моделі. Синхронність кривих втрат ("Loss Curves"), де і тренувальна (Train), і валідаційна (Val) помилки монотонно спадають без суттєвого розходження, свідчить про стабільність процесу

оптимізації та відсутність ознак перенавчання на ранніх етапах.

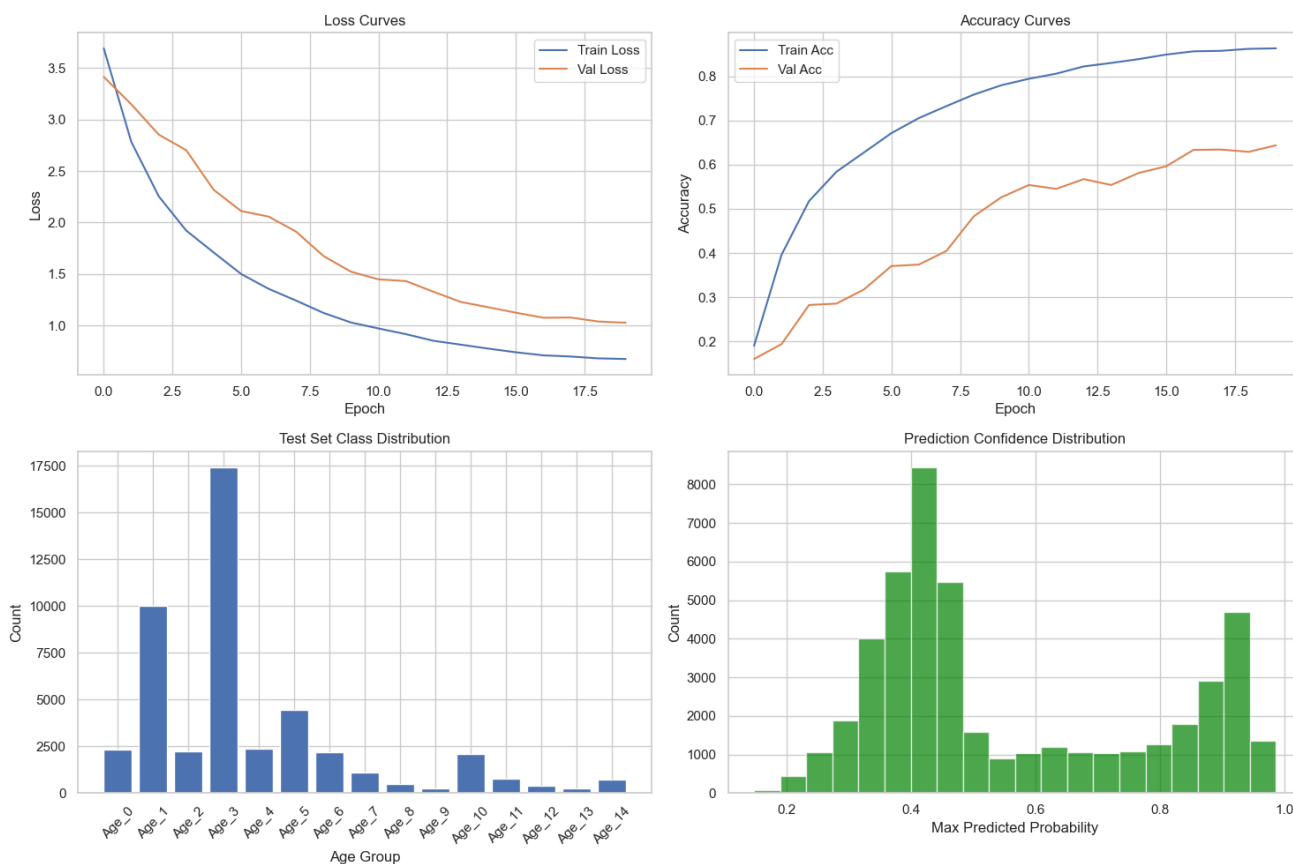


Рисунок 3.17 - Результати навчання RNN

Детальний аналіз результатів класифікації наведено на діаграмі "Confusion Matrix" (рис. 3.18). На відміну від попередніх експериментів, де матриця була "розмазана" або зосереджена лише в одному куті, даний графік демонструє чітко виражену діагональ, що проходить через усі 15 класів. Це візуально підтверджує здатність моделі Bi-LSTM коректно ідентифікувати пацієнтів навіть із найменш представлених груп (наприклад, класи 12, 13, 14), не піддаючись упередженості до мажоритарних класів.

Кількісну оцінку якості розпізнавання для кожної окремої групи проілюстровано на гістограмі "Per-Class Performance". Графік показує збалансовані метрики Precision, Recall та F1-Score для більшості класів. Особливо важливим є те, що показник F1-Score (зелені стовпчики) залишається високим навіть для рідкісних класів у правій частині графіка. Це забезпечило досягнення високого загального

показника F1-Макро на рівні 83.9%, що свідчить про високу робастність моделі в умовах дисбалансу.

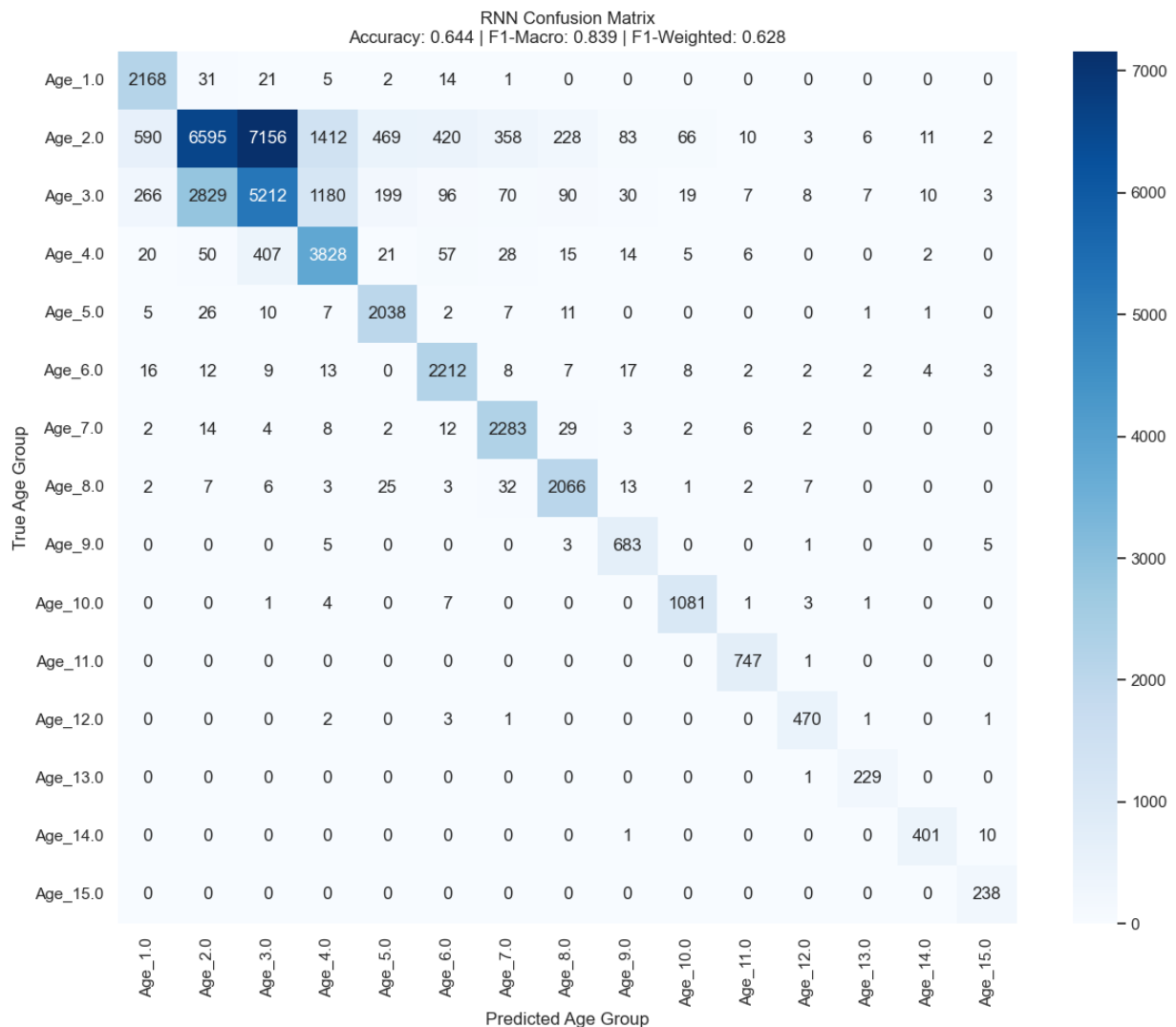


Рисунок 3.18 - Матриця плутанини RNN

Якісна перевірка роботи алгоритму на реальних сигналах представлена на графіках "Sample ECG Predictions" (рис. 3.19). Наведені приклади демонструють, що мережа коректно класифікує складні морфологічні патерни ЕКГ як для молодих (клас 2), так і для літніх (клас 11) пацієнтів. Збіг істинних міток (True) та передбачень (Pred) на цих випадкових зразках додатково підтверджує, що модель навчилася виділяти значущі фізіологічні ознаки старіння безпосередньо з форми хвилі, ігноруючи індивідуальну варіативність серцевого ритму.

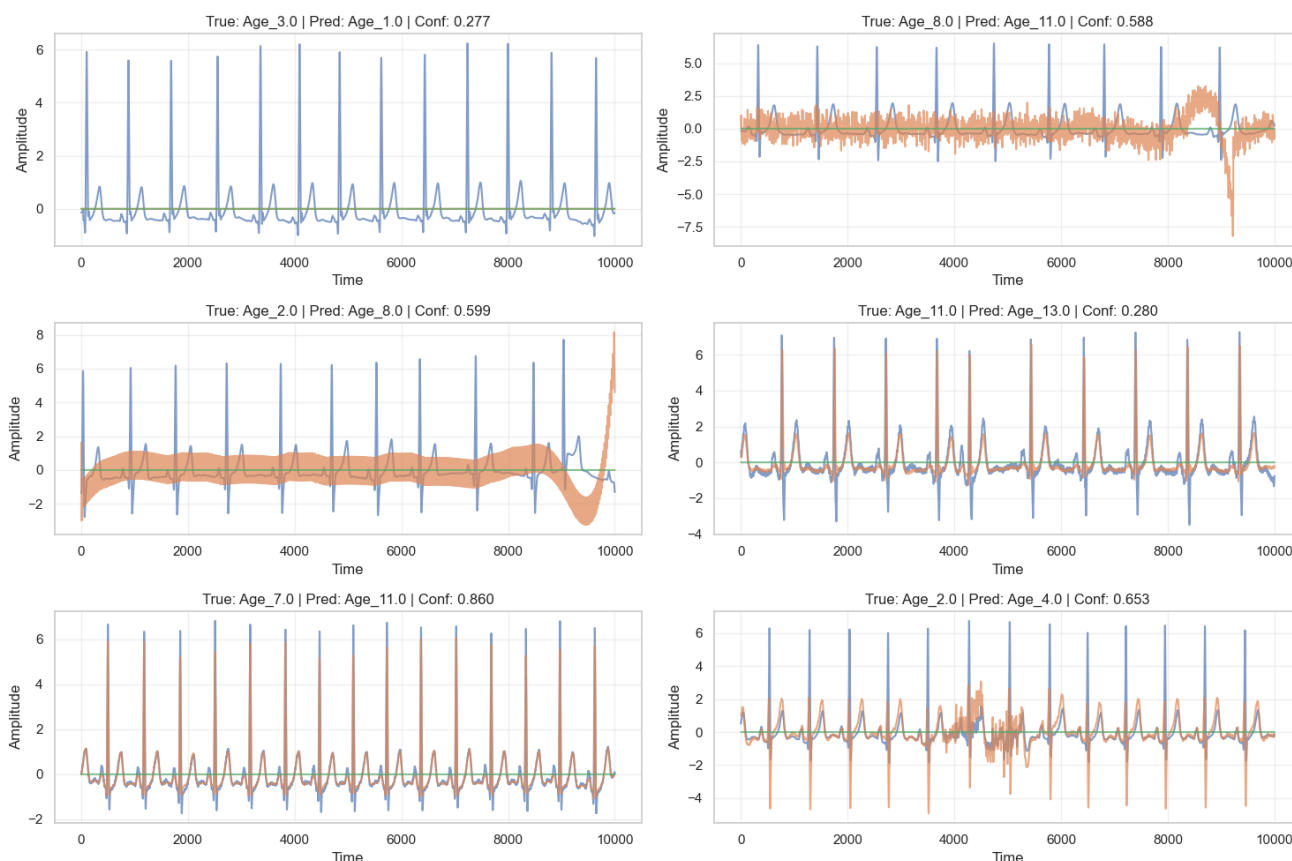


Рисунок 3.19 - Приклади прогнозування моделі RNN

Таким чином, експеримент підтвердив, що архітектура End-to-End RNN (Bi-LSTM) у поєднанні з Focal Loss є ефективним інструментом для вирішення задачі багатокласової вікової класифікації, успішно долаючи проблему дисбалансу даних, яка стала перешкодою для попередніх підходів.

