

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ, РОЗРОБКА ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
МЕТОДІВ АНАЛІЗУ ТА КЛАСИФІКАЦІЇ БАГАТОВИМІРНИХ СИГНАЛІВ
RESEARCH, DEVELOPMENT AND SOFTWARE IMPLEMENTATION OF
METHODS FOR ANALYSIS AND CLASSIFICATION OF
MULTIDIMENSIONAL SIGNALS

Виконав: студент(ка) 5 курсу, групи КНТ-219м
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

Лимарєв І.О.

(прізвище та ініціали)

Керівник Субботін С.О.

(прізвище та ініціали)

Рецензент Гофман Є.О.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Системи штучного інтелекту
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Лимарєва Ігоря Олеговича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження, розробка та програмна реалізація методів аналізу та класифікації багатовимірних сигналів. Research, Development and Software Implementation of Methods for Analysis and Classification of Multidimensional Signals

керівник проєкту (роботи) Субботін Сергій Олександрович, д.т.н., професор,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “3” листопада 2020 року № 301

2. Строк подання студентом проєкту (роботи) 5 грудня 2020 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми та постановка завдань дослідження. 2. Матеріали і методи. 3. Розробка архітектури програми. 4. Розробка програми. 5. Експлуатація, тестування та експериментальне дослідження програми. 6. Економічно-організаційна частина. 7. Охорона праці та безпека у надзвичайних ситуаціях. 8. Додатки.

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

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Субботін С.О., зав. каф. ПЗ		
Економічно-організаційна частина	Пожуєва Т.О., професор		
Охорона праці та безпека у надзвичайних ситуаціях	Коробко О.В., ст. викладач		
Нормоконтролер	Калініна М.В., асистент		

7. Дата видачі завдання “ 4 ” вересня 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тижні	Розділ 1
3	Розробка та удосконалення методів, моделей й алгоритмів вирішення задачі.	4–5 тижні	Розділ 2
4	Розробка архітектури програми.	6 тиждень	Розділ 3
5	Розробка програми.	7–8 тижні	Розділ 4
6	Тестування та експериментальне дослідження програмного забезпечення.	9 тиждень	Розділ 5
7	Розробка економічно-організаційної частини.	10 тиждень	Розділ 6
8	Розробка заходів з охорони праці та безпеки у надзвичайних ситуаціях.	11 тиждень	Розділ 7
9	Оформлення пояснювальної записки та документів до неї.	12-13 тиждень	Додатки
10	Нормоконтроль та рецензування.	14–15 тижні	
11	Захист роботи.	16 тиждень	

Студент(ка)

(підпис)

Лимарєв І.О.

(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Субботін С.О.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
123 с., 27 рис., 12 табл., 2 дод., 79 джерел.

БАГАТОВИМІРНИЙ СИГНАЛ, КОМП'ЮТЕРНА ПРОГРАМА,
КЛАСИФІКАЦІЯ, ЕЛЕКТРОМІОГРАФІЯ, НЕЙРОННА МЕРЕЖА.

Об'єкт дослідження – процес класифікації сигналів в системах діагностування.

Предмет дослідження – методи та програмні засоби класифікації багатовимірних сигналів для діагностування.

Мета роботи – створення методів програмної системи для автоматизації класифікації багатовимірних сигналів.

Матеріали, методи та технічні засоби: методи машинного навчання, методи глибокого навчання нейромереж, структурне та об'єктно-орієнтоване програмування, технологія електроміографії, мова програмування Python, персональний комп'ютер з процесором Intel Core 2 Duo під управлінням операційної системи Linux Ubuntu 20.04.

Результати. Створено методи для автоматичної класифікації багатовимірних сигналів та специфічні методи для попередньої обробки сигналів електроміографії. Розроблені методи реалізовано у бібліотеці функцій на мові програмування Python, які реалізують автоматичну класифікацію багатовимірних сигналів за даними діагностування.

Висновки. Розроблено методи та програмну систему для автоматичної класифікації багатовимірних сигналів, яка автоматизує та прискорює процес діагностування.

Галузь використання – діагностування та класифікація процесів за даними багатовимірних сигналів.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 123 pages, 27 figures, 12 tables, 2 appendixes, 79 sources.

MULTIDIMENSIONAL SIGNAL, COMPUTER PROGRAM, CLASSIFICATION, ELECTROMIOGRAPHY, NEURAL NETWORK.

Object of study is a software process of signal classification in diagnosis systems.

Subject of study are methods and software for classifying multidimensional signals for diagnosis.

The purpose of the work is to create a software system for automation of diagnostics of electromyography signals under the supervision of a doctor or for use by patients in the process of telemedicine.

Materials, methods, and tools: machine learning methods, deep learning methods, structural and object-oriented programming, electromyography technology, Python programming language, personal computer with Intel Core 2 Duo processor running Linux Ubuntu 20.04 operating system.

Results. The methods for automatic classification of multidimensional signals of electromyography is created. Developed methods are implemented in the library of functions in the Python programming language, which implement automatic diagnosis of the patient for the presence or absence of knee muscle injury.

Conclusions. The methods and software system has been developed for automatic classification of multidimensional signals, which automate and accelerate the diagnostic process.

The field of use is a diagnosis and classification of processes according to multidimensional signals.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	8
Вступ	9
1 Аналіз проблеми та постановка завдань дослідження	10
1.1 Види сигналів	10
1.2 Задача класифікації сигналів	11
1.3 Глибокі нейронні мережі у задачі класифікації багатовимірних сигналів	12
1.3.1 Багатошаровий перцептрон	14
1.3.2 Згорткова нейронна мережа	16
1.3.3 Аналіз і порівняння наскрізних моделей глибокого навчання у задачі класифікації багатовимірних сигналів	18
1. 4 Постановка завдань роботи	19
1.5 Висновки за розділом 1	20
2 Матеріали і методи	21
2.1 Методи виділення ділянок активності одновимірного ЕМГ сигналу ...	21
2.2 Методи виділення ділянок багатовимірного ЕМГ сигналу для синтезу НМ	25
2.3 Висновки за розділом 2	28
3 Розробка архітектури програми	30
3.1 Опис основних вимог до програми	30
3.2 Структура програмної системи та функціонування програми	30
3.3 Вибір мови програмування та інструментарію для розробки програми...	32
3.4 Вимоги до технічних і програмних засобів	34
3.5 Висновки за розділом 3	34
4 Розробка програми	35
4.1 Опис основних вимог до програми	35
4.1.1 Модуль введення-виведення даних	36
4.1.2 Модуль попередньої обробки даних	37

4.1.3 Модуль створення НМ	39
4.2 Інформаційне забезпечення	41
4.3 Технічні засоби	42
4.4 Висновки за розділом 4	42
5 Експлуатація, тестування та експериментальне дослідження програми	43
5.1 Опис застосування програми	43
5.2 Інструкція по роботі з програмою	44
5.2.1 Встановлення програми	44
5.2.2 Запуск програми	45
5.2.3 Методика роботи з програмою	46
5.3 Приклад роботи користувача з програмою	47
5.4 Методика тестування програми	49
5.5 Практичні завдання для тестування та експериментального дослідження програми	50
5.6 Результати тестування та експериментального дослідження програми....	52
5.7 Висновки за розділом 5	55
6 Організаційно-економічна частина	56
7 Охорона праці та безпека у надзвичайних ситуаціях.....	70
7.1 Аналіз потенційних небезпек	70
7.2 Заходи із забезпечення безпеки	71
7.3 Заходи з виробничої санітарії та гігієни праці.....	73
7.4 Заходи безпеки у надзвичайних ситуаціях.....	77
7.4.1 Заходи з пожежної безпеки	77
7.4.2 Заходи з цивільної безпеки	79
Висновки.....	84
Перелік джерел посилання	85
Додаток А Текст програми	94
Додаток Б Слайди презентації.....	114

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

COTE	– Collective Of Transformation-Based Ensembles;
DTW	– Dynamic Time Warping;
HIVE-COTE	– Hierarchical Vote Collective of Transformation-Based Ensembles;
ISO	– International Organization for Standardization;
MCM	– Move Split Merge;
SVM	– Support Vector Machine;
TWE	– Time Warp Edit;
ДБН	– державні будівельні норми;
ДСТУ	– державний стандарт України;
ЕМГ	– електроміографія;
ЄСВ	– єдиний соціальний внесок;
МОЗ	– Міністерство охорони здоров'я;
НАПБ	– нормативно-правовий акт з питань пожежної безпеки;
НМ	– нейронна мережа;
НПАОП	– нормативно-правові акти з охорони праці;
ОС	– операційна система;
ПЗ	– програмне забезпечення;
ПК	– персональний комп'ютер.

ВСТУП

Класифікація процесів за даними багатовимірного сигналу, отриманого з декілької фізичних датчиків та синхронізованих у часі є важливою задачею при побудові програмних систем для автоматичного діагностування. Побудова якісних моделей для класифікації таких сигналів дозволить автоматизувати процес їх аналізу та діагностування.

У дипломній роботі розглядається процес діагностування екземпляру вибірки за даними багатовимірного сигналу. Кожен екземпляр характеризується значеннями декількох параметрів у кожен момент часу. Для навчальної вибірки експерти розмічують екземпляри на їх приналежність до одного із заздалегіть визначених класів.

Побудова класифікатора, що навчається на даних такої вибірки допоможе у вирішенні практичної задачі. Розглянута робота методів класифікації та попередньої обробки даних для задач аналізу та класифікації пацієнтів за багатовимірними ЕМГ сигналами. Вони допомагають виділити інформативні проміжки багатовимірного сигналу діагностування, розширити початкову вибірку даних та здійснювати класифікацію з високою точністю.

В основі такої системи можна використати інтелектуальну модель керовану даними, наприклад, як одну із найпотужніших – глибоку нейромережеву модель, навчивши її прогнозувати майбутній рівень вібрації за попередніми, на основі зібраних історичних даних. Детально досліджуються наскрізні моделі глибокого навчання (англ. end-to-end deep learning models), що мають на меті виключити етап ручного пошуку процедур перетворення вхідного сигналу у процесі синтезу НМ, шляхом більш точної настройки класифікатора.

Таким чином, тема даної магістерської роботи – актуальна і пов'язана з розробкою нових і модифікацією існуючих методів класифікації багатовимірних сигналів на основі наскрізних нейромоделей глибокого навчання.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Види сигналів

Сигнал – це множина спостережень, які генеруються послідовно у часі. У випадку коли час змінюється неперервно, сигнал називається процесом, а коли змінюється дискретно – дискретним сигналом.

Спостереження дискретного сигналу, в моменти часу $t_1, t_2, \dots, t_N$, позначаються як $x(t_1), x(t_2), \dots, x(t_N)$ [1].

Сигнали розділяють на одновимірні та багатовимірні, за кількістю показників які вимірюються у кожен момент часу. Розділяють сигнали також на моментні та інтервальні за характером часового параметра. У інтервальних сигналів їх рівні характеризують значення показника за певні періоди часу. Натомість, кожен рівень моментного сигналу характеризує значення показника на певний момент часу.

Виділяють детерміновані і випадкові сигнали. Перші обчислюються на основі значень деякої не випадкової функції, другі є результатом реалізації деякої випадкової величини. За відстанню між моментами або інтервалами часу вимірювань виділяють рівновіддалені – коли моменти вимірювання значення показника або закінчення періодів слідує один за одним з рівними інтервалами часу та нерівномірні сигнали, для яких принцип рівних інтервалів часу не дотримується [2]. Також сигнали поділяються на стаціонарні, у яких середнє значення і дисперсія постійні, та на нестаціонарні [3]-[8].

У даній магістерській роботі ставиться задача дослідження процесу класифікації багатовимірних випадкових нестаціонарних дискретних моментних рівновіддалених сигналів.

1.2 Задача класифікації сигналів

Класифікація полягає у передбаченні правильного класу за певної точки даних за допомогою розміченого навчального набору. Для задачі класифікації одновимірних сигналів ця точка даних сама по собі являє собою сигнал, а для багатовимірних – множину сигналів. Сигнали зустрічаються у багатьох практичних задачах, починаючи від охорони здоров'я [9] та розпізнавання людської діяльності [10]-[11] до класифікації акустичних сцен [12] та кібербезпеки [13]. Різноманітність вибірок даних в UCR / UEA - найбільшому архіві вибірок сигналів та даних часових рядів [14]-[15] показує різні застосування класифікації сигналів.

Дослідниками, на даний час, запропоновано сотні методів вирішення задачі класифікації сигналів [16]. Одним із класичних є використання НМ класифікатора у поєднанні з функцією відстані. Зокрема, DTW при використанні з НМ. У [17] порівняно декілька метрик відстані, наприклад, TWE [18] та MCM [19] та продемонстровано, що не існує метрики відстані, яка значно перевершує DTW за точністю класифікації. Отже, останні дослідження були зосереджені на розробці методів класифікації, які значно перевершують НМ з DTW [20]-[27]. Зазвичай, ці підходи використовують або ансамбль дерев рішень (випадковий ліс) [26]-[27] або ансамбль різноманітних типів дискримінантних класифікаторів (SVM, НМ з декількома метриками відстаней) на одному або декількох просторах ознак [20]-[22], [24]-[25]. Більшість із цих підходів значно перевершують НМ з DTW [15] та мають одну загальну властивість - наявність фази трансформації, де сигнали перетворюються на новий простір ознак. Це мотивувало розробку ансамблю з класифікаторів під назвою COTE [20], який не лише поєднує різні класифікатори за однаковою трансформацією сигналу, але також об'єднує різні класифікатори протягом різних подань сигналу у часі. У [23] COTE розширили до HIVE-COTE, якому вдалось досягти значного поліпшення порівняно з COTE шляхом використання нової

ієрархічної структури з імовірнісним голосуванням, що включає два нових класифікатори та дві додаткові трансформації сигналів. Зараз HIVE-COTE вважається найсучаснішим універсальним алгоритмом класифікації сигналів [15], що було оцінено на 85 вибірках даних з архіву UCR / UEA.

Однак, HIVE-COTE є обчислювально-трудомістким і його важко використовувати для практичних задач навчання на великих даних. Цей метод вимагає навчання 37 класифікаторів, а також виконання перехресної перевірки кожного гіперпараметру цих алгоритмів. Отримане рішення, прийняте 37 класифікаторами, не може бути легко інтерпретоване експертами.

Нещодавні дослідження демонструють успіх глибокого навчання [31] у різних задачах класифікації, що також показало використання глибоких НМ для задачі класифікації сигналів [35]. Використання глибоких НМ дозволяє використовувати їх обчислювальну потужність для автоматичного виділення ознак сигналу, без етапу підбору індивідуальних процедур обробки сигналу та забезпечує уніфікований підхід до багатовимірних сигналів, який не залежить від кількості показників які вимірюються.

