НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

«До захисту допущено»
В.О. завідувача кафедри
_____ О.Л. Тимощук

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки 6.050101 "Комп'ютерні науки"

на тему: «Система підтримки прийняття рішень розпізнавання кардіограм методами штучного інтелекту»

Виконав:
студент IV курсу, групи КА-55
Кравченко Влада Андріївна
_____

Керівник:
професор кафедри ММСА,
д.т.н. Данилов В. Я. _____

Консультант з економічного розділу:
доцент, к.е.н. Шевчук О. А. _____

Консультант з нормоконтролю:
доцент, к.т.н. Коваленко А.Є. _____
Рецензент:
к.т.н. Мурга М.О _____

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

Київ – 2019 року

РЕФЕРАТ

Дипломна робота: 92 с., 31 рис., 15 табл., 4 додатки, 25 джерел.

ЕЛЕКТРОКАРДІОГРАМА, ЦИФРОВА ОБРОБКА СИГНАЛІВ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, МАШИННЕ НАВЧАННЯ, ГЛИБИННІ НЕЙРОННІ МЕРЕЖІ.

Об’єкт дослідження – набір даних каріограм пацієнтів з захворюваннями серця.

Мета роботи – розробка системи класифікації сигналів кардіограм для діагностики захворювань серця.

Предмет дослідження – засоби попередньої обробки цифрових сигналів та методи класифікації на основі згорткових нейронних мереж.

Система, представлена в даній роботі, може бути застосована у сфері кардіології для діагностування серцево-судинних захворювань людини.

ABSTRACT

The theme: «Support system for making decisions of recognition of cardiograms by methods of artificial intelligence».

Diploma: 92 p., 31 fig, 15 tabl., 4 appendixes, 25 sources.

ELECTROCARDIOGRAM, DIGITAL SIGNAL PROCESSING, CONVOLUTIONAL NEURAL NETWORKS, MACHINE LEARNING, DEEP NEURAL NETWORKS

The object of research – cardiogram dataset of patients with heart diseases MIT BIH Arrhythmia Database.

The purpose of the work - to develop a classification system for cardiograms signals to diagnose heart diseases.

The subject of research - ways of preliminary processing of digital signals and methods of classification on the basis of convolutional neural networks.

The system presented in this paper can be applied in the field of cardiology for the diagnosis of cardiovascular diseases of a person.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

ВП – вейвлет-перетворення

ЕКГ - електрокардіограма

ЗНМ – згорткова нейронна мережа

МОВ – метод опорних векторів

СППР – система підтримки прийняття рішень

СУБД – система управління базами даних

ЦОС – цифрова обробка сигналів

ШІ – штучний інтелект

ВСТУП

Серцево-судинні захворювання є основними причинами смертності у світі. На сайті Всесвітньої організації охорони здоров'я повідомляють, що у 2016 році від серцево-судинних захворювань у світі померло 17,9 млн осіб, це 31% усіх випадків смертей у світі. До переліку серцево-судинних захворювань входять інфаркт, інсульт, гіпертонія, хвороба периферичних артерій, ревматична вада серця, вроджена вада серця і серцева недостатність. За повідомленням прес-служби МОЗ України від 19 листопада 2018 року в Україні 67% смертей трапляються від серцево-судинних захворювань.

Професійний лікар-кардіолог може визначити наявність хвороби серця спираючись на розмову з пацієнтом та огляд. Проте для підтвердження діагнозу, встановлення важкості захворювання, а також для корекції лікування використовують спеціальні діагностичні дослідження.

Електрокардіограма є найбільш простим і разом з цим дуже інформативним методом обслідування роботи серця. ЕКГ – це запис електричної активності серця, аналіз якого дозволяє виявити різні види патології серця. Проте серед спеціалістів в галузі кардіології існують різні підходи щодо трактування показників ЕКГ, а також достатньо високим є процент помилок, пов’язаних з некоректною інтерпретацією ділянок сигналів ЕКГ, які містять артефакти (дефекти запису). Отже, існує потреба у нових методах обробки ЕКГ.

Для розв’язку даної проблеми було запропоновано цифрову обробку ЕКГ. Оскільки кардіограма - це електричний сигнал, набір значень, його можна представити у вигляді функції і, аналізуючи її, робити висновки про стан пацієнта і патології його серцево-судинної системи. Аналізу підлягають такі параметри ЕКГ, як міжзубцеві інтервали, частоти та безліч інших, що є достатньо трудо- та часомістким процесом для виконання людиною. Тому такі функції, як аналіз інтервалів та виявлення відхилень було запропоновано виконувати з використанням спеціальних програмних засобів, що значно

пришвидшують процес обробки ЕКГ для подальшого встановлення діагнозу лікарем-кардіологом.

Метою даної роботи є створення системи розпізнавання сигналів ЕКГ та їх аналізу для встановлення наявності або відсутності захворювань серця, а також класифікації таких форм захворювання серцево-судинної системи, як аритмії, у разі виявлення відхилень в роботі серця.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз вимог до системи розпізнавання ЕКГ. Постановка задачі

Метою дипломної роботи є розробка системи підтримки прийняття рішень розпізнавання кардіограм. Програма має виконувати попередню обробку сигналу ЕКГ та, основуючись на параметрах кардіограми, класифікувати захворювання серця у разі іхньої наявності.

Система, представлена в даній роботі, може бути застосована у сфері кардіології для діагностування таких серцево-судинних захворювань людини, як аритмії. Загалом аритмія характеризується порушенням ритму серця, що можна встановити завдяки аналізу параметрів сигналу ЕКГ. Програмна обробка надає значну перевагу у часі у порівнянні з ручною, що є безперечно дуже важливим фактором у питанні встановлення захворювань.

Необхідно розв’язати наступні задачі для досягнення мети:

- Вивчити існуючі підходи та системи для вирішення задачі обробки часових сигналів;
- Дослідити фізіологію серця та її зв’язок с електрокардіограмами;
- Дослідити методи обробки часових сигналів;
- Дослідити методи прийняття рішень;
- Реалізувати обробку сигналів кардіограм;
- Реалізувати блок прийняття рішень;
- Провести навчання системи;
- Проаналізувати одержані результати навчання;
- Розробити інтерфейс програми;
- Виконати тестування кінцевого програмного продукту.

1.2 Аналіз засобів обробки ЕКГ

Існують різні методи аналізу сигналу ЕКГ. Серед основних можна виділити наступні: безпосередній аналіз форми сигналу (визначення PQRST – комплексу, визначення RR-інтервалів тощо), використання нейронних мереж, застосування функціональних перетворень, що є чутливими до змін локальних властивостей сигналу (швидке перетворення Фур’є, вейвлет-перетворення).

Аналіз ЕКГ з використанням відповідних програмних засобів є достатньо поширеним способом на сьогоднішній день. Його можна розділити на три основних етапи: цифрова обробка сигналу, виділення ознак, аналіз одержаних параметрів.

Перший етап включає у себе переведення сигналу ЕКГ у цифрову форму, фільтрацію (видалення шумів та артефактів) і нормування сигналу. Шумом прийнято вважати високочастотні компоненти кардіосигналу, його видалення призводить до згладжування. Другий - розпізнавання характерних ділянок ЕКГ (зубці, сегменти, QRS-комплекс). Останній етап являє собою аналіз параметрів і взаємозв'язку розпізнаних ділянок і формування підсумкового висновку.

Застосування автоматичних алгоритмів розпізнавання дозволяють отримувати якісні результати.

У таблиці 1.1 наведено список програм для аналізу та інтерпретації сигналу ЕКГ від провідних світових компаній-виробників електрокардіографів з зазначенням їх ключових можливостей.

Таблиця 1.1 - Програми для аналізу ЕКГ

Назва	Виробник	Можливості
Heart Rate Variability Software	Schiller (Швейцарія)	Аналіз варіабельності серцевого ритму (ЕКГ)
Marquette 12SL	GE Healthcare (США)	Аналіз ЕКГ, унікальні критерії оцінки головних сегментів ЕКГ, автоматичне визначення аритмій
Cardiosoft	GE Healthcare (США)	Вимірювання інтервалів ЕКГ, аналіз основних зубців, аналіз аритмій
FP-804	Fukuda (Япония)	Основні виміри: ЧСС, RR, PR, QRS, час QT, QTc, електрична ось, SV, RV5(6); 120 типів кодів інтерпретації, 130 типів кодів Мінесоти

Основні проблеми аналізу ЕКГ:

- відсутність обліку деяких особливостей і відомостей, які не містяться в ЕКГ, але враховуються лікарем при інтерпретації сигналу;
- діагностичні помилки;

Є дані, що результати автоматичного аналізу ЕКГ програмою в 5-20% випадків не збігаються з лікарськими висновками. Система може робити помилки або не виявляти патології в сигналі.

- незручний інтерфейс.

1.3 Будова та функціонування серця, зв’язок з ЕКГ

На початку XX століття нідерландський фізіолог Ейнтховен винайшов новий спосіб діагностики роботи серця - електрокардіографію (ЕКГ). ЕКГ - це запис коливань різниці потенціалів, що виникають на поверхні збудливої тканини або в оточуючому серце провідному середовищі при поширенні хвилі збудження по серцю. Запис сигналу здійснюється з допомогою електрокардіографа і електродів, прикріплених до тіла пацієнта і включає в себе послідовність кардіоциклу.

З технічної точки зору, серце - це чотирикамерний насос з двома передсердями для збору крові і двома шлуночками для виштовхування крові. Скорочення шлуночків серця фіксуються на ЕКГ як відхилення від ізолінії. Ці відхилення називаються зубцями і позначаються літерами P, Q, R, S, T.

На рисунку 1.1 зображено порядок формування зубців ЕКГ внаслідок серцевих скорочень.

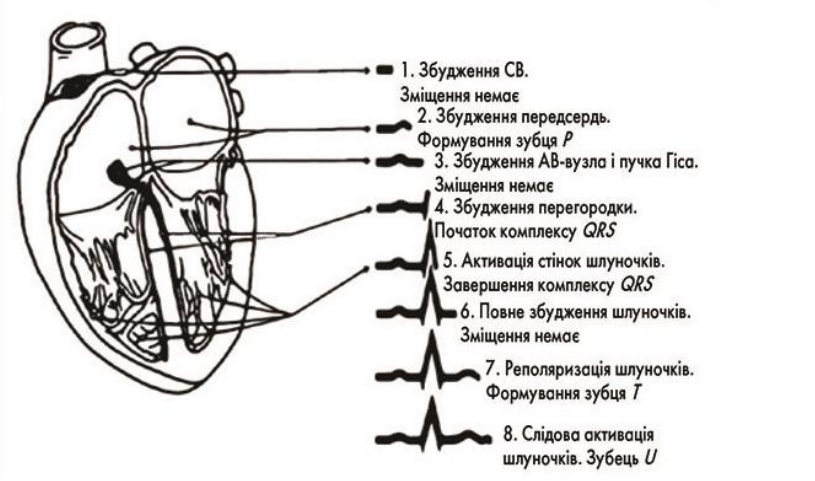


Рисунок 1.1 - Схема розповсюдження збудження по провідній системі серця та схематичне зображення кривої ЕКГ

До серця струм потрапляє через синусовий вузол, саме тому серцевий ритм у нормі називається синусовим. Вузол знаходиться у правому передсерді, через що першим збуджується саме воно, надалі імпульс переходить до лівого передсердя. Сигнали від збудження кожного передсердя фіксуються і внаслідок накладання вони утворюють Р зубець.

Три піки, Q, R та S разом утворюють QRS-комплекс, який відповідає за скорочення шлуночків. Першою збуджується міжшлуночкова перегородка, утворюючи зубець Q. Вектор збудження при цьому спрямований убік від реєструючого електрода, що дає на ЕКГ від'ємне значення. Далі збуджується верхівка серця. Вектор збудження спрямований прямо до датчика, що дає на графіку найбільший за амплітудою пік R. Останньою збуджується основа серця, вектор збудження спрямований проти датчика, значення негативне. На графіку утворюється пік S.

Після збудження шлуночків починається процес реполяризації, тобто відновлення серця перед початком нового циклу. На кардіограмі це відповідає інтервалу ST (рис. 1.2).

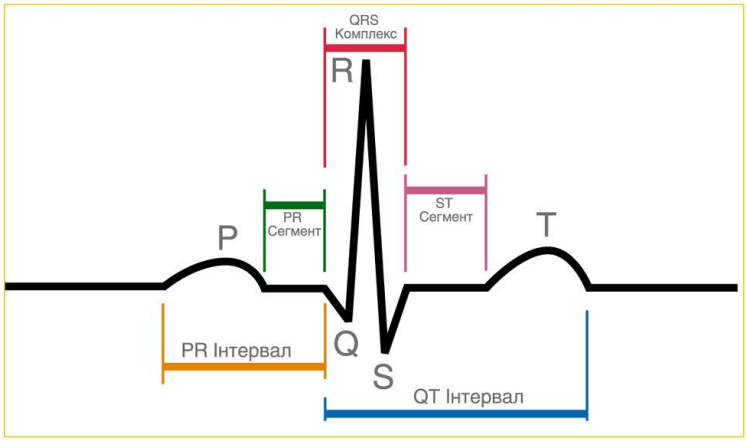


Рисунок 1.2 - Елементи ЕКГ в нормі

ЕКГ може виявити незвичайні закономірності і аномальні конфігурації хвиль і комплексів сигналу, які можуть надати лікарю інформацію про етіологію захворювання. У багатьох випадках діагноз може також істотно залежати від вимірювання певних періодів на ЕКГ, які визначаються розташуванням певних точок (піки, набори і зміщення P, QRS, T і U-хвиль). Анотація цих точок на ЕКГ і виявлення конкретних конфігурацій хвиль може виконуватися вручну кардіологом, або ці завдання можуть грунтуватися на комп'ютерних методах. Точність ручного розмежування залежить від навчання людини, і у різних лікарів можуть вийти різні результати. Оскільки

24-годинний Холтер містить в середньому 100 000 ударів - робота вручну з такими даними є складним завданням.

Одна з найпоширеніших патологій, що може бути виявлена на ЕКГ – аритмія. Це зміна основних електрофізіологічних властивостей серця (провідність, збудливість, автоматизм), яка веде до порушення роботи серця в цілому або його відділів.

Нормальною вважається частота скорочень 60-90 в хвилину (тобто поява зубців Р) і рівні часові інтервали між ними. У разі, якщо інтервали варіюються менше, ніж на 0,12 с, говорять про неправильний синусовий ритм. Якщо різниця між найбільшим і найменшим інтервалом перевищує 0,12 с, то мова йде про аритмію (рис. 1.3).

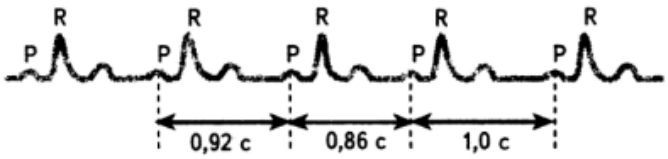


Рисунок 1.3 - Крива ЕКГ хворого на аритмію

За Європейським стандартом при підозрі на аритмію, лікар повинен встановити порушення ритму, визначити його характер, причину і визначити подальше лікування. Для цього, лікар використовує методи фізикальної діагностики, електрокардіограму, добове моніторування ЕКГ (по Холтеру) і ін.

1.5. Висновки до розділу

В даному розділі було проведено аналіз вимог до створюваної системи, розглянуто існуючі програмні засоби, за допомогою яких проводиться діагностика захворювань серця на основі ЕКГ, проведено їх аналіз та порівняння. Також було розглянуто предметну область задачі, а саме анатомію та фізіологію серця людини і її зв’язок з електрокардіограмами.

РОЗДІЛ 2 МАТЕМАТИЧНІ ОСНОВИ РОБОТИ

2.1 Дослідження існуючих методів обробки часових сигналів

Дані ЕКГ є неперервним сигналом. Для обробки на комп'ютері потрібно перевести сигнал в цифрову форму. Один із способів зробити це - рівномірно по часу виміряти значення сигналу на певному проміжку часу і ввести отримані значення амплітуд в комп'ютер. Якщо робити вимірювання досить часто, то за отриманим дискретним сигналом можна буде досить точно відновити вигляд вихідного безперервного сигналу (рис 2.1).

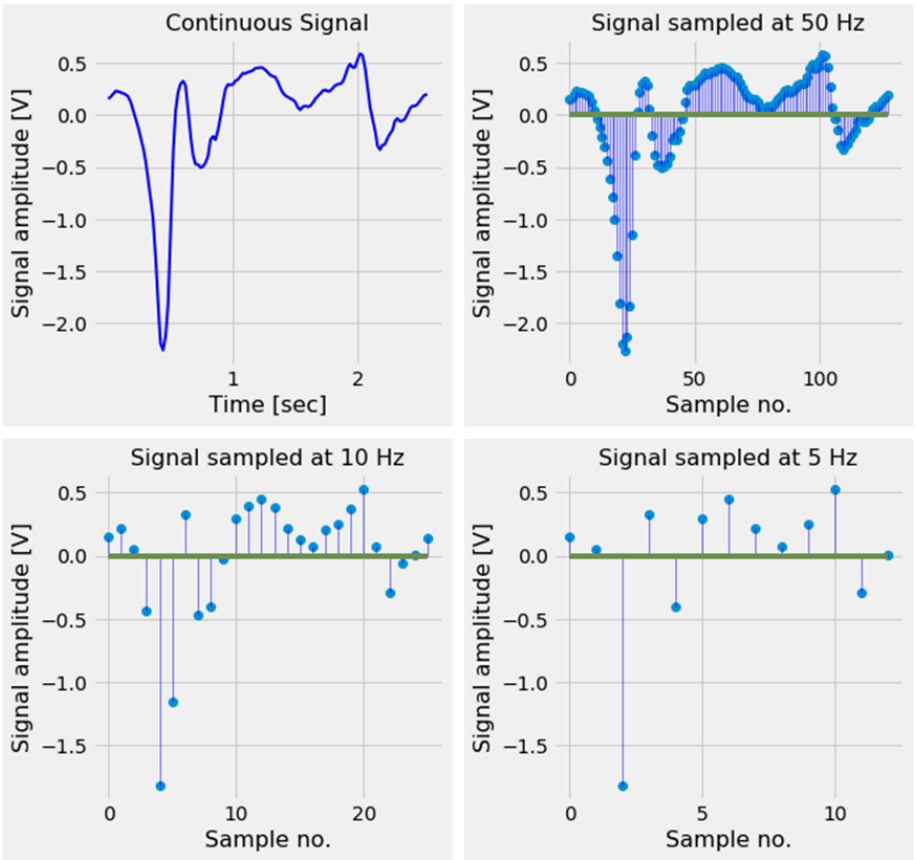


Рисунок 2.1 - Безперервний сигнал обробляють на різних частотах. Коли сигнал обробляється при занадто низькій частоті дискретизації, цифровий сигнал більше не можна відновити до вигляду початкового аналогового сигналу.