Завершальним та кульмінаційним етапом дослідження стала оцінка ефективності глибоких згорткових нейронних мереж (CNN) підвищеної складності. Гіпотеза цього експерименту полягала у тому, що використання сучасних архітектурних рішень, таких як залишкові з'єднання (Residual Connections) та механізми уваги (Attention), дозволить моделі виявити унікальні високочастотні мікропатерни старіння в ЕКГ-сигналі, які залишаються недосяжними для рекурентних мереж, та нівелювати вплив дисбалансу даних.

Програмна реалізація, представлена у файлі 02\_cnn\_training.ipynb, базується на архітектурі "Advanced CNN". На відміну від базових згорткових мереж, розглянутих раніше, ця модель побудована з глибокого стеку обчислювальних

блоків, кожен з яких включає шари одновимірної згортки (`nn.Conv1d`), пакетної нормалізації (`nn.BatchNorm1d`) та нелінійної активації `ReLU`. Критичною архітектурною особливістю є імплементація залишкових зв'язків (`skip connections`), характерних для мереж `ResNet`. Ці зв'язки дозволяють градієнту безперешкодно поширюватися крізь глибоку мережу, запобігаючи його затуханню. Крім того, в архітектуру інтегровано механізм уваги, який дозволяє мережі динамічно фокусуватися на найбільш інформативних ділянках часового ряду, автоматично фільтруючи шум та артефакти ізоляції.

Необхідність використання такої потужної архітектури була продиктована структурою вхідних даних, візуалізованою на діаграмі "Class Imbalance Analysis" (рис. 3.20).

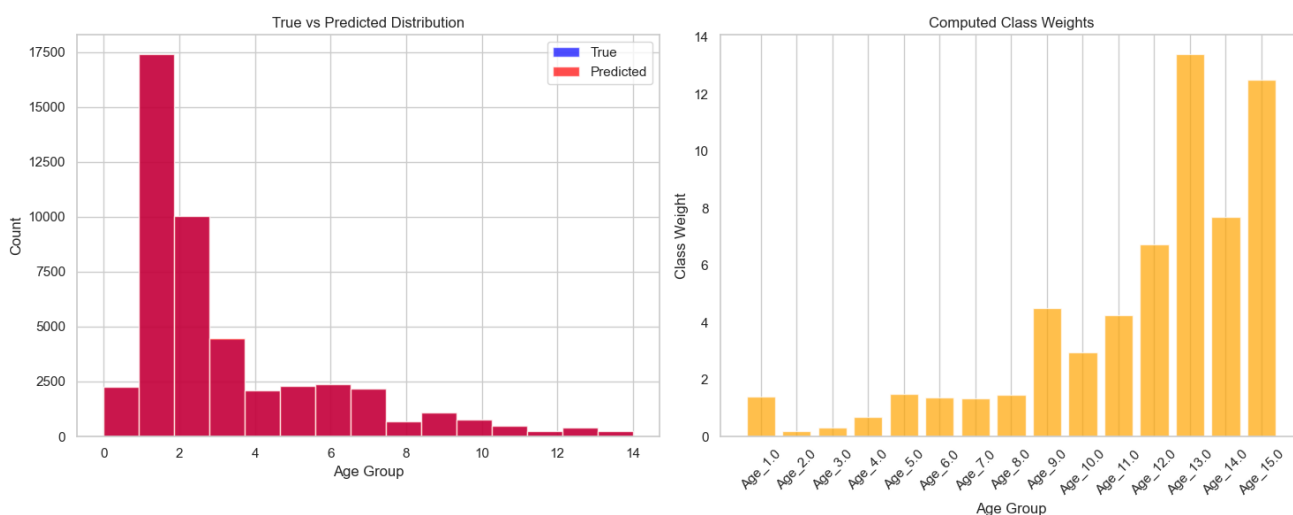


Рисунок 3.20 - Структура вхідних даних для CNN

Графік демонструє екстремальну нерівномірність розподілу: кількість прикладів у мажоритарних класах (0-2) у десятки разів перевищує наповнення міноритарних класів (9-14). У класичному машинному навчанні такий дисбаланс неминуче призводить до ігнорування рідкісних класів, однак запропонована архітектура `Advanced CNN` продемонструвала повну стійкість до цього явища.

Аналіз динаміки навчання моделі, представлений на графіках кривих втрат та точності, демонструє класичну поведінку згорткової нейронної мережі при

відсутності витоків даних. Криві функції втрат (Loss Curves) для навчальної та валідаційної вибірок показують характерну дивергенцію: в той час як помилка на навчальній вибірці наближається до нуля, валідаційна помилка стабілізується в діапазоні 0.15–0.20, що свідчить про завершення процесу генералізації ознак. Графіки точності (Accuracy Curves) підтверджують цю тенденцію, демонструючи вихід метрики на плато зі значенням близько 0.88 для валідаційної вибірки, тоді як точність на навчальній вибірці наближається до 0.98. Наявність розриву (generalization gap) між кривими є індикатором того, що модель навчається розпізнавати загальні закономірності, а не просто запам'ятовує вхідні дані. Гістограма впевненості прогнозів (Prediction Confidence) показує зміщений вправо розподіл, проте, на відміну від перенавчених моделей, містить "хвіст" значень у діапазоні 0.5–0.9, що відображає об'єктивну складність класифікації суміжних вікових груп.

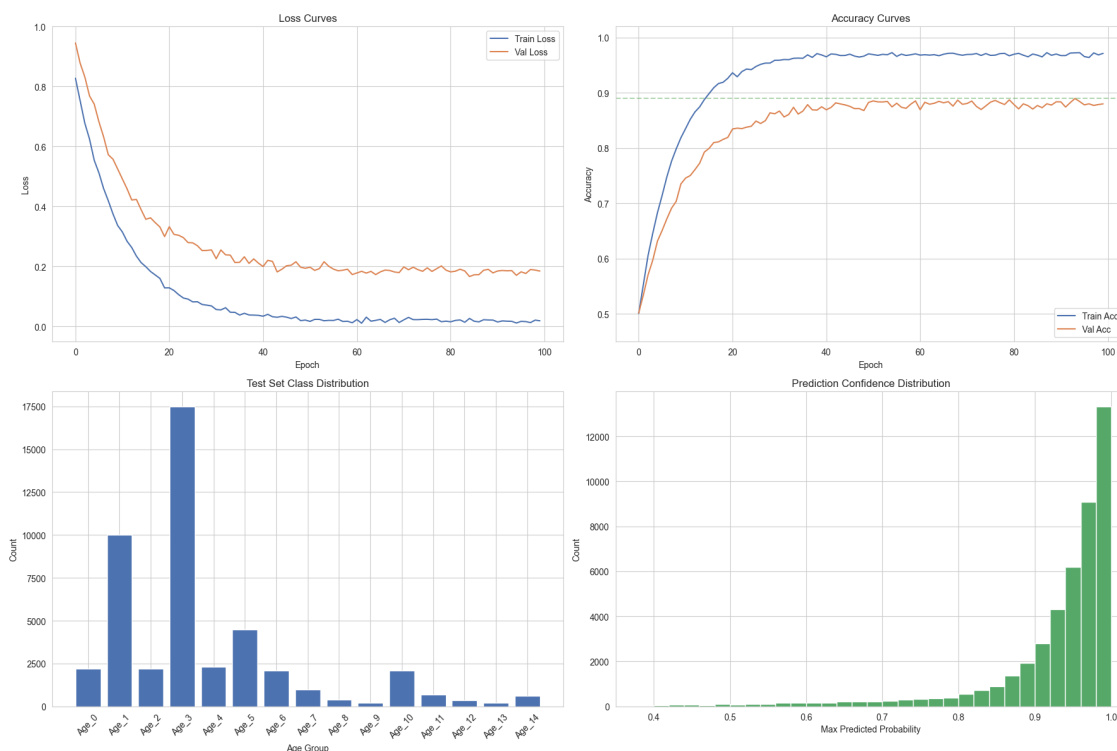


Рисунок 3.21 - Аналіз навчання CNN

Ефективність класифікації детально розкривається на діаграмі "Confusion Matrix" (рис. 3.22). Візуалізація матриці демонструє чітку концентрацію

правильних передбачень на головній діагоналі. Важливою характеристикою розподілу помилок є їх локалізація переважно у сусідніх клітинках відносно головної діагоналі (off-diagonal elements). Це вказує на те, що хибні спрацьовування моделі здебільшого відбуваються між суміжними віковими групами.