1.3 Глибокі нейронні мережі у задачі класифікації багатовимірних сигналів

Глибока НМ - це композиція L параметричних функцій, що називаються шарами, де кожен шар вважається поданням вхідного домену [31]. Один шар l_i , такий як $i \in 1 \dots L$, містить нейрони, які обчислюють один елемент вихідного шару.

Шар l_i приймає в якості вихідних даних вихідне значення свого попереднього шару l_{i-1} і застосовує нелінійність для обчислення власного виходу. Результат цих нелінійних перетворень контролюється набором параметрів θ_i для кожного шару, які називаються вагами. Отже, з

урахуванням входу x , НМ виконує наступні обчислення для прогнозування класу за формулою (1.1):

$$f_L = f_{L-1}(\theta_{L-1}, f_{L-2}(\theta_{L-2}, \dots, f_1(\theta_1, x))), \quad (1.1)$$

де f_i - нелінійність застосована до шару l_i .

Під час навчання глибока НМ отримує певну кількість відомих розмічених вхідних даних (тренувальний набір). Спочатку ваги ініціалізуються випадковим чином [16]. Після ініціалізації, здійснюється прямий прохід через модель: обчислюється функція f від вхідного сигналу x . Результатом є вектор, чийі компоненти - це оціночні ймовірності x , що належать до кожного класу. Використовуючи цей вектор обчислюється функція втрат моделі. Потім, використовуючи метод градієнтного спуску [16], ваги оновлюються під час зворотного поширення помилки. Таким чином, шляхом ітеративного прямого проходження сигналу через модель з подальшим зворотним розповсюдженням помилки, параметри моделі оновлюються таким чином що функція втрат моделі мінімізується для навчальної вибірки даних.

Під час тестування імовірнісний класифікатор перевіряється на тестових даних, які не використовувались під час навчання моделі. Прогноз відповідає класу, імовірність якого максимальна. Однією з переваг глибоких НМ перед неімовірнісними класифікаторами (наприклад, НМ з DTW) є те, що перші приймають імовірнісне рішення [28], таким чином дозволяючи виміряти впевненість передбачення моделі. Для вирішення задачі класифікації сигналів було запропоновано кілька архітектур глибокого навчання, які можна розділити на дві групи: моделі глибоких НМ з ознаками розробленими вручну (англ. *hand engineered features*) та наскрізні моделі глибокого навчання. На відміну від ознак розроблених вручну, підхід наскрізного глибокого навчання має на меті виключити етап ручного пошуку процедур перетворення вхідного сигналу, шляхом більш точної настройки

класифікатора [29]. Далі будуть розглянуті дві основні наскрізні моделі глибоких НМ для вирішення задачі класифікації багатовимірних сигналів: багат шаровий перцептрон та згорткова НМ.

1.3.1 Багат шаровий перцептрон

Багат шаровий перцептрон - це найпростіша традиційна архітектура для глибокого навчання НМ. Це такий тип архітектури НМ, у якому кожен нейрон у шарі l_i пов'язаний з кожним нейроном шару l_{i-1} (рис. 1.1). Ці з'єднання моделюються вагами НМ. Загальна форма застосування нелінійності до вхідного сигналу X за формулою (1.2):

$$A_{l_i} = f(w_{l_i} \cdot X + b), \quad (1.2)$$

де w_{l_i} – набір вагів з довжиною та кількістю вимірюваних показників відповідно до розмірності сигналу X ;

b – зміщення;

A_{l_i} – активація нейронів шару l_i .

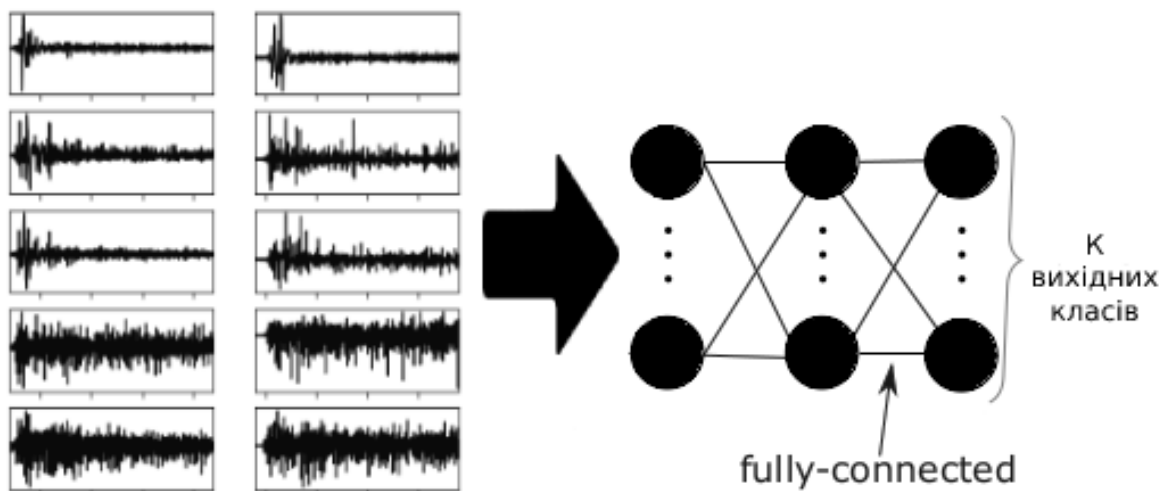


Рисунок 1.1 – Багат шаровий перцептрон для задачі класифікації багатовимірних сигналів

Однією з перешкод у використанні багат шарового перцептрона для даних сигналів є те, що ця модель не підтримує будь-яку часову інваріантність, що продемонстровано на Рис. 1.1. Іншими словами, кожне значення параметру у кожен момент часу має власну вагу, і інформація про час втрачається, значення сигналу обробляються незалежно один від одного.

Для задачі класифікації сигналів останній шар глибокої НМ, як правило, є дискримінаційним шаром, який приймає як вхідний сигнал значення активації попереднього шару і дає розподіл ймовірностей за класами, наявними у наборі даних. У більшості підходів до глибокого навчання НМ для задачі класифікації сигналів застосовується повнозв'язний шар з функцією активації softmax. Результат softmax функції за формулою (1.3):

$$\hat{Y}_j = \frac{e^{A_{L-1} \cdot w_j + b_j}}{\sum_{k=1}^K e^{A_{L-1} \cdot w_k + b_k}}, \quad (1.3)$$

де $\hat{Y}_j$ – імовірність того, що X належить до класу j із K класів навчального набору даних.

Набори вагів у рівняннях (1.2–1.3) слід налаштовувати автоматично, використовуючи алгоритм оптимізації, який мінімізує цільову функцію витрат. Для апроксимації похибки певного заданого значення ваг слід визначити диференційовану функцію втрат, яка кількісно визначає цю помилку. Найбільш популярною функцією втрат у глибоких НМ для завдання класифікації є категоріальна кросс-ентропія за формулою (1.4):

$$L(X) = -\sum_{j=1}^K Y_i \log \hat{Y}_j, \quad (1.4)$$

де L – функція втрат або вартості при класифікації сигналу X .

Середню функція втрат при класифікації всього навчального набору можна визначити наступним чином за формулою (1.5):

$$J(\Omega) = \frac{1}{N} \sum_{n=1}^N L(X_n), \quad (1.5)$$

де Ω – набір вагів, що вивчаються глибокою НМ (у цьому випадку, ваги w з рівнянь 1.2–1.3). Функція втрат мінімізується для набору Ω за допомогою методу градієнтного спуску за формулою (1.6):

$$\omega = \omega - \alpha \frac{\partial J}{\partial \omega} \mid \forall \omega \in \Omega, \quad (1.6)$$

де α – швидкість навчання оптимізаційного алгоритму (англ. learning rate).

Шляхом віднімання часткової похідної, модель автоматично налаштовує свої параметри ω з метою досягати локального мінімуму J у випадку нелінійного класифікатора. Слід зазначити, що у випадку коли часткову похідну неможливо безпосередньо обчислити щодо певного параметра ω , застосовується ланцюгове правило похідної, що є основною ідеєю алгоритму зворотного розповсюдження помилки [16].

1.3.2 Згорткова нейронна мережа

З моменту успіху НМ AlexNet [30] у задачі класифікації зображень, глибокі згорткові НМ успішно застосовувалися у багатьох практичних задачах, в тому числі для аналізу сигналів [20]. Згортку (англ. convolution) можна розглядати як застосування фільтра зі зсувом до сигналу. На відміну від зображень, для сигналів застосовуються одновимірні фільтри (час) замість двовимірних (ширини та висоти). Загальна форма застосування згортки для моменту часу t за формулою (1.7):

$$C_t = f(\omega \cdot X_{t-l/2:t+l/2} + b) \mid \forall t \in [1, T], \quad (1.7)$$

де C – результат згортки застосований до сигналу X довжини T з фільтром ω довжини l , параметром зміщення b та нелінійною функцією f .

Результат згортки вхідний сигнал X можна розглядати як інший сигнал C , який пройшов процес фільтрації.

Найпопулярнішою функцією нелінійності у глибоких згорткових НМ є ReLU за формулою (1.8):

$$f(x) = \max(0, x). \quad (1.8)$$

На відміну від багат шарового перцептрону, однакова згортка (з однаковими значеннями параметрів фільтру w та b) буде використана для пошуку результату для всіх часових позначок $t \in [1, T]$. Це важлива властивість згорткових НМ, що називається розподілом ваги, це дозволяє їм налаштовувати фільтри, які інваріантні щодо часового виміру (рис. 1.2).

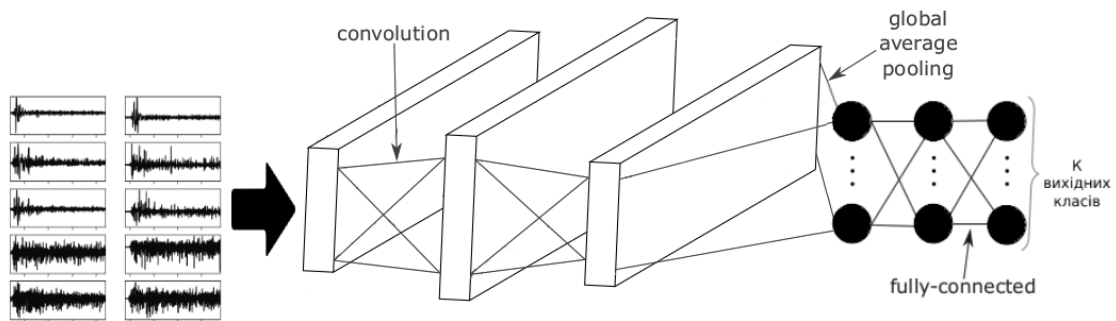


Рисунок 1.2 – Згорткова нейронна мережа для задачі класифікації багатовимірних сигналів

Для того, щоб автоматично налаштувати фільтр, згортка повинна супроводжуватися класифікатором, якому зазвичай передую операція об'єднання (англ. pooling), яка може бути локальною або глобальною. Локальне об'єднання (англ. local pooling), таке як середнє (англ. average

pooling), або максимальне об'єднання (англ. max-pooling), приймає введений сигнал і зменшує його довжину шляхом агрегування ковзним вікном часового ряду. За допомогою операції глобального об'єднання (англ. global pooling) сигнал агрегується протягом усього виміру часу, що у результаті дає єдину результуючу величину. Іншими словами, це еквівалентно до застосування локального об'єднання з довжиною ковзного вікна, що дорівнює довжині вхідного сигналу.

Також, деякі архітектури глибоких НМ включають шари нормалізації, що допомагає мережі швидше сходитися. Для багатовимірних сигналів – операція нормалізації виконується по кожному вимірюваному параметру, таким чином запобігаючи внутрішньому коваріантному зсуву для міні-батчу даних [32]. Інший тип нормалізації [33] виконує нормалізацію для кожного екземпляру сигналу, таким чином налаштовуючи середнє значення та стандартне відхилення кожного екземпляру навчального набору для кожного шару за допомогою градієнтного спуску. Цей підхід називається нормалізацією екземплярів та імітує вивчення параметрів z-нормування для навчальних даних сигналів. Остатній шар приймає представлення вхідного сигналу (результат згортки) і обчислює розподіл ймовірностей за змінними класу в наборі даних. Зазвичай цей шар складається з операції softmax, подібно до багат шарового перцептрона. Процес навчання також ідентичний навчанню багат шарового перцептрона [16].

1.3.3 Аналіз і порівняння наскрізних моделей глибокого навчання у задачі класифікації багатовимірних сигналів

У [34]-[35] було використано багат шаровий перцептрон для класифікації сигналів. Основною проблемою у використанні багат шарового перцептрона для даних сигналів є те, що ця НМ не підтримує будь-яку часову інваріантність. Натомість глибокі згорткові НМ є більш ефективними, так як вони використовують часово-інваріантні фільтри для необроблених вхідних

даних сигналу [35]. Та є найбільш широко застосовуваною архітектурою для задачі класифікації сигналів, що зумовлено їх робастністю та відносно невеликим часом навчання в порівнянні з аналогічними архітектурами, такими як багатошаровий перцептрон або рекурентні НМ.

Декілька архітектур глибоких згорткових НМ було запропоновано у [36] ефективність яких була підтверджена на архіві UCR / UEA. У [37] пропонується виявляти порушення міокарда на основі сигналів електрокардіографії з використанням глибокої згорткової НМ. У [29] пропонується використання глибокої згорткової НМ для розпізнавання людської діяльності за допомогою переносних датчиків, що замінює попередній підхід з ручною розробкою ознак.

Також ефективність показали гібридні архітектури глибоких НМ для задачі класифікації сигналів на наборах даних з архіву UCR / UEA, де глибокі згорткові НМ поєднувалися з іншими типами архітектур, такими як закриті рекурентні блоки (англ. gated recurrent units) [38] та механізм уваги [39].

1.4 Постановка завдань роботи

На основі проведеного аналізу сучасних методів вирішення задачі класифікації багатовимірних сигналів, найбільш перспективними є наскрізні моделі глибокого навчання. Так як їх використання не вимагає ручного пошуку процедур перетворення вхідного сигналу та дає змогу сконцентруватися на точному налаштуванні НМ класифікатора.

Серед наскрізних моделей глибокого навчання більш ефективними є глибокі згорткові НМ та гібридні архітектури глибоких НМ, де глибокі згорткові НМ поєднуються з іншими типами архітектур, так як вони використовують часово-інваріантні фільтри, на відміну від багатошарового перцептрона, який не підтримує будь-яку часову інваріантність. Тому у даному дослідженні, у якості класифікатора багатовимірних сигналів для використання було обрано модель глибокої згорткової НМ.

Так як, на якість класифікації впливає не лише архітектура класифікатора, а і якість навчального набору даних, постає задача виділення із загального сигналу його найбільш інформативних ділянок. У даній магістерській роботі ставиться практична задача класифікації сигналів ЕМГ з м'язів колін.