Процес виміру величини сигналу через рівні проміжки часу називається рівномірною (за часом) дискретизацією. Багато пристроїв для введення даних

в комп'ютер здійснюють дискретизацію. Наприклад, звукова карта дискретизує сигнал з мікрофона, сканер дискретизує сигнал, що надходить з фотоелемента. В результаті дискретизації безперервний (аналоговий) сигнал переводиться в послідовність чисел. Пристрій, що виконує цей процес, називається аналогово-цифровим перетворювачем (АЦП, analogue-to-digital converter, ADC). Частота, з якою АЦП виробляє виміри аналогового сигналу і видає його цифрові значення, називається частотою дискретизації.

2.1.1. Перетворення Фур’є

Будь-яка безперервна функція може бути розкладена в інтеграл Фур'є. Сенс цього розкладання полягає в тому, що функція представляється у вигляді суми «ряду» синусоїд з різними амплітудами. Коефіцієнти (амплітуди) при синусоїдах називаються спектром функції. У відносно гладких функцій спектр швидко убуває (з ростом частоти коефіцієнти швидко прямують до нуля). Для відносно «ламаних» функцій спектр убуває повільно, тому що для подання розривів і «зламів» функції потрібні синусоїди з великими частотами.

Перетворення Фур'є (Fourier transform) - це розкладання функцій на синусоїди та косинусоїди. Існує кілька видів перетворення Фур'є.

1. Неперіодичний безперервний сигнал можна розкласти в інтеграл Фур'є.

2. Періодичний безперервний сигнал можна розкласти в нескінченний ряд Фур'є.

3. Неперіодичний дискретний сигнал можна розкласти в інтеграл Фур'є.

4. Періодичний дискретний сигнал можна розкласти в кінцевий ряд Фур'є.

Комп'ютер здатний працювати тільки з обмеженим обсягом даних, отже, реально він здатний обчислювати тільки останній вид перетворення Фур'є. Розглянемо його докладніше.

Нехай дискретний сигнал x [n] має період N точок. В цьому випадку його можна представити у вигляді кінцевого ряду (тобто лінійної комбінації) дискретних синусоїд:

$$x[n] = \sum_{k=0}^{N/2} C_k \cos \frac{2\pi k(n+\varphi_k)}{N} \quad (2.1)$$

Еквівалентний запис (кожен косинус розкладаємо на синус і косинус, але тепер - без фази):

$$x[n] = \sum_{k=0}^{N/2} A_k \cos \frac{2\pi kn}{N} + \sum_{k=0}^{N/2} B_k \sin \frac{2\pi kn}{N} \quad (2.2)$$

Базисні синусоїди мають кратні частоти. Перший член ряду (k = 0) – це константа, звана постійною складовою (DC offset) сигналу. Найперша синусоїда (k = 1) має таку частоту, що її період збігається з періодом самого вихідного сигналу. Найбільш високочастотна складова (k = N / 2) має таку частоту, що її період дорівнює двом відлікам. Коефіцієнти $A_к$ і $B_к$ називаються спектром сигналу (spectrum). Вони показують амплітуди синусоїд, з яких складається сигнал. Крок за частотою між двома сусідніми синусоїдами з розкладання Фур'є називається частотним дозволом спектра.

Для кожного сигналу можна однозначно визначити коефіцієнти $A_к$ і $B_к$. Знаючи ці коефіцієнти, можна однозначно відновити вихідний сигнал, обчисливши суму ряду Фур'є в кожній точці. Розкладання сигналу на синусоїди (тобто отримання коефіцієнтів) називається прямим перетворенням Фур'є. Зворотний процес - синтез сигналу по синусоїді - називається зворотним перетворенням Фур'є (inverse Fourier transform).

Спектральне подання сигналу повністю еквівалентне самому сигналу. Між ними можна переміщатися, використовуючи пряме і зворотне перетворення Фур'є.

Розкладання сигналів в ряди Фур'є мають ряд недоліків, які привели до появи швидкого перетворення Фур'є і стимулювали розвиток вейвлет-перетворення. Основні з них:

- обмежена інформативність аналізу сигналів в зв'язку з тим, що в частотній області немає чіткого виявлення особливостей сигналів по всьому частотному діапазону спектра;
- перепади сигналів з нескінченною крутизною неможливо відобразити за допомогою гармонійних базисних функцій розкладання, тому що для цього потрібна нескінченна кількість членів ряду;
- перетворення Фур'є не дає уявлення про локальні особливості сигналу, а відображає глобальні відомості про частоти. Немає можливості аналізувати частотні характеристики сигналу в певний момент часу.

Наприклад, перетворення Фур'є не відрізняє сигнал з сумою двох синусоїд від сигналу, де синусоїди йдуть послідовно одна за одною з тими ж частотами. Використовуючи перетворення Фур'є, можна працювати з даними тільки в одній області (часовій або частотній). Одночасно отримати інформацію, використовуючи класичне перетворення Фур'є не можна (немає можливості отримати інформацію про те, в який момент часу які частоти присутні в сигналі).

Обчислення перетворень Фур'є вимагає дуже великої кількості множень (близько N^2) і обчислень синусів. Існує спосіб виконати ці перетворення значно швидше: приблизно за $N * log_2 N$ операцій множення. Цей спосіб називається швидким перетворенням Фур'є (ШПФ, FFT, fast Fourier transform). Він заснований на тому, що серед множників (синусів) є багато значень, що повторюються (в силу періодичності синуса). Алгоритм ШПФ групує доданки з однаковими множниками, значно скорочуючи число множень. В результаті швидкодія ШПФ може в сотні разів перевершувати швидкодію стандартного алгоритму (в залежності від N). При цьому слід підкреслити, що алгоритм

ШПФ є точним, він навіть точніший від стандартного, тому що скорочуючи число операцій, він призводить до менших помилок округлення.

2.1.2 Вейвлет аналіз

Вейвлет (англ. Wavelet) - це «маленька хвиля», енергія якої кінцева і локально сконцентрована в часі. Вейвлети використовуються для представлення сигналів аналогічно тому, як ряди Фур'є використовують синусоїду для розкладання сигналу (рис. 2.2). При вейвлет-розкладанні всі функції виходять з однієї шляхом її зрушень і розтягувань по осі часу.

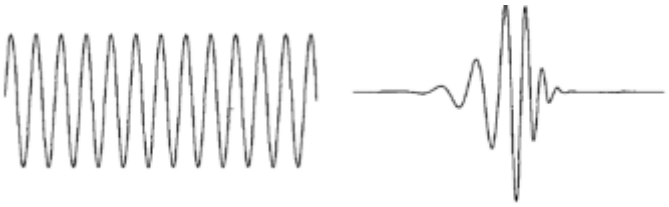


Рисунок 2.2 - Синусоїда та вейвлет

В загальному вигляді вейвлет-перетворення функції f(t) визначається виразами:

$$W_f(a,b) = \int_{-\infty}^{\infty} f(t)\psi^*_{a,b}(t)dt \qquad (2.3)$$

де $\psi^*_{a,b}(t)$ - комплексно-спряжені функції сімейства вейвлет-функцій $\psi_{a,b}(t)$виведених з материнського вейвлета $\psi\ (t)$:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{2}}\psi(\frac{t-b}{a}) \qquad (2.4)$$

де a – параметр розтягування (масштабу);

b – параметр зсуву.

Параметр масштабу a вейвлет-перетворення можна порівняти з частотним параметром перетворення Фур'є.

Оскільки вейвлет локалізується в часі, ми можемо помножити наш сигнал на вейвлет в різних місцях у часі. Ми починаємо з початку нашого сигналу і повільно переміщуємо вейвлет до кінця сигналу. Ця процедура також відома як згортка (рис. 2.3). Після того, як ми зробили це для вихідного (материнського) вейвлету, ми можемо його масштабувати так, щоб він ставав більшим і повторював процес.

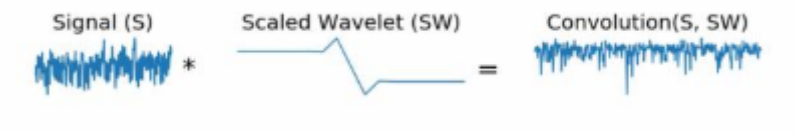


Рисунок 2.3 - Згортка сигналу з використанням вейвлету

Таким чином, вейвлет-перетворення переводить сигнал з часового подання у частотно-часове. Вибір материнського вейвлета $\psi(t)$ зазвичай визначається формою вейвлета, яка повинна бути ближче до форми QRS-комплексу. На рисунку 2.4 зображено різні види вейвлетів.

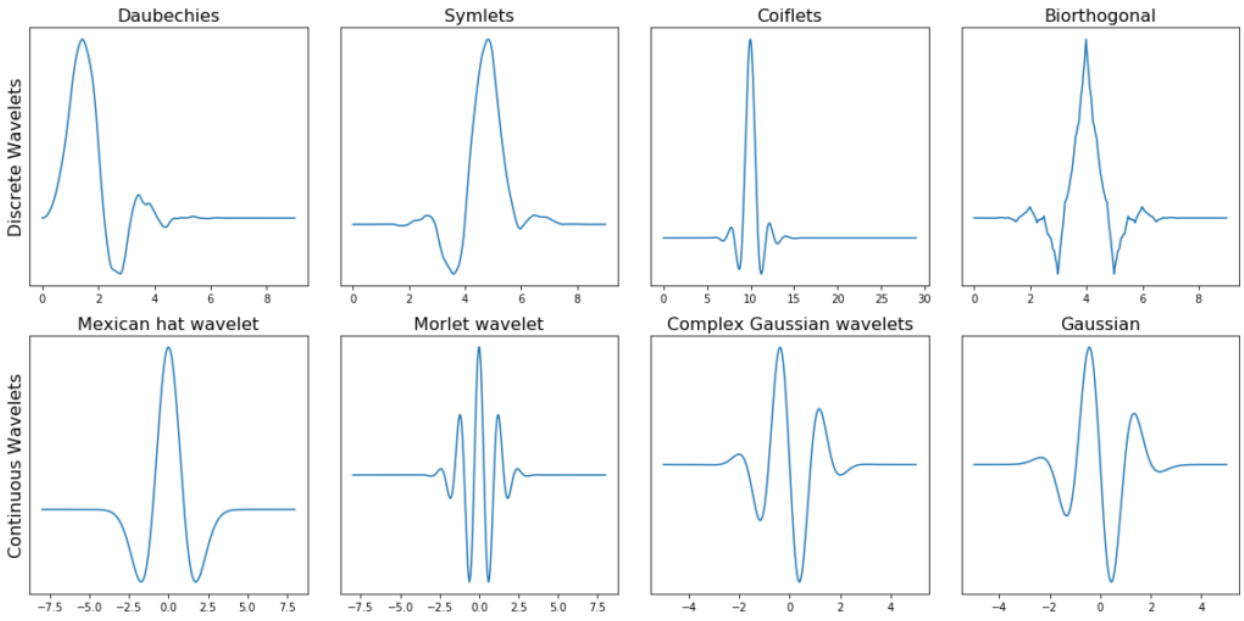


Рисунок 2.4 - Різні види вейвлетів

Кожен тип вейвлетів має різну форму, гладкість і компактність. Оскільки існує лише дві математичних умови, які вейвлет повинен задовольняти, легко згенерувати новий тип вейвлету.

Двома математичними умовами є так звані обмеження нормалізації та ортогоналізації: вейвлет повинен мати 1) кінцеву енергію і 2) локалізацію в часі.

Вибір оптимального вейвлету для аналізу ЕКГ базується на його здатності визначати правильне положення координат дев’яти точок кардіоциклу: початок, пік та зміщення Т-зубця, QRS-комплесу та Р-зубця. В існуючих роботах аналізу ЕКГ використовувались біортогональні вейвлети з компактним носієм, використовуючи масштаби, кратні степеню двійки, а також гаусові вейвлети.

Таблиця 2.1 - Порівняння гаусових та біортогональних вейвлетів

Критерій	Гаусові вейвлети (Gaus)	Біортогональні вейвлети (bior)		
		Bior1.1	Bior1.3	Bior1.5
Ортогональний аналіз	-	+	+	+
Наявність компактного носія	-	+	+	+
Можливість поновлення	Не гарантується	+	+	+
Симетрія	+	+	+	+
Можливий вейвлет-аналіз	CWT без використання швидких алгоритмів	CWT і DWT з використанням швидких алгоритмів		
Точність визначення координат точок ЕКГ-сигналу, %	91-92	93-94	95-96	98-99

Найкращим вейвлет-носієм, який задовольняє раніше описаним вимогам, згідно з отриманими результатами, є біортогональний вейвлет «bior1.5».

В якості масштабу, що використовується для правильного визначення положення дев’яти координат точок ЕКГ-сигналу, було використано 15 масштаб для визначення QRS-сигналу та 41 масштаб для визначення P і T зубців. Масштаби 15 і 41 забезпечують найбільшу точність у визначенні цих зубців. Вейвлет «bior1.5» в масштабах 15 та 41 представлено на рисунку 2.5.

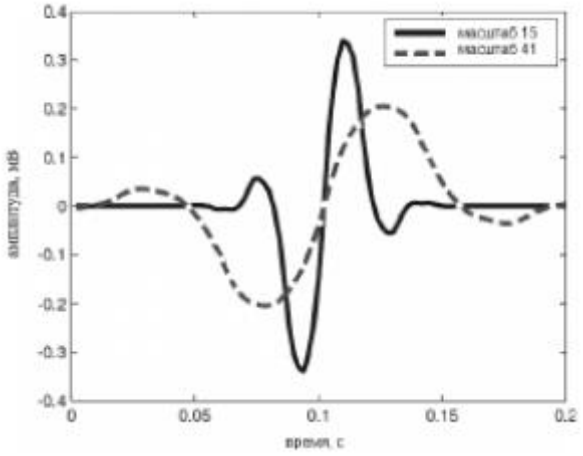


Рисунок 2.5 - Вейвлет «bior1.5» в масштабах 15 та 41

Способи представлення результатів вейвлет-перетворення можуть бути ріноманітними. Вейвлет-спектр виглядає як поверхня в тривимірному просторі. Вид цієї поверхні відображає зміни у часі компонентів різного масштабу. Замість тривимірного зображення часто використовують:

- проекцію поверхні на площину із зображенням ізорівнів (відображають зміни інтенсивності коефіцієнтів ВП на різних часових масштабах);
- картини локальних екстремумів поверхонь, який визначає структуру локалізації максимумів і мінімумів аналізованого сигналу.

Приклад зображення набору послідовних гармонік з частотою 30, 20, 10 і 5 Гц і поверхню спектра його ВП зображені на рисунку 2.6.

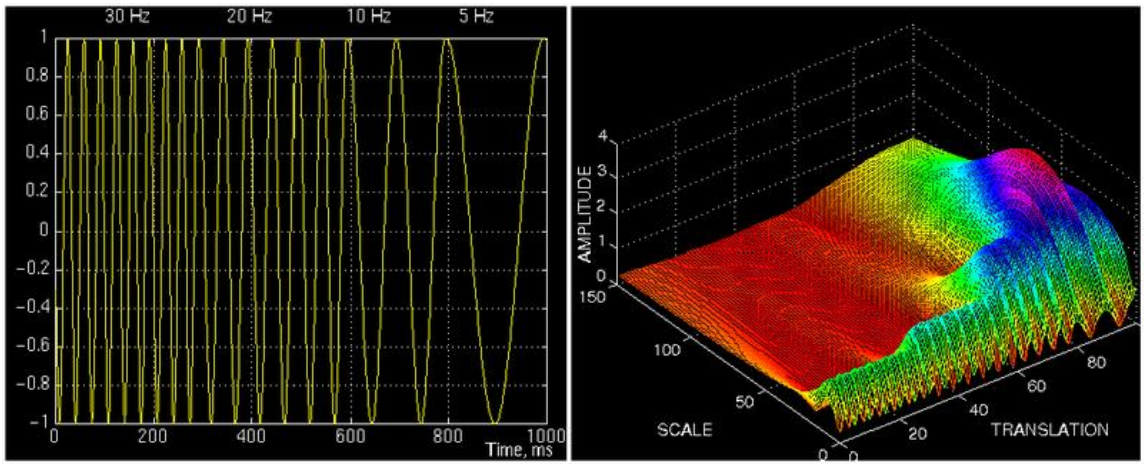


Рисунок 2.6 - Нестаціонарний модельний сигнал та відповідна поверхня ВП цього сигналу

Приклад зображення ліній рівня вейвлет-поверхні фізичного процесу зображено на рисунку 2.7. В сигналі помітна наявність високочастотної складової, що відображено в спектрі.

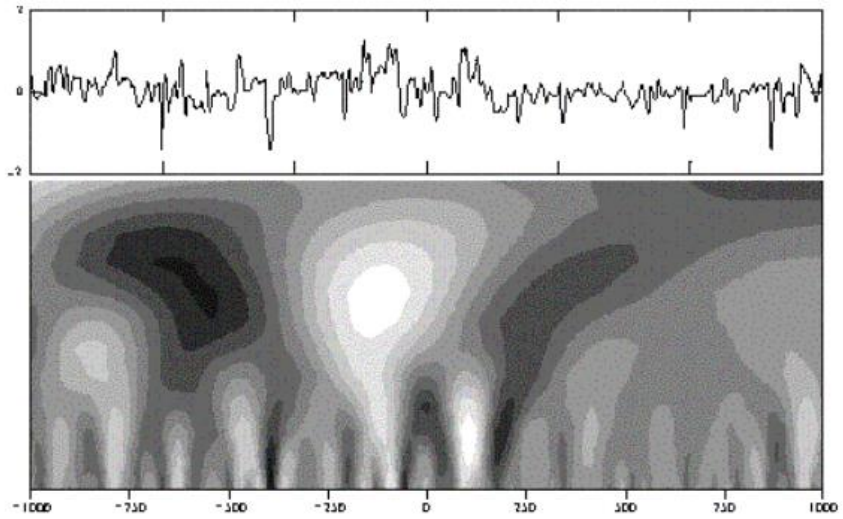


Рисунок 2.7 - Зображення ліній рівня вейвлет-поверхні фізичного процесу

Обгрунтування застосування вейвлетів для виділення QRS-комплексу полягає в тому, що вейвлет-перетворення є гарним інструментом для аналізу перехідних, нестаціонарних (тих, що змінюються в часі) явищ або, іншими словами, може характеризувати локальну регулярність сигналів (рис. 2.8).