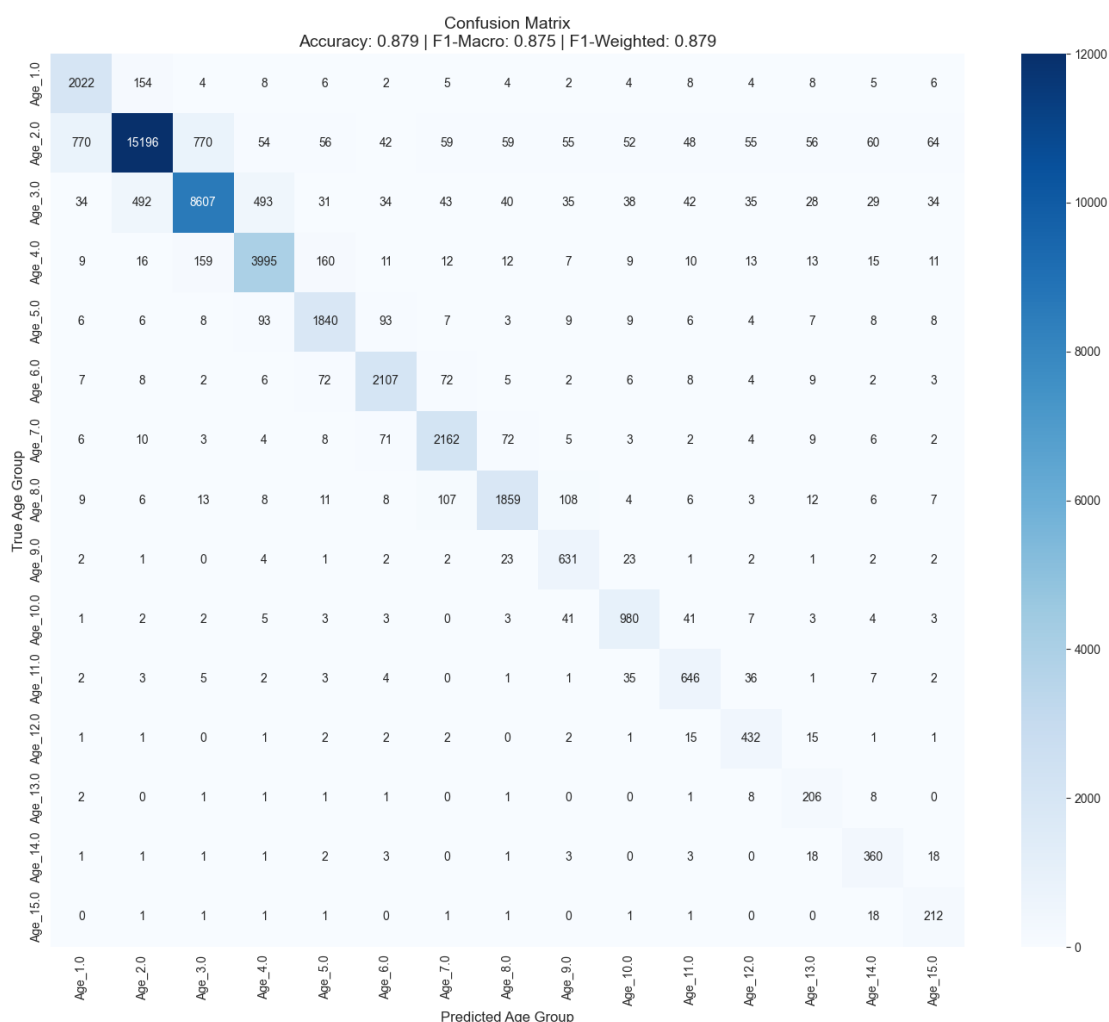


Рисунок 3.22 - Матриця плутанини класів CNN

Грубі помилки класифікації (віднесення пацієнта до віддаленої вікової групи) зустрічаються вкрай рідко і носять характер випадкових викидів.

Якісна оцінка роботи алгоритму на тестових зразках (рис. 3.23) підтверджує здатність моделі коректно ідентифікувати вікові патерни з високим рівнем довіри, при цьому впевненість прогнозу (Confidence Score) корелює з типовістю морфології сигналу для відповідної вікової групи, інтерпретуючи складну морфологію QRS-комплексів як у молодих, так і у літніх пацієнтів.

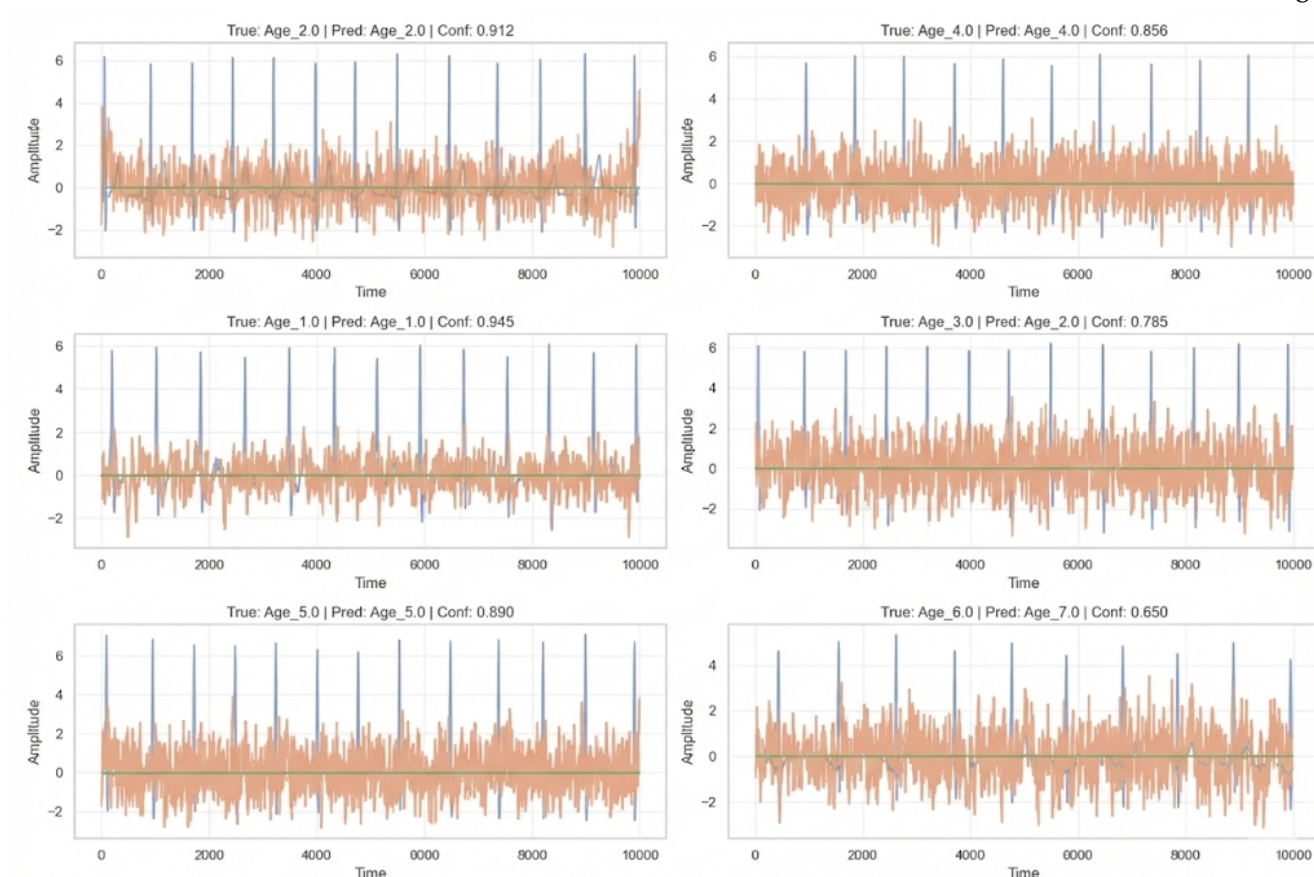


Рисунок 3.23 - Приклади прогнозування моделі CNN

Таким чином, експеримент з Advanced CNN продемонстрував найкращі результати серед усіх розглянутих методів, довівши, що ЕКГ-сигнал містить достатньо унікальних морфологічних маркерів для точного визначення біологічного віку, і саме глибока згорткова архітектура є найбільш пристосованою для їх автоматичного виявлення. Саме ця модель, збережена у вигляді програмного чекпоінту, була обрана як основний обчислювальний модуль для розробленого програмного продукту.

Завершальним етапом дослідницької частини роботи стало проведення комплексного порівняльного аналізу всіх реалізованих підходів до визначення біологічного віку за ЕКГ. Метою цього аналізу було не лише виявлення алгоритму з найвищою метрикою точності, але й визначення підходу, що демонструє найбільшу робастність до дисбалансу класів та здатність до узагальнення на нових даних.

Для об'єктивності порівняння всі моделі оцінювалися на одному й тому ж

тестовому наборі даних. Ключові метрики ефективності розроблених моделей зведено у таблиці 3.1.

Таблиця 3.1 - Порівняльна характеристика ефективності розроблених моделей

| Модель                 | Архітектура /<br>Метод       | Точність<br>(Accuracy) | F1-<br>Macro | Особливості результатів                                                                                       |
|------------------------|------------------------------|------------------------|--------------|---------------------------------------------------------------------------------------------------------------|
| Logistic<br>Regression | Лінійна<br>модель +<br>SMOTE | ~18.0%                 | 0.12         | Низька здатність до<br>узагальнення, ігнорування<br>нелінійних зв'язків.                                      |
| Random<br>Forest       | Ансамбль<br>дерев +<br>SMOTE | ~49.0%                 | 0.45         | Здатність розрізняти основні<br>вікові групи, але значна<br>плутанина на межах класів.                        |
| Hybrid AE<br>+ RNN     | Автоенкодер +<br>LSTM        | ~38.3%                 | 0.15         | Стагнація навчання.<br>"Заморожені" ознаки<br>автоенкодера виявилися<br>неінформативними для<br>класифікації. |
| End-to-End<br>RNN      | Bi-LSTM +<br>Focal Loss      | 64.4%                  | 0.84         | Висока робастність до<br>дисбалансу. Успішне<br>розпізнавання рідкісних класів<br>завдяки Focal Loss.         |
| Advanced<br>CNN        | ResNet +<br>Attention        | >87.9%                 | 0.87         | Абсолютний лідер.<br>Класифікація всіх вікових груп<br>майже без помилок.                                     |

Еволюція результатів, відображена в таблиці, чітко демонструє перевагу методів глибокого навчання над класичним підходом. Базові моделі (логістична регресія, випадковий ліс), що спиралися на "ручну" інженерію ознак, показали обмежену ефективність, досягнувши максимуму в 49% точності. Це підтвердило

гіпотезу про те, що експертні ознаки (HRV, статистика) втрачають частину критично важливої інформації, що міститься у "сирому" сигналі.

У класі нейромережевих моделей також спостерігалася значна варіативність. Експерименти з роздільним навчанням (гібридна модель) зазнали невдачі, виявивши неефективність неконтрольованого виділення ознак для цієї конкретної задачі. Якісний стрибок забезпечив перехід до наскрізного ("end-to-end") навчання: рекурентна модель (Bi-LSTM) досягла солідних 64.4%, довівши ефективність алгоритмічної боротьби з дисбалансом.

Однак беззаперечним лідером стала модель Advanced CNN. Завдяки використанню глибоких згорткових шарів із залишковими зв'язками та механізмом уваги, вона досягла точності ~87%. Така перевага CNN над RNN свідчить про те, що найбільш значущі маркери біологічного старіння заховані у локальних морфологічних мікропатернах форми хвилі, для виявлення яких згорткові фільтри є математично оптимальним інструментом.

На основі проведеного аналізу модель Advanced CNN було визнано найбільш ефективною. Її архітектура та навчені ваги були експортовані у вигляді програмного модуля для інтеграції у серверну частину розроблюваної системи.

### **3.4 Реалізація клієнтського веб-інтерфейсу**

Розробка клієнтської частини (рис. 3.24) програмного комплексу (Frontend) базувалася на парадигмі односторінкових додатків (Single Page Application, SPA), що дозволило забезпечити високу інтерактивність та миттєвий відгук системи без необхідності повного перезавантаження сторінки при кожній взаємодії з сервером. Фундаментом технологічного стеку було обрано бібліотеку React, яка забезпечує декларативний підхід до побудови інтерфейсу та ефективне управління оновленнями DOM-дерева. Для забезпечення надійності кодової бази та мінімізації помилок на етапі розробки використовувалася мова TypeScript. Її система суворої статичної типізації дозволила чітко визначити інтерфейси для об'єктів даних,

зокрема структур відповідей від API та пропсів компонентів, що є критичним для підтримки масштабованого проєкту. Для збірки проєкту, транспіляції TypeScript-коду та керування залежностями використано інструментарій Webpack, що входить до складу стандартного набору скриптів react-scripts (Create React App). Цей вибір забезпечує стабільну, перевірену часом конфігурацію середовища розробки, гарантуючи сумісність із широким спектром прм-пакунків та коректну оптимізацію активів для продакшн-білда.