Ставляться наступні завдання роботи: 1) проведення аналізу існуючих методів виділення ділянок активності з одновимірного ЕМГ сигналу; 2) розробка методу знаходження ділянок найбільшої активності багатовимірного ЕМГ сигналу, для подальшого їх використання для навчання та тестування класифікатора; 3) розробка методу знаходження оптимальної довжини вхідного шару глибокої НМ, використовуючи інформацію про довжину виділених ділянок активності для навчального набору даних; 4) проведення експериментів на основі реальних даних.

1.5 Висновки за розділом 1

Автоматична класифікація стану людини або технічного об'єкту за даними сигналів є важливою задачею, яка може допомогти лікарям та технічним спеціалістам економити час, який раніше вони витрачали на ручну класифікацію. Так, як у практичних задачах все частіше стає складно характеризувати комплексний процес за одним сигналом або параметром спостереження, то все частіше ставиться задача класифікації багатовимірних сигналів, що характеризуються вимірюванням більше ніж одного параметра в кожен момент часу. Класифікація такого виду сигналів є важкою задачею для людей також. Для вирішення цієї проблеми перспективно використовувати методи математичної статистики, цифрової обробки сигналів, а також один із напрямків обчислювального інтелекту, що активно розвивається – теорію штучних глибоких НМ.

У результаті проведення досліджень проблемної області необхідно розробити методи та реалізувати на їх основі програмний продукт, який

дозволить виділяти ділянки активності та проводити класифікацію стану пацієта на основі багатовимірного ЕМГ сигналу, експериментально перевірити роботу розроблених методів і зробити висновки про ефективність їх використання у відношенні до існуючих рішень.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Методи виділення ділянок активності одновимірного ЕМГ сигналу

У процесі побудови класифікатору на основі ЕМГ сигналів, необхідно виділити із загального сигналу проміжки м'язової активності, що саме і характеризують стан м'язів, та відділити його від проміжків спокою. Так, як саме за значеннями сигналу на проміжках м'язової активності проводиться діагностування лікарем. Більшість лікарів виконують це ручним візуальним способом [40]-[41]. Результат такої процедури наведено на рис. 2.1.

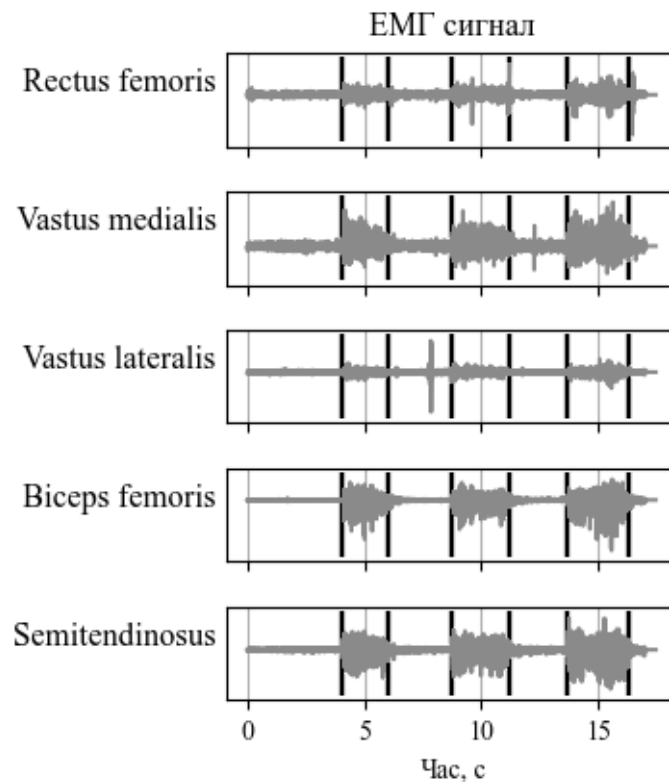


Рисунок 2.1 – Результат процедури виділення проміжків активності із ЕМГ сигналу для вимірів з п'яти м'язів коліна

На даний час, існує декілька математичних методів для автоматичного виділення періодів активності з одновимірного ЕМГ сигналу. Вони базуються на трансформації вхідних сигналів, статистичних методах

та методах трешолдингу [42]-[48]. Наведемо опис цих методів для їх порівняння, подальшої реалізації та використання при синтезі НМ. Так, як ці методи не мають загальноприйнятих назв, то вони будуть названі відповідно до особливостей їх роботи.

Метод простого трешолдингу [42].

Метод ідентифікує точку активації там де середнє значення заданої кількості вимірів m перевищило базовий рівень активності сигналу на задану кількість стандартних відхилень.

Метод подвійного трешолдингу [43].

Метод ідентифікує точку активації там де середнє значення заданої кількості вимірів m перевищило базовий рівень активності сигналу на задану кількість стандартних відхилень, у випадку якщо хоча б r_0 вимірів перевищило це значення.

Метод трешолдингу з використанням тривалості активації та кількості успішних вимірів [44].

Метод ідентифікує точку активації там де середнє значення заданої кількості вимірів m перевищило базовий рівень активності сигналу на задану кількість стандартних відхилень, у випадку якщо попередня точка активації, була виявлена не раніше чим за s секунд до поточної (або не була виявлена зовсім). Точка активації вважається виявленою, якщо не менше f вимірів до неї перевищило цей поріг.

Метод трешолдингу з використанням амплітудного уточнення [45].

Метод ідентифікує початкову точку активації там де середнє значення заданої кількості вимірів m перевищило базовий рівень активності сигналу на задану кількість стандартних відхилень. Після знаходження початкової точки активації, проводиться пошук менших амплітуд у a вимірюваннях до знайденого, у яких величина перевищує другий менший заданий поріг t . Серед них, точка з найбільшим значенням амплітуди ідентифікується як точка активації.

Метод трешолдінгу з використанням енергетичного оператора Тиджера – Кайзера та тривалості активації [46].

До ЕМГ сигналу застосовується енергетичний оператор Тиджера – Кайзера для дискретного випадку за формулою (2.1):

$$\psi[x(n)] = x^2(n) - x(n+1)x(n-1). \quad (2.1)$$

Метод ідентифікує точку активації там де значення оператора перевищило його базовий рівень на задану кількість стандартних відхилень, у випадку якщо попередня точка активації, була виявлена не раніше чим за s секунд до поточної (або не була виявлена зовсім).

Метод подвійного адаптивного трешолдінгу [47].

Для ЕМГ сигналу обчислюється ковзне середнє за допомогою часового вікна із заданою шириною w .

Метод ідентифікує точку активації там де середнє значення заданої кількості вимірів m перевищило ковзне середнє для цієї точки, яке у свою чергу перевищило базовий рівень активності сигналу на задану кількість стандартних відхилень.

Метод трешолдінгу з використанням максимальної ймовірності локальної дисперсії [48].

Метод ідентифікує точку активації там де середнє значення максимальної ймовірності локальної дисперсії заданої кількості вимірів m перевищило її базовий рівень на задану кількість стандартних відхилень. Середнє значення максимальної ймовірності локальної дисперсії (англ. maximum-likelihood estimate of the variance) обчислюється за формулою (2.2):

$$\hat{\sigma}_x^2 = \frac{1}{m} \left[\sum_{i=1}^m x_i^2 - \frac{1}{m} \left(\sum_{i=1}^m x_i \right)^2 \right]. \quad (2.2)$$

Усі вищенаведені методи також можуть використовуватися для ідентифікації точок деактивації ЕМГ сигналу з точністю до знаків порівнянь.

Значення трешолдінгу рекомендується обирати у діапазоні від одного до трьох стандартних відхилень, з огляду на те, що ЕМГ сигнал є гаусівським процесом та у залежності від зашумленості сигналу.

Однак, усі ці методи призначені для одновимірного ЕМГ сигналу. Для практичних задач синтезу НМ є потреба у визначенні проміжків активностей багатовимірного ЕМГ сигналу та оптимальної довжини ділянки сигналу у екземплярі набору даних, що надходить до НМ.

2.2 Методи виділення ділянок багатовимірного ЕМГ сигналу для синтезу НМ

Вищенаведені методи демонструють ефективність для виділення періодів активності з одновимірного ЕМГ сигналу. Тому є підстава вважати, що вони будуть ефективними і для багатовимірного ЕМГ сигналу через їх однакову природу. Однак, зазвичай ЕМГ сигнали з різних м'язів мають різний рівень амплітуд. Тому необхідно провести попереднє їх приведення до єдиного масштабу. Для цього будемо використовувати *rest* - проміжок ЕМГ сигналу без м'язової активності (зазвичай перші декілька секунд вимірювання). Наведемо запропонований метод виділення періодів активності з багатовимірного ЕМГ сигналу.

Метод виділення періодів активності з багатовимірного ЕМГ сигналу пропонується використовувати для формування навчальної вибірки та подальшого синтезу НМ.

Етап ініціалізації.

Задати набір синхронізованих у часі одновимірних ЕМГ сигналів $\{x_1(t), x_2(t), \dots, x_n(t)\}$.

Етап масштабування.

Для кожного одновимірного ЕМГ сигналу, розрахувати за формулою (2.3):

$$f(t) = \frac{x(t) - \overline{rest}}{\sigma(rest)}, \quad (2.3)$$

де $\overline{rest}$ - середнє значення проміжку сигналу *rest*;

$\sigma(rest)$ - середнє квадратичне відхилення проміжку сигналу *rest*.

Етап об'єднання.

Об'єднати набір одновимірних ЕМГ сигналів у багатовимірний сигнал за формулою (2.4):

$$X(t) = \begin{bmatrix} f_1(t) \\ f_2(t) \\ \dots \\ f_n(t) \end{bmatrix}. \quad (2.4)$$

Етап усереднення.

Розрахувати середнє значення для кожної точки багатовимірного сигналу за формулою (2.5):

$$s(t) = \frac{\sum_{i=1}^n X_{it}}{n}. \quad (2.5)$$

Етап виділення періодів активності.

Налаштувати та застосувати один із методів наведених у п. 2.1 до сигналу $s(t)$ для виділення точок активації та деактивації. Отриманий набір пар точок $onsets = \{(t_{1s}, t_{1f}), (t_{2s}, t_{2f}), \dots, (t_{ms}, t_{mf})\}$ відображає усереднені

періоди активності багатовимірного ЕМГ сигналу та є кінцевим результатом роботи запропонованого методу.

Запропонований методу виділення періодів активності з багатовимірного ЕМГ сигналу може бути використаний для формування навчальної вибірки для подальшого синтезу НМ. Однак, довжина виділених проміжків багатовимірного ЕМГ сигналу може бути різною, а НМ модель вимагає однакової розмірності вхідної матриці. Тому пропонується метод знаходження оптимальної довжини ділянки активації сигналу для навчального набору даних та формування навчальної вибірки.

Метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних пропонується використовувати для формування навчальної вибірки з екземплярами однакової розмірності для подальшого синтезу НМ.

Етап ініціалізації.

Отримати набори пар точок активації та деактивації багатовимірного ЕМГ сигналу $L = \{onsets_1, onsets_2, ..., onsets_c\}$ для навчального набору даних розміру c , застосувавши наведений вище метод виділення періодів активності з багатовимірного ЕМГ сигналу, або вручну.

Етап усереднення екземплярів.

Розрахувати середнє значення довжини проміжку активації для кожного екземпляру навчального набору даних за формулою (2.6):

$$l_{onset} = \frac{\sum_{i=1}^m |t_{is} - t_{if}|}{m}, \quad (2.6)$$

де m – кількість пар точок у екземплярі даних;

t_s – точка активації;

t_f – точка деактивації.

Етап усереднення набору даних.

Розрахувати середнє значення довжини проміжку активації для усього навчального набору даних за формулою (2.7):

$$D = \frac{\sum_{i=1}^c l_i}{c} . \quad (2.7)$$

Етап перерахунку початкових наборів пар точок.

Отримати набори пар точок з однаковою оптимальною довжиною проміжку активації D для усього набору даних. Для цього додати до значення кожної точки активації цю довжину (так, як активація ЕМГ сигналу є більш інформативною ніж деактивація). Отримані набори є кінцевим результатом роботи запропонованого методу.

Отримані набори пар точок можуть бути використані для отримання однакових за розмірністю фрагментів з вхідного багатовимірного ЕМГ сигналу для синтезу НМ.

1.3 Висновки за розділом 2

На основі проведеного аналізу предметної області та існуючих методів для автоматичного виділення періодів активності з одновимірного ЕМГ сигналу було розроблено методи виділення періодів активності з багатовимірного ЕМГ сигналу та знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних.

Ділянки сигналів отримані методом виділення періодів активності з багатовимірного ЕМГ сигналу є інформативними для побудови класифікатору на їх основі. Він надає альтернативу процедурам ручного та полу-автоматичного (автоматичне виділення періодів активності з набору одновимірних ЕМГ сигналів та подальше їх узгодження вручну) виділення періодів активності з багатовимірного ЕМГ сигналу.

Метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних дає можливість автоматичного використання отриманих за його результатами однакових за розмірністю фрагментів з вхідного багатовимірного ЕМГ сигналу для синтезу сучасних моделей машинного навчання, що вимагають однакової розмірності вхідної матриці, у тому числі глибоких наскрізних НМ.

Для експериментальної перевірки якості роботи розроблених методів необхідно реалізувати їх програмно, протестувати та зробити висновки про ефективність їх використання.

3 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

3.1 Опис основних вимог до програми

У ході аналізу предметної області було з'ясовано, що для вирішення поставленої задачі необхідно розробити бібліотеку програмних функцій, який надасть можливість виділення періодів активності з багатовимірного ЕМГ сигналу, знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору та синтезу на їх основі НМ для класифікації пацієнтів на наявність чи відсутність у них травм груп м'язів.

Розроблені методи, що реалізовані у вигляді програмної бібліотеки повинні мати високу швидкодію для забезпечення обчислювальної ефективності. Також повинен бути розроблений зрозумілий та уніфікований інтерфейс виклику цих функцій. Так, як навчання нейромоделей вимагає великих обчислювальних можливостей та системного часу, то програмна бібліотека, що розроблюється повинна максимально повно використовувати апаратні ресурси комп'ютера на якому будуть запущені її програмні модулі.

3.2 Структура програмної системи та функціонування програми

Програмна бібліотека повиненна включати в себе: модуль введення-виведення даних – для завантаження вхідних даних та виведення результатів класифікації, модуль попередньої обробки даних – для попередньої обробки та виділення періодів активності з одновимірних та багатовимірних ЕМГ сигналів та модуль створення НМ – для знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору, синтезу, навчання та тестування НМ моделей.