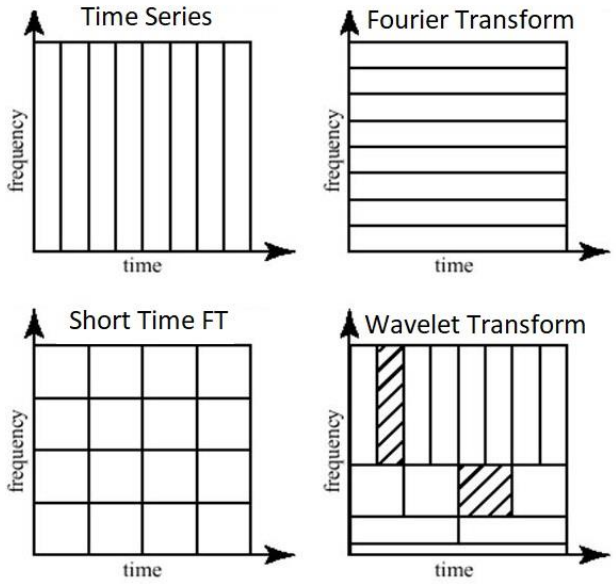


Рисунок 2.8 - Перетворення часових сигналів за допомогою Фур'є перетворень, швидких Фур'є перетворень та вейвлет-перетворень

Так як сигнал ЕКГ є нестаціонарним, як і більшість медичних сигналів, то зазначена особливість вейвлет-аналізу може використовуватися для того, щоб відрізнити хвилі ЕКГ (в тому числі і QRS-комплекс) від серйозних шумів та артефактів.

Вейвлет-перетворення має:

- Для малих значень частоти високу роздільну здатність в частотній області і низьку роздільну здатність у часовій області;
- Для великих значень частоти низьку роздільну здатність в частотній області та високу роздільну здатність в часовій.

Іншими словами, вейвлет-перетворення надає компроміс: у масштабах, в яких цікаві функції залежні від часу, він має високу роздільну здатність у часовій області і в масштабах, в яких цікаві частотно-залежні функції, має високу роздільну здатність в частотній області.

2.2. Дослідження існуючих методів прийняття рішень

Класифікація - один з розділів машинного навчання, присвячений вирішенню наступної задачі. Є безліч об'єктів (ситуацій), розділених деяким чином на класи. Визначено кінцеву множину об'єктів, для яких відомо, до яких класів вони належать. Ця множина називається навчальною вибіркою. Класова приналежність інших об'єктів не відома. Потрібно побудувати алгоритм, здатний класифікувати довільний об'єкт з початкової множини.

Класифікувати об'єкт - значить, вказати номер (або найменування класу), до якого належить даний об'єкт.

Класифікація об'єкта - номер або найменування класу, що видається алгоритмом класифікації в результаті його застосування до даного конкретного об'єкту.

Представимо задачу в формалізованому вигляді:

Дано: Множина об’єктів $T = \{t_1, t_2, \dots, t_n\}$ – навчальна вибірка

$t_i \to \{x_1, x_2, \dots, x_m, y\}$

$X = \{x_1, x_2, \dots, x_m\}$ – незалежні змінні (атрибути)

$C_h = \{c_{h1}, c_{h2}, \dots, c_{hq_h}\}$ - значення, які приймає x_h

y – залежна змінна

$V = \{v_1, v_2, \dots, v_s\}$ – значення, які приймає y

Знайти: спрогнозувати значення y при нових значеннях незалежних змінних.

У машинному навчанні завдання класифікації відноситься до розділу навчання з учителем. Існує також навчання без учителя, коли поділ об'єктів навчальної вибірки на класи не задається, і потрібно класифікувати об'єкти тільки на основі їх подібності між собою. У цьому випадку прийнято говорити про завдання кластеризації або таксономії, і класи називати, відповідно, кластерами або таксонами.

Перед нами стоїть задача класифікації: знаючи показники ЕКГ в нормі та ті, які будуть одержані в разі наявності хвороби, треба визначити за даними кардіограми, є чи немає хвороби серця у пацієнта. У разі, якщо виявлено патологію, визначити вид захворювання. Для такої задачі об'єктом є кардіограма, класами ж є стани здоров'я.

Було розглянуто основні методи класифікації, які використовуються для розв'язання різноманітних класів задач, серед яких є і задачі медичної діагностики.

2.2.1. Нейронні мережі

Нейронна мережа - алгоритм машинного навчання, що представляє собою математичну модель, побудовану за принципом організації та функціонування біологічних нейронних мереж - мереж нервових клітин живого організму. Це поняття виникло при вивченні процесів, що протікають в мозку, і при спробі змоделювати ці процеси.

Нейронні мережі складаються з обчислювальних одиниць - нейронів, пов'язаних один з одним синапсами (ваговими коефіцієнтами). У тому випадку, коли мережа складається з великої кількості нейронів, вводять термін шару.

Структура найпростішої одношарової нейронної мережі наступна (рис. 2.9): вхідний шар нейронів приймає масив вхідних значень $\{x_1, x_2, \dots, x_n\}$ і передає його з ваговими коефіцієнтами $\{w_1, w_2, \dots, w_n\}$ на суматор, результат складання проходить через функцію активації H (S), яка обмежує амплітуду вихідного сигналу нейрона. Зазвичай нормалізований діапазон амплітуд виходу нейрона належить інтервалу [0,1] або [-1,1]

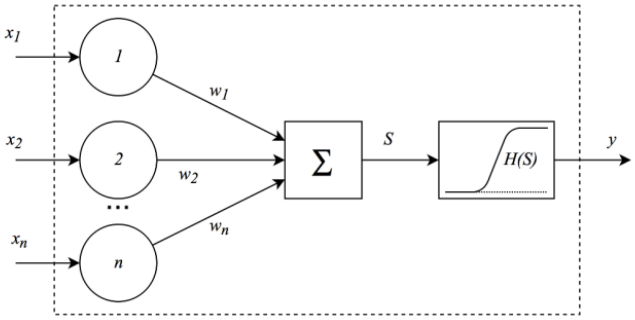


Рисунок 2.9 - Структура найпростішої одношарової нейронної мережі

Роботу мережі можна описати так:

$$S = \sum_{i=1}^{n} x_i w_i \quad (2.5)$$

$$y = H(S) \quad (2.6)$$

Активаційні функції можуть мати різний вигляд: в таблиці 2.2 наведено найбільш поширені варіанти.

Таблиця 2.2 - Види функцій активації

Назва	Формула	Область значень
Порогова	$\Psi = \mathrm{F}(\psi) = \begin{cases} 0, \psi < \Theta, \\ 1, \psi \geq \Theta. \end{cases}$	0, 1
Сигмоїда	$\Psi = \mathrm{F}(\psi) = \frac{1}{1+e^{-\psi}}$	(0, 1)
Лінійна	$\mathrm{F}(\psi) = \psi$	(-∞, ∞)
Лінійна з насиченням	$\Psi = \mathrm{F}(\psi) = \begin{cases} -1, \psi \leq -1, \\ \psi, -1 < \psi < 1, \\ 1, \psi \geq 1. \end{cases}$	(-1, 1)
Гіперболічний тангенс	$\Psi = \mathrm{F}(\psi) = \frac{e^{\psi}-e^{-\psi}}{e^{\psi}+e^{-\psi}}$	(-1, 1)

Виділяють три основних функції активації: лінійну (випрямлену), сигмоїду (логістичну) та гіперболічний тангенс. Відмінності цих функцій полягають у діапазонах значень.

1) Функція активації сигмоїда (рис. 2.10) належить до класу безперервних функцій і приймає на вході довільне дійсне число, а на виході дає дійсне число в інтервалі від 0 до 1. Зокрема, великі (по модулю) негативні числа перетворюються в нуль, а великі позитивні - в одиницю. Історично сигмоїда знаходила широке застосування, оскільки її вихід добре інтерпретується, як рівень активації нейрона: від відсутності активації (0) до повністю насиченою активації (1).

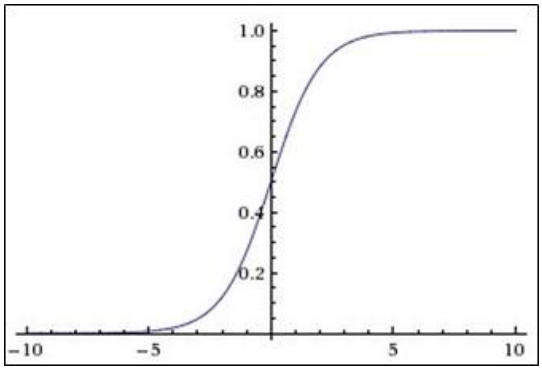


Рисунок 2.10 - Функція активації сигмоїда

Сигмоїдальна функція є:

1. безперервною;
2. монотонно зростаючою;
3. диференційовною.

2) Вибір гіперболічного тангенсу (рис. 2.11) обумовлено наступними причинами:
   - симетричні активаційні функції, типу гіперболічного тангенса забезпечують більш швидку збіжність, ніж стандартна логістична функція;
   - функція має безперервну першу похідну;
   - функція має просту похідну, яка може бути обчислена через її значення, що дає економію обчислень.

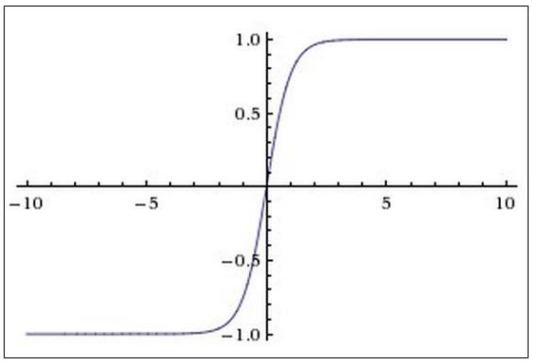


Рисунок 2.11 - Функція активації гіперболічний тангенс

3) Відомо, що нейронні мережі здатні наблизити як завгодно складну функцію, якщо в них досить шарів і функція активації є нелінійною. Функції активації типу сигмоідної або тангенціальної є нелінійними, але призводять до проблем із загасанням або збільшенням градієнтів. Однак можна використовувати і набагато простіший варіант - випрямлену лінійну функцію активації (rectified linear unit, ReLU) (рис. 2.12), яка виражається формулою:

$$f(s) = \max(0, s) \tag{2.7}$$

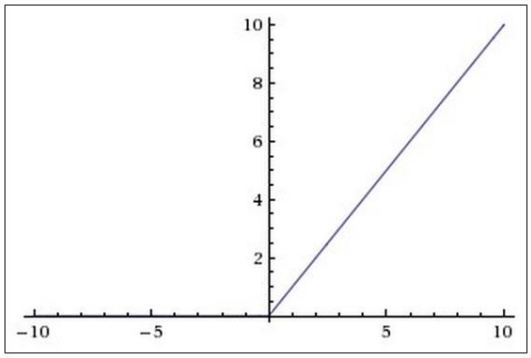


Рисунок 2.12 - Функція активації ReLU

Переваги використання ReLU:

– її похідна дорівнює або одиниці, або нулю, і тому не може статися розростання або загасання градієнтів, тому що помноживши одиницю на дельту помилки ми отримаємо дельту помилки, якщо ж ми б використовували іншу

функцію, наприклад, гіперболічний тангенс, то дельта помилки могла, або зменшитися, або зрости, або залишитися такою ж, тобто, похідна гіперболічного тангенса повертає число з різним знаком і величиною, що може сильно вплинути на загасання або розростання градієнта. Більш того, використання даної функції приводить до проріджування вагів;

– обчислення сигмоїд і гіперболічного тангенса вимагає виконання ресурсомістких операцій, таких як зведення в степінь, в той час як ReLU може бути реалізований за допомогою простого порогового перетворення матриці активацій в нулі;
– відсікає непотрібні деталі в каналі при негативному виході.

З недоліків можна відзначити, що ReLU не завжди достатньо надійна і в процесі навчання може виходити з ладу («вмирати»). Наприклад, великий градієнт, що проходить через ReLU, може привести до такого оновлення ваг, що даний нейрон ніколи більше не активується. Якщо це станеться, то, починаючи з даного моменту, градієнт, що проходить через цей нейрон, завжди буде дорівнювати нулю. Відповідно, даний нейрон буде необоротно виведений з ладу. Наприклад, при дуже великій швидкості навчання (learning rate), може виявитися, що до 40% ReLU «мертві» (тобто, ніколи не активуються). Ця проблема вирішується за допомогою вибору належної швидкості навчання.

Нейронні мережі мають здатність до навчання (тренування). Технічно навчання полягає у визначенні вагових коефіцієнтів між нейронами. При навчанні мережі пропонуються різні зразки образів із зазначенням того, до якого класу вони відносяться. Зразок, як правило, представляється як вектор значень ознак. При цьому сукупність усіх ознак повинна однозначно

визначати клас, до якого належить зразок. У разі, якщо ознак недостатньо, мережа може співвіднести один і той же зразок з декількома класами, що невірно. Після закінчення навчання мережі їй можна пред'являти невідомі раніше образи і отримувати відповідь про належність до певного класу.

Структура нейронної мережі тісно пов’язана з алгоритмами навчання, які використовуються. В загальному випадку можна виділити три фундаментальних класи нейромережевих архітектур:

1) Одношарові мережі прямого розповсюдження:

В такій мережі існує вхідний шар вузлів джерела, від якого інформація передається на вихідний шар нейронів, проте не навпаки. Така мережа називається одношаровою, при цьому під єдиним шаром мається на увазі шар обчислювальних елементів (нейронів). При підрахунку кількості шарів до уваги не беруться вузли джерела, оскільки вони не виконують обчислень.

2) Багатошарові мережі прямого розповсюдження

Даний клас характеризується наявністю одного або декількох прихованих шарів, вузли яких називаються прихованими нейронами. Їхньою функцією є посередництво між зовнішнім вхідним сигналом та виходом нейронної мережі. Така мережа дозволяє виділяти глобальні властивості даних за рахунок наявності додаткових синаптичних зв’язків та підвищення рівня взаємодії нейронів.

3) Рекурентні мережі

Рекурентна нейронна мережа відрізняється від мережі прямого розповсюдження наявністю принаймні одного зворотнього зв’язку. Наявність зворотніх зв’язків чинить безпосередній вплив на здатність таких мереж до навчання та на їхню продуктивність.

Серед існуючих архітектур нейронних мереж для вирішення задач класифікації виділяють **згорткові нейронні мережі** (англ. *convolutional neural network, CNN*). Структура мережі - односпрямована (без зворотних зв'язків), принципово багатошарова.

ЗНМ складається з різних видів шарів: згорткові (convolutional) шари, субдискретизуючі (subsampling, підвибірка) шари і шари «звичайної» нейронної мережі - персептрона, відповідно до рисунку 2.13.

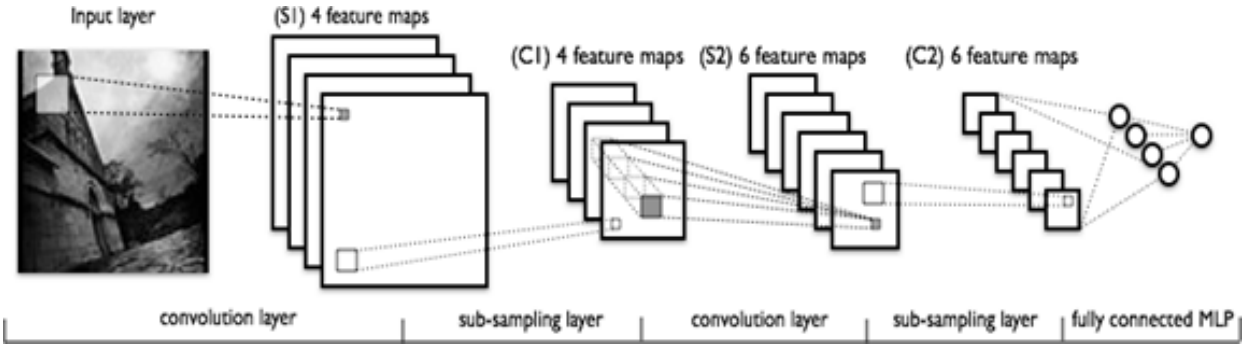


Рисунок 2.13 - Топологія згорткової нейронної мережі

Перші два типи шарів (convolutional, subsampling), чергуючись між собою, формують вхідний вектор ознак для багатошарового персептрона.

Згорткові мережі є вдалою серединою між біологічно правдоподібними мережами і звичайним багатошаровим персептроном. На сьогоднішній день кращі результати в розпізнаванні зображень отримують з їх допомогою. В середньому точність розпізнавання таких мереж перевершує звичайні ШНМ на 10-15%. ЗНМ - це ключова технологія Deep Learning.

Аргументувати вибір такої архітектури нейронної мережі, як згорткова, можна тим, що розмір ЗНМ є порівняно невеликим та не залежить від розміру навчальних даних. При навчанні для кожного зразка на вхід згорткової нейронної мережі потрапляє не все зображення m×n, а тільки її частина $r \times c$, наприклад 10 × 10, ця частина називається фільтром або ядром і вона рухається по великому зображенню. Така згорткова нейронна мережа має фіксовану кількість нейронів на вхідному та вихідному шарах $r \times c = 10 \times 10 = 100$, а кількість вагів при задачі виділення, наприклад, k = 50 ознак буде дорівнювати $2 \times r \times c \times k = 2 \times 10 \times 10 \times 50 = 10000$.

Для кожного великого зображення m × n беремо маленькі зображення $r \times c$ для навчання. Їх кількість рівна (m – r + 1) × (n – c +1).

Суть згортки зображено на рисунку 2.14.

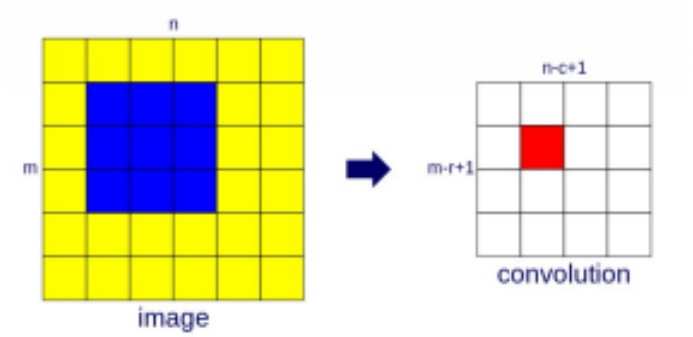


Рисунок 2.14 – Згортка

При використанні згорткової нейронної мережі виникає проблема – велика кількість виділених ознак. Для однієї великої картинки $m \times n$ буде $(m\text{-}r + 1) \times (n\text{-}c + 1)$ маленьких картинок $r \times c$, і кількість виділених ознак рівна $k \times (m - r + 1) \times (n - c + 1) = 50 \times (100 - 10 + 1) \times (100 - 10 + 1) = 414\ 050$. Використання такої величезної кількості ознак для класифікації виявилося неефективним. Для зменшення розміру простору ознак проводимо субдискретизацію (англ. pooling), розділивши карту ознак, отриманих від згорткової нейронної мережі, на фіксовану кількість частин p, див. рис. 2.15, на цьому рисунку $p = pm \times pn = 2 \times 2 = 4$ і на кожній частині обчислюється її максимальне значення (англ. max pooling) або середнє значення (англ. mean pooling).