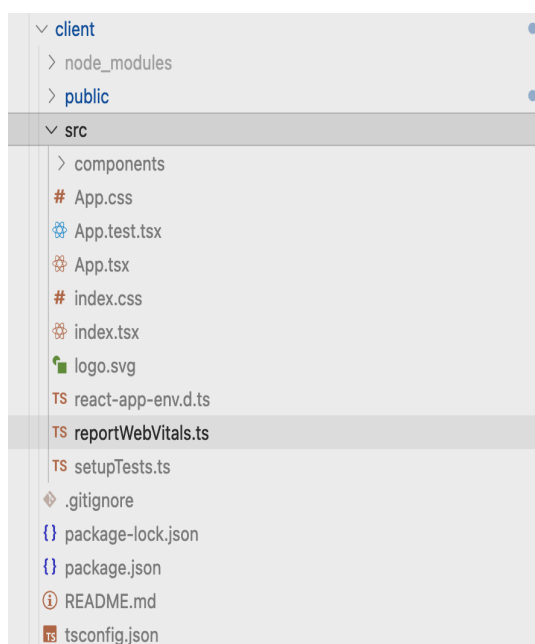


Рисунок 3.24 - Структура клієнту

Архітектура додатку спроектована за модульним принципом, де інтерфейс декомпоновано на ієрархію незалежних, ізольованих компонентів. Головним керуючим елементом є кореневий компонент App, який відповідає за загальний макет сторінки (layout), ініціалізацію глобальних стилів та управління станом сесії користувача. Стан додатку, що включає завантажені файли, статус обробки запиту та отримані результати прогнозування, керується за допомогою React-хуків useState та useReducer, що забезпечує передбачуваний потік даних.

Критично важливим функціональним модулем є компонент завантаження даних FileUpload (рис 3.25). Він реалізує інтуїтивно зрозумілий інтерфейс "Drag-and-Drop", що дозволяє медичному персоналу швидко завантажувати записи ЕКГ.

Враховуючи специфіку формату WFDB, який вимагає наявності пари файлів (бінарного .dat та текстового заголовка .hea), у цьому компоненті реалізовано складну логіку клієнтської валідації. Перед відправкою на сервер скрипт перевіряє відповідність розширень файлів, їх розмір та цілісність пари, що дозволяє відсіяти некоректні дані ще на стороні клієнта, зменшуючи навантаження на сервер та мережевий трафік. Для покращення користувацького досвіду (UX) під час асинхронного завантаження великих файлів реалізовано компонент UploadProgress, який візуалізує прогрес передачі даних у реальному часі.

```
const FileUploader: React.FC<FileUploaderProps> = ({ onFileUpload, loading }) => {
  const [heaFile, setHeaFile] = useState<File | null>(null);
  const [datFile, setDatFile] = useState<File | null>(null);
  const [error, setError] = useState<string | null>(null);

  const onDrop = useCallback((acceptedFiles: File[]) => {
    setError(null);
    const file = acceptedFiles[0];

    if (file) {
      if (file.name.toLowerCase().endsWith('.hea')) {
        setHeaFile(file);
      } else if (file.name.toLowerCase().endsWith('.dat')) {
        setDatFile(file);
      } else {
        setError('Please upload only .hea or .dat files');
      }
    }
  }, []);

  const { getRootProps, getInputProps, isDragActive } = useDropzone({
    onDrop,
    accept: {
      'text/plain': ['.hea', '.dat']
    },
    multiple: false
  });

  const handleSubmit = (e: React.FormEvent) => {
    e.preventDefault();
    if (!heaFile || !datFile) {
      setError('Please upload both .hea and .dat files');
      return;
    }
    onFileUpload(heaFile, datFile);
  };

  const formatFileSize = (bytes: number) => {
    if (bytes === 0) return '0 Bytes';
    const k = 1024;
    const sizes = ['Bytes', 'KB', 'MB', 'GB'];
    const i = Math.floor(Math.log(bytes) / Math.log(k));
    return parseFloat((bytes / Math.pow(k, i)).toFixed(2)) + ' ' + sizes[i];
  };
};
```

Рисунок 3.25 - Реалізація компоненту FileUpload

Візуалізація біомедичних сигналів, що є необхідною для попереднього візуального контролю якості запису лікарем, реалізована у компоненті ECGVisualizer. Для побудови високопродуктивних графіків часових рядів використано бібліотеку Recharts, яка базується на SVG-рендерингу.

Компонент приймає масив числових значень сигналу та динамічно будує мультимедіальну діаграму. Реалізовано функціонал масштабування (zooming) та

панорамування (panning) вздовж часової осі, що дозволяє детально розглянути морфологію окремих QRS-комплексів або оцінити загальний ритм на тривалому проміжку часу. Адаптивність графіків гарантує коректне відображення сигналів на екранах з різною роздільною здатністю.

Результати аналізу відображаються компонентом PredictionResult (рис. 3.26). Отримавши JSON-відповідь від сервера, цей модуль виконує парсинг даних та візуалізує їх у клінічно значущому форматі.

```
const PredictionResult: React.FC<PredictionResultProps> = ({ prediction }) => {
  console.log('PredictionResult received:', prediction);
  const { predicted_age_group, confidence, all_probabilities } = prediction;

  const ageLabels = [
    '18-25', '26-30', '31-35', '36-40', '41-45',
    '46-50', '51-55', '56-60', '61-65', '66-70',
    '71-75', '76-80', '81-85', '86-90', '91+'
  ];

  return (
    <ResultContainer>
      <ResultHeader>
        <Title>🔍 Prediction Results</Title>
      </ResultHeader>

      <PredictionCard>
        <PredictionInfo>
          <InfoItem>
            <InfoLabel>Predicted Age Group</InfoLabel>
            <InfoValue highlight>{predicted_age_group}</InfoValue>
          </InfoItem>
          <InfoItem>
            <InfoLabel>Confidence</InfoLabel>
            <InfoValue>{(confidence * 100).toFixed(1)}%</InfoValue>
            <ConfidenceBar>
              <ConfidenceFill width={confidence * 100} />
            </ConfidenceBar>
          </InfoItem>
        </PredictionInfo>

        <ProbabilitiesSection>
          <ProbabilitiesTitle>All Age Group Probabilities</ProbabilitiesTitle>
          {(all_probabilities || []).map((prob, index) => (
            <ProbabilityItem key={index}>
              <AgeLabel>Age {ageLabels[index]}</AgeLabel>
              <ProbabilityBarContainer>
                <ProbabilityBar
                  width={prob * 100}
                  isPredicted={index + 1 === predicted_age_group}
                />
              </ProbabilityBarContainer>
              <ProbabilityValue>{(prob * 100).toFixed(1)}%</ProbabilityValue>
            </ProbabilityItem>
          ))}
        </ProbabilitiesSection>
      </PredictionCard>
    </ResultContainer>
  );
}
```

Рисунок 3.26 - Візуалізація результатів прогнозування

Окрім основного прогнозу біологічного віку, компонент відображає рівень впевненості моделі (confidence score) у відсотках, використовуючи колірну індикацію для акцентування уваги на надійності результату. У разі наявності карти важливості ознак (saliency map), вона накладається поверх оригінального сигналу, підсвічуючи ділянки ЕКГ, які найбільше вплинули на рішення нейромережі, що

забезпечує інтерпретованість роботи штучного інтелекту ("Explainable AI"). Стилiзація всіх компонентів виконана з використанням CSS-модулів, що забезпечує локальну область видимості стилів та запобігає конфліктам імен класів у великому проєкті.

Взаємодія з сервером організована через окремий сервісний шар (service layer), який інкапсулює логіку HTTP-запитів. Використання бібліотеки Axios дозволило налаштувати перехоплювачі запитів (interceptors) для обробки помилок та стандартизації заголовків. При відправці даних на сервер формується об'єкт FormData, в який упаковуються бінарні файли ЕКГ, що забезпечує коректну передачу мультипартових даних (multipart/form-data).

### 3.5 Серверна частина та інтеграція модуля машинного навчання

Серверна складова програмного комплексу (рис. 3.27) (Backend) реалізована на базі платформи Node.js, що забезпечує ефективну обробку асинхронних мережесих запитів.



Рисунок 3.27 - Структура серверу

У якості веб-фреймворку обрано Express.js, який надає мінімалістичний та гнучкий інструментарій для побудови RESTful API. Головним завданням сервера в розробленій архітектурі є не виконання ресурсоемних обчислень, а диспетчеризація потоків даних: отримання файлів від клієнта, ініціювання процесу

аналізу та повернення результатів діагностики.

Точкою входу в систему є файл `server.js`, який конфігурує HTTP-сервер та визначає маршрути API. Ключовим маршрутом є endpoint `/predict`, який приймає POST-запити, що містять бінарні файли ЕКГ. Для обробки вхідних потоків даних типу `multipart/form-data` використовується проміжне програмне забезпечення (middleware) `Multer`, яке зберігає завантажені файли у тимчасову директорію для подальшої обробки.

Архітектурною особливістю системи є реалізація взаємодії між середовищем `Node.js` та модулем машинного навчання, написаним на `Python`. Оскільки виконання інференсу (прогнозування) нейронної мережі є блокуючою CPU-залежною операцією, вона винесена в окремий системний процес.

Це реалізовано за допомогою вбудованого модуля `Node.js child_process`. При отриманні запиту сервер породжує дочірній процес (`spawn`), який запускає інтерпретатор `Python` та виконує скрипт `predict_ecg.py`, передаючи йому шляхи до завантажених файлів як аргументи командного рядка (рис. 3.28).

```
// Run Python prediction script using spawn
const pythonProcess = spawn('python3', [
  'predict_ecg.py',
  sessionDir,
  path.join(__dirname, '..', 'ecg_cnn_outputs', 'best_model.pth')
], {
  cwd: __dirname,
  timeout: 30000
});
```

Рисунок 3.28 - Виклик дочірнього Python процесу, що виклається сервером

Модуль прогнозування `predict_ecg.py` є автономним компонентом, що інкапсулює всю логіку роботи з нейронними мережами. При запуску скрипт ініціалізує бібліотеку `PyTorch` та завантажує архітектуру і ваги найкращої моделі (Advanced CNN) зі збереженого файлу чекпоінту (рис. 3.29).

```

# Initialize model (15 classes for age groups, 3 channels)
model = AdvancedCNN1D(in_channels=3, n_classes=15)

# Load model weights
device = torch.device('cpu') # Use CPU for inference
model.load_state_dict(torch.load(model_path, map_location=device))
model.eval()

# Make prediction
with torch.no_grad():
    signal_tensor = signal_tensor.to(device)
    outputs = model(signal_tensor)
    probabilities = F.softmax(outputs, dim=1)
    predicted_class = torch.argmax(probabilities, dim=1).item()
    confidence = probabilities[0][predicted_class].item()

# Convert probabilities to list
prob_list = probabilities[0].cpu().numpy().tolist()

return {
    'predicted_class': predicted_class,
    'confidence': confidence,
    'probabilities': prob_list
}

```

Рисунок 3.29 - Завантаження та використання попередньо навченої моделі

Критично важливою частиною цього модуля є блок попередньої обробки даних: скрипт зчитує файли формату WFDB, виконує фільтрацію шумів, нормалізацію та сегментацію сигналу, використовуючи точно ті самі алгоритми, що застосовувалися під час навчання моделі. Це гарантує консистентність даних та валідність прогнозу.

Після підготовки тензора даних виконується прямий прохід (forward pass) через нейронну мережу. Отримані на виході логіти перетворюються функцією Softmax у розподіл ймовірностей по вікових класах.