Так як данна програмна бібліотека, насамперед, призначена для дослідників у сфері обробки сигналів та обчислювального інтелекту, то однією із вимог є відкритий програмний код бібліотеки та можливість

легкого перевикористання та доповнення її функцій. Це дозволить не обмежувати користувача у засобах обраної мови програмування та без додаткових зусиль використовувати програмну бібліотеку для власних досліджень та розробки ПЗ. Такої гнучкості можна досягнути використовуючи при розробці скриптову мову програмування.

Користувач повинен мати можливість завантажити вхідні дані сигналів діагностування у вигляді файлів формату .txt, .xls, або .xlsx та розділити їх на навчальну та тестову вибірки. Виділити періоди активності з багатовимірних ЕМГ сигналів шляхом застосування, запропонованого у п. 2.2, методу. У ході застосування цього методу, користувач може обрати найбільш оптимальний для своїх даних метод виділення періодів активності з одновимірною ЕМГ сигналу, що є складовою частиною даного методу. У бібліотеці реалізовано усі методи, що описано у п. 2.1.

Далі, користувач може знайти оптимальну довжину ділянки активації багатовимірною ЕМГ сигналу для навчального набору та розділити таким чином вхідний набір даних. Синтезувати та навчити на їх основі НМ класифікатор. Отриману модель користувач може використовувати для діагностики нових пацієнтів. Результат прогнозу користувач повинен мати можливість отримати у вигляді файлу формату .txt або .xlsx та шляхом виводу на екран. Для вирішення поставлених задач було спроектовано структурну схему програмної бібліотеки (рис. 3.1) та схему її функціонування (рис. 3.2).



Рисунок 3.1 – Структурна схема програмної бібліотеки

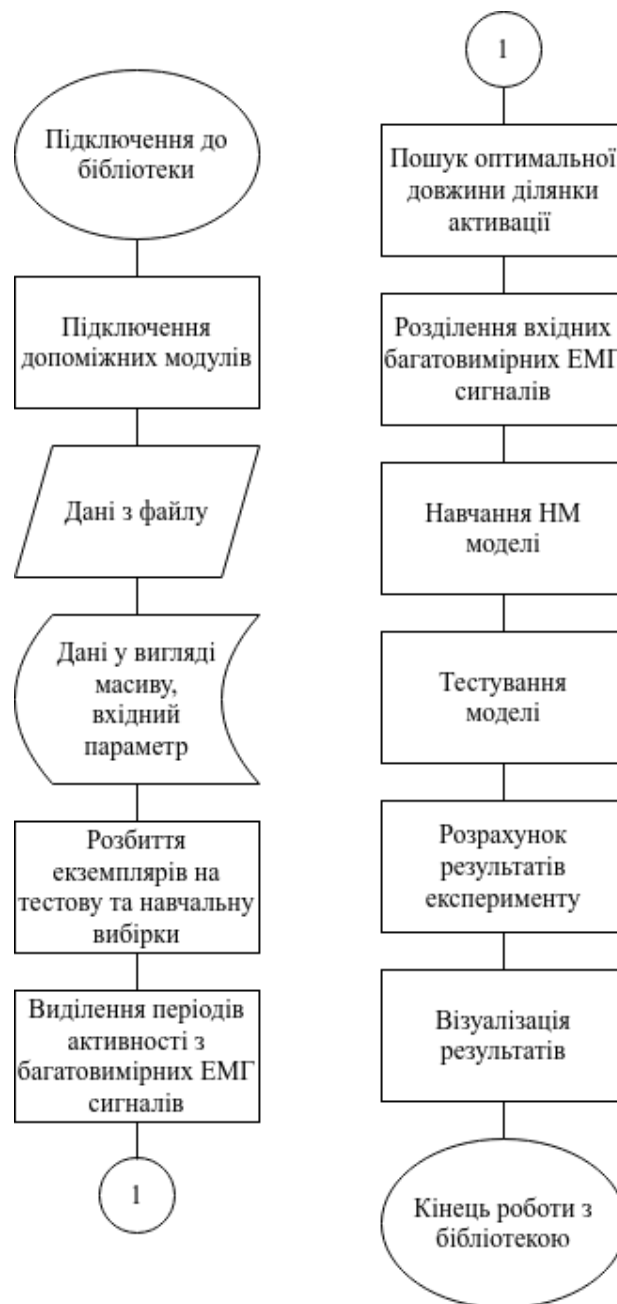


Рисунок 3.2 – Схема функціонування програмної бібліотеки

3.3 Вибір мови програмування та інструментарію для розробки програми

На основі проведеного аналізу вимог до програмної бібліотеки було вирішено для розробки програми використати скриптову мову програмування. Для вибору мови програмування для розробки проведемо

порівняння найбільш популярних скриптових мов програмування. Результати порівняння наведено в табл. 3.1.

Таблиця 3.1 — Порівняння найбільш популярних скриптових мов програмування

Можливості	Скриптові мови програмування						
	VBScript	JavaScript	Python	Tcl	Ruby	PHP	Perl
1	2	3	4	5	6	7	8
Імперативна парадигма	+	+	+	+	+	+	+
Об'єктно-орієнтована парадигма	+	+	+	+	+	+	+
Статична типізація	+	-	-	-	-	-	-
Динамічна типізація	+	+	+	+	+	+	+
Неявне приведення типів	+	+	-	+	-	+	+
Підтримка алгебраїчних типів даних	-	+	+	+	+	+	+
«Прибирання сміття» при керуванні пам'ятю	+	+	+	+	+	+	+

Продовження таблиці 3.1

1	2	3	4	5	6	7	8
Наявність бібліотек з відкритим кодом для роботи з багатовимірними матрицями	-	+	+	-	+	+	-
Наявність бібліотек з відкритим кодом для візуалізації	-	+	+	-	+	-	-

За результатами порівняння найбільш популярних скриптових мов програмування у якості мови програмування для розробки програмного застосунку було обрано мову Python. Використання цієї мови програмування дозволить розробити програмну бібліотеку, яка буде зручною у використанні за рахунок наявності широкого інструментарію для інтелектуального аналізу даних з відкритим кодом для даної мови програмування. Це дасть широкі можливості для використання методів реалізованих у розроблених програмній бібліотеці для розробки ПЗ, у тому числі на стадії швидкого прототипування. Дослідник повинен не бути обмежений в інструментах обробки даних, так як доповнення існуючої програми власними модулями або сторонніми програмними бібліотеками буде доступно прямо під час роботи з нею, без необхідності перезавантажувати, перезбирати або компілювати бібліотеку.

У якості інтегрованого середовища розробки буде використано PyCharm [49], як найбільш розвинене середовище для розробки програм мовою програмування Python на теперішній час. Для завантаження та попередньої обробки даних багатовимірних сигналів буде використано

бібліотеку Pandas [50]. Для застосування операцій лінійної алгебри, реалізації існуючих методів для автоматичного виділення періодів активності з одновимірного ЕМГ сигналу, а також для реалізації запропонованих методів виділення періодів активності з багатовимірного ЕМГ сигналу та знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних буде використано бібліотеку NumPy [51]. Для синтезу, навчання та тестування НМ моделі буде застосовано бібліотеку TensorFlow [52]. Для візуалізації вибірок даних та результатів роботи НМ моделі буде використано бібліотеку Matplotlib [53].

3.4 Вимоги до технічних і програмних засобів

Для функціонування програмної бібліотеки необхідна наявність встановленого інтерпретатора Python 3.8 зі встановленими у робочому оточенні необхідними бібліотеками. Інтерпретатор може бути встановлений на такі ОС, як Windows, Mac OS, GNU/Linux. Висуваються такі мінімальні вимоги до апаратного забезпечення: персональний IBM-сумісний комп'ютер з процесором Intel Core 2 Duo, оперативною пам'яттю 4 Гб, монітором, клавіатурою та маніпулятором типу «миша».

3.5 Висновки за розділом 3

На основі проведеного аналізу предметної області та розроблених методів попередньої обробки сигналів діагностування було висунуто вимоги до програмної бібліотеки, яка буде розроблена для автоматизації процесу класифікації багатовимірних сигналів в НМ моделі. На їх основі було розроблено структуру програми. Для його розробки було обрано скриптову мову програмування Python. Також було висунуто мінімальні вимоги до технічних і програмних засобів для стабільного використання та функціонування програмної бібліотеки.

4 РОЗРОБКА ПРОГРАМИ

4.1 Опис основних вимог до програми

На основі аналізу предметної області та розроблених методів було розроблено архітектуру програмної бібліотеки, що буде розроблена. Програмна бібліотека буде складатися з трьох модулів: модуль введення-виведення даних, модуль попередньої обробки даних та модуль створення НМ. Дана архітектура дозволить задовільнити всі вимоги, що було поставлено до програмної бібліотеки.

Завдяки гнучкості обраних інструментів розробки, а також розробленим програмним інтерфейсам, користувач зможе з легкістю змінювати існуючі та додавати нові модулі для розширення та модифікації функціональності бібліотеки, що є вкрай важливим для дослідницького ПЗ. Проведемо опис функціонування кожного модуля програми та їх взаємодії один з одним.

4.1.1 Модуль введення-виведення даних

Задля реалізації можливості використання програмної бібліотеки для експериментів з різними вибірками даних багатовимірних сигналів до неї було включено модуль введення-виведення даних.

Завантаження вибірки даних з файлу реалізує метод `read_file(mypath)` з програмного модуля `main.py`. Завантаження даних відбувається з файлів формату `.txt`, `.xls`, або `.xlsx`.

Так як на теперішній час не існує чітких стандартів форматів запису файлів з багатовимірними сигналами, то разом з програмним застосунком пропонується набір файлів у форматі `.xlsx`. Внутрішня структура яких повністю відповідає вимогам які висуває модуль введення-виведення даних. Ці файли містять вибірку даних, що є багатовимірними ЕМГ сигналами з позначками міток класів - результатів медичного діагностування. Дані цієї

вибірки детально описані у р. 5. Вони будуть слугувати користувачам при знайомстві з програмною бібліотекою, а також будуть слугувати прикладами при потребі приведення власної вибірки до необхідного формату, або при необхідності зміни чи доповнення модуля введення-виведення даних.

Виведення результатів роботи програми може відбуватися, як на екранну форму користувача, так і у файл з результатами. Результируючий файл має такий самий формат і структуру, як і файли для завантаження вибірок. Збереження вибірки у файл реалізує метод `save_file(my_path)` з програмного модуля `main.py`. Збереження даних відбувається з файлів формату `.txt`, або `.xlsx`.

Розробка даного модуля дозволила автоматизувати процес завантаження вибірки та виведення результатів проведених експериментів у файл.

4.1.2 Модуль попередньої обробки даних

Задля реалізації можливості використання програмної бібліотеки для експериментів з різними вибірками даних багатовимірних сигналів до неї було включено модуль попередньої обробки даних.

За допомогою даного модулю, користувач має можливість виділити періоди активності з багатовимірних ЕМГ сигналів шляхом застосування, запропонованого у п. 2.2, методу. Це реалізовано у методі `md_onsets_detection(X, od_method)` з програмного модуля `md_detection.py` що приймає, відповідно, вибірку даних та метод виділення періодів активності з одновимірного ЕМГ сигналу зі своїми внутрішніми параметрами.

У ході застосування цього методу, користувач може обрати найбільш оптимальний для своїх даних метод виділення періодів активності з одновимірного ЕМГ сигналу, що є складовою частиною данного методу. У данному модулі реалізовано усі методи, що описано у п. 2.1. Ці методи реалізовано у вигляді набору програмних модулів, кожен з яких відповідає за

певний метод. Сигнатури методів відрізняються відповідно до гіперпараметрів, які необхідні для їх застосування. Назви методів відповідають назвам програмних модулів, у яких вони реалізовані.

Метод простого трешолдінгу реалізовано у програмному модулі `hodges_bui.py`. Метод вимагає вказання значення кількості вимірів m , які перевищили базовий рівень активності сигналу на певну кількість стандартних відхилень, яку теж треба задати користувачу.

Метод подвійного трешолдінгу реалізовано у програмному модулі `bonato.py`. Метод вимагає вказання значення кількості вимірів m , які перевищили базовий рівень активності сигналу на певну кількість стандартних відхилень, яку теж треба задати користувачу, у випадку якщо хоча б задана кількість - r_0 вимірів перевищила це значення.

Метод трешолдінгу з використанням тривалості активації та кількості успішних вимірів реалізовано у програмному модулі `lidieth.py`. Метод вимагає вказання значення кількості вимірів m , які перевищили базовий рівень активності сигналу на задану кількість стандартних відхилень, яку теж треба задати користувачу. Точка активації ідентифікується у випадку якщо попередня точка активації, була виявлена не раніше чим за задану кількість - s секунд до поточної (або не була виявлена зовсім). Точка активації вважається виявленою, якщо не менше за задану кількість - f вимірів до неї перевищило вказаний поріг.

Метод трешолдінгу з використанням амплітудного уточнення реалізовано у програмному модулі `abbink.py`. Метод вимагає вказання значення кількості вимірів m , які перевищили базовий рівень активності сигналу на задану кількість стандартних відхилень, яку теж треба задати користувачу. Після знаходження початкової точки активації, проводиться пошук менших амплітуд у заданій кількості - a вимірюваннях до знайденого, у яких величина перевищує другий менший заданий поріг t . Серед них, точка з найбільшим значенням амплітуди ідентифікується як точка активації.

Метод трешолдінгу з використанням енергетичного оператора Тиджера – Кайзера та тривалості активації реалізовано у програмному модулі `solnik.py`. Метод вимагає вказання значення кількості вимірів m , де значення енергетичного оператора перевищило його базовий рівень на задану кількість стандартних відхилень, яку теж треба задати користувачу. Точка активації ідентифікується у випадку якщо попередня точка активації, була виявлена не раніше чим за задану кількість - s секунд до поточної (або не була виявлена зовсім).

Метод подвійного адаптивного трешолдінгу реалізовано у програмному модулі `silva.py`. Для сигналу обчислюється ковзне середнє за допомогою часового вікна із заданою шириною w . Метод вимагає вказання значення кількості вимірів m , які перевищили ковзне середнє для цієї точки, яке у свою чергу перевищило базовий рівень активності сигналу на задану кількість стандартних відхилень, яку теж треба задати користувачу.

Метод трешолдінгу з використанням максимальної ймовірності локальної дисперсії реалізовано у програмному модулі `londral.py`. Метод вимагає вказання значення кількості вимірів m , які перевищили базовий рівень максимальної ймовірності локальної дисперсії сигналу на задану кількість стандартних відхилень, яку теж треба задати користувачу.

Розробка даного модуля дозволила користувачу проводити експерименти та підбирати оптимальні методи та їх гіперпараметри для виділення періодів активності з одновимірного ЕМГ сигналу. Цей налаштований метод, використовується як аргумент для запропонованого методу виділення періодів активності з багатовимірного ЕМГ сигналу, що також реалізовано у даному програмному модулі.