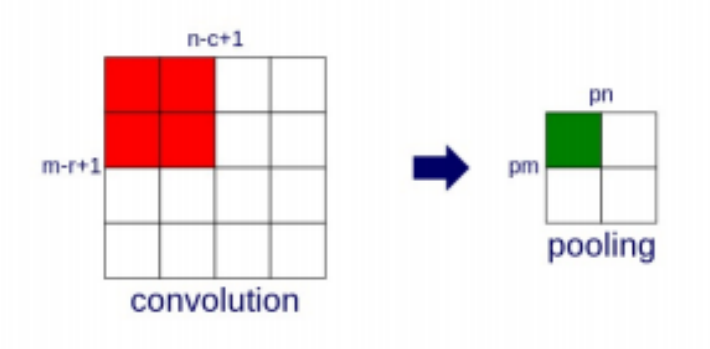


Рисунок 2.15 – Субдискретизація

До переваг згорткової нейромережі можна віднести:

1) Зменшення кількості навчальних параметрів і підвищення швидкості навчання порівняно з повнозв’язною нейронною мережею.

2) Можливість розпаралелювання обчислень і реалізації алгоритмів навчання мережі на графічних процесорах (GPU).

3) Стійкість до зсуву позиції об'єкта у вхідних даних. При навчанні згорткова нейронна мережа зсувається по частинах об'єкта. Тому навчальні ознаки не залежать від позиції «важливих частин», наприклад голова собаки (див. рис 2.16, згорткова нейронна мережа виділяє однакові ознаки для двох картинок, хоча позиції собак на цих картинках різні). Таким чином, ця властивість згорткової нейронної мережі допомагає підвищувати якість класифікації.

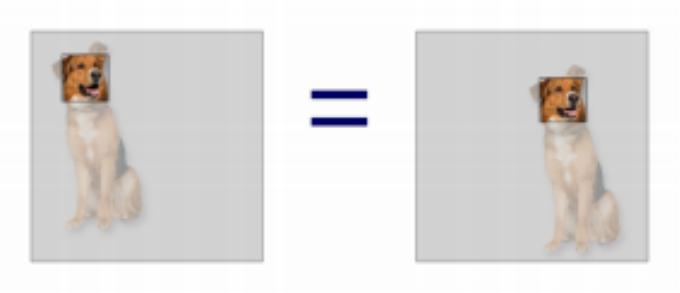


Рисунок 2.16 - Стійкість до зсуву об’єкта у вхідних даних

До недоліків згорткової нейромережі можна віднести:

1) Занадто багато змінних параметрів мережі; незрозуміло, для якої задачі і обчислювальної потужності які потрібні налаштування. Так, до варійованих параметрів можна віднести: кількість шарів, розмірність ядра згортки для кожного з шарів, кількість ядер для кожного з шарів, крок зсуву ядра при обробці шару, необхідність шарів субдискретизації, ступінь зменшення ними розмірності, функція по зменшенню розмірності (вибір максимуму, середнього і т. п.), передавальна функція нейронів, наявність і параметри вихідної повнозв’язної нейромережі на

виході згорткової. Всі ці параметри істотно впливають на результат, але вибираються дослідниками емпірично. Існує кілька вивірених і прекрасно працюючих конфігурацій мереж, але не вистачає рекомендацій, за якими потрібно будувати мережу для нового завдання.

2.2.2. Машина опорних векторів

МОВ - це вид мережі з позитивним зворотним зв'язком, запропонованої Вапніком, і є одним з найбільш ефективних інструментів в задачі класифікації. Цей метод має дуже гарну здатність до узагальнення. На відміну від проблеми навчання класичних нейронних мереж, де мінімізована функція помилки нелінійна по відношенню до оптимізованих змінних множини потенційних мінімумів, МОВ призводить до квадратичного програмування з лінійними залежностями і може визначити чітко виражений глобальний мінімум.

По суті, МОВ - це лінійний механізм, що працює в просторі характеристик з високою роздільною здатністю, створеному шляхом нелінійного перетворення початкового N-мірного вхідного вектора x в K-мірний простір характеристик (K > N) з використанням функції φ (x). Рівняння гіперплощини, що розділяє два різних класи в N-вимірному просторі,наведено нижче:

$$y(x) = w^T\varphi(x) + \omega_0 = \sum_{j=1}^{K} \omega_j\varphi_j(x) + \omega_0 \quad (2.8)$$

$$\varphi(x) = [\varphi_1(x),\ \varphi_2(x), \dots, \varphi_K(x)]^T \quad (2.9)$$

$$w = [\omega_1,\ \omega_2, \dots, \omega_K]^T \quad (2.10)$$

де w – ваговий вектор мережі;

w_0 – похибка.

Коли виконується y (x)> 0, вхідний вектор x належить до одного класу, а коли y (x) <0 - до іншого. Усі математичні операції в режимі навчання і

тестування виконуються з використанням так званої керн-функції $K(x_i, x)$, що визначається як внутрішній скалярний добуток:

$$\varphi(x), K(x) = \varphi^T(x_i)\varphi(x) \tag{2.11}$$

Найбільш відомі керн-функції - це лінійна, Гауссіан, поліноміальна і сплайн-функція. Основним завданням навчання в МОВ є поділ тренувальних векторів x_i на 2 класи, що визначаються приймаючими значеннями $d_i = 1$ або $d_i = -1$, з максимальною роздільною здатністю. Далі виникає завдання максимізації квадратичної функції Q (x):

$$Q(\alpha) = \sum_{i=1}^{p} \alpha_i - 0{,}5 \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j d_i d_j K(x_i, x_j) \tag{2.12}$$

При цьому

$$\sum_{i=1}^{p} \alpha_i d_i \tag{2.13}$$

$$0 \le \alpha_i \le C \tag{2.14}$$

Змінні a_i - це коефіцієнти Лагранжа, d_i відносяться до приймаючих значень, що відповідають вхідним векторам x_i, С – константа регуляризації, яку визначає користувач, а р – кількість пар навчальних даних (x_i, d_i). Розв’язок цих двох задач відносно коефіцієнтів Лагранжа дозволяє визначити оптимальний ваговий вектор , w_{opt} мережі МОВ

$$w_{opt} = \sum_{i=1}^{N_{sv}} \alpha_i d_i \varphi \quad (x_i) \tag{2.15}$$

N_{sv} – це кількість опорних векторів (векторів x_i, для яких коефіцієнти Лагранжа відмінні від нуля). Підставивши розв'язок w_{opt} до рівняння у(х), отримаємо вираз вихідного сигналу у(х) мережі МОВ як функції кернів

$$y(x) = \sum_{i=1}^{N_{sv}} \alpha_i d_i K(x_i, x) + \omega_0 \quad (2.16)$$

Додатнє значення y(x) відповідає 1 (об'єкт належить до досліджуваного класу), а від'ємне значення -1 (об'єкт належить до протилежного класу).

Основний параметр МОВ - це константа регуляризації C. Вона контролює співвідношення між шириною роздільного кордону, що впливає на складність методу, і нерозділених точок на етапі навчання мережі. Маленьке значення C призводить до розширення роздільного кордону і збільшення кількості нерозділених точок на фазі навчання. Велике значення C дозволяє досягти мінімальної кількості помилок класифікації, вузького роздільного кордону і меншої кількості опорних векторів. Занадто велике значення C призводить до втрати узагальнюючої здатності мережі, що навчається. Для нормалізованих вхідних сигналів зі значеннями в інтервалі (-1, 1) константа регуляризації зазвичай набагато більше 1 і визначається емпіричним шляхом використання зразкового набору даних.

Важливим досягненням МОВ є перетворення завдання навчання в задачу квадратичного програмування. Для такого завдання оптимізації існує безліч ефективних алгоритмів навчання, які майже у всіх випадках призводять до глобального мінімуму цільової функції і вибору найкращого набору параметрів мережі. Іншим досягненням МОВ є гарна узагальнююча здатність, яка майже не залежить від кількості тренувальних зразків.

Хоча МОВ розділяє дані на 2 класи, розпізнати більшу число класів також нескладно шляхом застосування або методу «один проти одного» або методу «один проти всіх». Метод «один проти одного» зазвичай більш ефективний. У цьому методі незалежно навчається безліч локальних двокласових класифікаторів, що дозволяє вибрати найкращий класифікатор. Для M класів необхідно навчити M (M-1) / 2 двокласових, заснованих на МОВ системи розпізнавання.

2.3 Критерії якості оцінки рішення задачі

Основою перевірки є тестова вибірка, в якій проставлено відповідність між документами і їх класами.

Чисельна оцінка якості алгоритму (Accuracy):

У найпростішому випадку такою метрикою може бути частка документів, за якими класифікатор прийняв правильне рішення.

$$Accuracy = \frac{P}{N} \tag{2.17}$$

де P - кількість документів, за якими класифікатор прийняв правильне рішення;

N - розмір навчальної вибірки.

Проте, у цієї метрики є одна особливість яку необхідно враховувати. Вона присвоює всім документам однакову вагу, що може бути не коректно в разі якщо розподіл документів в навчальній вибірці сильно зміщена в бік якогось одного або декількох класів. В цьому випадку у класифікатора є більше інформації по цих класах і відповідно в рамках цих класів він буде приймати більш адекватні рішення. На практиці це призводить до того, що ви маєте accuracy, скажімо, 80%, але при цьому в рамках якогось конкретного класу класифікатор працює з рук геть погано не розпізначаючи правильно навіть третину документів.

Один вихід з цієї ситуації полягає в тому щоб навчати класифікатор на спеціально підготовленому, збалансованому корпусі документів. Мінус цього рішення в тому що ви відбираєте у класифікатора інформацію про відносну частоту документів. Ця інформація при інших рівних може виявитися дуже доречною для прийняття правильного рішення.

Інший вихід полягає в зміні підходу до формальної оцінці якості.

Точність і повнота:

Точність (precision) і повнота (recall) є метриками, які використовуються при оцінці здебільшого алгоритмів вилучення інформації. Іноді вони використовуються самі по собі, іноді в якості базису для похідних метрик, таких як F-міра або R-Precision. Суть точності і повноти дуже проста.

Точність системи в межах класу - це частка документів, що дійсно належать даному класу щодо всіх документів, які система віднесла до цього класу. Повнота системи - це частка знайдених классфікатором документів належать класу щодо всіх документів цього класу в тестовій вибірці.

Ці значення легко розрахувати на підставі таблиці контингентні, яка складається для кожного класу окремо.

У таблиці міститься інформація, скільки разів система прийняла вірне і скільки разів невірне рішення за документами заданого класу. А саме:

- TP - істино-позитивне рішення;
- TN - істино-негативне рішення;
- FP - хибно-позитивного рішення;
- FN - хибно-негативне рішення.

Тоді, точність і повнота визначаються наступним чином:

$$Precision = \frac{TP}{TP+FP} \tag{2.18}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.19}$$

Confusion Matrix:

На практиці значення точності і повноти набагато більш зручніше розраховувати з використанням матриці неточностей (confusion matrix). У разі якщо кількість класів відносно невелика (не більше 100-150 класів), цей підхід дозволяє досить наочно представити результати роботи класифікатора.

Матриця неточностей - це матриця розміру N на N, де N - це кількість класів. Стовпці цієї матриці резервуються за експертними рішеннями, а рядки за рішеннями класифікатора. Коли ми класифікуємо документ з тестової

вибірки ми інкрементіруем число, що стоїть на перетині рядка класу, який повернув класифікатор і стовбця класу до якого дійсно відноситься документ.

На малюнку 2.17 зображено структуру матриці неточностей.

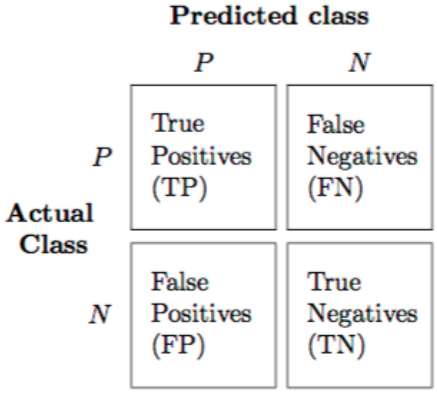


Рисунок 2.17 - Структура матриці неточностей

Маючи таку матрицю точність і повнота для кожного класу розраховується дуже просто. Точність дорівнює відношенню відповідного діагонального елемента матриці і суми всього рядка класу. Повнота - відношенню діагонального елемента матриці і суми всього стовпчика класу.

Результуюча точність класифікатора розраховується як арифметичне середнє його точності по всіх класах. Те ж саме з повнотою. Технічно цей підхід називається macro-averaging.

2.4 Порівняння методів класифікації кардіограм

За останні кілька років було запропоновано багато методів автоматичної класифікації сигналів ЕКГ. Наприклад, Пан та Томпкінс запропонували алгоритм виявлення QRS-комплексу на основі цифрового аналізу нахилу, амплітуди і ширини. У 2004 році вчені Шуі та ін. використовували вейвлет-перетворення (WT) для виявлення аритмії.

Однак ці методи потребують виділення ознак вручну, часто втрачаючи багато важливої інформації. Тому для виявлення сигналів ЕКГ впроваджуються такі методи машинного навчання, як метод опорних векторів (SVM) та штучні нейронні мережі (ANNs).

У 2016 році вченими Кіраніаз та ін. було запропоновано новий спосіб моніторингу для конкретного пацієнта з використанням одновимірної

згорткової нейромережі (1D-CNNs). Пізніше на основі цього методу було запропоновано використання двовимірної згорткової нейронної мережі (2D-CNNs), яка дозволяє робити класифікацію на основі зображень побудованих на основі сигналів ЕКГ.

У таблиці 2.3 наведено порівняння методів автоматичної класифікації сигналів ЕКГ.

Таблиця 2.3 – Порівняння існуючих методів класифікації ЕКГ

Метод	Класифікатор	Чутливість	Точність
Jun et al.	2D CNN	97.85	99.05
Kiranyaz et al.	1D CNN	68.8	96.4
Melgani and Bazi	SVM	93.83	91.67
Ubeyli et al.	RNN	98.15	98.06

За оцінками чутливості та точності класифікації бачимо, що найкращим за одержаними результатами є метод, заснований на використанні двовимірної згорткової нейронної мережі.

Отже, для реалізації системи розпізнавання сигналів ЕКГ в даній роботі використовується саме двовимірна згорткова нейронна мережа.

2.5 Алгоритм розв’язку задачі

В якості метода для цифрової обробки сигналу було обрано вейвлет-перетворення, оскільки серед розглянутих досліджень визначення параметрів ЕКГ саме з використанням вейвлет-перетворення дало найкращі результати.

Для побудови класифікатора було обрано згорткову нейронну мережу з огляду на потребу опрацювання великого об’єму даних електрокардіограм.

Алгоритм розв’язку задачі розпізнавання кардіограм, що пропонується в цій роботі, є наступним:

1. Підготовка даних до програмної обробки: завантаження датасетів.
2. Виконання попередньої обробки даних: використання неперервного вейвлет-перетворення для переведення часового сигналу у частотно-часовий і його представлення у вигляді спектрограми.
3. Визначення параметрів згорткової нейронної мережі.

4. Побудова нейронної мережі
5. Оцінка отриманих результатів.

2.6 Висновки до розділу

У другому розділі було розглянуто основні засоби цифрової обробки часових сигналів, до яких належать сигнали ЕКГ, а саме перетворення Фур’є та вейвлет-перетворення. Для методу вейвлет-перетворення було проведено порівняльний аналіз його засобів, а саме двох найчастіше використовуваних вейвлетів для виявлення точок сигналу ЕКГ.

Для виконання етапу класифікації даних було розглянуто такі засоби класифікації, як нейронні мережі (в тому числі детально було описано принцип роботи згорткових нейронних мереж), метод опорних векторів.

Було розглянуто і наведено основні критерії оцінки якості розв’язку поставленої задачі розпізнавання кардіограм, серед яких чисельна оцінка якості алгоритму, точність і повнота, матриця неточностей.

У даному розділі було проведено порівняння існуючих методів класифікації ЕКГ та зроблено висновок щодо найкращого з них.

Також у другому розділі наведено алгоритм розв’язку задачі, який реалізовується в даній роботі.

РОЗДІЛ 3 АРХІТЕКТУРА ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ

3.1 Аналіз існуючих систем та пакетів для цифрової обробки сигналів і побудови систем класифікації даних

Неперервний сигнал, яким є сигнал кардіограми, може бути представлений в цифровому форматі, тобто його можна дискретизувати у часі та квантувати за рівнем. Після проведення оцифровки сигнал може бути оброблений за допомогою комп’ютерних методів, які реалізовано в різноманітних математичних системах.

3.1.1 Mathematica

Система комп'ютерної алгебри, яка широко використовується в наукових, інженерних, математичних і комп'ютерних областях. Mathematica підтримує процедурне програмування із застосуванням стандартних операторів управління виконанням програми (цикли і умовні переходи), і об'єктно-орієнтований підхід. Mathematica допускає відкладені обчислення. Також в системі Mathematica можна задавати правила роботи з тими чи іншими виразами.

Основні аналітичні можливості:

- розв’язання систем поліноміальних і тригонометричних рівнянь і нерівностей, а також трансцендентних рівнянь, що зводяться до них;
- розв’язання рекурентних рівнянь;
- спрощення виразів;
- знаходження меж;
- інтегрування і диференціювання функцій;
- знаходження кінцевих і нескінченних сум і добутків;
- розв’язання диференціальних рівнянь і рівнянь в часткових похідних;

- перетворення Фур'є і Лапласа, а також Z-перетворення;
- перетворення функції в ряд Тейлора, операції з рядами Тейлора: додавання, множення, композиція, отримання зворотної функції;
- вейвлетний аналіз.

Система також здійснює чисельні розрахунки: визначає значення функцій (в тому числі спеціальних) з довільною точністю, здійснює поліноміальну інтерполяцію функції від довільного числа аргументів по набору відомих значень, розраховує ймовірності.

Також в систему закладено лінійно-алгебраїчні можливості - робота з матрицями (додавання, множення, знаходження оберненої матриці, множення на вектор, обчислення експоненти, взяття визначника), пошук власних значень і власних векторів.

Система представляє результати як в алфавітно-цифровій формі, так і у вигляді графіків. Зокрема, реалізовано побудову графіків функцій, в тому числі параметричних кривих і поверхонь; побудова геометричних фігур (ламаних, кіл, прямокутників та інших); побудова і маніпулювання графами. Крім того, реалізовано відтворення звуку, графік якого задається аналітичною функцією або набором точок.