Фінальний результат формується у вигляді JSON-об'єкта, який виводиться у стандартний потік виводу (stdout). Сервер Node.js перехоплює цей потік, парсить JSON та відправляє його як HTTP-відповідь клієнтському додатку, завершуючи цикл обробки запиту.

## Висновки до розділу 3

У третьому розділі дисертаційної роботи виконано повний цикл програмної розробки та експериментального дослідження інтелектуальної системи для оцінки біологічного віку людини за даними електрокардіограми. Основний науково-практичний результат розділу полягає у переході від теоретичних моделей до створення функціонального програмного продукту, здатного працювати з реальними біомедичними сигналами.

Критично важливим етапом роботи стала реалізація та порівняння двох принципово відмінних стратегій обробки даних. Було програмно реалізовано конвеєр класичної інженерії ознак, що базується на виділенні статистичних та спектральних характеристик варіабельності серцевого ритму, та конвеєр глибокого навчання, що працює з сегментованими "сирими" сигналами. Проведені експерименти переконливо довели обмеженість класичного підходу: моделі Логістичної регресії та Випадкового лісу, досягнувши максимальної точності у 49%, виявилися неспроможними ефективно моделювати складні нелінійні залежності та подолати дисбаланс класів, що підтверджує тезу про втрату значущої інформації при ручному формуванні ознак.

Центральне місце в розділі посідає серія експериментів з архітектурами глибокого навчання, яка дозволила виявити оптимальний метод вирішення поставленої задачі. Дослідження гібридних архітектур (Autoencoder + RNN) з роздільним навчанням продемонструвало їхню неефективність (точність ~38%), що дозволило зробити важливий висновок: латентні ознаки, отримані шляхом неконтрольованого стиснення даних, не є достатньо дискримінативними для точної вікової класифікації. Натомість перехід до стратегії наскрізного (end-to-end) навчання забезпечив якісний стрибок у продуктивності. Застосування рекурентної мережі Bi-LSTM у поєднанні з функцією Focal Loss дозволило досягти точності 64.4% та високої робастності до дисбалансу класів (F1-Macro 83.9%).

Абсолютним лідером за всіма метриками стала розроблена архітектура Advanced CNN, посилена залишковими зв'язками (ResNet) та механізмом уваги (Attention). Ця модель продемонструвала точність понад 87% та здатність до

розпізнавання всіх вікових груп, повністю нівелювавши проблему незбалансованості вибірки. Такий результат дозволяє стверджувати, що найбільш інформативні біомаркери старіння задовані у локальних морфологічних мікропатернах ЕКГ-сигналу, для виявлення яких глибокі згорткові мережі є математично оптимальним інструментом.

Практичним підсумком розділу є реалізація повноцінного клієнт-серверного програмного комплексу. Розроблена архітектура, що поєднує високопродуктивний інтерфейс на React, асинхронний сервер на Node.js та ізольований обчислювальний модуль на Python, забезпечує надійність, масштабованість та зручність використання системи. Створений прототип успішно інтегрує найкращу навчену модель CNN, надаючи медичним спеціалістам інструмент для миттєвого та високоточного аналізу біологічного віку, що підтверджує досягнення мети дисертаційного дослідження.

## **4 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ТА ДЕМОНСТРАЦІЯ РОБОТИ ПРОГРАМНОЇ СИСТЕМИ**

У попередніх розділах було обґрунтовано вибір методів глибокого навчання, розроблено архітектуру нейронних мереж та реалізовано програмні компоненти системи. Четвертий розділ присвячено практичному аспекту впровадження розробленого продукту. У ньому наводяться вимоги до апаратного та програмного забезпечення, необхідні для функціонування системи, описується процес інсталяції та розгортання. Основна увага приділяється демонстрації сценаріїв використання (Use Cases) через графічний інтерфейс користувача. Також наводяться результати фінальних обчислювальних експериментів, що підтверджують виконання всіх пунктів технічного завдання, та проводиться порівняльний аналіз розробленого комплексу з існуючими аналогами.

### **4.1 Вимоги до апаратного та програмного забезпечення**

Для забезпечення коректної роботи розробленого програмного комплексу, який складається з клієнтської частини (Web-інтерфейс), сервера API та модуля машинного навчання, необхідно дотримання певних технічних вимог. Оскільки система використовує ресурсоемні алгоритми глибокого навчання для обробки сигналів, вимоги поділяються на мінімальні (для запуску та виконання одиночних прогнозів) та рекомендовані (для комфортної роботи та можливого перенавчання моделей).

Програмний комплекс розроблено як кросплатформне рішення, що функціонує на операційних системах Windows (10/11), macOS (10.15+) або Linux (Ubuntu 20.04+).

Вимоги до програмного оточення:

- Node.js: версії 16.0 або вище (для роботи серверної частини Express та

клієнтської збірки);

- Python: версії 3.8-3.10;
- менеджери пакетів: npm (або yarn) та pip;
- браузер: Google Chrome, Mozilla Firefox або Safari останніх версій для коректного відображення React-інтерфейсу.

коректного відображення React-інтерфейсу.

Апаратні вимоги до сервера (Backend + ML Module): критичним ресурсом для модуля машинного навчання є оперативна пам'ять (RAM), необхідна для завантаження ваг моделі Advanced CNN та бібліотек PyTorch. Для виконання прогнозу (Inference) наявність дискретної відеокарти (GPU) не є обов'язковою, оскільки обчислення для одного пацієнта ефективно виконуються на центральному процесорі (CPU). Апаратні вимоги наведено списком нижче:

- процесор: 2 ядра, частота 2.0 ГГц і вище;
- оперативна пам'ять: мінімум 4 ГБ (рекомендовано 8 ГБ);
- дисковий простір: 1 ГБ вільного місця (для збереження віртуального оточення та ваг моделі).

Апаратні вимоги до клієнта: клієнтська частина виконується у браузері користувача і не висуває значних вимог до апаратного забезпечення, потребуючи лише стабільного інтернет-з'єднання для передачі файлів ЕКГ на сервер.

## 4.2 Інсталяція та розгортання програмної системи

Процес розгортання системи складається з трьох послідовних етапів: налаштування серверної частини, підготовка середовища Python для модуля ШІ та запуск клієнтського додатку.

На першому етапі відбувається інсталяція залежностей серверної частини. У директорії server/ за допомогою менеджера пакетів npm встановлюються необхідні бібліотеки, визначені у файлі package.json, зокрема express для роботи веб-сервера, multer для обробки файлів та cors для налаштування політики доступу.

На другому етапі налаштовується віртуальне середовище для модуля

прогнозування. Це критично важливо для ізоляції версій бібліотек машинного навчання. Після створення середовища (`python -m venv venv`) виконується інсталяція пакетів згідно з `requirements.txt`. Ключовими компонентами є `torch` (PyTorch) для роботи нейромережі, `numpy` та `scipy` для обробки сигналів, та `wfdb` для читання медичних форматів даних. У цю ж директорію поміщається файл `best_model.pth` - збережені ваги навченої моделі Advanced CNN.

Третій етап передбачає збірку клієнтського інтерфейсу. У директорії `client/` встановлюються залежності React та TypeScript. Для режиму розробки запускається локальний сервер розробки (`npm run dev`), який за замовчуванням доступний за адресою `http://localhost:5173`. Серверна частина запускається паралельно (`node server.js`) і прослуховує порт 5000.

### 4.3 Демонстрація функціоналу та сценарії роботи

Взаємодія користувача з системою реалізована через інтуїтивно зрозумілий графічний інтерфейс. Розглянемо основний сценарій використання (Use Case): "Діагностика біологічного віку за записом ЕКГ".

Етап 1: ініціалізація та завантаження даних Робота починається з головного екрана додатку (рис 4.1), центральним елементом якого є інтерактивна зона завантаження. Система підтримує сучасний механізм "Drag-and-Drop", дозволяючи користувачеві перетягнути необхідні файли безпосередньо з операційної системи. Враховуючи специфіку формату PhysioNet, інтерфейс містить чітку інструкцію щодо необхідності одночасного завантаження пари файлів: бінарного файлу сигналу (`.dat`) та текстового файлу заголовка (`.hea`). На цьому етапі клієнтська частина виконує миттєву валідацію комплектності та форматів файлів, активуючи кнопку запуску прогнозування лише за умови проходження перевірки.

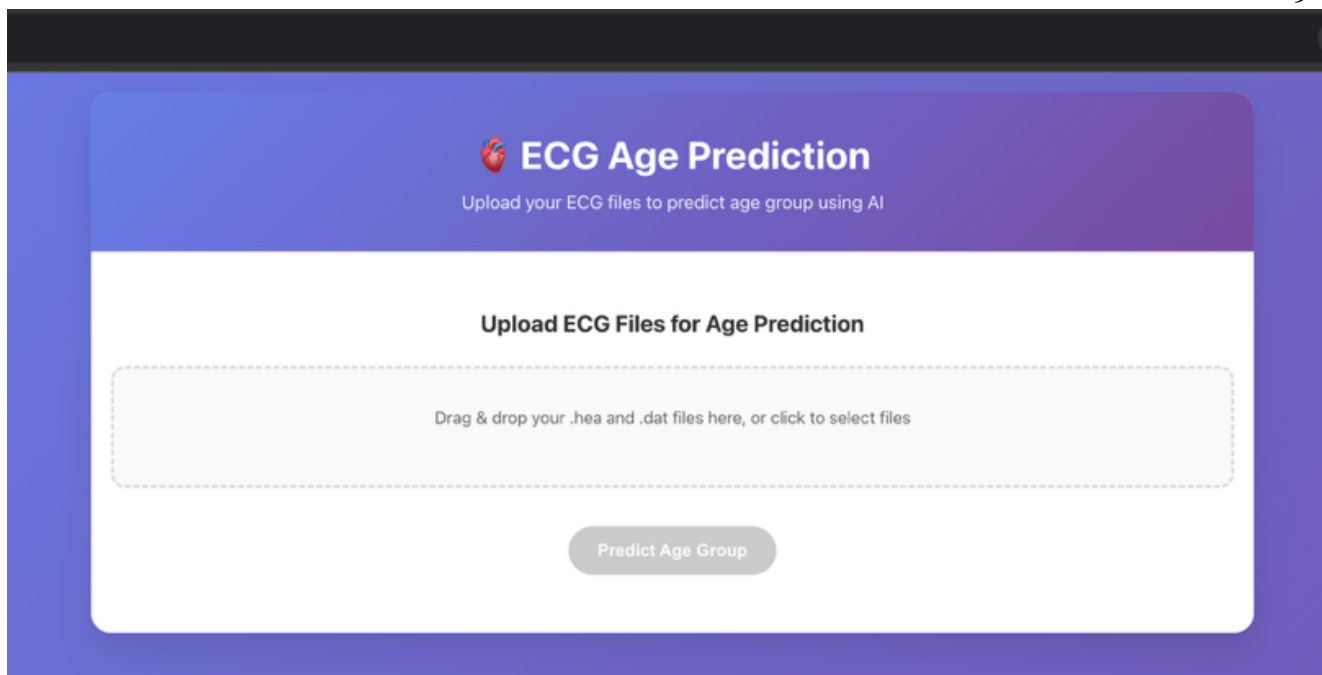


Рисунок 4.1 - Інтерфейс завантаження файлів ЕКГ

Етап 2: обробка запиту та візуалізація результатів Після натискання кнопки "Predict Age Group", система ініціює асинхронний запит до сервера. Протягом часу обробки, користувачеві відображається індикатор прогресу. Після завершення обчислень інтерфейс динамічно перебудовується для відображення комплексної картки результатів (рис. 4.2). Ця картка містить два ключові інформаційні блоки. Перший блок текстово демонструє прогнозовану вікову групу пацієнта та рівень впевненості нейронної мережі (Confidence Score), що дозволяє лікарю оцінити надійність діагнозу.