4.1.3 Модуль створення НМ

Задля реалізації можливості використання програмної бібліотеки для знаходження оптимальної довжини ділянки активації багатовимірного

ЕМГ сигналу для навчального набору та подальшого синтезу, навчання та тестування НМ моделей до неї було включено модуль створення НМ. У процесі розробки програмної бібліотеки, було реалізовано програмні інтерфейси для взаємодії даного модуля з модулями введення-виведення даних та попередньої обробки даних.

Першим етапом роботи з даним модулем програмної бібліотеки є застосування запропонованого методу знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу `md_tsol_calc(onsets)` з програмного модуля `md_detection.py` для навчального набору даних, пропонується використовувати для формування навчальної вибірки з екземплярами однакової розмірності для подальшого синтезу НМ. Вхідними даними до цього методу є вибірка даних та точки, визначені запропонованим методом виділення періодів активності з багатовимірного ЕМГ сигналу, що реалізовано у модулі попередньої обробки даних цієї програмної бібліотеки. Особливо треба зауважити, що перед використанням даного методу треба розділити вхідну вибірку даних на навчальну та тестову і використовувати для отримання значення оптимальної довжини ділянки активації сигналу саме навчальну вибірку. Адже, у випадку використання всього набору даних, буде порушуватись принцип невикористання тестової вибірки при синтезі та навчанні НМ. Це не дасть змоги адекватно оцінити якість її роботи.

Згорткова НМ модель була реалізована у класі `Classifier_MDCNN` модуля `main.py`. За замовчуванням, НМ має три згорткових шара з шарами батч-нормалізації між ними, шар глобального максимального об'єднання та кінцевий повнозв'язний шар з функцією активації `softmax`. При потребі, користувач програмної бібліотеки може легко змінювати архітектуру НМ шляхом зміни, додавання, або видалення змінних відповідного класу, що легко дозволяє робити обрана мова програмування.

Також у даному класі реалізовано методи для динамічного зменшення швидкості навчання моделі при знаходженні на плато функції помилки НМ, на основі даних моніторингу функції втрат, збереження її вагових

коефіцієнтів на найкращій епосі навчання моделі та візуалізації результатів навчання. Додатково реалізовано завантаження та вивантаження моделі до файлу на диску.

Розробка даного модуля дозволила користувачу синтезувати, навчати та тестувати НМ моделі для класифікації багатовимірних сигналів. Окремо реалізовано метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних, що робить можливим використання НМ для сигналів такого типу. Сукупність цих методів, реалізованих у даному модулі, дає можливість проводити експериментальну оцінку роботи НМ моделі.

4.2 Інформаційне забезпечення

Програмна бібліотека була розроблена для дослідження діагностичних моделей, що керуються даними, а саме НМ моделей для класифікації багатовимірних сигналів. Тому важливою частиною її розробки була реалізація інформаційного забезпечення.

Завантаження даних відбувається з файлів формату .txt, .xls, або .xlsx. Кожен файл є окремим екземпляром початкової вибірки. Колонки файлу відповідають різним параметрам багатовимірного сигналу, рядки відповідають моментам часу, коли ці параметри були виміряні. Кожен файл, на самому початку, містить код істинного класу екземпляру даних.

Вихідні дані зберігаються у ідентичному форматі .txt, або .xlsx. Замість істинного класу, на самому початку файлу, містяться результати роботи НМ моделі для відповідного екземпляру у вигляді: порядковий номер активації – мітка класу. Так, як у загальному вигляді, передбачається наявність у одного екземпляра початкової вибірки одного, або декількох проміжків активації сигналу.

4.3 Технічні засоби

До складу технічних засобів для сталого функціонування програмної бібліотеки висуваються такі вимоги: персональний IBM-сумісний комп'ютер з процесором Intel Core 2 Duo, оперативною пам'яттю 4 Гб, 19'' монітором, клавіатурою та маніпулятором типу «миша».

У випадку використання користувачем розроблених програмних інтерфейсів для синтезу більш глибоких НМ моделей та навчання на великій кількості даних високодискретних багатовимірних сигналів йому може знадобитися додаткове технічне забезпечення, наприклад графічна карта з необхідними характеристиками, або процесор з певною частотою роботи та кількістю ядер. Детальніше, характеристики необхідних технічних засобів користувач може дізнатися у програмній документації бібліотеки TensorFlow [43], яка була використана у бібліотеці для роботи з НМ моделями.

4.4 Висновки за розділом 4

На основі розробленої програмної архітектури було реалізовано програмну бібліотеку для проведення досліджень моделей глибоких НМ у застосуванні до задачі класифікації багатовимірних сигналів. Програма складається з трьох модулів: модулю введення-виведення даних, модулю попередньої обробки даних та модулю створення НМ.

Було реалізовано існуючі та запропоновані специфічні математичні методи для обробки даних одновимірних та багатовимірних ЕМГ сигналів. Після обробки вхідних даних цими методами, вони є готовими для навчання та тестування НМ моделі на їх основі. Також було розроблено інформаційне забезпечення для завантаження вибірок даних та збереження результатів експериментів, описано склад необхідних технічних засобів та ситуації у яких його може бути розширено.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

5.1 Опис застосування програми

На основі розробленої архітектури, було розроблено програмну бібліотеку для проведення досліджень у сфері обчислювального інтелекту. Дослідник може використовувати програмний застосунок для дослідження роботи НМ моделей та методів попередньої обробки даних у вирішенні задач класифікації багатовимірних сигналів на запропонованій або власних вибірках даних.

Програмна бібліотека реалізує повний цикл роботи інженера з аналізу даних з вибіркою багатовимірних сигналів. Починаючи від завантаження вибірки, до вивантаження результатів її класифікації на диск. У якості класифікаційної моделі пропонується глибока згорткова НМ, що є наскрізною моделлю глибокого навчання та реалізує часово-інваріантні фільтри для необроблених вхідних даних сигналу. Тому у випадку коли вхідні сигнали є інформативним на протязі всього періоду своєї тривалості та мають однакову розмірність, вони не вимагають попередньої обробки перед завантаженням у модель.

У протилежному випадку, необхідно проводити попередню обробку даних. Це може допомогти збільшити вхідну вибірку шляхом розділення сигналу на декілька нових, які можуть розглядатися як окремі екземпляри з міткою класу із початкової вибірки. Також це допоможе очистити її від неінформативних фрагментів та прибрати зсуви у часі між різними екземплярами вибірки під час вимірювання сигналів.

Так, як у дипломній роботі розглядалась задача класифікації багатовимірного ЕМГ сигналу, то у програмній бібліотеці було розроблено специфічні існуючі та запропоновані математичні методи для вирішення цієї задачі.

У модулі попередньої обробки даних реалізовано існуючі методи виділення періодів активності з одновимірного ЕМГ сигналу та запропонований метод виділення періодів активності з багатовимірного ЕМГ сигналу. У модулі створення НМ реалізовано запропонований метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору для отримання даних, придатних для синтезу на їх основі НМ моделі. В разі потреби, дослідник може опускати деякі методи, або замінювати їх на специфічні для свої даних.

5.2 Інструкція по роботі з програмою

Для того щоб розпочати роботу з розробленою програмною бібліотекою досліднику потрібно її встановити у своєму робочому оточенні. Для зручності користувача, розроблену програмну бібліотеку було реалізовано у вигляді рір – пакету. Це така система керування пакетами, яка використовується для встановлення та керування програмними пакетами, які написані на мові програмування Python.

Розроблена програмною бібліотека є кросплатформовою та може бути використана на будь-якій з популярних ОС. Якщо у дослідника є можливість вибору ОС для роботи, то рекомендується використовувати Linux Ubuntu 20.04. У випадку роботи з програмною бібліотекою на даній ОС, гарантується її швидке становлення, налаштування, запуск, а також простота розширення функціоналу при приєднанні додаткових користувацьких модулів. Далі буде наведена інструкція для користування програми під управлінням даної ОС.

5.2.1 Встановлення програми

У випадку використання Linux Ubuntu 20.04 у системі після встановлення уже доступна потрібна версія інтерпретатора Python. Для

роботи з програмною бібліотекою, у системне або віртуальне оточення Python необхідно встановити бібліотеки, що вона використовує ці процедури виконуються шляхом запуску наступної команди з командного рядка ОС: `pip install pandas numpy matplotlib`. Якщо дослідник бажає навчати НМ використовуючи процесор необхідно виконати: `pip install tensorflow`; у випадку використання графічної карти: `pip install tensorflow-gpu`.

Далі треба встановити розроблену бібліотеку шляхом виконання: `pip install nn-md-classificator`, або шляхом звичайного переносу її файлів у віртуальне оточення Python з необхідними встановленими бібліотеками. Після здійснення цих кроків, користувач отримує можливість повністю вільно використовувати усі можливості данної програмної бібліотеки.

Для того щоб розпочати роботу з розробленою програмною бібліотекою досліднику потрібно її встановити у своєму робочому оточенні. Для зручності користувача, розроблену програмну бібліотеку було реалізовано у вигляді `pip` – пакету [54]. Це така система керування пакетами, яка використовується для встановлення та керування програмними пакетами, які написані на мові програмування Python.

5.2.2 Запуск програми

Після закінчення процедури встановлення, необхідно імпортувати всю програмну бібліотеку у `.py` файл, наприклад: `import nn_md_classificator as nc`; або будь-яку із її частин, наприклад: `from nn_md_classificator.main import Classifier_MDCNN`.

Для зручного опанування розробленої бібліотеки, у її основному програмному модулі `main.py` реалізовано стандартний сценарій її використання. Запустити його можна шляхом запуску наступної команди з командного рядка ОС: `python3 main.py`. Після здійснення процедури імпорту, бібліотека, або її імпортована частина знаходиться у повністю робочому стані та не потребує додаткових системних налаштувань.

5.2.3 Методика роботи з програмою

Робота з програмною бібліотекою розділяється на 4 етапи: етап завантаження вибірки, етап попередньої обробки даних, етап налаштування НМ моделі та етап отримання і збереження результатів експерименту.

Під час етапу завантаження вибірки необхідно обрати директорію із заздалегіть приведеними до необхідного формату файлами з даними вибірки за допомогою файлової системи ОС, вказати шлях та завантажити їх.

Під час етапу попередньої обробки даних необхідно обрати методи обробки одновимірних сигналів, за кожним із параметрів багатовимірного сигналу. Після цього об'єднати результати за допомогою методу обробки багатовимірних сигналів. У разі роботи з вирішення задачі класифікації багатовимірного ЕМГ сигналу рекомендується звернути увагу на реалізовані у програмній бібліотеці існуючі методи для автоматичного виділення періодів активності з одновимірного ЕМГ сигналу. А також на запропоновані методи виділення періодів активності з багатовимірного ЕМГ сигналу та знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних для застосування цих даних у наступному етапі роботи з програмною бібліотекою.

Під час етапу налаштування НМ моделі необхідно налаштувати критерій зупинки навчання нейромоделі, гіперпараметри алгоритму навчання нейромоделі та архітектуру нейромоделі. Після налаштування треба запустити процедуру навчання та тестування нейромоделі.

Під час етапу отримання і збереження результатів експерименту, результати можна отримати у вигляді текстових результатів, які можна зберігти у файл, або у графічному форматі.

На будь якому етапі є можливим доповнення програмного застосунку своїми додатковими методами через те, що формат програмної бібліотеки передбачає комбінування заздалегіть реалізованих методів з користувацькими, через власні програмні інтерфейси.

5.3 Приклад роботи користувача з програмою

Для того щоб розпочати роботу з розробленою програмною бібліотекою досліднику потрібно її встановити у своєму робочому оточенні.

Після закінчення процедури встановлення, необхідно імпортувати всю програмну бібліотеку у .py файл, наприклад: `import nn_md_classifier as nc;` або будь-яку із її частин, наприклад: `from nn_md_classifier.main import Classifier_MDCNN.`

Розглянемо типовий сценарій роботи користувача з програмною бібліотекою. Нехай користувач має вибірку даних у форматі .xlsx, у якій кожен файл містить набір синхронізованих у часі ЕМГ сигналів з різних м'язів однієї м'язової групи пацієнта. Тому він може розглядати цей набір одновимірних сигналів діагностування, як один багатовимірний сигнал. Також користувач має мітки класу для кожного пацієнта. Приклад фрагменту файлу даних наведено на рис. 5.1.

0.0405	3.882941636409E-06	-5.3763805198421E-06	8.66194687261861E-06	-3.21587227746659E-05	1.93151448705055E-05
0.0414	6.87805099655E-06	-1.4761979471118E-05	2.21901888090992E-05	-7.8242689512376E-05	5.21377156888282E-05
0.0423	1.322782389419E-06	-1.4767525004294E-05	2.26539893644065E-05	-4.50733886232037E-05	4.70423063865759E-05
0.0432	-1.923580477167E-06	-7.7740486011366E-06	1.54574100765461E-05	3.25368256306563E-05	8.07887443520678E-06
0.0441	-1.409243868832E-06	-2.6258767859132E-06	8.79042921876834E-06	6.06555038670354E-05	-2.38025723653919E-05
0.045	-2.163385608082E-06	-9.7023193542147E-07	1.79358279971363E-06	4.78991162762091E-05	-4.33220839415959E-05
0.0459	3.052573980073E-06	-4.4711993043584E-06	-2.7311295973592E-06	2.45868201001898E-05	-5.04692527448298E-05
0.0468	6.557921991299E-06	-6.4757587154044E-06	-7.45485549362749E-06	-4.3137792897293E-05	-3.14693963906024E-05
0.0477	3.998735022241E-06	-5.1098312757636E-06	-1.1004222555085E-05	-0.000129154119146	1.0408395096141E-05
0.0486	1.371454437868E-05	-5.2439918087414E-06	-7.47563203978399E-06	-9.39466468653618E-05	4.12020494171567E-05
0.0495	1.522283623601E-05	-3.1734836157677E-06	-8.23943931498452E-06	1.36918837260925E-05	3.34671787678855E-05
0.0504	-5.971695845564E-06	1.59631069025235E-06	-1.34892903329006E-05	1.42265081995975E-05	4.1384396486758E-07
0.0513	-9.988976999687E-06	4.74174228945375E-06	-1.13929940058077E-05	-3.85800424615679E-05	-2.65717204018477E-05
0.0522	-7.053806791666E-06	6.45825263435087E-06	-1.14080040682564E-05	-1.36912800133965E-05	-3.63881602017351E-05
0.0531	-1.443407759533E-05	6.61470575502772E-06	-1.39669876840621E-05	-1.14960044720482E-05	-2.98390419450464E-05
0.054	-1.169257817464E-05	6.66242226525396E-06	-1.36289905303414E-05	-7.58702893131319E-05	-3.13083482888308E-06
0.0549	-1.664017973479E-06	1.15422418724249E-05	-1.25455025235265E-05	-5.07700200890205E-05	2.94716325117248E-05
0.0558	1.149232614025E-05	1.64840635873346E-05	-7.2077108469812E-06	2.52388548152965E-05	2.61416172072488E-05
0.0567	2.062581491883E-05	1.18440647920155E-05	-1.97639146049222E-07	2.8959346654029E-05	-1.85904713657308E-05
0.0576	1.607146325391E-05	1.39490790259975E-06	6.60304582292116E-06	1.95356681940458E-05	-5.29737564821671E-05
0.0585	1.965401519161E-06	-3.7111170209951E-06	1.20705078954709E-05	5.35013902461546E-05	-4.97322213736588E-05

Рисунок 5.1 – Приклад фрагменту файлу даних багатовимірних сигналів

Користувач обирає директорію з даними вибірки за допомогою файлової системи ОС, вказує шлях та завантажує їх, шляхом використання методу `read_file(my_path)` із модулю введення-виведення даних.