В системі Mathematica наявні функції розрахунку аналогових та цифрових фільтрів, функції обробки сигналів, які можна використовувати з зображеннями, аудіо та іншими даними, в тому числі і сигналами електрокардіограм. Зокрема можуть бути використані такі функції як:

- WarpingDistance [s1, s2] - дає мінімальну відстань для будь-якої відповідності між еталонною послідовністю s1 і послідовністю запиту s2.

  Дана функція може бути застосована для порівняння двох сигналів ЕКГ (рис. 3.1).

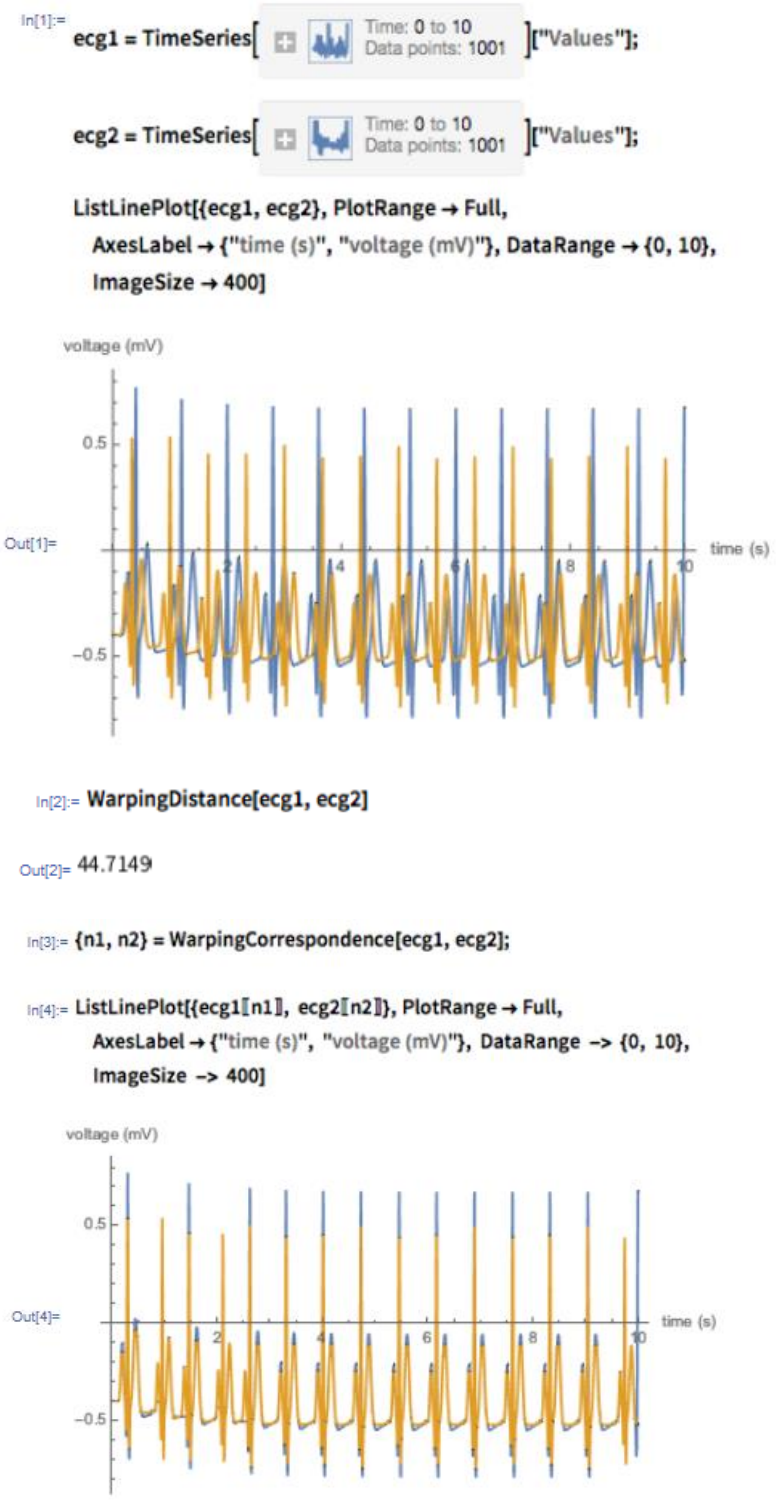


Рисунок 3.1. Порівняння двох сигналів ЕКГ засобами системи Mathematica

– WaveletThreshold для вейвлет-згладжування, видалення шуму та компресії сигналу (рис. 3.2)

In[4]:= **{ListLinePlot[data], ListLinePlot[InverseWaveletTransform[thr]]}**

```
{ListLinePlot[data], ListLinePlot[InverseWaveletTransform[thr]]}
```

Copy to clipboard.

Out[4]=

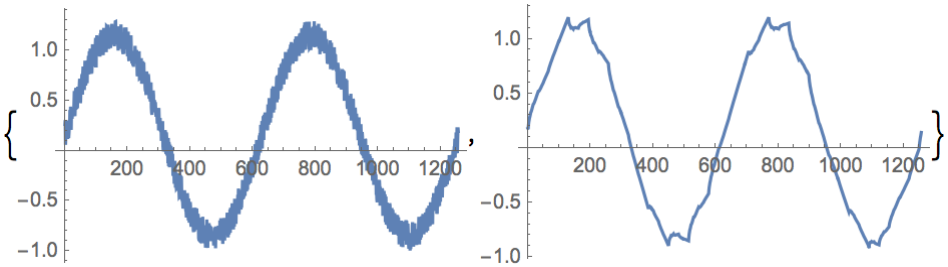


Рисунок 3.2. Вейвлет-обробка сигналу в системі Mathematica

3.1.2 Matlab

Пакет прикладних програм для вирішення задач технічних обчислень і однойменна мова програмування, що використовується в цьому пакеті. Мова Matlab є високорівневою інтерпретовною мовою програмування, що включає засновані на матрицях структури даних, широкий спектр функцій, інтегроване середовище розробки, об'єктно-орієнтовані можливості та інтерфейси до програм, написаних на інших мовах програмування.

Matlab надає користувачеві велику кількість функцій для аналізу даних, які покривають майже всі області математики, зокрема:

- матриці і лінійна алгебра - алгебра матриць, лінійні рівняння, власні значення і вектори, сингулярності, факторизація матриць та інші;
- многочлени і інтерполяція - корені многочленів, операції над многочленами і їх диференціювання, інтерполяція і екстраполяція кривих та інші;
- обробка даних - набір спеціальних функцій, включаючи побудову графіків, оптимізацію, пошук нулів, чисельне інтегрування (в квадратурі) та інші.

Також Matlab підтримує безліч зовнішніх інтерфейсів, за допомогою яких організовано взаємодію з іншими програмами і периферійними

пристроями. Серед можливих інтерфейсів є COM (Component Object Model), з його допомогою можна маніпулювати як клієнтами і серверами. Пакет Matlab в Microsoft Windows надає доступ до програмної платформи .NET Framework. Пакет Matlab містить функції, які дозволяють йому отримувати доступ до інших додатків середовища Windows, так само як і цим програмам отримувати доступ до даних Matlab, за допомогою технології динамічного обміну даними (DDE).

Саме для цифрової обробки сигналу існують розширення Signal Processing Toolbox, DSP System Toolbox, Image Processing Toolbox, Wavelet Toolbox, Communications System Toolbox - набори функцій і об'єктів, що дозволяють вирішувати широкий спектр завдань обробки сигналів, зображень, проектування цифрових фільтрів і систем зв'язку (рис. 3.3).

Розширення DSP SystemToolbox призначене для моделювання систем обробки сигналів. Наприклад: радарів, сонарів, систем управління, медичних приладів.

У DSP SystemToolbox реалізовано більшість існуючих алгоритмів для ЦОС. Так само присутній інструментарій для аналізу спектра сигналу, логічний аналізатор і візуалізації результату. використовуючи DSP SystemToolbox спільно з MatlabCoder і SimulinkCoder можна генерувати код на C / C ++.

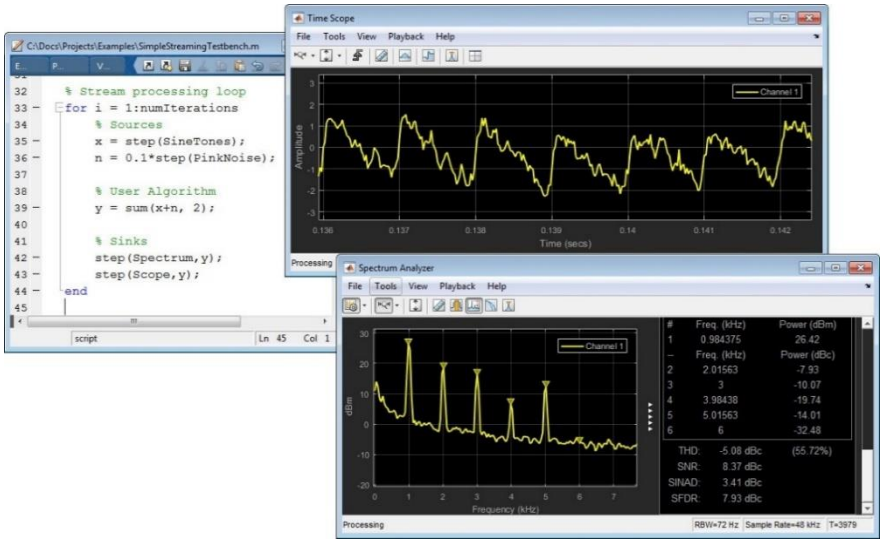


Рисунок 3.3. Реалізація ЦОС в системі Матлаб

Також в Матлаб э розширення Neural Network Toolbox - інструменти для синтезу та аналізу нейронних мереж (рис. 3.4).

Рисунок 3.4. Використання розширення Neural Network Toolbox

3.1.3 AnyLogic

Пакет AnyLogic призначений для розробки та дослідження імітаційних моделей. Він має зручний інтерфейс користувача, в ньому застосовується об’єктно-орієнтований підхід, як мова програмування використовується Java. Наявність компілятора Java не лише розширює можливості у створенні моделей, але і надає можливість створювати аплети, які відкриваються в будь-якому браузері та можуть розміщуватись на веб-сайтах.

- AnyLogic підтримує такі методи імітаційного моделювання, як:
- агентне моделювання (рис.3.5);
- дискретно-подійне моделювання (рис. 3.6);
- системна динаміка (рис. 3.7).

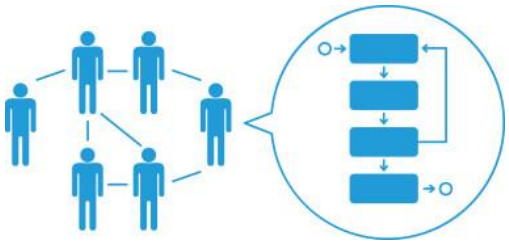


Рисунок 3.5 - Агентне моделювання

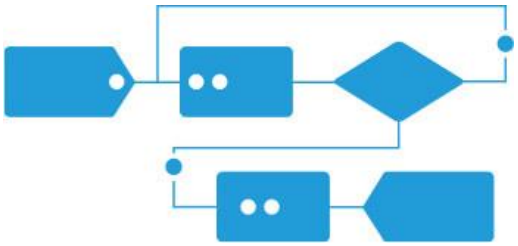


Рисунок 3.6 - Дискретно-подійне моделювання

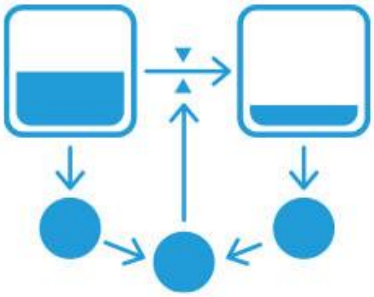


Рисунок 3.7 - Системна динаміка

Даний пакет надає можливості розробки моделей виробництва, бізнес-процесів, управління активами та проектами, екологічних систем, а також моделей в сфері фармацевтики та охорони здоров'я.

AnyLogic працює в таких оперційних системах, як Windows, Linux, Mac OS.

В пакеті можна моделювати процеси з використанням графічних інструментів (рис.3.8):

– Stock & Flow Diagrams (діаграма потоків та накопичувачів) застосовується для розробки моделей із використанням методу системної динаміки.
– Statecharts (карти станів) використовується здебільшого в агентних моделях для визначення поведінки агентів. Також використовується

в дискретно-подійному моделюванні, наприклад для симуляції машинних відмов.

– Action charts (блок-схеми) використовуються для побудови алгоритмів. Застосовуються в дискретно-подійному моделюванні та агентному моделюванні.
– Process flowcharts (діаграми процесів) основна конструкція, що використовується для визначення процесів в дискретно-подійному моделюванні.

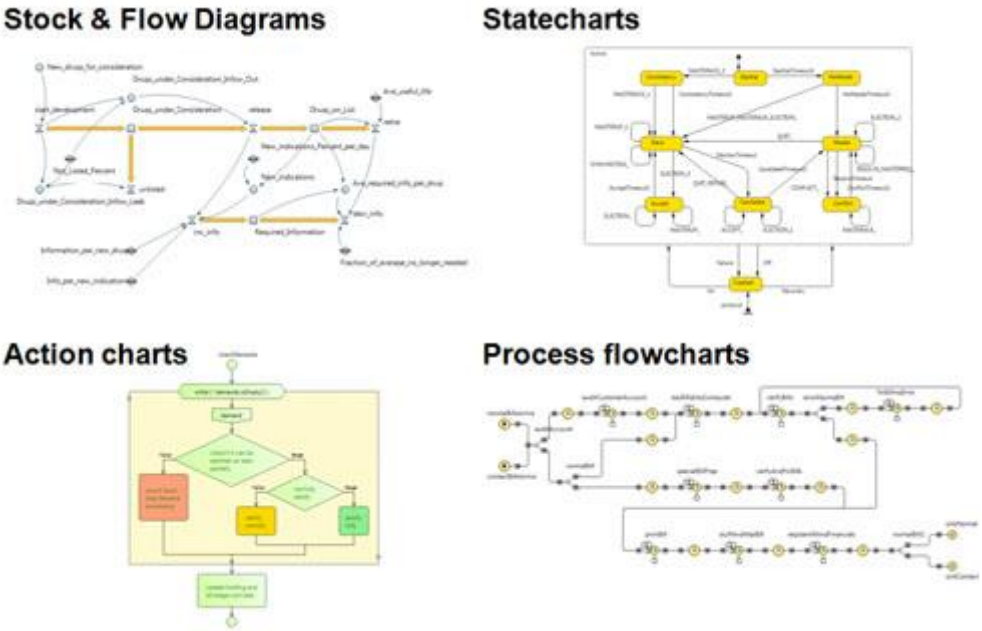


Рисунок 3.8. Графічні інструменти пакету AnyLogic

Також пакет надає можливість AnyLogic Cloud – онлайн-сервіс, який дозволяє запускати моделі в браузері, проводити експерименти в хмарі, налаштовувати відображення результатів та аналізувати їх онлайн.

3.1.4 Python

Python не є пакетом для обробки сигналів, це мова програмування загального призначення, стандартна бібліотека якого включає в себе багато

корисних функцій. Крім стандартних бібліотек, існує ряд спеціалізованих, за допомогою яких можна проводити цифрову обробку сигналів:

- NumPy для роботи з числами, векторами, масивами; містить методи для створення, змінювання форми, множення та розрахунку детермінанту матриць, розв’язання лінійних рівнянь;
- SciPy розширює можливості NumPy, включає методи лінійної алгебри та методи для роботи з імовірнісними розподілами, інтегральними обчисленнями та перетвореннями Фур’є;
- Matplotlib для візуалізації даних: 2D і 3D графіка, графіки, діаграми, спектральні діаграми;
- Pandas – представляє структури даних та інструменти для аналізу та обробки даних, в тому числі невпорядкованих, немаркованих.

Для аналізу даних в Python ефективним і зручним є використання бібліотек (рис. 3.9):

- Scikit-learn – заснована на NumPy і SciPy та надає алгоритми для машинного навчання й інтелектуального аналізу даних: класифікації, кластеризації, регресії;
- Theano – використовується для оцінки та покращення математичних виразів;
- TensorFlow – використовується для налаштування, тренування та застосування нейронних мереж з багаточисельними наборами даних.

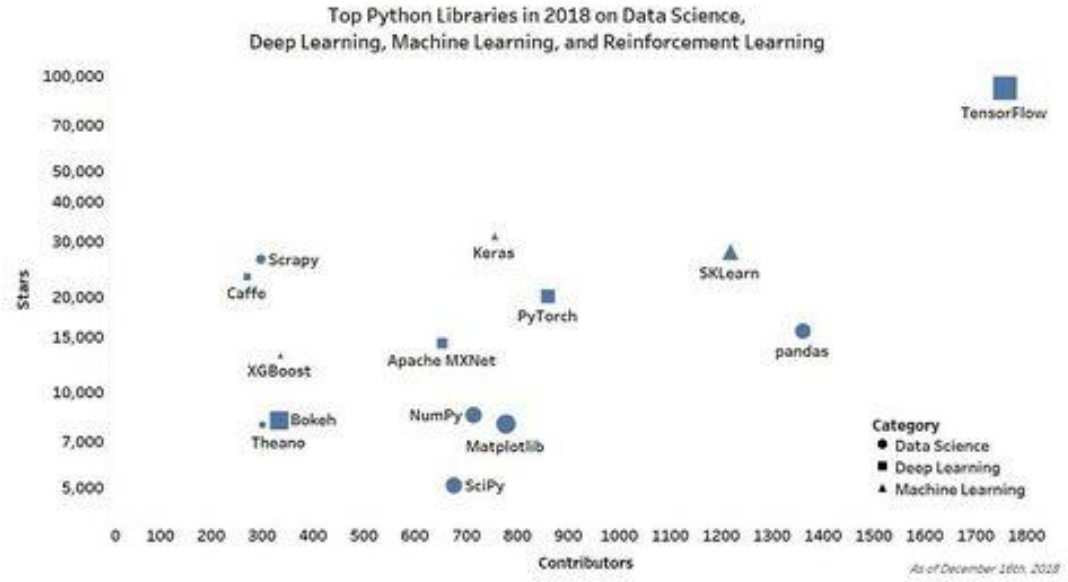


Рисунок 3.9. Найбільш часто використовувані бібліотеки мови Python для машинного навчання

Для Python розроблені пакети для доступу до різних СУБД: Oracle, MySQL, PostgreSQL, Sybase, Firebird (Interbase), Informix, Microsoft SQL Server і SQLite. На платформі Windows доступ до БД можливий через ADO (ADOdb).

3.2 Обґрунтування вибору платформи та мови реалізації

Розглянуті системи та пакети надають широкий спектр можливостей для обробки інформації та побудови систем для роботи з нею, в тому числі інструменти для ЦОС та побудови системи класифікації даних.