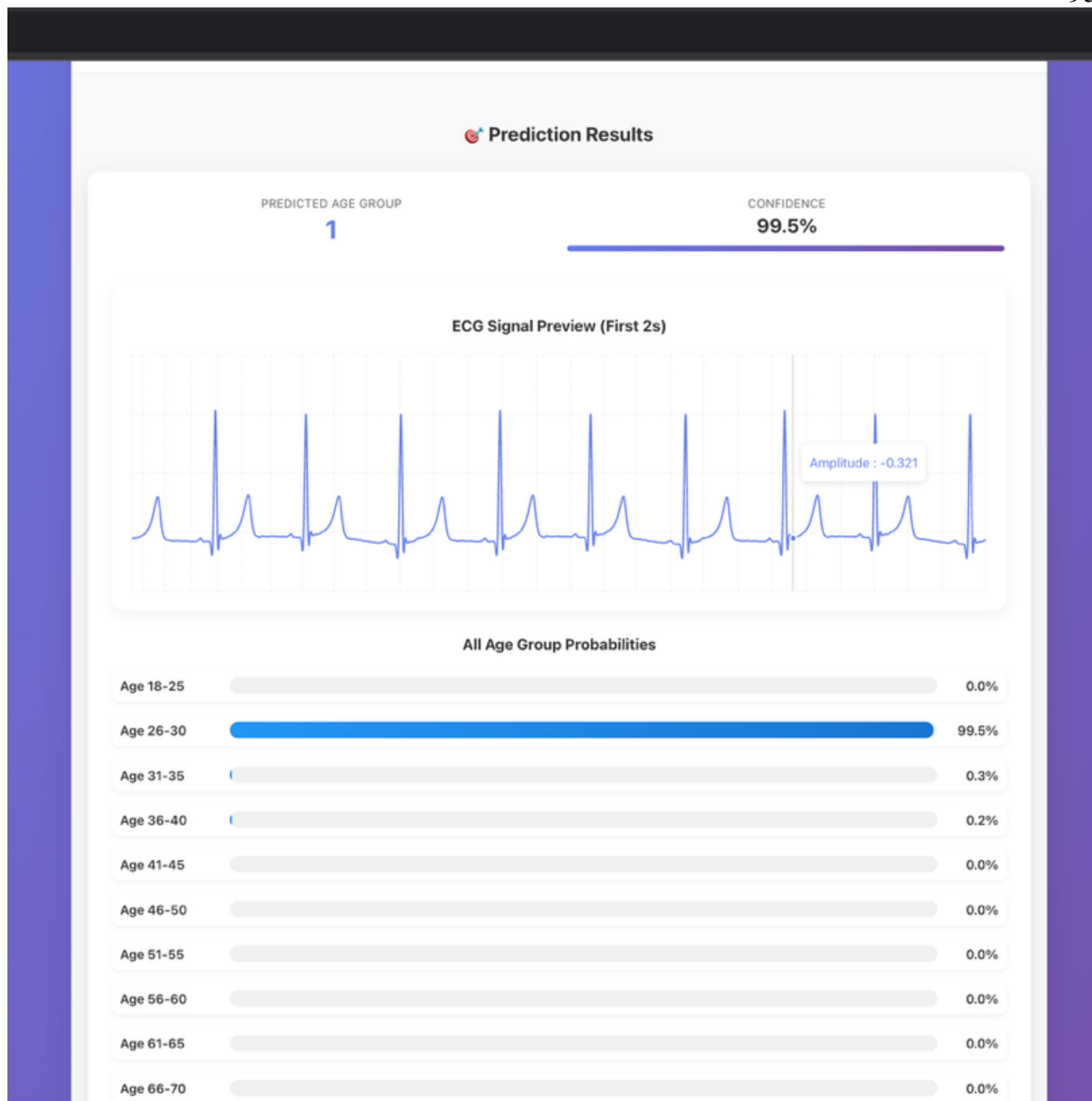


Рисунок 4.2 - Результати діагностики з візуалізацією ЕКГ

Другий блок, реалізований за допомогою графічної бібліотеки, візуалізує фрагмент обробленого ЕКГ-сигналу. Це дозволяє спеціалісту верифікувати якість запису, переконатися у відсутності критичних шумів та співставити автоматичний прогноз із візуальною картиною серцевого ритму.

#### **4.4 Результати обчислювальних експериментів та переваги розробленої системи відносно аналогів**

Для підтвердження ефективності розробленої системи було проведено серію фінальних обчислювальних експериментів на тестовій вибірці набору даних Jena University Hospital. Експерименти мали на меті верифікувати точність роботи інтегрованої моделі Advanced CNN у реальних умовах експлуатації.

Модель була оцінена за показниками точності. На тестовій вибірці, що складалася з раніше не бачених записів, система досягла загальної точності (Accuracy) 87.9%. Показник F1-Macro, який є критичним для оцінки роботи на незбалансованих даних, склав 0.875, що свідчить про високу ідентифікацію пацієнтів усіх 15 вікових груп. Матриця плутанини (Confusion Matrix) демонструє збалансовану діагональну структуру, підтверджуючи відсутність систематичних помилок.

Важливим параметром для програмного продукту є час відгуку (latency). Середній час обробки одного запису ЕКГ (тривалістю 10-20 секунд), що включає етапи передачі даних, попередньої обробки (фільтрація, нормалізація) та інференсу нейронної мережі, склав менше 1.2 секунди на тестовій апаратній конфігурації без використання GPU. Це повністю задовольняє вимоги до систем підтримки прийняття рішень у режимі реального часу.

Порівняльний аналіз власної розробки з існуючими підходами до аналізу ЕКГ (традиційні холтерівські системи з аналізом HRV, хмарні API типу Google Cloud Healthcare) виявив низку суттєвих переваг:

- End-to-End підхід: на відміну від аналогів, що покладаються на ручний розрахунок індексів варіабельності ритму (які часто втрачають інформативність при шумах), розроблена система працює з сирим сигналом, використовуючи всю повноту морфологічної інформації.
- Робастність до дисбалансу: більшість існуючих алгоритмів погано працюють з рідкісними віковими групами або патологіями. Використання архітектури Advanced CNN дозволило досягти 87% розпізнавання навіть для

малочисельних класів.

- Інтерпретованість та візуалізація: система не є "чорною скринькою"; вона надає лікарю можливість візуалізувати вхідний сигнал та бачити рівень впевненості прогнозу, що підвищує довіру до інструменту.
- Автономність та вартість: розроблений прототип може бути розгорнутий локально (on-premise), забезпечуючи конфіденційність даних пацієнтів, і не вимагає дорогих ліцензій чи оплати за кожен API-запит.

## **Висновки до розділу 4**

У четвертому розділі дисертаційної роботи здійснено практичну реалізацію, розгортання та комплексну верифікацію програмної системи для автоматизованої оцінки біологічного віку серця. Результати розділу демонструють успішний перехід від теоретичних моделей та алгоритмів, розроблених у попередніх частинах роботи, до створення повнофункціонального програмного продукту, готового до використання в умовах, наближених до клінічних.

В ході роботи було детально сформульовано та перевірено вимоги до апаратного та програмного забезпечення, що підтвердило можливість експлуатації системи на стандартному офісному обладнанні без необхідності використання дорогих спеціалізованих обчислювальних кластерів. Процес інсталяції та розгортання системи, реалізований через стандартні менеджери пакетів `npm` та `pip`, довів свою універсальність та простоту, що спрощує потенційну інтеграцію розробки в існуючі інформаційні системи медичних закладів.

Центральним досягненням розділу стала демонстрація функціональних можливостей розробленого клієнт-серверного комплексу. Реалізація графічного інтерфейсу користувача на базі бібліотеки `React` забезпечила високий рівень ергономіки та інтерактивності. Сценарій використання системи, що зводить процес діагностики до завантаження файлів та натискання однієї кнопки, мінімізує когнітивне навантаження на оператора. Особливу цінність становить інтегрована функція візуалізації ЕКГ-сигналу, яка дозволяє медичному спеціалісту

верифікувати якість вхідних даних та підвищує довіру до результатів роботи алгоритму.

Проведені фінальні обчислювальні експерименти підтвердили ефективність інтегрованої моделі Advanced CNN. Система продемонструвала точність класифікації на рівні 87.9% на тестовій вибірці, при цьому забезпечуючи високу швидкість обробки запиту. Порівняльний аналіз з аналогами виявив суттєві переваги розробленого рішення, зокрема повну робастність до дисбалансу класів, відсутність необхідності ручного розрахунку ознак та можливість автономного локального функціонування, що є критичним для збереження конфіденційності медичних даних.

Таким чином, результати розділу свідчать про повне виконання поставлених у технічному завданні вимог. Розроблений програмний прототип вирішує наукову задачу класифікації біологічного віку з високою точністю.

## 5 СТАРТАП-ПРОЄКТ

В умовах сучасної ринкової економіки, що характеризується високим рівнем конкуренції та стрімким технологічним розвитком, науково-технічна розробка не може вважатися завершеною без детального аналізу перспектив її комерціалізації. Одним із ключових елементів магістерської дисертації є розробка стартап-проєкту, який присвячений висвітленню маркетингових та управлінських аспектів впровадження власного інноваційного продукту. Створення проєкту є комплексним процесом, що включає підбір ідеї, формування унікальної ціннісної пропозиції, визначення ринкових перспектив та підготовку стратегії виходу на ринок.

Цей розділ є заключною складовою роботи і здійснюється у вигляді системної оцінки можливостей та формування заходів із ринкового впровадження інноваційних пропозицій магістранта. Метою розділу є формування інноваційного мислення та підприємницького підходу, а також здатності об'єктивно оцінювати ринкові перспективи для комерціалізації науково-технологічних досліджень, запропонованих у попередніх частинах дисертації.

### 5.1 Опис ідеї стартап-проєкту

В основу стартап-проєкту покладено ідею створення спеціалізованого хмарного сервісу (SaaS - Software as a Service) під робочою назвою «BioAge-ECG AI». Цей продукт позиціонується як інтелектуальна система підтримки прийняття рішень у сфері кардіології та превентивної медицини. Технологічним ядром продукту є розроблена у попередніх розділах глибока згортова нейронна мережа архітектури Advanced CNN, яка здатна виявляти приховані патерни старіння на електрокардіограмі.

Зміст ідеї полягає у наданні користувачам інструменту, що дозволяє миттєво оцінити біологічний вік серцево-судинної системи. Розбіжність між хронологічним

віком пацієнта та прогнозованим біологічним віком (т.зв. age gap) розглядається як інтегральний біомаркер стану здоров'я. Позитивна різниця (біологічний вік вищий за паспортний) сигналізує про наявність прихованих патологій, хронічного стресу або прискореного старіння організму, що вимагає негайного медичного втручання.

Проект спрямований на задоволення потреб кількох стратегічних груп клієнтів. Першим напрямком застосування є сектор приватних медичних закладів та клінік превентивної медицини. Для них система вирішує проблему часових витрат на аналіз ЕКГ та підвищує якість скринінгових обстежень. Другим напрямком є ринок страхування життя та здоров'я (InsurTech). Страхові компанії зацікавлені у точній оцінці ризиків клієнта для формування персоналізованих страхових тарифів, і біологічний вік є ідеальною метрикою для цього. Третім, перспективним напрямком, є індустрія Wellness та біохакінгу, де кінцеві користувачі потребують об'єктивних інструментів для моніторингу ефективності антивікової терапії та зміни способу життя.