Під час етапу попередньої обробки користувач обрає метод обробки одновимірних сигналів, за кожним із параметрів багатовимірного сигналу за характеристиками сигналів та власним дослідницьким досвідом, та підбирає оптимальні гіперпараметри цього методу. На цьому етапі він використовує один з наступних модулів програмної бібліотеки, в залежності від обраного методу: `hodges_bui.py`, `bonato.py`, `lidiert.py`, `abbink.py`, `solnik.py`, `silva.py`, `londral.py` у якості модулю попередньої обробки даних, так як всі вони мають однаковий інтерфейс з точністю до набору їх гіперпараметрів. Приклад результатів попередньої обробки даних наведено на рис. 5.2.

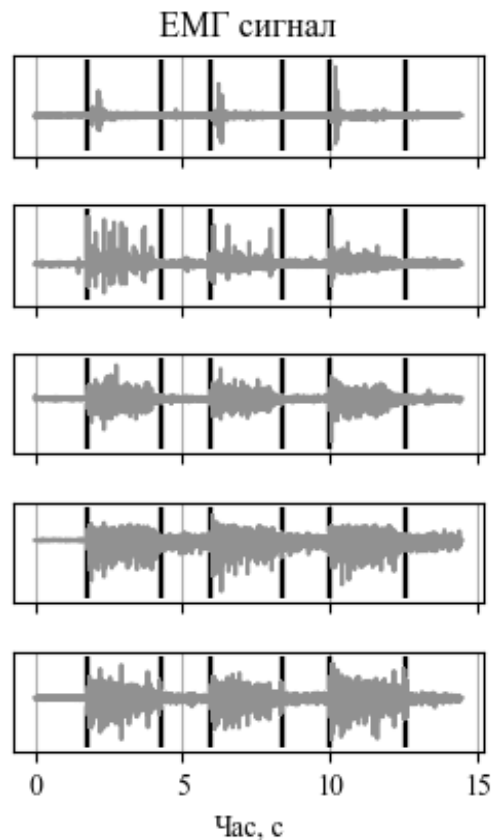


Рисунок 5.2 – Приклад результатів попередньої обробки даних

Після цього користувач використовує запропонований метод виділення періодів активності з багатовимірного ЕМГ сигналу `md_onsets_detection(X, od_method)` з програмного модуля `md_detection.py` та у

результаті отримує набір точок активації та деактивації багатовимірною сигналу. В результаті даного етапу користувач зміг виділити інформативний сигнал та розширити початкову вибірку у три рази (виявилось, що кожен багатовимірний сигнал має три ділянки активації).

Далі, користувач знаходить оптимальну довжини ділянки активації багатовимірною ЕМГ сигналу для навчального набору даних `md_tsol_calc(onsets)` з програмного модуля `md_detection.py` для застосування цих даних у наступному етапі роботи з програмною бібліотекою.

Під час етапу налаштування НМ моделі користувач налаштовує критерій зупинки навчання нейромоделі, гіперпараметри алгоритму навчання нейромоделі та архітектуру нейромоделі шляхом налаштування класу `Classifier_MDCNN` з програмного модуля `main.py`. Після налаштування він запускає процедуру навчання `train(X_train, Y_train)` та тестування нейромоделі `test(X_test, Y_test)`, що є методами цього класу.

Під час етапу отримання і збереження результатів експерименту, користувач отримує результати у текстовій та графічній формі. За потребою, результати експерименту можна зберегти у файл.

5.4 Методика тестування програми

Ручне тестування програмної бібліотки можливе шляхом перевірки усіх функцій які вона надає. У разі проведення ручного тестування, необхідно протестувати: функції попередньої обробки одновимірних та багатовимірних сигналів, функцію знаходження оптимальної довжини ділянки активації багатовимірною ЕМГ сигналу, функцію встановлення кількості епох навчання нейромоделі, функціональну можливість встановлення порогового значення допустимої помилки при навчанні, функції встановлення швидкості навчання та константи часу пошуку, функцію побудови архітектури нейромоделі, функцію запуску процедури навчання та тестування нейромоделі, функціональну можливість зупинки

процесу навчання, функціональну можливість отримання результатів експерименту у текстовій (з можливістю збереження на диск) та графічних формах.

5.5 Практичні завдання для тестування та експериментального дослідження програми

Для експериментальної перевірки розроблених методів та для тестування розробленої програмної бібліотеки, з її допомогою було вирішено практичну задачу класифікації багатовимірних сигналів діагностування. А саме, задачу класифікації пацієнтів за даними багатовимірних ЕМГ сигналів з груп їх м'язів.

Проводиться вимірювання синхронізованих у часі ЕМГ сигналів у пацієнтів, що можуть бути об'єднані у багатовимірний ЕМГ сигнал. Лікарем-травматологом проводиться діагностування пацієнта та віднесення його до одного з класів із набору заздалегіть визначених класів (розмічення даних). Метою є побудова класифікаційної моделі, що буде автоматично класифікувати нових пацієнтів на приналежність до одного із класів з високою точністю класифікації.

У якості вибірки даних була використана вибірка зібрана у 10 футболістів професійної футбольної команди Alcorcon B, що у сезоні 2019/2020 змагалася у третьому дивізіоні іспанської футбольної ліги. Вимірювання проводилися у жовтні 2019 року та у лютому 2020 року. ЕМГ сигнали вимірювались з п'яти м'язів коліна, а саме: rectus femoris, vastus medialis, vastus lateralis, biceps femoris, semitendinosus, на лівій та правій ногах футболіста, та є синхронізованими у часі у межах вимірювань з однієї ноги.

Під час кожного вимірювання, спортсмени виконували три рази фізичну вправу, що називається міст на одній нозі. Футболісти проходили діагностування у лікаря-травматолога команди, який розділив їх на два

класи: травмованих та здорових. П'ять футболістів виявилися травмованими, а п'ять – здоровими.

Так, як маємо справу з задачею бінарної класифікації, то введемо специфічні метрики якості до неї, щоб найбільш повно оцінити якість роботи класифікаційної моделі за формулами (5.1 - 5.3):

$$Accuracy = \frac{TN + TP}{TN + FN + TP + FP}, \quad (5.1)$$

де TP – кількість істинно позитивних екземплярів за результатами класифікації;

TN – кількість істинно негативних екземплярів;

FP – кількість помилково позитивних екземплярів;

FN – кількість помилково негативних екземплярів.

$$PPV = \frac{TP}{TP + FP}. \quad (5.2)$$

$$TPR = \frac{TP}{TP + FN}. \quad (5.3)$$

Для автоматичного виділення періодів активності з одновимірною ЕМГ сигналу було обрано метод трешолдінгу з використанням тривалості активації та кількості успішних вимірів [44]. Після цього було застосовано запропонований метод виділення періодів активності з багатовимірною ЕМГ сигналу. Виділення періодів активності відбувалося окремо для багатовимірних ЕМГ сигналів з кожної ноги, так як сигнали з різних ніг не є синхронізованими у часі. Після виділення, багатовимірні ЕМГ сигнали з обидвох ніг були об'єднані у один багатовимірний сигнал, так як з умов

проведення експерименту впливає, що вони характеризують виконання однієї і тієї ж фізичної вправи. Таким чином вдалось прибрати зсуви у часі між різними сигналами всередині екземпляру та між екземплярами вибірки.

У результаті, початкову вибірку було розширено до 60 екземплярів, де кожна ділянка активації – окремий екземпляр вибірки (по 2 вимірювання на кожного із 10 футболістів, кожне з яких містить по 3 ділянки активації). Отримана вибірка була випадковим чином, з дотриманням балансу класів, розділена на навчальну вибірку з 40 екземплярів, та на тестову з 20.

Після цього було застосовано метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних. Із початкових багатовимірних ЕМГ сигналу було отримано нові екземпляри даних однакової розмірності матриці. Ці данні є придатними для побудови класифікаційної НМ моделі.

У якості НМ моделі було обрано глибоку згорткову НМ, що має три згорткових шара з шарами батч-нормалізації між ними, шар глобального максимального об'єднання та кінцевий повнозв'язний шар з функцією активації softmax та є моделлю за замовчуванням у розробленій програмній бібліотеці.

Функція втрат нейронної мережі розраховувалась за формулою (1.4). Навчання нейромережі проводилось на протязі 200 епох. Далі будуть наведені результати роботи НМ моделі для задачі класифікації багатовимірного сигналу.

5.6 Результати тестування та експериментального дослідження програми

Для оцінки якості класифікації багатовимірного сигналу для різних архітектур НМ та вибору найкращої було проведено експериментальне дослідження їх застосування та результатів роботи для навчальної та тестової вибірок. неперотестування розробленої програмної бібліотеки, з її допомогою

було вирішено практичну задачу класифікації багатовимірних сигналів діагностування.

Архітектура НМ моделі інкрементивно ускладнювалась протягом експерименту, шляхом додавання додаткових шарів різних типів. Розмірності згорткових шарів збільшуються починаючи від 64 нейронів для першого шару, та подвоюючись для кожного наступного згорткового шару НМ. У термінах глибокого навчання НМ, це називається поглибленням НМ. Для оцінки якості роботи НМ моделі на тестовій вибірці даних використовувалися метрики якості роботи класифікаційної моделі для задачі бінарної класифікації. Результати експериментів наведено в табл. 5.1.

Таблиця 5.1 — Результати роботи НМ моделі на тестовій вибірці даних для задачі бінарної класифікації

№ НМ моделі	К-ть згорткових шарів	Наявність шарів батч- нормалізації	Ассурасу	PPV	TPR
1	2	-	0.80	0.80	0.73
2	2	+	0.90	0.90	0.75
3	3	-	0.85	0.90	0.82
4	3	+	1.00	1.00	1.00
5	4	-	0.85	0.80	0.82
6	4	+	0.95	0.90	0.9
7	5	-	0.85	0.80	0.82
8	5	+	0.90	0.80	0.9

За результатами експериментів, можна зробити висновок, що використані та розроблені методи попередньої обробки даних у сукупності з використанням, у якості класифікатора, глибокої згорткової НМ демонструють високу якість класифікації багатовимірного сигналу.

Результати показують, що найкращі метрики якості класифікації демонструє глибока згорткова НМ, що має три згорткових шара. Більш глибокі моделі НМ демонструють гіршу якість. Це пояснюється тим, що навчальна вибірка є не досить великою для навчання моделей такої складності.

Також видно, що важливим є застосування шарів батч-нормалізації після кожного згорткового шару, яке значно підвищує якість класифікації.

Графік функції втрат на кожній епосі навчання НМ моделі, що має найбільшу точність класифікації наведено на рис. 5.3.

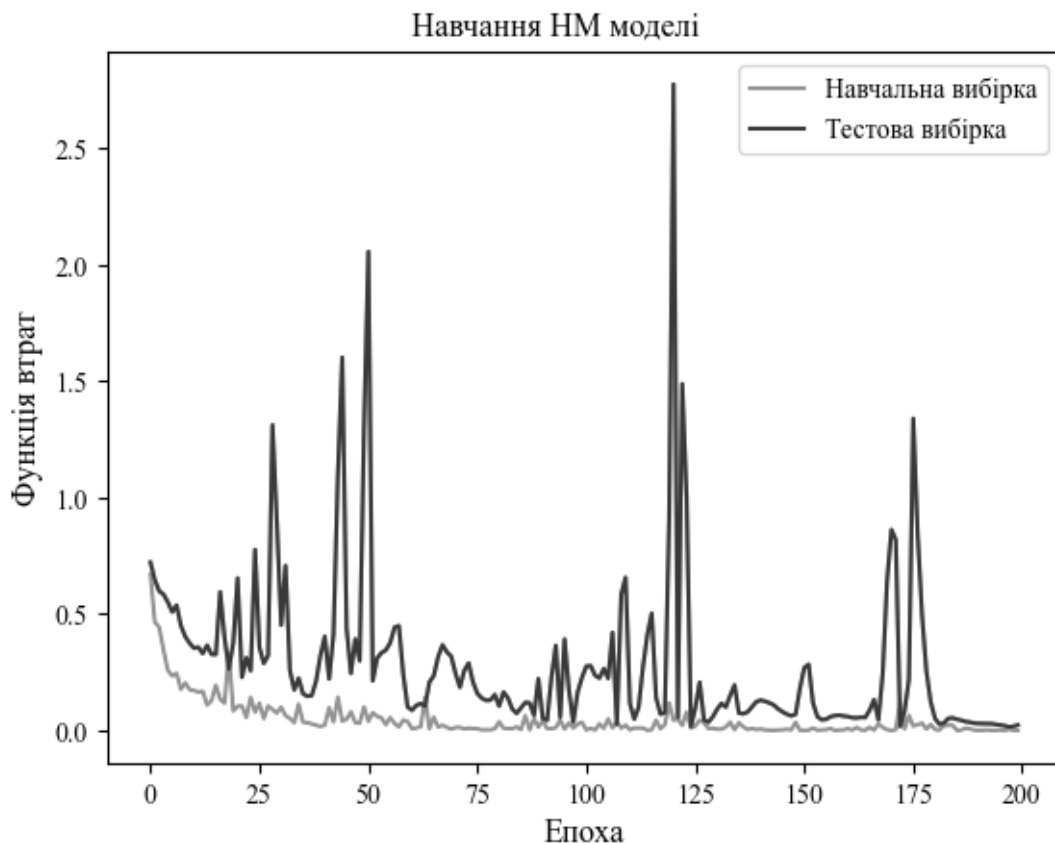


Рисунок 5.3 – Функції втрат на кожній епосі навчання найбільш точної НМ моделі

За результатами аналізу функції втрат НМ моделі, можна зробити висновок, що її навчання проходить успішно. Обраної кількості епох

навчання достатньо, щоб значення функцій втрат навчальної та тестової вибірки стали приблизно рівними.