Проте такі системи як Mathematica та Matlab мають обмеження у використанні, оскільки у безкоштовній академічній версії вони не надають можливість використання існуючого функціоналу повністю.

Розглянутий пакет AnyLogic не було обрано для реалізації проекту з тої причини, що в якості мови програмування в ньому використовується мова Java, яка не надає достатньо високого рівня швидкодії при реалізації проектів машинного навчання.

Для реалізації СППР розпізнавання кардіограм було обрано мову Python, оскільки вона знаходиться у вільному доступі, а також надає можливість

використання великої кількості бібліотек, в яких реалізовано функції та методи, що використовуються в даній роботі.

3.3 Аналіз вимог користувача до програми

Оскільки розроблена система розпізнавання кардіограм має на меті подальше використання медичними працівниками, є потреба в розробці зручного інтерфейсу користувача, який буде інтуїтивно зрозумілим у експлуатації.

Отже, можна сформувати наступні вимоги до програми:

- Можливість завантаження до системи даних електрокардіограми конкретного пацієнта;
- Відображення сигналу ЕКГ, який було завантажено;
- Відображення результатів аналізу ЕКГ на екрані;

Основні етапи роботи програми зображено на рисунку 3.10.

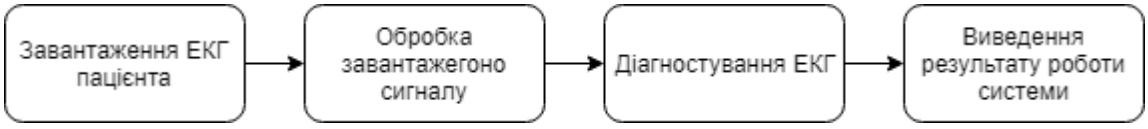


Рисунок 3.10 – Основні етапи роботи програми

3.4 Аналіз архітектури системи

На рисунку 3.11 наведено архітектуру побудованої в роботі системи.

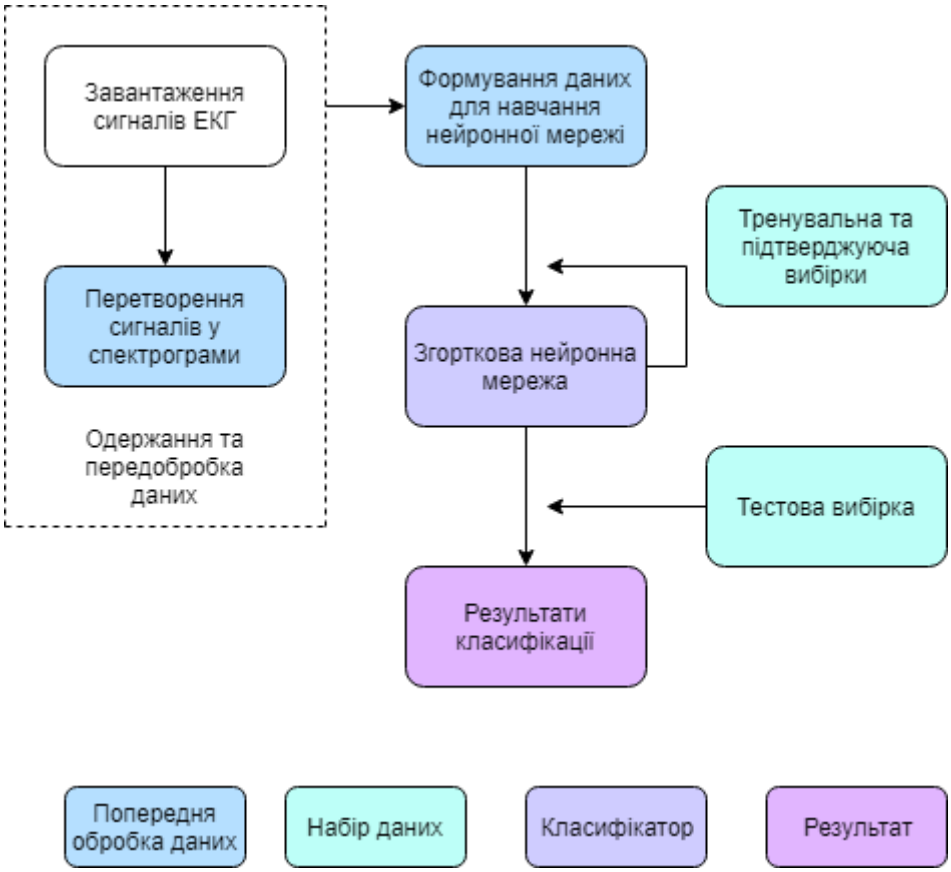


Рисунок 3.11 – Архітектура системи

Оскільки початково сигнал ЕКГ є одновимірним, а саме часовим сигналом, є потреба у переведенні його у двовимірну частотно-часову форму. Для цього використовується неперервне вейвлет-перетворення, за допомогою якого сигнал ЕКГ можна представити у вигляді спектрограми. Спектрограма дає краще розуміння динамічної поведінки системи, а також дозволяє виділяти різні сигнали системи. Так ЕКГ вимірювання людей із здоровим серцем будуть мати відмінні спектрограми від тих, що мають певне захворювання.

При аналізі роботи серцево-судинної системи людини з використанням ЕКГ зазвичай розглядають три основних серцевих відведення, тобто три складові сигналу, одержані з різних ділянок серця.

Оскільки кожен сигнал має три складові, відповідно він матиме три спектрограми. Для того, щоб передати цей сигнал згортковій нейронній мережі три спектрограми поміщаються одна на одну, внаслідок чого утворюється одне зображення з трьома каналами (рис 3.12).

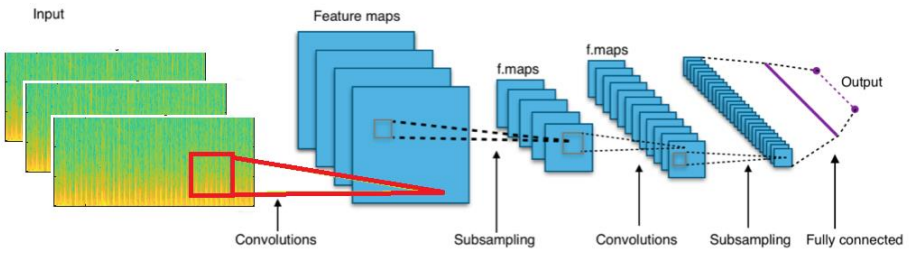


Рисунок 3.12 – Утворення триканального зображення для подачі на вхід нейронної мережі

Далі зображення подаються на вхід нейронної мережі, архітектуру якої зображено на рисунку 3.13. та описано у таблиці 3.1.

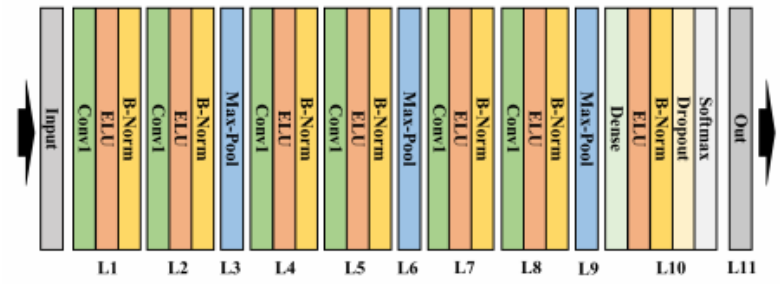


Рисунок 3.13 - Архітектура згорткової нейронної мережі

Таблиця 3.1 – Архітектура згорткової нейронної мережі

Шар	Тип	Розмір ядра	Крок	Фільтр ядра	Вхідний розмір
1	Conv2D	3*3	1	64	128*128*3
2	Conv2D	3*3	1	64	128*128*64
3	Pool	2*2	2		128*128*64
4	Conv2D	3*3	1	128	64*64*64
5	Conv2D	3*3	1	128	64*64*128
6	Pool	2*2	2		64*64*128
7	Conv2D	3*3	1	256	32*32*128
8	Conv2D	3*3	1	256	32*32*256
9	Pool	2*2	2		32*32*256
10	Full			2048	16*16*256
11	Out			8	2048

Оцінку результатів роботи ЗНМ наведено у таблиці 3.2. Усі показники є високими, тому можна вважати, що реалізація мережі для класифікації ЕКГ є вдалою.

Таблиця 3.2 – Оцінки результатів роботи ЗНМ

AUC	Accuracy	Specificity	Sensitivity
0.914	92.10	90.56	94.13

3.5 Керівництво користувача

При запуску програми відкривається головне вікно. Активною є лише клавіша «Завантажити ЕКГ» (рис.3.14), за допомогою якої можна відкрити на екрані вимірювання певного пацієнта, а саме три відведення кардіограм. Також у вікні з’явиться інформація про ПІБ пацієнта, його вік, а також дату зняття ЕКГ (рис. 3.15).

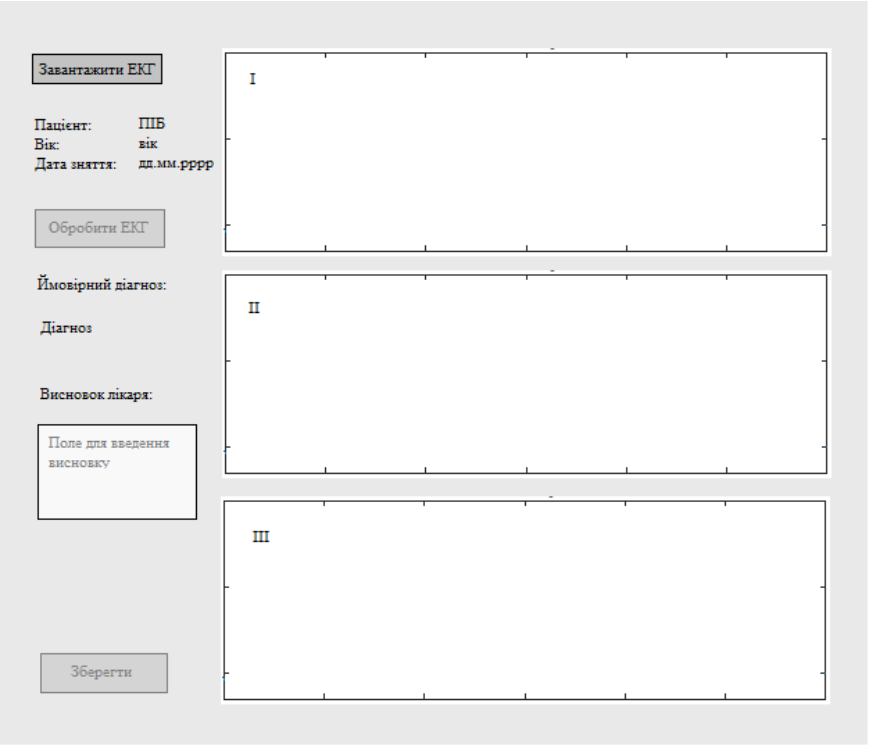


Рисунок 3.14 – Діалогове вікно програми при запуску

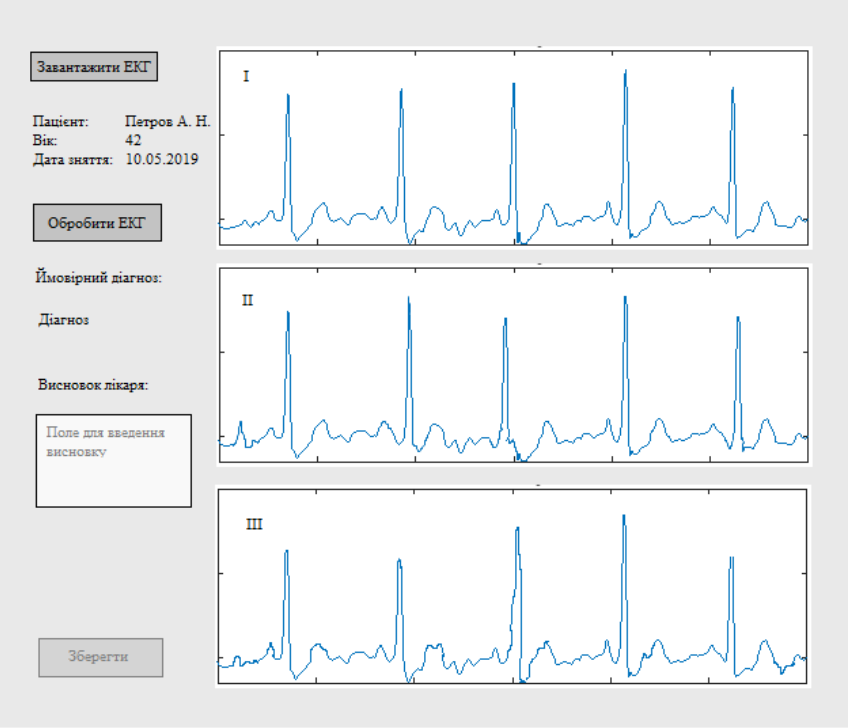


Рисунок 3.15 – Завантаження даних ЕКГ пацієнта

При натисканні клавіші «Обробити ЕКГ» виконується обробка завантаженого сигналу. На екран виводиться ймовірний діагноз пацієнта. Також є можливість введення користувачем-лікарем висновку до проведеної діагностики. Активною стає клавіша «Зберегти» в нижній частині вікна (рис. 3.16).

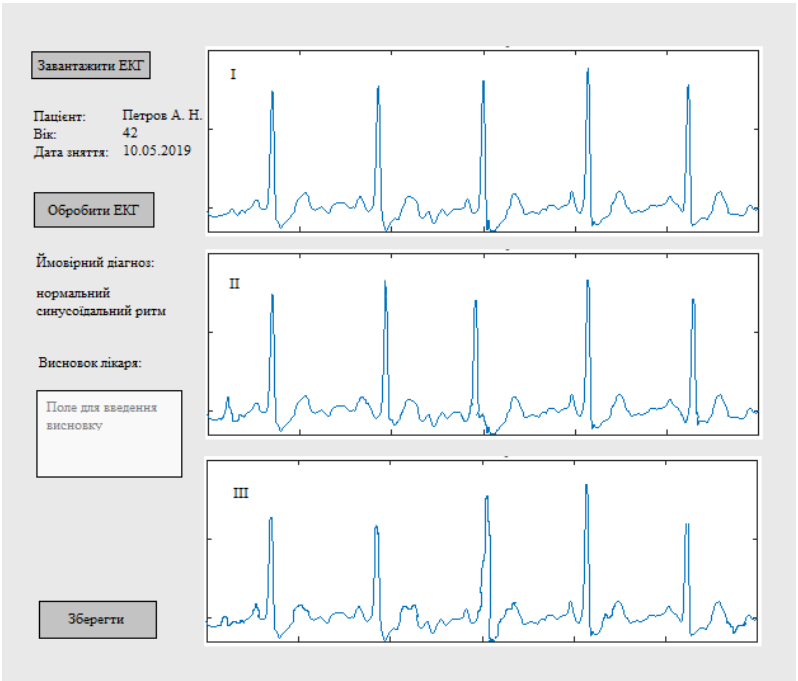


Рисунок 3.16 – Встановлення ймовірного діагнозу

Після збереження поточних даних можна завантажити іншу ЕКГ за допомогою клавіші «Зберегти ЕКГ».

3.6 Висновки до розділу

У даному розділі було розглянуто існуючі системи та пакети для цифрової обробки сигналу та побудови систем класифікації даних, на основі чого було обгрунтовано обрано мову Python як засіб розробки системи.

Проведено аналіз вимог користувача програми, сформульовано основні пункти, наведено основні етапи роботи програми. Також в даному розділі було описано архітектуру реалізованої системи, детально розглянуто архітектуру модуля класифікації. В цьому розділі наведено поетапне керівництво користувача програми з графічним відображенням кожного з етапів.

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

4.1 Постановка завдання проектування

У даному розділі проводиться оцінка основних характеристик програмного продукту, призначеного для підтримки прийняття рішень розпізнавання сигналів кардіограм. Програмний продукт був розроблений за допомогою мови програмування Python у середовищі розробки Jupyter Notebook. Інтерфейс користувача створений за допомогою технології Qt. Програмний продукт призначено для використання на персональних комп'ютерах під управлінням операційних систем MacOS, Linux, Windows.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті методу ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізі функцій програмного продукту й виявленні усіх витрат на реалізацію цих функцій.

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0– розробка програмного продукту, який виконує обробку цифрових сигналів електрокардіограм та їх класифікацію для створення системи підтримки прийняття рішення встановлення захворювання серця. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір мови програмування;

F_2 – вибір методу цифрової обробки сигналу;

F_3 – вибір методу класифікації

F_4 – інтерфейс користувача.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування Python;

б) мова програмування MatLab;

Функція F_2:

а) вейвлет-перетворення;

б) перетворення Фур’є.

Функція F_3:

а) згорткова нейронна мережа;

б) метод опорних векторів.

Функція F_4:

а) інтерфейс користувача, створений за технологією GUIDE**;**

б) інтерфейс користувача, створений за технологією Qt.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1).

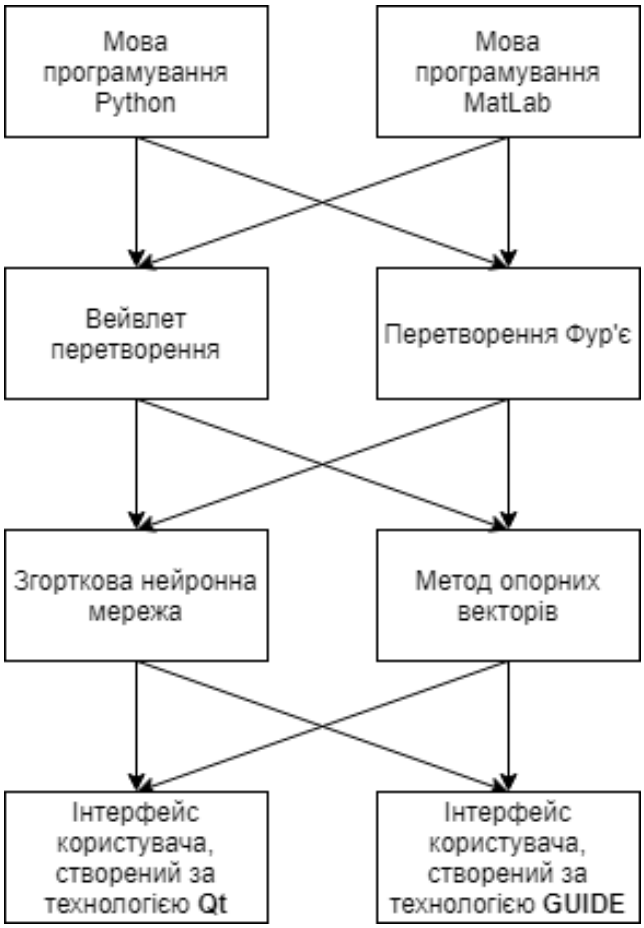


Рисунок 4.1 Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП. На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

Таблиця 4.1 - Позитивно-негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	А	Знаходиться у вільному доступі, є багато бібліотек для обробки даних та побудови нейромереж	Можуть виникати складнощі при встановленні на ПК з ОС Windows.
	Б	Є багато доступної технічної документації, інтуїтивно зрозуміла мова	Академічна версія має обмеження у використанні
F2	А	Доступний для вивчення і реалізації	Довго виконується для великих об’ємів даних
	Б	Швидкий, ефективний	Складний у вивченні
F3	А	Дає високі показники у задачах класифікації	Метод не підтверджено теоретично
	Б	Пристосований для розв’язання даного класу задач	Менш ефективний
F4	А	Широкі можливості для реалізації	Об’єм програмного коду може бути великим
	Б	Легкий у створенні	Відсутність кросплатформеності

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція F1:

Оскільки розрахунки проводяться з великими об’ємами вхідних даних, то бажаною є наявність стандартних бібліотек для обчислень. Важливою є можливість повноформатного безкоштовного використання. До того ж, їх використання знижує тривалість виконання програм, тому варіант б) має бути відкинутий.