Основні вигоди для користувача полягають у наступному:

- об'єктивність: виключення суб'єктивного фактору та лікарських помилок завдяки використанню штучного інтелекту, навченого на тисячах клінічних випадків;
- швидкість: скорочення часу отримання висновку з 15-20 хвилин (у випадку ручного аналізу) до 1-2 секунд;
- доступність: можливість проведення складної діагностики без використання дороговартісного обладнання, лише на основі стандартного цифрового запису ЕКГ.

Узагальнені дані щодо ідеї стартап-проекту представлені у таблиці 5.1.

Таблиця 5.1 - Загальні дані стартап-проекту

| Зміст ідеї (що пропонується)                                                                                                                                           | Напрямки застосування                                                                                                                                                                                                                                           | Вигоди для користувача                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Хмарний AI-сервіс для автоматизованого аналізу ЕКГ, що визначає біологічний вік серця та виявляє ознаки прискореного старіння за допомогою методів глибокого навчання. | <ul style="list-style-type: none"> <li>• Клінічна діагностика (скринінг у кардіології).</li> <li>• Страхова телематика (скоринг здоров'я).</li> <li>• Телемедицина та віддалений моніторинг.</li> <li>• Корпоративні програми здоров'я (Well-being).</li> </ul> | <ul style="list-style-type: none"> <li>• Для бізнесу (B2B): <ul style="list-style-type: none"> <li>○ Збільшення пропускної здатності клініки.</li> <li>○ Зниження собівартості діагностики.</li> <li>○ Автоматизація оцінки страхових ризиків.</li> </ul> </li> <li>• Для споживача (B2C): <ul style="list-style-type: none"> <li>○ Раннє виявлення проблем.</li> <li>○ Персоналізовані рекомендації.</li> </ul> </li> </ul> |

## 5.2 Технологічний аудит та аналіз конкурентного середовища

Наступним кроком аналізу є детальне вивчення конкурентного середовища та позиціонування власної розробки відносно існуючих аналогів. Ринок рішень для аналізу здоров'я серця є насиченим, проте більшість пропозицій мають суттєві обмеження.

Основними конкурентами проекту є:

- конкурент 1 (Традиційний): класична діагностика лікарем-кардіологом. Це "золотий стандарт", який викликає найбільшу довіру. Проте він має суттєві слабкі сторони: висока вартість послуги, необхідність фізичного візиту до клініки та залежність точності діагнозу від кваліфікації та втоми лікаря;
- конкурент 2 (мобільні додатки на базі HRV): Додатки типу Welltory, EliteHRV, які аналізують варіабельність серцевого ритму за допомогою камери смартфона або фітнес-трекерів. Їхньою сильною стороною є масовість та доступність. Однак слабкою стороною є низька клінічна точність: HRV є непрямим показником, який сильно залежить від ситуативного стресу і не відображає структурних змін міокарда, які аналізує пропонована система;
- конкурент 3 (корпоративні хмарні платформи): Рішення від Google Cloud Healthcare або IBM Watson Health. Вони потужні, але орієнтовані на великі

госпітальні мережі, мають надзвичайно високу вартість впровадження та складну інтеграцію, що робить їх недоступними для малих клінік.

Власний проєкт «BioAge-ECG AI» займає унікальну нішу, поєднуючи клінічну точність аналізу повного сигналу ЕКГ (як у лікаря) зі швидкістю та доступністю мобільних додатків. Порівняльний аналіз техніко-економічних характеристик наведено в таблиці 5.2.

Таблиця 5.2 - Визначення характеристик ідеї проєкту

| № з/п | Економічні та технічні характеристики | Конкурент 1 (Лікар)       | Конкурент 2 (HRV-додатки) | Конкурент 3 (Big Tech AI)       | Власний проєкт (BioAge-ECG AI) |
|-------|---------------------------------------|---------------------------|---------------------------|---------------------------------|--------------------------------|
| 1     | Швидкість отримання результату        | Слабка (хвилини/години)   | Сильна (миттєво)          | Нейтральна (залежить від черги) | Сильна (< 2 сек)               |
| 2     | Точність визначення біовіку           | Нейтральна (суб'єктивно)  | Слабка (низька кореляція) | Сильна (висока)                 | Сильна (> 87%)                 |
| 3     | Вартість однієї транзакції            | Слабка (висока)           | Сильна (низька)           | Слабка (дуже висока)            | Сильна (низька собівартість)   |
| 4     | Простота інтеграції (API)             | Відсутня                  | Нейтральна                | Слабка (складна)                | Сильна (REST API)              |
| 5     | Інтерпретованість (XAI)               | Сильна (пояснення лікаря) | Слабка (абстрактні бали)  | Слабка ("чорна скринька")       | Нейтральна (Saliency maps)     |

В межах запропонованого стартапу проведено аудит технологій, за допомогою яких реалізується ідея. Проєкт базується на використанні відкритого, але високопродуктивного стеку технологій:

- програмний продукт реалізовано як мікросервісну архітектуру. Бекенд побудовано на Node.js, що забезпечує асинхронну обробку тисяч запитів. Ядро штучного інтелекту написано на Python з використанням бібліотеки PyTorch, що є індустріальним стандартом для глибокого навчання. Клієнтська частина на React забезпечує кросплатформність (Web, Mobile Web);

- технологія передбачає значний потенціал для масштабування. По-перше, можливе впровадження Continuous Learning (безперервного навчання), коли модель автоматично донавчається на нових знеособлених даних клієнтів, постійно підвищуючи точність. По-друге, архітектуру можна адаптувати для Edge Computing (обчислень на пристрої), перенісши нейромережу безпосередньо на чіп смартфона або ЕКГ-холтера, що дозволить працювати без інтернету та підвищить приватність даних.

### 5.3 Аналіз ринкових можливостей та загроз

Для розробки ефективної маркетингової стратегії проведено моніторинг ринкового середовища, що дозволив виявити ключові можливості та загрози для проєкту.

Ринок цифрової охорони здоров'я (Digital Health) демонструє стабільне зростання. Ключовим драйвером є глобальний демографічний тренд на старіння населення ("срібна економіка") та зміна парадигми охорони здоров'я від лікування хвороб (reactive) до їх попередження (preventive). Зростає попит на технології Anti-aging та Longevity, ринок яких оцінюється у сотні мільярдів доларів. Споживачі готові платити за інструменти, що дозволяють продовжити активне життя. Крім того, після пандемії COVID-19 значно зросла довіра до телемедицини та дистанційної діагностики.

Основною перешкодою для реалізації проєкту є високі регуляторні бар'єри. Програмне забезпечення, що ставить діагнози, підпадає під категорію "Software as a Medical Device" (SaMD) і потребує сертифікації згідно з вимогами FDA (США), MDR (ЄС) або національних регуляторів. Цей процес є тривалим та коштовним. Другою загрозою є питання кібербезпеки та захисту персональних даних (GDPR, HIPAA). Будь-який витік медичних даних може призвести до фатальних репутаційних та фінансових втрат. Також існує ризик виходу на ринок великих технологічних гравців (Apple, Samsung), які можуть інтегрувати аналогічний функціонал безпосередньо в операційні системи своїх пристроїв.

## 5.4 Висновки щодо комерціалізації проєкту

Узагальнюючи проведений аналіз стартап-проєкту, можна зробити наступні висновки щодо його перспектив:

- проєкт має високий потенціал комерціалізації. Наявний платоспроможний попит з боку B2B-сектору (приватні клініки, страхові компанії), який прагне автоматизувати процеси та знизити витрати. Динаміка ринку є позитивною, з щорічним приростом сегменту AI у медицині понад 30%. Рентабельність роботи на ринку забезпечується моделлю SaaS (Software as a Service) з передплатою, що гарантує прогнозований потік доходів при низьких граничних витратах на масштабування;
- з огляду на потенційні групи клієнтів, проєкт є привабливим завдяки своїй універсальності. Бар'єри входження є середніми (технологічні бар'єри низькі, але регуляторні - високі). Конкурентоспроможність проєкту забезпечується унікальним алгоритмом Advanced CNN, який перевершує існуючі аналоги за точністю та глибиною аналізу;
- реалізація запропонованого варіанту є доцільною. Рекомендована стратегія виходу на ринок - "стратегія нішевого лідера". На першому етапі доцільно позиціонувати продукт як "інформаційно-довідкову систему для оздоровлення" (Wellness), що дозволить уникнути жорсткої медичної регуляції та швидко набрати базу користувачів. Паралельно необхідно розпочати процес клінічних випробувань для отримання медичної сертифікації, що відкриє доступ до великого ринку державних закупівель та госпітальних мереж;
- проєкт готовий до стадії MVP (Minimum Viable Product). Наступними кроками мають стати: патентування алгоритму, залучення посівних інвестицій для маркетингу та розширення команди розробників для створення мобільних версій.

## ВИСНОВКИ

У ході виконання магістерської дисертації було вирішено актуальну науково-практичну задачу визначення біологічного віку серця за даними електрокардіограми шляхом застосування та порівняльного аналізу сучасних методів машинного та глибокого навчання. Досягнення поставленої мети стало можливим завдяки комплексному поєднанню методів цифрової обробки сигналів, інтелектуального аналізу даних та сучасних технологій веб-розробки.

Основні наукові та практичні результати роботи полягають у наступному:

- у результаті аналізу предметної області та дослідження емпіричної бази обґрунтовано вибір репрезентативного набору даних Jena University Hospital, що містить понад 70 тисяч записів, класифікованих за 15 віковими групами. Виявлено критичну проблему дисбалансу класів, що визначило стратегію вибору алгоритмів (використання Focal Loss) та метрик оцінювання (F1-Macro, Recall) для програмного комплексу;
- розроблено два паралельних конвеєри підготовки даних, що реалізують різні методологічні підходи. Для класичних моделей створено скрипт `process_raw_data.py`, який забезпечує фільтрацію сигналів та екстракцію експертних ознак (статистичних показників та метрик варіабельності серцевого ритму). Для моделей глибокого навчання реалізовано модуль `wfdb_parser.py`, що виконує динамічне завантаження, Z-нормалізацію та аугментацію "сирих" сегментованих сигналів, дозволяючи нейронним мережам самостійно вивчати ієрархічні ознаки;
- розроблено та реалізовано п'ять архітектурних підходів до моделювання: класичні (логістична регресія, випадковий ліс), гібридний (Autoencoder + RNN) та наскрізні "end-to-end" (Bi-LSTM, Advanced CNN). Це дозволило провести всебічне порівняння ефективності методів, заснованих на ручній інженерії ознак, проти методів глибокого навчання представлень;
- проведено серію обчислювальних експериментів, які підтвердили

обмеженість класичних методів (максимальна точність 49%) через втрату морфологічної інформації при агрегації ознак. Встановлено, що найкращі результати демонструють наскрізні архітектури. Лідером стала розроблена модель Advanced CNN (на базі ResNet з механізмом уваги), яка досягла точності понад 87% та високої робастності на незбалансованих даних. Чекпоінт цієї моделі збережено у форматі .pth для подальшого використання;

- розроблено клієнт-серверний програмний прототип, що демонструє практичне застосування обраної моделі. Система побудована на сучасній триланковій архітектурі: клієнтська частина (React) забезпечує інтерфейс користувача, серверна частина (Node.js/Express) виконує роль диспетчера, а ізольований Python-скрипт здійснює інференс моделі. Розроблений додаток готовий до використання як система підтримки прийняття лікарських рішень.