5.7 Висновки за розділом 5

Було проведено опис застосування розробленої програмної бібліотеки. Складено інструкцію для її використання, яка включає: інструкції з встановлення, запуску та методику роботи з програмною бібліотекою. Було наведено приклад роботи користувача з програмною бібліотекою та методику її тестування.

За її допомогою була вирішена практична задача класифікації пацієнтів за даними багатовимірних ЕМГ сигналів з груп їх м'язів. За допомогою розробленої програмної бібліотеки була підтверджена ефективність розроблених методів виділення періодів активності з багатовимірного ЕМГ сигналу та знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу. Також була підтверджена ефективність застосування глибокої згорткової для класифікації таких багатовимірних сигналів.

6 ОРГАНІЗАЦІЙНО-ЕКОНОМІЧНА ЧАСТИНА

Автоматична ідентифікація та класифікація стану здоров'я людини за даними біомедичних сигналів є важливою задачею, яка може допомогти лікарям економити час, який раніше вони витрачали на ручну класифікацію таких сигналів для постановки діагнозу пацієнту. Тема дипломної роботи присвячена класифікації багатовимірних ЕМГ сигналів, що характеризують м'язову активність. Так, як у сучасних задачах біомедичної діагностики складно характеризувати комплексний процес за одним сигналом або параметром спостереження, то все частіше ставиться задача класифікації багатовимірних сигналів, що характеризуються вимірюванням більше ніж одного параметра в кожен момент часу.

На даний час, на ринку існують рішення лише для класифікації одновимірних ЕМГ сигналів, що вимагає від лікаря кінцевого аналізу отриманих результатів для остаточної класифікації стану пацієнта та не враховують кореляцію між різними параметрами багатовимірного сигналу. Рішення, запропоноване у даній дипломній роботі, дозволяє усунути ці недоліки та здійснювати класифікацію багатовимірних ЕМГ сигналів на більш високому рівні автоматизації. Опис ідеї наведено у табл. 6.1.

Таблиця 6.1 – Опис ідеї

Зміст ідеї	Напрямки застосування	Вигоди для споживачів (користувачів)
1	2	3
Програмний застосунок, який на основі аналізу багатовимірних сигналів ЕМГ, проводить класифікацію пацієнтів на хворих та здорових.	1. Використання лікарем для економії часу, який раніше втрачався на ручну класифікацію таких сигналів.	Уникнення втрати робочого часу лікарем та ймовірності людської помилки при аналізі великої кількості числових даних.

Продовження таблиці 6.1

1	2	3
	2. Використання лікарем для телемедицини пацієнтів.	Використання у сукупності з переносними ЕМГ сенсорами для діагностики пацієнтів, без особистої їх присутності.
	3. Використання пацієнтом для регулярного моніторингу стану власних м'язів без постійної участі лікаря.	Економія коштів на оплату послуг лікаря при кожному сеансі моніторингу стану м'язів. Можливість накопичення великої статистики спостережень з використанням переносних ЕМГ сенсорів для призначення оптимального лікування у лікаря.

На основі наведеного опису ідеї, а також результатів аналізу можливих напрямків застосування та вигоди для споживачів [55], можна зробити висновок, що потенційний ринком для впровадження данного програмного застосунку є ринок медичного ПЗ для аналізу ЕМГ. Визначивши потенційний ринок необхідно послідовно проаналізувати його

характеристики, щоб окреслити ринкові можливості запуску реалізації ідеї. Попередню характеристику потенційного ринку наведено у табл. 6.2.

Таблиця 6.2 – Попередня характеристика потенційного ринку

№	Показники стану ринку (найменування)	Характеристика
1	2	3
1	Головні конкуренти	Biometrics Ltd – підтримує основні операції для аналізу одновимірного ЕМГ сигналу: обчислення випрямлення, середньоквадратичного значення, інтеграції, середньої частоти, медіанної частоти.
		EMGworks – підтримує запис ЕМГ сигналів, використовуючи широкий спектр готових до використання протоколів збору даних, побудову графіків та різні варіанти аналізу одновимірного ЕМГ сигналу.
		EMG Analysis – підтримує широкий спектр потужних методів аналізу з використанням методів швидкого перетворення Фур'є та класичних методів для аналізу одновимірного ЕМГ сигналу.

Продовження таблиці 6.2

1	2	3
2	Динаміка ринку (якісна оцінка)	зростає
3	Наявність обмежень для входу (характер обмежень)	Лояльність покупців, ефект масштабу, мережевий ефект.
4	Специфічні вимоги до стандартизації та сертифікації	Вимоги до стандартизації та сертифікації регулюються наказом МОЗ України "Про введення відомчої сертифікації програмних продуктів та програмних компонентів апаратно-програмних комплексів медичного призначення " від 21.02.1994.

Визначивши попередні характеристики потенційного ринку необхідно на їх основі визначити потенційні групи клієнтів, їх характеристики, та сформувати орієнтовний перелік вимог до товару для кожної групи. Попередню характеристику потенційних клієнтів ринку наведено у табл. 6.3.

Таблиця 6.3 – Попередня характеристика потенційних клієнтів

№	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів (користувачів)
1	2	3	4	5
1	Автоматизація класифікації стану пацієнта за даними сигналів ЕМГ.	Лікарі травматологи, пацієнти.	Лікарі можуть діагностувати вручну, а пацієнти – ні.	Точність класифікації повинна бути високою.

Продовження таблиці 6.3

1	2	3	4	5
2	Теледіагностика м'язів пацієнта без особистої присутності лікаря.	Лікарі травматологи – ортопеди, пацієнти.	Лікарі мають кваліфікацію проводити вимірювання ЕМГ сигналів та інтерпретувати їх результати, а пацієнти – ні.	Пацієнт повинен мати можливість легко проводити вимірювання ЕМГ сигналів. Кінцевий результат програми не повинен вимагати подальшої інтерпретації
3	Накопичення результатів вимірювань та діагностики для кожного пацієнта.	Лікарі травматологи – ортопеди, пацієнти.	—	Можливості перегляду всієї історії результатів вимірювань та діагностики для пацієнта.

Визначивши попередні характеристики потенційного ринку та груп клієнтів, доцільно провести SWOT-аналіз середовища реалізації інноваційного проєкту для виявлення сильних і слабких сторін, можливостей і загроз при реалізації проєкту. SWOT-аналіз наведено у табл. 6.4.

Таблиця 6.4 – SWOT-аналіз

Сильні сторони:	Слабкі сторони:
- іноваційна ідея в основі проєкту; - унікальність та ефективність методів покладених в основу; - невеликий штат працівників.	- обмежена цільова аудиторія; - потреба у додатковому обладнанні для використання продукту; - недостатньо високий прибуток.
Можливості:	Загрози:
- вдосконалення програмного застосунку для більш детальної постановки діагнозу; - вдосконалення програмного застосунку для зручнішої взаємодії з користувачем; - вихід на міжнародний ринок; - інтеграція з іншими продуктами на ринку.	- бойкот використання продукту лікарями; - поява нових конкурентів; - зростаючий конкурентний тиск; - зниження репутації; - банкрутство.

На основі попередньо визначених характеристик потенційного ринку, груп клієнтів та SWOT-аналізу доцільно розробити ринкову стратегію. Першим кроком, це передбачає визначення стратегії охоплення ринку: опису цільових груп потенційних споживачів. Вибір цільових груп потенційних споживачів наведено у табл. 6.5.

Таблиця 6.5 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Державні медичні клініки	середня	середній	висока	низька
2	Приватні медичні клініки	висока	високий	висока	середня
3	Професійні спортивні клуби	висока	високий	середня	середня
4	Аматорські спортивні клуби	низька	низький	низька	середня
5	Фізичні особи, які потребують моніторингу та діагностики стану м'язів	висока	середній	низька	середня
Які цільові групи обрано: приватні медичні клініки, професійні спортивні клуби, фізичні особи, які потребують моніторингу та діагностики стану м'язів.					

На основі обраних цільових груп потенційних споживачів треба ідентифікувати стейкхолдерів, здійснити їх пріоритизацію та побудувати карту стейкхолдерів, яка дозволить візуалізувати картину їх взаємозв'язків. Карту стейкхолдерів проєкту наведено на рис. 6.1.

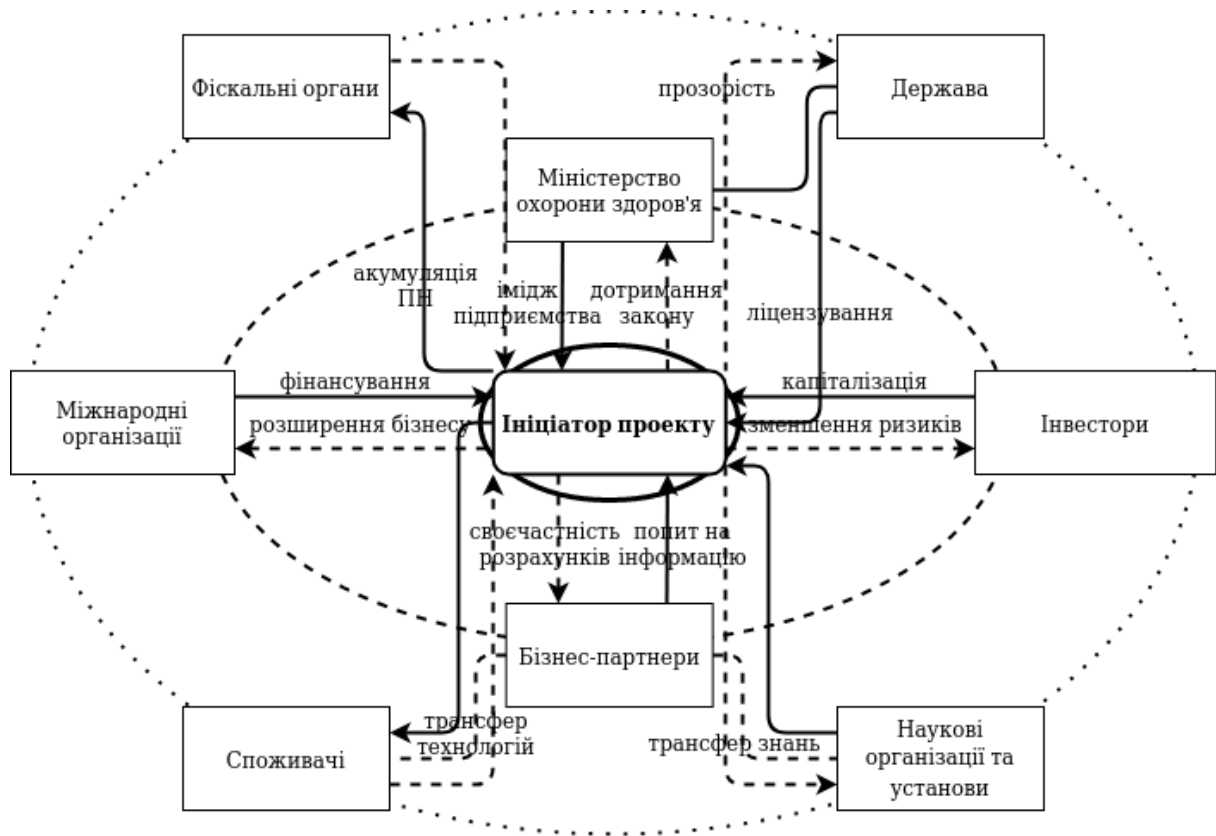


Рисунок 6.1 – Карта стейкхолдерів проекту

З метою планування фонду оплати праці робітників, залучених до реалізації проекту, необхідно визначити витрати на оплату праці з урахуванням балансу робочого часу одного працівника та суму ЄСВ. Склад, чисельність та фонд заробітної плати для виробничих працівників наведено у табл. 6.6, а для адмінперсоналу у табл. 6.7.

Таблиця 6.6 – Склад, чисельність та фонд заробітної плати виробничих працівників

Категорії працівників	Наявна чисельність, осіб		Тарифна ставка за розрядом виконуваних робіт, грн / год	Ефективний фонд робочого часу, год	Тарифний заробіток, грн.	Преміальний відсоток до тарифного заробітку, %	Розмір премії, грн.	Річний фонд заробітної плати, грн.	ЄСВ, грн.
	за зміну	на добу							
1. Основні працівники	4	4	120	160	19200	20	3840	230400	4224
2. Допоміжні працівники	-	-	-	-	-	-	-	-	-
3. Черговий та ремонтний персонал	-	-	-	-	-	-	-	-	-
Разом виробничих працівників	4	4	-	-	-	-	15360	921600	16896

Таблиця 6.7 – Склад, чисельність та фонд заробітної плати адмінперсоналу

Посада	Кількість осіб	Посадовий оклад, грн.	Преміальний відсоток до окладу, %	Сума премії, грн.	Місячна заробітна плата, грн.	Річний фонд оплати праці, грн.	ЄСВ, грн.
Директор	1	25000	20	5000	30000	360000	6600
Разом управлінського персоналу	1	25000	-	5000	30000	360000	6600

Так, як результатом реалізації проєкту є нематеріальний актив – ПЗ, то не має сенсу розраховувати матеріальні витрати та вартість спожитих послуг у розрахунку на одиницю продукції. Розробка ПЗ не потребує використання сировини. Вона вимагає лише використання електроенергії для живлення комп'ютерів співробітників під час роботи.

Потужність блоків живлення використовуваних комп'ютерів дорівнює 500 Вт. Одночасно використовуються 5 комп'ютерів, 8 годин у день, на протязі 20 робочих днів. Сума використаної електроенергії дорівнює 400 кВт. Згідно тарифам ПАТ «Запоріжжяобленерго» на листопад 2020 року, вартість цієї електроенергії для споживачів I класу напруги становить 680 грн.

Після розрахунку вартості спожитих послуг необхідно визначити річну суму амортизації з урахуванням первісної вартості основних засобів. Розрахунок амортизації наведено у табл. 6.8.

Таблиця 6.8 – Розрахунок амортизації

Група основних засобів	Норма амортизації, %	Первісна вартість ОЗ на 01.01, грн	Надійшло ОЗ		Вибуло ОЗ		Сума, грн.
			дата	пер. вар.	дата	пер. вар.	
Комп'ютерна техніка	4	75000	-	-	-	-	3000
Разом	-	75000	-	-	-	-	3000

Для визначення витрат на поточний ремонт, приймемо їх вартість у розмірі 2% від балансової вартості основних засобів, що дорівнює 1500 грн. Після підрахунку усіх витрат проекту можемо підрахувати собівартість продукту. Розрахунок кошторису витрат наведено у табл. 6.9.