Функція F2:

Оскільки однією з основних задач продукту є цифорова обробка сигналів, то зручнішим є той варіант, який надає більше можливостей та вищу швидкість. Отже, варіант а) має бути відкинутий

Функція F3:

Ефективність є більш важливим критерієм, тому варіант б) має бути відкинутий.

Функція F4:

Інтерфейс користувача не відіграє велику роль у даному програмному продукті, тому вважаємо варіанти а) та б) гідними розгляду.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

F1а – F2б – F3а – F4а

F1а – F2б – F3а – F4б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів ПП

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

X1 – швидкодія мови програмування;

X2 – об’єм пам’яті для збереження даних;

X3 – час обробки даних;

X4 – потенційний об’єм програмного коду.

X1: Відображає час, необхідний для виконання програмного коду.

X2: Відображає об’єм пам’яті в оперативній пам’яті ПК, необхідний для збереження та обробки даних під час виконання програми.

X3: Відображає час, який витрачається на дії.

X4: Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 - Основні параметри ПП

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	Оп/мс	19000	11000	2000
Об’єм пам’яті для збереження даних	X2	Мб	32	16	8
Час обробки даних алгоритмом	X3	мс	800	420	60
Потенційний об’єм програмного коду	X4	кількість рядків коду	2000	1500	1000

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис.4.2 – рис. 4.5.

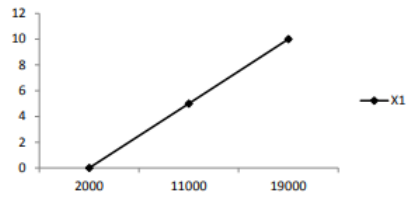


Рисунок 4.2 – Бальна оцінка швидкодії мови програмування

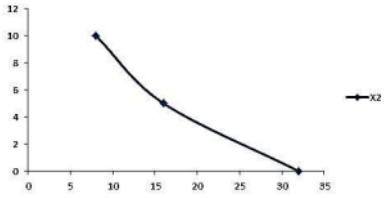


Рисунок 4.3 – Бальна оцінка об’єму пам’яті, що займається даними

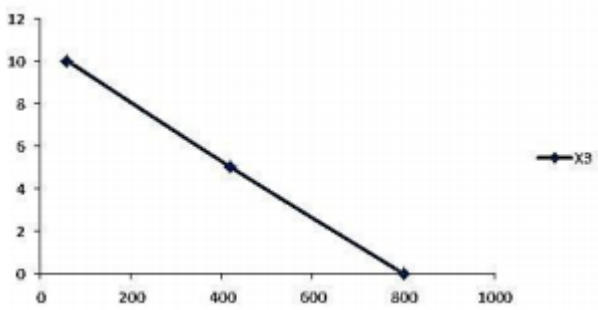


Рисунок 4.4 – Бальна оцінка часу обробки даних

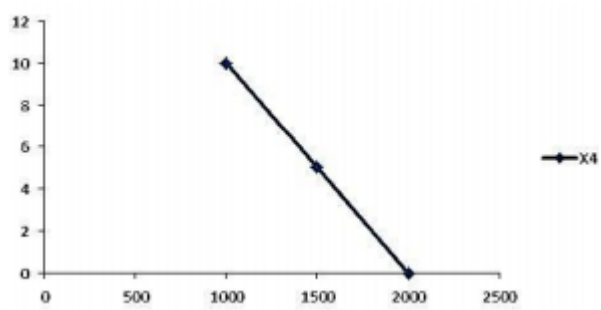


Рисунок 4.5 – Бальна оцінка точності розв’язок потенційного об’єму програмного коду

4.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 - Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	2	1	1	2	1	1	1	9	-7	49
X2	Об’єм пам’яті для збереження даних	Мб	3	2	3	1	2	3	2	16	0	0
X3	Час обробки даних алгоритмом	Мс	4	4	4	4	3	4	4	27	11	121
X4	Потенційний об’єм програмного коду	кіль-кість рядків коду	1	2	1	2	2	2	2	12	-4	16
	Разом		10	9	9	9	8	8	9	64	0	186

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 64, \quad (4.1)$$

де N – число експертів;

n – кількість параметрів;

R_{ij} – ранг.

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 16. \quad (4.2)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T. \quad (4.3)$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 186. \quad (4.4)$$

Де Δ_i^2 – відхилення;

N - число експертів.

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12 \cdot 186}{7^2(4^3-4)} = 0.75 > W_k = 0{,}67. \quad (4.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 - Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	<	>	<	<	<	<	0.5
X1 і X3	<	<	<	<	<	<	<	<	0.5
X1 і X4	>	<	=	<	<	<	<	<	0.5
X2 і X3	<	<	<	<	<	<	<	<	0.5
X2 і X4	>	=	>	<	=	>	=	>	1.5
X3 і X4	>	>	>	>	>	>	>	>	1.5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j. \\ 0.5 \text{ при } X_i < X_j \end{cases} \quad (4.6)$$

З отриманих числових оцінок переваги складемо матрицю А=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \quad (4.7)$$

де $b_i = \sum_{i=1}^{N} a_{ij}$.

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх. На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{вi}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \quad (4.8)$$

де $b_i' = \sum_{i=1}^{N} a_{ij} b_j$.

Таблиця 4.5 - Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітерація		Друга ітерація		Третя ітерація	
	1	2	3	4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1,0	0,5	0,5	0,5	2.5	0.178	6.25	0.122	39.06	0.052
X2	0,5	1,0	0,5	1,5	3.5	0.250	12.25	0.240	150.06	0.200
X3	0,5	0,5	1,0	1,5	3.5	0.250	12.25	0.240	150.06	0.200
X4	0,5	1,5	1,5	1,0	4.5	0.321	20.25	0.397	410.06	0.547
Всього:					14	1	51	1	749.24	1

4.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2* (об'єм пам'яті для збереження даних) та *X1* (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X3 (*час обробки даних) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 800 мс або варіанту б) 80мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j}\,, \qquad (4.9)$$

де n – кількість параметрів;

$K_{вi}$– коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	11000	5.8	0.052	0,3016
F2(X2)	А	16	4.9	0.200	0,98
F3(X3)	Б	420	4.7	0.200	0,94
F4(X4)	А	1500	5.4	0.547	2,9538
	Б	1100	1.3	0.547	0,71

За даними з таблиці 4.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \quad (4.10)$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 0{,}3016 + 0{,}98 + 0{,}94 + 2{,}9538 = 5{,}1754,$$
$$K_{K2} = 0{,}3016 + 0{,}98 + 0{,}94 + 0{,}71 = 2{,}9316.$$

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \quad (4.11)$$

де T_P – трудомісткість розробки ПП;

$K_П$ – поправочний коефіцієнт;

$K_{СК}$ – коефіцієнт на складність вхідної інформації;

K_M – коефіцієнт рівня мови програмування;

$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.8. Тоді загальна трудомісткість

програмування першого завдання дорівнює: T_1 = 90·1.7·0.8 = 122.4 людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $T_Р$ =19 людино-днів, $К_П$ =1.42,$К_{СК}$ = 1,$К_{СТ}$ =0.8.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

T_I = (122.4 + 21,584 + 4.8 + 21,584) · 8 = 1362,944 людино-годин.

T_{II} = (122.4 + 21,584 + 6.91 + 21,584) · 8 = 1379.744 людино-годин.

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 8000 грн., один фінансовий аналітик з окладом 12000грн. Визначимо середню зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} \text{ грн.}, \quad (4.12)$$

де М – місячний оклад працівників;

T_m – кількість робочих днів тиждень;

t – кількість робочих годин в день.

$$С_Ч = \frac{8000 + 8000 + 12000}{3 \cdot 21 \cdot 8} = 55.55 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою:

$$С_{ЗП} = С_Ч \cdot Т_i \cdot К_Д, \quad (4.13)$$

де $С_Ч$– величина погодинної оплати праці програміста;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП} = = 55.55 \cdot 1362,944 \cdot 1.2 = 90853,85$ грн.

II. $С_{ЗП} = 55.55 \cdot 1379.744 \cdot 1.2 = 91973,74$ грн.

Відрахування на єдиний соціальний внесок в залежності від групи ризику (II клас) становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 90853,85 \cdot 0.22 = 19987,847$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 91973,74 \cdot 0.22 = 20234,74$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 8000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 8000 \cdot 0,2 = 19200 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 19200 \cdot (1 + 0.2) = 23040 \text{ грн.}$$

Відрахування на єдиний соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 23040 \cdot 0,22 = 5068,8 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 20% та вартості ЕОМ – 10000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.2 \cdot 10000 = 2300 \text{ грн.},$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 10000 \cdot 0.05 = 575 \text{ грн.},$$

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4 \text{ годин},$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706.4 \cdot 0{,}32 \cdot 0.3 \cdot 2.7515 = 450{,}73 \text{ грн.},$$

Де $N_С$ – середньо-споживча потужність приладу;

$К_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 10000 \cdot 0,67 = 6700 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н, \quad (4.14)$$

$$С_{ЕКС} = 23040 + 5068,8 + 2300 + 575 + 450,73 + 6700 = 38592,74 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 38592,74 / 1706.4 = 22,61 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М\text{-}Г} \cdot Т, \quad (4.15)$$

I. $С_М = 22,61 \cdot 1362,944 = 31274.24$ грн.

II. $С_М = 22,61 \cdot 1379.744 = 30987.48$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67, \quad (4.16)$$

I. $С_Н = 90853,85 \cdot 0,67 = 60872,08$ грн.

II. $С_Н = 91973,74 \cdot 0,67 = 61622,41$ грн.

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_{М} + С_{Н}, \quad (4.17)$$

I. $С_{ПП} = 90853,85 + 19987,84 + 33174.05 + 60872,08 = 201452,71$ грн.

II. $С_{ПП} = 91973,74 + 20234,74 + 33582.99 + 61622,41 = 208798,18$ грн.

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕРj} = К_{Кj} / С_{Фj}, \quad (4.18)$$

$$К_{ТЕР1} = 5.285 / 201452,71 = 0,26 \cdot 10^{-4},$$
$$К_{ТЕР2} = 5.947 / 208798,18 = 0,28 \cdot 10^{-4}.$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 0,26 \cdot 10^{-4}$.

4.8 Висновки до розділу

У даній роботі було проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР} = 0,24 \cdot 10^{-4}$.

Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Python;
- вейвлет-перетворення;
- згорткова нейронна мережа;
- інтерфейс користувача, створений за технологією Qt.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, непоганий функціонал і швидкодію.

ВИСНОВКИ

У ході виконання даної роботи було проведено аналіз вимог до системи розпізнавання ЕКГ. Було зроблено огляд вже існуючих систем розпізнавання кардіограм, визначено їх основний функціонал. Розглянуто основні аспекти анатомії та фізіології серця (будова серця, серцеві скорочення, стан серця здорової людини та тієї, що має захворювання тощо) і їх зв'язок з ЕКГ (формування зубців, комплексів, інтервалів серцевих скорочень).

Детально розглянуто два найбільш часто використовуваних засоби цифрової обробки сигналу, а саме перетворення Фур'є та вейвлет-перетворення. Наведено переваги та недоліки кожного з засобів. Названо причини обрання саме вейвлет-перетворення як засобу ЦОС в цій роботі.

Було зроблено огляд методів класифікації даних, а саме нейронних мереж (в тому числі і згорткових) та методу опорних векторів. Виконано порівняння існуючих систем, що використовують ці методи, на основі чого було обрано метод класифікації, який реалізовується в цій роботі, а саме згорткова нейронна мережа.

Виконано аналіз систем та пакетів для цифрової обробки сигналу та побудови систем класифікації даних, їх функціоналу та можливостей. Перевагу було віддано мові програмування Python за рахунок того, що вона знаходиться у вільному доступі, а також надає можливість використання готових бібліотек машинного навчання.

Проведено аналіз вимог користувача до інтерфейсної частини системи. Побудовано діаграму, яка описує архітектуру реалізованої системи в цілому. Окремо детально розглянуто архітектуру модуля класифікації, а саме згорткової нейронної мережі. Описано одержані результати роботи системи. Наведено покрокове керівництво користувача програми.

ПЕРЕЛІК ПОСИЛАНЬ

1. Mind-Reading Machines: Automated Inference of Complex Mental States [Електронний ресурс] – URL: affect.media.mit.edu/projectpages/esp/elkaliouby-Phd.pdf/ (дата звернення 08.05.2019)
2. Electrocardiography - Wikipedia [Електронний ресурс] – URL: en.wikipedia.org/wiki/Electrocardiography/ (дата звернення 08.05.2019)
3. Interwoven W. "Un nouveau galvanometre" / Interwoven W .- Berlin, 1901.- 625. 35.Naming of the Waves in the ECG, With a Brief Account of Their Genesis [Електронний ресурс] – URL: http://circ.ahajournals.org/content/98/18/1937?download=true (дата звернення 08.05.2019)
4. Adaptive baseline wander removal in the ECG: Comparative analysis with cubic spline technique [Електронний ресурс] – URL: https://ieeexplore.ieee.org/document/269426/ (дата звернення 08.05.2019)
5. Comparative study of FIR and IIR filters for the removal of Baseline noises from ECG signal [Електронний ресурс] – URL: https://ieeexplore.ieee.org/document/269426/ (дата звернення 08.05.2019)
6. An Optimized Filter System For Eliminating 50 Hz Interference from High Resolution ECG [Електронний ресурс] – URL: https://www.ncbi.nlm.nih.gov/pubmed/7772709 (дата звернення 08.05.2019)
7. Frequency-Domain Digital Filtering Techniques for the Removal Power Line Noise with Application to the Electrocardiogram [Електронний ресурс] – URL: https://www.sciencedirect.com/science/article/pii/001048099090035B (дата звернення 08.05.2019)
8. Filtering Electrocardiographic Signals using an unbiased and normalized adaptive noise reduction system [Електронний ресурс] – URL:

https://www.medengphys.com/article/S1350-4533(08)00046-5/abstract (дата звернення 08.05.2019)

9. Detection of Fetal ECG with IIR Adaptive Filtering and Genetic Algorithm [Електронний ресурс] – URL: http://www.masys.url.tw/AU/2015SP/BMSDD/HW/HWfinalMaternal_Fetal_ECG/xx-Kam.pdf (дата звернення 08.05.2019)
10. FIR Filter Design Analysis for Power Line Interference in ECG Signals [Електронний ресурс] – URL: http://www.ijirst.org/articles/IJIRSTV1I6081.pdf (дата звернення 08.05.2019)
11. Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition [Електронний ресурс] – URL: https://arxiv.org/pdf/1410.4281.pdf (дата звернення 08.05.2019)
12. Recurrent neural network - Wikipedia [Електронний ресурс] – URL: en.wikipedia.org/wiki/Recurrent_neural_network / (дата звернення 08.05.2019)
13. Long Short-Term Memory [Електронний ресурс] – URL: http://www.bioinf.jku.at/publications/older/2604.pdf (дата звернення 08.05.2019)
14. Deep Learning in Neural Networks: An Overview [Електронний ресурс] – URL: https://arxiv.org/pdf/1404.7828.pdf (дата звернення 08.05.2019)
15. Multilingual Language Processing From Bytes [Електронний ресурс] – URL: https://arxiv.org/pdf/1512.00103.pdf (дата звернення 08.05.2019)
16. Convolutional neural network - Wikipedia [Електронний ресурс] – URL: en.wikipedia.org/wiki/Convolutional_neural_network/ (дата звернення 08.05.2019)
17. Згорткова нейронна мережа - Вікіпедія [Електронний ресурс] – URL: en.wikipedia.org/wiki/Згорткова_нейронна_мережа/ (дата звернення 08.05.2019)

18. Cardiologist-level arrhythmia detection with convolutional neural networks [Електронний ресурс] – URL: https://arxiv.org/pdf/1707.01836.pdf (дата звернення 08.05.2019) 1
19. Precision and recall - Wikipedia [Електронний ресурс] – URL: en.wikipedia.org/wiki/Precision_and_recall/ (дата звернення 08.05.2019)
20. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization [Електронний ресурс] – URL: http://jmlr.org/papers/v12/duchi11a.html (дата звернення 08.05.2019)
21. On Baseline Drift Suppressing in ECG Recording [Електронний ресурс] – URL: http://www.ijreat.org/Papers%202015/Issue13/IJREATV3I1012.pdf (дата звернення 08.05.2019)
22. Пашин В.П. Функционально-стоимостный анализ конструкторско-технологических решений. - К.: РДЭНТП «Знание» УССР, 1989. - 22с.
23. Пашін В.П. Оцінка конкурентоспроможності електронних пристроїв на стадії проектування. - К. Економічний вісник НТУУ „КПІ", 2006. - №3. с. 252-255.
24. Пашин В.П. Управление качеством изделий на основе функционально-стоимостного анализа. - К.: «Технология и организация производства», 1989. - №1. с. 17-19.
25. Пашін В. П. Методичні вказівки до виконання економіко-організаційного розділу дипломних проектів (робіт) бакалаврів і спеціалістів для студентів Інституту прикладного системного аналізу: Навч. посібник / Пашін В. П., Романов В. В., Єгорова Н. В – К.: НТУУ “КПІ”, 2011. – 118 с.