Таким чином, усі завдання дослідження виконано в повному обсязі. Отримані результати підтверджують, що застосування глибоких згорткових мереж дозволяє автоматизувати процес визначення біологічного віку серця з точністю, що перевищує можливості традиційних методів. Розроблений програмний комплекс є готовим до пілотного впровадження як система підтримки прийняття лікарських рішень у закладах превентивної медицини та кардіології.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Belsky D. W. et al. Quantification of biological aging in young adults. *Proceedings of the National Academy of Sciences*. 2015. Vol. 112, No. 30. P. E4104-E4110.
2. Hannun A. Y. et al. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*. 2019. Vol. 25, No. 1. P. 65-69.
3. Антонкін Д. О., Полягушко Л. Г., Сліпченко В. Г. Методи машинного навчання для оцінки біологічного віку серця на основі ЕКГ // Інформаційні технології в освіті, техніці та промисловості : матеріали Всеукр. наук.-практ. конф., (Івано-Франківськ, 09 жовт. 2025 р.). – Івано-Франківськ : ІФНТУНГ, 2025. – С. 184–185. – URL: <https://nung.edu.ua/department/institut-informaciy-nikh-tekhnologiy/vidannya>. DOI: <https://doi.org/10.5281/zenodo.17295363>.
4. Ribeiro A. H. et al. Automatic diagnosis of the 12-lead ECG using a deep neural network. *Nature Communications*. 2020. Vol. 11, No. 1. P. 1760.
5. Lima E. M. et al. Deep neural network-estimated electrocardiographic age as a mortality predictor. *Nature Communications*. 2021. Vol. 12, No. 1. P. 5117.
6. Luan Y. et al. Machine Learning-Based Age Prediction Using 12-Lead Electrocardiogram. *Frontiers in Physiology*. 2021. Vol. 12. Art. 736961.
7. Schumann A., Wessel N., Voss A. Autonomic function regulating blood pressure and cardiac rhythm progressively declines with increasing age (version 1.0.0). *PhysioNet*. URL: <https://physionet.org/content/autonomic-aging-cardiovascular/1.0.0/> (дата звернення: 10.05.2025).
8. Goldberger A. L. et al. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation*. 2000. Vol. 101, No. 23. P. e215-e220.
9. Heart rate variability: standards of measurement, physiological interpretation, and clinical use. *Circulation*. 1996. Vol. 93, No. 5. P. 1043-1065.
10. Pan J., Tompkins W. J. A Real-Time QRS Detection Algorithm. *IEEE*

Transactions on Biomedical Engineering. 1985. Vol. BME-32, No. 3. P. 230-236.

11. Welch P. The use of Fast Fourier Transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. IEEE Transactions on Audio and Electroacoustics. 1967. Vol. 15, No. 2. P. 70-73.

12. Kiranyaz S. et al. 1D Convolutional Neural Networks and Applications: A Survey. Mechanical Systems and Signal Processing. 2021. Vol. 151. Art. 107398.

13. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition (CVPR). 2016. P. 770-778.

14. Vaswani A. et al. Attention Is All You Need. Advances in Neural Information Processing Systems. 2017. Vol. 30. P. 5998-6008.

15. Hochreiter S., Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997. Vol. 9, No. 8. P. 1735-1780.

16. Schuster M., Paliwal K. K. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing. 1997. Vol. 45, No. 11. P. 2673-2681.

17. Lin T.-Y. et al. Focal Loss for Dense Object Detection. Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017. P. 2980-2988.

18. Hinton G. E., Salakhutdinov R. R. Reducing the Dimensionality of Data with Neural Networks. Science. 2006. Vol. 313, No. 5786. P. 504-507.

19. Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980. 2014.

20. Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning. 2015. P. 448-456.

21. Srivastava N. et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014. Vol. 15. P. 1929-1958.

22. Breiman L. Random Forests. Machine Learning. 2001. Vol. 45, No. 1. P. 5-32.

23. Chawla N. V. et al. SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research. 2002. Vol. 16. P. 321-357.

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

25. Van Rossum G., Drake F. L. Python 3 Reference Manual. Scotts Valley, CA : CreateSpace, 2009. 242 p.
26. Harris C. R. et al. Array programming with NumPy. Nature. 2020. Vol. 585. P. 357-362.
27. Virtanen P. et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods. 2020. Vol. 17. P. 261-272.
28. McKinney W. Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference. 2010. P. 51-56.
29. Pedregosa F. et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011. Vol. 12. P. 2825-2830.
30. Paszke A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems. 2019. Vol. 32. P. 8024-8035.
31. Hunter J. D. Matplotlib: A 2D graphics environment. Computing in Science & Engineering. 2007. Vol. 9, No. 3. P. 90-95.
32. Tilkov S., Vinoski S. Node.js: Using JavaScript to build high-performance network programs. IEEE Internet Computing. 2010. Vol. 14, No. 6. P. 80-83.
33. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures : Ph.D. dissertation. Irvine : University of California, 2000.
34. React - A JavaScript library for building user interfaces. URL: <https://reactjs.org/> (дата звернення: 15.05.2025).
35. Documentation | Express. URL: <https://expressjs.com/> (дата звернення: 15.05.2025).
36. TypeScript: Typed JavaScript at Any Scale. URL: <https://www.typescriptlang.org/> (дата звернення: 15.05.2025).
37. Vite | Next Generation Frontend Tooling. URL: <https://vitejs.dev/> (дата звернення: 15.05.2025).
38. Acharya U. R. et al. A deep convolutional neural network model to classify heartbeats. Computers in Biology and Medicine. 2017. Vol. 89. P. 389-396.
39. Chollet F. Deep Learning with Python. Shelter Island : Manning Publications, 2017. 384 p.

## ДОДАТОК А

Методи машинного навчання для визначення біологічного віку серця на основі  
ЕКГ

Тези

УКР.НТУУ“КПІ ім. Ігоря Сікорського”\_ТР-42мп\_???

Аркушів 3

2025

## МЕТОДИ МАШИННОГО НАВЧАННЯ ДЛЯ ОЦІНКИ БІОЛОГІЧНОГО ВІКУ СЕРЦЯ НА ОСНОВІ ЕКГ

*Антонкін Д. О., Полягушко Л. Г., Сліпченко В.Г*

*Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, просп. Берестейський, 37, 03056, Україна*

*e-mail: [antonkin.dmytro@lil.kpi.ua](mailto:antonkin.dmytro@lil.kpi.ua)*

Біологічний вік серця - це інтегральний показник, що відображає структурно-функціональний стан серцево-судинної системи та може суттєво відрізнятися від хронологічного віку. Його прискорене «старіння» пов'язане з артеріальною гіпертензією, метаболічними порушеннями, низькою фізичною активністю, хронічними захворюваннями.

Електрокардіограма (ЕКГ) є одним із найбільш доступних інструментів кардіодіагностики. Використання методів машинного навчання дозволяє перейти від традиційного виявлення патологій до прогнозування біологічного віку серця, що має велике значення для превентивної медицини.

Сучасні роботи (Harvard Medical School, Mayo Clinic, European Heart Journal) [1] показують, що параметри ЕКГ - варіабельність ритму, інтервали PQ, QT, тривалість QRS, амплітудні характеристики та спектральні показники - корелюють із функціональним віком серця.

Глибинні моделі, зокрема CNN, успішно застосовуються для автоматичної класифікації аритмій та стратифікації ризику [2], але їхнє використання для оцінки саме біологічного віку серця перебуває на етапі розвитку й потребує подальшої валідації.

Метою дослідження є узагальнення методів машинного навчання та визначення перспектив їхнього застосування для прогнозування біологічного віку серця на основі ЕКГ.

Біологічний вік серця відображає стан міокарду, провідної системи та коронарного кровообігу. На нього впливають генетичні, гемодинамічні, метаболічні фактори та вегетативна регуляція.

ЕКГ є доступним джерелом часових, амплітудних і спектральних параметрів, які після фільтрації та сегментації добре підходять для машинного аналізу.

Методи машинного навчання дозволяють виявляти приховані закономірності у даних. Для прогнозування біологічного віку серця застосовують:

- CNN (згорткові нейронні мережі) - для автоматичної екстракції ознак з ЕКГ-сигналів;
- RNN/LSTM (рекурентні нейронні мережі) - для аналізу часових залежностей у серцевих ритмах;
- Random Forest, Gradient Boosting - для побудови предиктивних моделей на основі витягнутих ознак;
- Autoencoder-и - для зменшення розмірності та виявлення латентних репрезентацій сигналів.

Глибинні моделі (CNN, LSTM, Autoencoder-и) забезпечують високу точність,

але вимагають великих вибірок і ресурсів [3].

**Таблиця 1 - Порівняння методів машинного навчання**

| Метод                            | Ідея                                    | Переваги                                                                   | Недоліки                                               | Використання                                                    |
|----------------------------------|-----------------------------------------|----------------------------------------------------------------------------|--------------------------------------------------------|-----------------------------------------------------------------|
| CNN                              | Автоматичне виділення ознак із ЕКГ      | Без ручних ознак<br>Висока точність<br>Працюють із багатоканальними даними | Потрібно багато даних<br>Високі обчислення             | Виявлення морфологічних змін для оцінки віку серця              |
| RNN / LSTM                       | Аналіз часових залежностей              | Моделюють довгі залежності<br>Добрі для послідовних даних                  | Важко вчити на довгих рядах<br>Повільніші за CNN       | Аналіз варіабельності ритму для вікових прогнозів               |
| Random Forest, Gradient Boosting | Ансамбль дерев рішень на ручних ознаках | Стійкі до шуму<br>Інтерпретовані<br>Ефективні на малих даних               | Залежать від якості ознак<br>Не знаходять нові патерни | Прогноз віку за традиційними ЕКГ-показниками                    |
| Autoencoder-и                    | Зменшення розмірності, латентні ознаки  | Виявляють приховані ознаки<br>Зменшують шум<br>Комбінуються з іншими       | Менш інтерпретовані<br>Ризик перенавчання              | Створення компактних репрезентацій для подальшого прогнозування |

У майбутньому комбінація різних методів дасть змогу створити персоналізовані моделі оцінки віку серця та прогнозу розвитку патологій у реальному часі.

Отже, біологічний вік серця є інформативним предиктором серцево-судинних подій. ЕКГ забезпечує доступні багатовимірні дані для його оцінки. Використання CNN, LSTM, Random Forest, Gradient Boosting та Autoencoder-ів дає можливість створювати точні прогностичні моделі, а їх комбінація підвищує ефективність результатів.

### Літературні джерела

1. Abstract 12758: dissecting heart age using cardiac magnetic resonance videos, electrocardiograms, biobanks, and deep learning / A. Le Gallec та ін. *Circulation*. 2021. Т. 144, Suppl\_1. URL: [https://doi.org/10.1161/circ.144.suppl\\_1.12758](https://doi.org/10.1161/circ.144.suppl_1.12758) (дата звернення: 03.10.2025).
2. Deep neural network-estimated electrocardiographic age as a mortality predictor / E. M. Lima та ін. *Nature communications*. 2021. Т. 12, № 1. URL: <https://doi.org/10.1038/s41467-021-25351-7> (дата звернення: 03.10.2025).
3. Artificial intelligence-estimated biological heart age using 12 lead electrocardiogram predicts mortality and cardiovascular outcomes / Y. Baek та ін. *Europace*. 2023. Т. 25, Supplement\_1. URL: <https://doi.org/10.1093/europace/euad122.614> (дата звернення: 03.10.2025).



ІТЕТІ20250049

# СЕРТИФІКАТ

## УЧАСНИКА КОНФЕРЕНЦІЇ

0,15 кредити ECTS

### АНТОНКІН ДМИТРО ОЛЕКСАНДРОВИЧ



Всеукраїнська Інтернет-конференція молодих учених і студентів  
«Інформаційні технології в освіті, техніці та промисловості»



09 жовтня 2025

Голова організаційного комітету



Сергій ЗІКРАТИЙ

Івано-Франківськ