Додаток А Лістинг програми

```
activities_description = {
    1: 'SVEB',
    2: 'N',
    3: 'VEB',
    4: 'F',
    5: 'SBR',
    6: 'VT'}
def read_signals(filename):
    with open(filename, 'r') as fp:
        data = fp.read().splitlines()
        data = map(lambda x:
x.rstrip().lstrip().split(), data)
        data = [list(map(float, line)) for line in data]
    return data
def read_labels(filename):
    with open(filename, 'r') as fp:
        activities = fp.read().splitlines()
        activities = list(map(lambda x: int(x)-1,
activities))
    return activities
def randomize(dataset, labels):
    permutation =
np.random.permutation(labels.shape[0])
    shuffled_dataset = dataset[permutation, :, :]
    shuffled_labels = labels[permutation]
    return shuffled_dataset, shuffled_labels
INPUT_FILES_TRAIN = ['first_lead_train.txt',
'second_lead_train.txt', 'third_lead_train.txt']
INPUT_FILES_TEST = ['first_lead_test.txt',
'second_lead_test.txt', 'third_lead_test.txt']
LABELFILE_TRAIN = 'y_train.txt'
LABELFILE_TEST = 'y_test.txt'
train_signals, test_signals = [], []
for input_file in INPUT_FILES_TRAIN:
    signal = read_signals(input_file)
    train_signals.append(signal)
train_signals =
np.transpose(np.array(train_signals), (1, 2, 0))
for input_file in INPUT_FILES_TEST:
    signal = read_signals(input_file)
    test_signals.append(signal)
test_signals =
np.transpose(np.array(test_signals), (1, 2, 0))
train_labels =
read_labels(LABELFILE_TRAIN)
test_labels = read_labels(LABELFILE_TEST)
[no_signals_train, no_steps_train,
no_components_train] = np.shape(train_signals)
[no_signals_test, no_steps_test,
no_components_test] = np.shape(test_signals)
no_labels = len(np.unique(train_labels[:]))
print("The train dataset contains {} signals, each
one of length {} and {} components
".format(no_signals_train, no_steps_train,
no_components_train))
print("The test dataset contains {} signals, each
one of length {} and {} components
".format(no_signals_test, no_steps_test,
no_components_test))
print("The train dataset contains {} labels, with
the following distribution:\n
{}".format(np.shape(train_labels)[0],
Counter(train_labels[:])))
print("The test dataset contains {} labels, with
the following distribution:\n
{}".format(np.shape(test_labels)[0],
Counter(test_labels[:])))
uci_har_signals_train, uci_har_labels_train =
randomize(train_signals, np.array(train_labels))
uci_har_signals_test, uci_har_labels_test =
randomize(test_signals, np.array(test_labels))
scales = range(1,128)
waveletname = 'morl'
train_size = 7000
train_data_cwt = np.ndarray(shape=(train_size,
127, 127, 3))

for ii in range(0,train_size):
    if ii % 1000 == 0:
        print(ii)
    for jj in range(0,3):
        signal = uci_har_signals_train[ii, :, jj]
        coeff, freq = pywt.cwt(signal, scales,
waveletname, 1)
        coeff_ = coeff[:,:127]
        train_data_cwt[ii, :, :, jj] = coeff_

test_size = 1000
test_data_cwt = np.ndarray(shape=(test_size,
127, 127, 3))
for ii in range(0,test_size):
    if ii % 100 == 0:
        print(ii)
    for jj in range(0,3):
        signal = uci_har_signals_test[ii, :, jj]
        coeff, freq = pywt.cwt(signal, scales,
waveletname, 1)
        coeff_ = coeff[:,:127]
        test_data_cwt[ii, :, :, jj] = coeff_
x_train = train_data_cwt
y_train = list(uci_har_labels_train[:train_size])
x_test = test_data_cwt
y_test = list(uci_har_labels_test[:test_size])
img_x = 127
img_y = 127
img_z = 3
num_classes = 6
batch_size = 16
epochs = 10
# reshape the data into a 4D tensor -
(sample_number, x_img_size, y_img_size,
num_channels)
# because the MNIST is greyscale, we only have
a single channel - RGB colour images would
have 3
input_shape = (img_x, img_y, img_z)
# convert the data to the right type
#x_train = x_train.reshape(x_train.shape[0],
img_x, img_y, img_z)
```

```
#x_test = x_test.reshape(x_test.shape[0], img_x,
img_y, img_z)
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')
# convert class vectors to binary class matrices -
this is for use in the
# categorical_crossentropy loss below
y_train = keras.utils.to_categorical(y_train,
num_classes)
y_test = keras.utils.to_categorical(y_test,
num_classes)
model = Sequential() model.add(Conv2D(64,
(3,3),strides = (1,1), input_shape =
IMAGE_SIZE +
[3],kernel_initializer='glorot_uniform'))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(Conv2D(64, (3,3),strides =
(1,1),kernel_initializer='glorot_uniform'))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(MaxPool2D(pool_size=(2, 2),
strides= (2,2))) model.add(Conv2D(128,
(3,3),strides =
(1,1),kernel_initializer='glorot_uniform'))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(Conv2D(128, (3,3),strides =
(1,1),kernel_initializer='glorot_uniform'))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(MaxPool2D(pool_size=(2, 2),
strides= (2,2))) model.add(Conv2D(256,
(3,3),strides =
(1,1),kernel_initializer='glorot_uniform'))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(Conv2D(256, (3,3),strides =
(1,1),kernel_initializer='glorot_uniform'))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(MaxPool2D(pool_size=(2, 2),
strides= (2,2))) model.add(Flatten())
model.add(Dense(2048))
model.add(keras.layers.ELU())
model.add(BatchNormalization())
model.add(Dropout(0.5)) model.add(Dense(7,
activation='softmax'))
model.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])
model.fit(x_train, y_train,
batch_size=batch_size,
        epochs=epochs, verbose=1,
        validation_data=(x_test, y_test),
        callbacks=[history])
train_score = model.evaluate(x_train, y_train,
verbose=0)
print('Train loss: {}, Train accuracy:
{}'.format(train_score[0], train_score[1]))
test_score = model.evaluate(x_test, y_test,
verbose=0)
print('Test loss: {}, Test accuracy:
{}'.format(test_score[0], test_score[1]))
```

Додаток Б Ілюстративний матеріал

СИСТЕМА ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ РОЗПІЗНАВАННЯ КАРДІОГРАМ МЕТОДАМИ ШТУЧНОГО ІНТЕЛЕКТУ

Кравченко Влада

ОБ'ЄКТ, ПРЕДМЕТ ТА МЕТА ДОСЛІДЖЕННЯ

- Об'єкт дослідження – набір даних каріограм пацієнтів з захворюваннями серця.
- Предмет дослідження – засоби попередньої обробки цифрових сигналів та методи класифікації на основі згорткових нейронних мереж.
- Мета роботи – розробка системи класифікації сигналів кардіограм для діагностики захворювань серця.

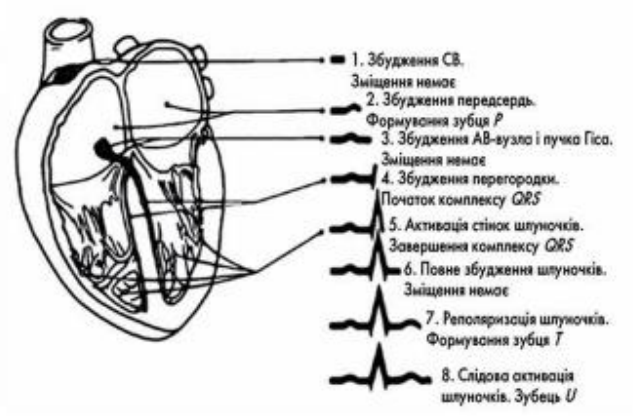
ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

- Дослідити фізіологію серця та її зв’язок с електрокардіограмами;
- Вивчити існуючі підходи та системи для вирішення задачі обробки часових сигналів;
- Дослідити методи обробки часових сигналів;
- Дослідити методи прийняття рішень;
- Реалізувати обробку сигналів кардіограм;
- Реалізувати блок прийняття рішень;
- Провести навчання системи.

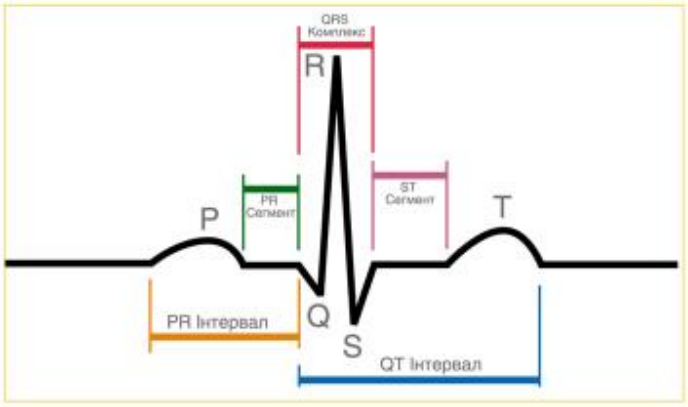
ЕЛЕКТРОКАРДІОГРАМА

- ЕКГ - це запис коливань різниці потенціалів, що виникають на поверхні збудливої тканини або в оточуючому серце провідному середовищі при поширенні хвилі збудження по серцю.

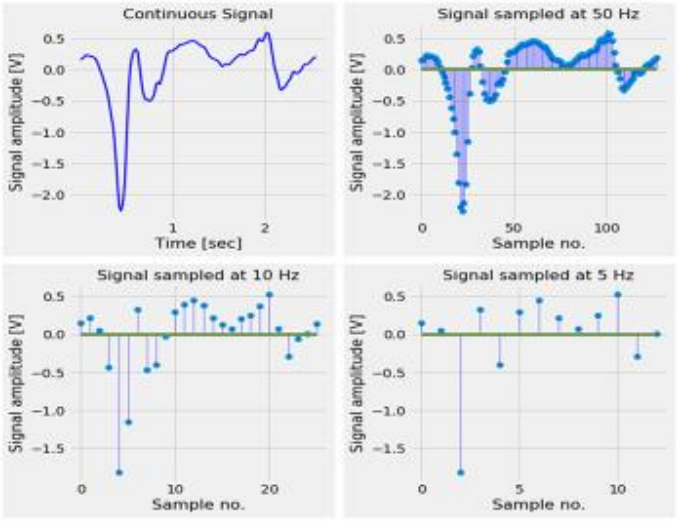
Функціонування серця, зв’язок з ЕКГ



ФУНКЦІОНУВАННЯ СЕРЦЯ, ЗВ’ЯЗОК З ЕКГ



ОБРОБКА ЧАСОВОГО СИГНАЛУ

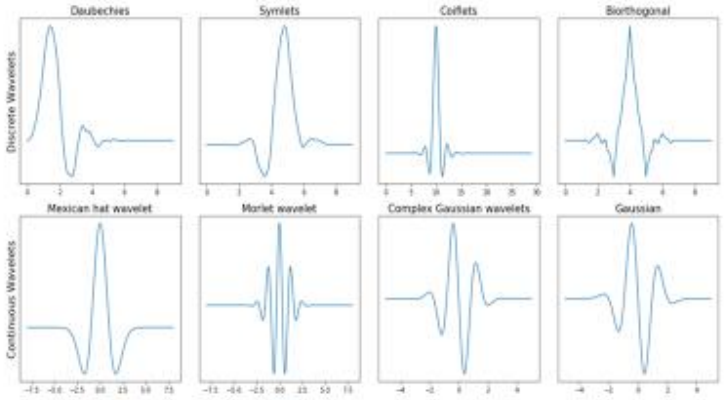


ПЕРЕТВОРЕННЯ ФУР’Є

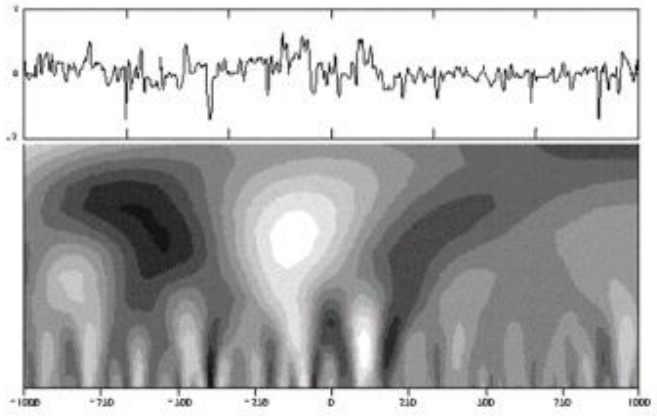
- Періодичний дискретний сигнал можна розкласти в кінцевий ряд Фур'є.
- Сенс цього розкладання полягає в тому, що функція представляється у вигляді суми «ряду» синусоїд з різними амплітудами.

ВЕЙВЛЕТ ПЕРЕТВОРЕННЯ

- Вейвлет (англ. Wavelet) - це «маленька хвиля», енергія якої кінцева і локально сконцентрована в часі.



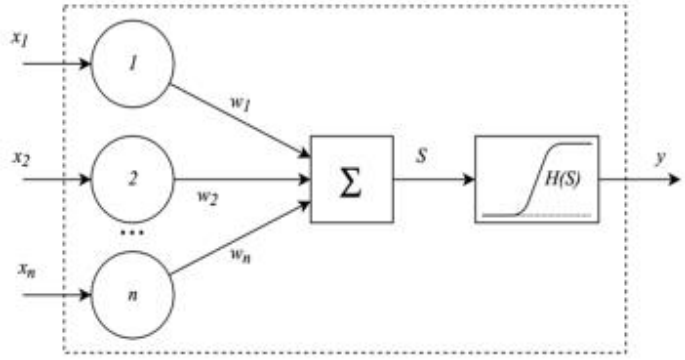
ПРЕДСТАВЛЕННЯ РЕЗУЛЬТАТІВ ВЕЙВЛЕТ-ПЕРЕТВОРЕННЯ



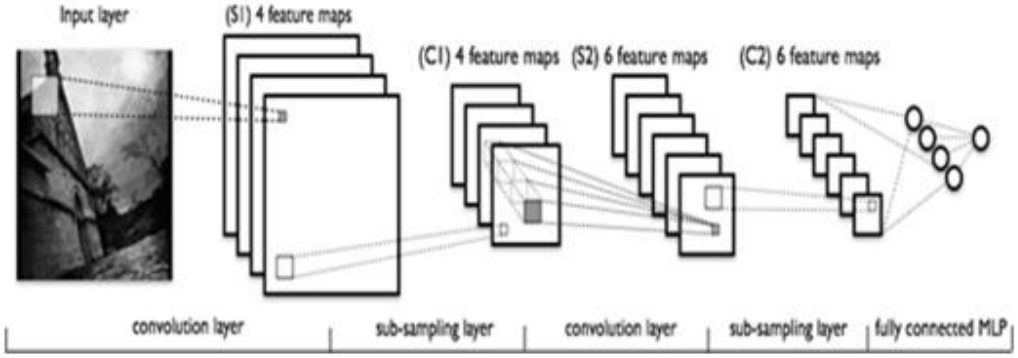
ЗАДАЧА КЛАСИФІКАЦІЇ

- Дано: Множина об'єктів $T = \{t_1, t_2, \dots, t_n\}$ – навчальна вибірка
- $t_i \rightarrow \{x_1, x_2, \dots, x_m, y\}$
- $X = \{x_1, x_2, \dots, x_m\}$ – незалежні змінні (атрибути)
- $C_h = \{c_{h1}, c_{h2}, \dots, c_{hq_h}\}$ - значення, які приймає x_h
- y – залежна змінна
- $V = \{v_1, v_2, \dots, v_s\}$ – значення, які приймає y
- Знайти: спрогнозувати значення y при нових значеннях незалежних змінних.

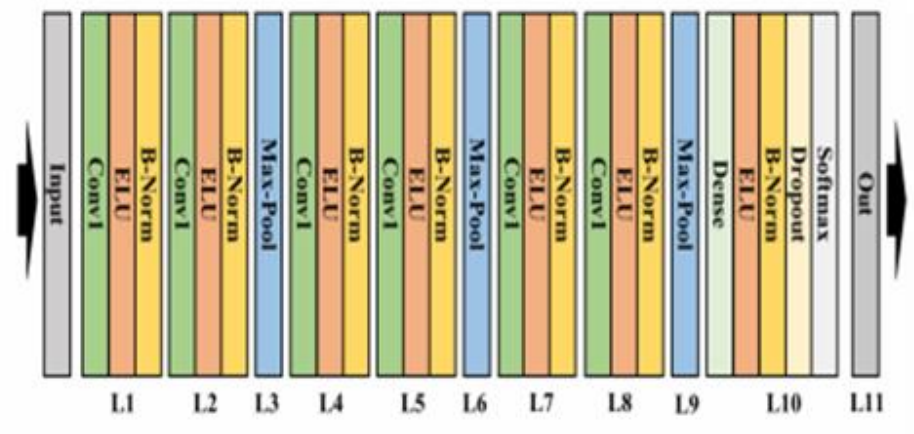
ОДНОШАРОВА НЕЙРОННА МЕРЕЖА

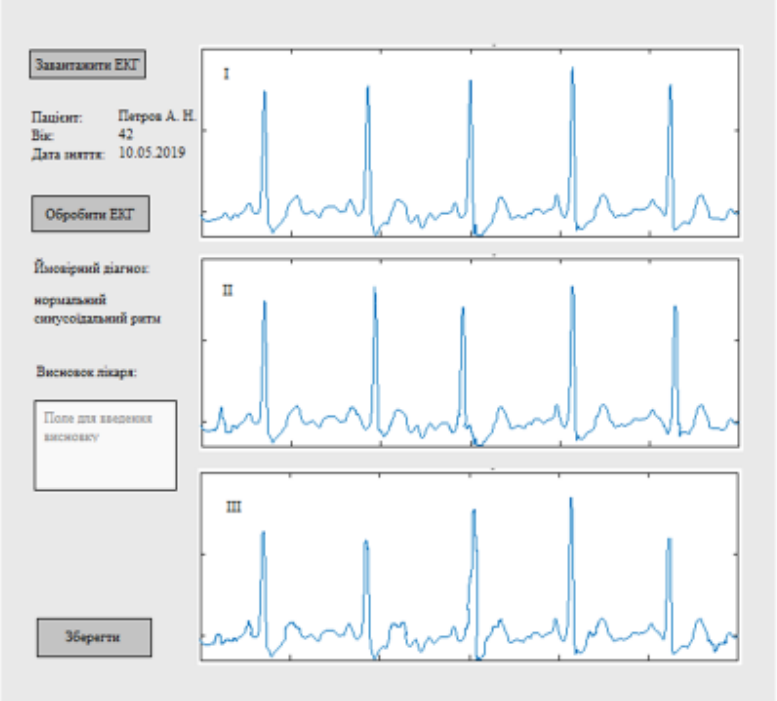


ЗГОРТКОВА НЕЙРОННА МЕРЕЖА



АРХІТЕКТУРА ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ





ДЯКУЮ ЗА УВАГУ!