Таблиця 6.9 – Кошторис витрат

Калькуляційні статті	Витрати, грн
1	2
Сировина та матеріали	-
Електропостачання	680
Водопостачання	-
Теплопостачання	-
інші	-
Разом	680
Заробітна плата основних виробничих працівників	76800
ЄСВ	23496
Амортизація	3000

Продовження таблиці 6.9

1	2
Витрати на утримання та експлуатацію основних засобів, поточний ремонт	1500
Загальновиробничі витрати	-
Виробнича собівартість	98196
Адміністративні витрати	30000
Інші витрати	-
Повна собівартість	128876

Підрахувавши кошторис витрат та повну собівартість, необхідно визначити очікуваний дохід та розрахувати фінансовий результат за реалізованим проєктом. У розрахунку приймемо середню вартість реалізації однієї копії програми за 1000 грн. Розрахунок фінансового результату наведено у табл. 6.10.

Таблиця 6.10 – Розрахунок фінансового результату

№ періоду	Обсяги реалізації	Витрати, грн.	Ціна, грн.	Дохід, грн.	Фінансовий результат, грн.
1	2	3	4	5	6
1	10	138876	10000	100000	-38876
2	20	58876	12000	240000	181124
3	30	30000	15000	450000	420000

Продовження таблиці 6.10

1	2	3	4	5	6
Разом	50	128876	-	790000	562248

За результатами розрахунку фінансового результату видно, що вже під час другого періоду реалізації продукту може бути отриманий прибуток. Однак, успішній реалізації проєкту можуть завадити існуючі ризики. Наведемо можливий перелік ризиків, а також шляхи їх упередження та мінімізації.

Можливі ризики проєкту: несприйняття продукту професійною спільнотою лікарів, невідповідність вимог користувачів розробленому ПЗ та зміна вимог з часом.

Для уникнення ризику бойкоту продукту з боку лікарів, необхідно поряд з рекламою переваг продукту також проводити роз'яснювальну компанію серед лікарів. Ключовими її тезами повинні бути пояснення, що розроблене ПЗ не має на меті відібрати роботу у лікарів, а навпаки, що воно допоможе лікарям більш ефективно виконувати свою роботу, уникнути рутинної роботи, та обслуговувати більшу кількість пацієнтів. Також, потрібно наголошувати, що даний продукт зручно використовувати у процесі телемедицини, що допоможе лікарю охопити більш широке коло пацієнтів.

Для мінімізації інших ризиків, пов'язаних з процесом розробки та вдосконалення ПЗ, пропонується замість, до недавнього часу, найбільш розповсюдженої моделі планування та розробки програмного забезпечення - каскадної моделі, використовувати гнучкі методології розробки ПЗ. Наприклад: Agile, Dynamic Systems Development Method, Extreme programming, Scrum та інші. Усі вони базуються на ітеративній розробці, у якій вимоги та рішення еволюціонують через співпрацю між командою та клієнтами.

У результаті виконання організаційно-економічної частини дипломної роботи була проаналізована ідея проєкту, складена характеристика потенційного ринку та клієнтів, проведено SWOT-аналіз, вибір цільових груп потенційних споживачів та складено карту стейкхолдерів проєкту. Було розраховано склад, чисельність та фонд заробітної плати виробничих працівників та адмінперсоналу, вартість спожитих послуг, амортизацію основних засобів, кошторис витрат та фінансового результату впровадження проєкту. Також було наведено можливий перелік ризиків і шляхи їх упередження та мінімізації.

Результати проведеного аналізу та розрахунків свідчать, що розроблений продукт буде користуватися попитом на сучасному ринку серед різних груп клієнтів. Він має свої сильні сторони та можливості для входу у визначені сегменти ринку, шляхом націлення на певні групи клієнтів та через взаємодію з іншими стейкхолдерами проєкту. Розрахунки основних економічних статей реалізації проєкту та прогнозованого прибутку свідчить, що реалізація ПЗ для класифікації багатовимірних ЕМГ сигналів є економічно вигідною.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

В дипломній роботі розглядається процес розробки ПЗ для аналізу та класифікації багатовимірних ЕМГ сигналів. Ціллю цього розділу є огляд питань пов'язаних з охороною праці та цивільної безпеки при впровадженні цього процесу. Роботи при його виконанні проводяться у офісі.

7.1 Аналіз потенційних небезпек

Розробка ПЗ для аналізу та класифікації багатовимірних ЕМГ сигналів виконується в приміщенні, розмірами 7 м x 5 м x 3 м (довжина A , ширина B , висота H).

Небезпечним фактором при проведенні робіт в офісі є небезпека ураження електричним струмом, внаслідок недотримання правил роботи з електроприладами.

Шкідливими та небезпечними факторами на робочому місці є:

- а) недостатня освітленість робочої зони;
- б) відсутність або недолік природного світла;
- в) підвищена яскравість світла;
- г) знижена контрастність;
- д) підвищений рівень шуму;
- е) підвищений рівень вібрації;
- є) підвищена або знижена вологість повітря;
- ж) підвищена або знижена рухливість повітря;
- з) підвищена або знижена іонізація повітря;
- и) підвищений рівень електромагнітних випромінювань;
- і) підвищений рівень статичної електрики;
- й) нервово-психічні перевантаження:
 - 1) розумове перенапруження;
 - 2) перенапруження аналізаторів;

- 3) розумове перенапруження;
- 4) монотонність праці;
- 5) емоційні перевантаження.

7.2 Заходи із забезпечення безпеки

Приміщення офісу, в якому перебувають програмісти, відноситься до приміщень без підвищеної небезпеки ураження електричним струмом.

Обладнання, що використовується в цих приміщеннях є споживачем електроенергії, що живиться від змінного струму 220 В від мережі з заземленою нейтраллю, та відноситься до електроустановок до 1000 В закритого виконання. За способом захисту людини від ураження електричним струмом воно відповідає ГОСТ 12.2.007.0-75* (2001) «ССБТ. Изделия электротехнические. Общие требования безопасности» [56] І (стаціонарні комп'ютери) та ІІ (освітлювальні прилади, кондиціонери, опалювальні пристрої, ноутбуки, сканери) класу захисту.

Комп'ютер є однофазним споживачем електроенергії, що живиться від змінного струму 220 В від мережі із заземленою нейтраллю. Він відноситься до електроустановок до 1000 В закритого виконання, всі струмопровідні частини знаходяться в кожухах. За способом захисту людини від ураження електричним струмом, комп'ютер та периферійна техніка повинні відповідати 1 класу захисту.

Для виключення можливості ураження електричним струмом передбачено:

- організаційні заходи — вивчення та атестація знань правил електробезпеки;
- технічні заходи — устрій системи заземлення електроспоживаючого обладнання згідно «Правилам улаштування електроустановок» [57]: забезпечення недоступності струмопровідних частин для випадкового доторкання, використання ізоляції, використання

методів колективного захисту від ураження електричним струмом. А саме: захисного заземлення, занулення та автоматичного відключення, періодичної перевірки опору заземлення, контролю та профілактиці пошкоджень ізоляції.

Згідно «Правилам улаштування електроустановок» [57] виконані наступні групи заходів з електробезпеки.

Конструктивні заходи забезпечують захист від випадкового дотику до струмопровідних частин за допомогою їх ізоляції та захисних оболонок.

Згідно до ГОСТ 12.1.009-76 (1999) «ССБТ. Электробезопасность. Термины и определения» [58] у приладах II класу захисту використовується подвійна ізоляція - електрична ізоляція, що складається з робочої і додаткової ізоляції.

Так як згідно з НПАОП 40.1-1.32-01 «Правила устройства электроустановок. Электрооборудование специальных установок» [59] офісні приміщення більшості своїй відносяться до класу пожежебезпечної зони - Па (приміщення, в яких містяться тверді горючі речовини), то передбачений ступінь захисту ізоляції обладнання – IP44.

Схемно-конструктивні заходи призначені для забезпечення захисту від ураження електричним струмом при дотику до металевих оболонок, які можуть опинитися під напругою в результаті аварій.

Згідно з ГОСТ 12.1.030-81 (2001) «ССБТ. Электробезопасность. Защитное заземление, зануление» [60] у приміщеннях галузі управління персоналом влаштовується занулення.

Експлуатація електроустановок і електроустаткування проводиться відповідно до НПАОП 40.1-1.01-97 «Правила безопасной эксплуатации электроустановок» [61] та НПАОП 40.1-1.21-98 «Правила безопасной эксплуатации электроустановок потребителей» [62].

7.3 Заходи з виробничої санітарії та гігієни праці

Відповідно до ДБН В2.5-28:2018 «Природне і штучне освітлення» [63] в офісних приміщеннях використовується природне та штучне освітлення. Природне освітлення здійснюється через світлові прорізи, які забезпечують коефіцієнт природної освітленості не нижче 1,5%.

Для захисту від прямих сонячних променів, які створюють прямі та відбиті відблиски на поверхні екранів і клавіатури, використовуються сонцезахисні пристрої, на вікнах встановлені жалюзі або штори. Штучне освітлення в приміщенні здійснено системою загального рівномірного освітлення. Як джерела штучного освітлення в приміщенні застосовуються люмінесцентні лампи. Значення освітленості на поверхні робочого столу в зоні розміщення документів становить 300-500 лк.

Рівні звукового тиску в октавних смугах частот, рівні звуку та еквівалентні рівні звуку на робочих місцях у приміщення нормуються згідно ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [64] та ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку» [65].

Зниження рівня шуму в приміщенні здійснено за допомогою:

- використання більш сучасного обладнання;
- розташування принтерів та різноманітного устаткування колективного користування на значній відстані від більшості робочих місць працівників;
- переведення жорсткого диску в режим сну, або гібернації, якщо комп'ютер не працює протягом визначеного часу;
- використання блоків живлення ПК з вентиляторами на гумових підвісках.

Приміщення офісу, в якому перебувають співробітники галузі управління персоналом, відноситься до приміщень без підвищеної небезпеки ураження електричним струмом.

Показники мікроклімату в офісі відповідають встановленим санітарно-гігієнічним вимогам ДСН 3.3.6-042-99 «Санітарні норми мікроклімату виробничих приміщень» [66], ГОСТ 12.1.005-88 (1991) «ССБТ. Общие санитарно-гигиенические требования к воздуху рабочей зоны» [69] і ГН2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень» [68]. Роботи програмістів в офісі, належить до категорії Іб – легка робота, тому встановлені наступні оптимальні значення параметрів мікроклімату:

а) у холодний період року:

- 1) температура від 22 до 24 С;
- 2) відносна вологість: 40-60%;
- 3) швидкість переміщення повітря: 0,1 м/с;

б) у теплий період року:

- 1) температура від 23 до 25 С;
- 2) відносна вологість: 40-60%;
- 3) швидкість переміщення повітря: 0,2 м/с.

Оптимальні рівні позитивних (n^+) і негативних (n^-) іонів у повітрі приміщення з ПК нормовані згідно ГН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень» [68] і становлять: $n^+=1500-30000$ (шт. на 1 см³); $n^- = 3000-5000$ (шт. на 1 см³). Підтримка оптимального рівня легких позитивних і негативних аероіонів у повітрі на робочих місцях забезпечено за допомогою біполярних коронних аероіонізаторів.

Загальні ергономічні вимоги встановлено ДСТУ ISO 9241-1:2003 «Ергономічні вимоги до роботи з відеотерміналами в офісі. Частина 1. Загальні положення» [69]. Організація робочого місця передбачає: правильне розміщення робочого місця у виробничому приміщенні, вибір

ергономічно обґрунтованого робочого положення, виробничих меблів з урахуванням антропометричних характеристик людини, раціональне компонування обладнання на робочих місцях, врахування характеру та особливостей трудової діяльності.

Розрахуємо кількість комп'ютеризованих робочих місць, оснащених ВДТ, яку можна встановити у даному приміщенні і як їх розташувати відповідно до встановлених норм та правил з охорони праці. Розміри приміщення: довжина $A = 7$ м, ширина $B = 5$ м, висота $H = 3$ м.

Відповідно до НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин» [70] є неприпустимим розташування приміщень, призначених для роботи з ВДТ у підвалах та цокольних поверхах. Також забороняється розташування вибухонебезпечних приміщень категорії А і Б (НАПБ Б.03.002-2007 «Нормы определения категорий помещений, зданий и наружных установок по взрывопожарной и пожарной безопасности» [71]) та виробництв з мокрими технологічними процесами поряд з приміщеннями, де розташовуються комп'ютери, а також над такими приміщеннями, або під ними. Окрім того, виробничі приміщення для роботи з ВДТ не повинні межувати з приміщеннями, у яких рівень шуму і вібрації перевищує допустимі значення.

Вибране приміщення відповідає вищезазначеним вимогам, тому перейдемо до наступного етапу, а саме, до визначення кількості комп'ютеризованих робочих місць, що можна розмістити в даному приміщенні.

Оскільки площа приміщення становить $S_{\text{пр}} = 35 \text{ м}^2$, а площа, на якій розташовується одне робоче місце з ВДТ, відповідно до вимог ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [64] повинна становити не менше $6,0 \text{ м}^2$, то в даному приміщенні можна розмістити щонайбільше п'ять комп'ютеризованих робочих місць. Такої кількості достатньо для технічного оснащення офісу.

Перевіримо, чи відповідає це число нормативу щодо мінімального об'єму приміщення на одне робоче місце з ВДТ, відповідно до вимог ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [64] ($V_{P.M.Min} = 20 \text{ м}^3$). Об'єм приміщення становить 105 м^3 , а об'єм, що припадає на одне комп'ютеризоване робоче місце - $V_{P.M.} = 21 \text{ м}^3$. Таким чином норматив щодо об'єму приміщення на одне робоче місце з ВДТ виконується. Тому розміщуємо в даному приміщенні п'ять комп'ютеризованих робочих місць.

Планування розміщення комп'ютеризованих робочих місць у приміщенні проводимо із врахуванням наступних вимог:

- робочі місця з ВДТ розміщуються на відстані не менше 1 м від стіни зі світловими прорізами;
- між бічними поверхнями ВДТ має бути не менше за 1,2 м;
- відстань між тильною поверхнею одного ВДТ та екраном іншого не повинна бути меншою за 2,5 м;
- прохід між рядами робочих місць має бути не меншим 1 м.

Необхідно також врахувати розміри меблів на комп'ютеризованих робочих місцях, зокрема робочого столу. Відповідно до НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин» [64] рекомендовані розміри столу для робочого місця з ВДТ становлять: висота – 725 мм, ширина – 600-1400 мм, глибина – 800-1000 мм. Приймаємо, що робочий стіл має такі розміри: ширина – 1200 мм, глибина – 800 мм.

Найкраще розмістити комп'ютеризовані робочі місця рядами вздовж стіни з вікнами. Це дасть змогу унеможливити дзеркальне відбиття на екрані ВДТ джерел природного світла (вікон) та потрапляння останніх у поле зору операторів, що погіршує умови їх зорової роботи.

На рис. 7.1 наведено план офісного приміщення з комп'ютеризованими робочими місцями.

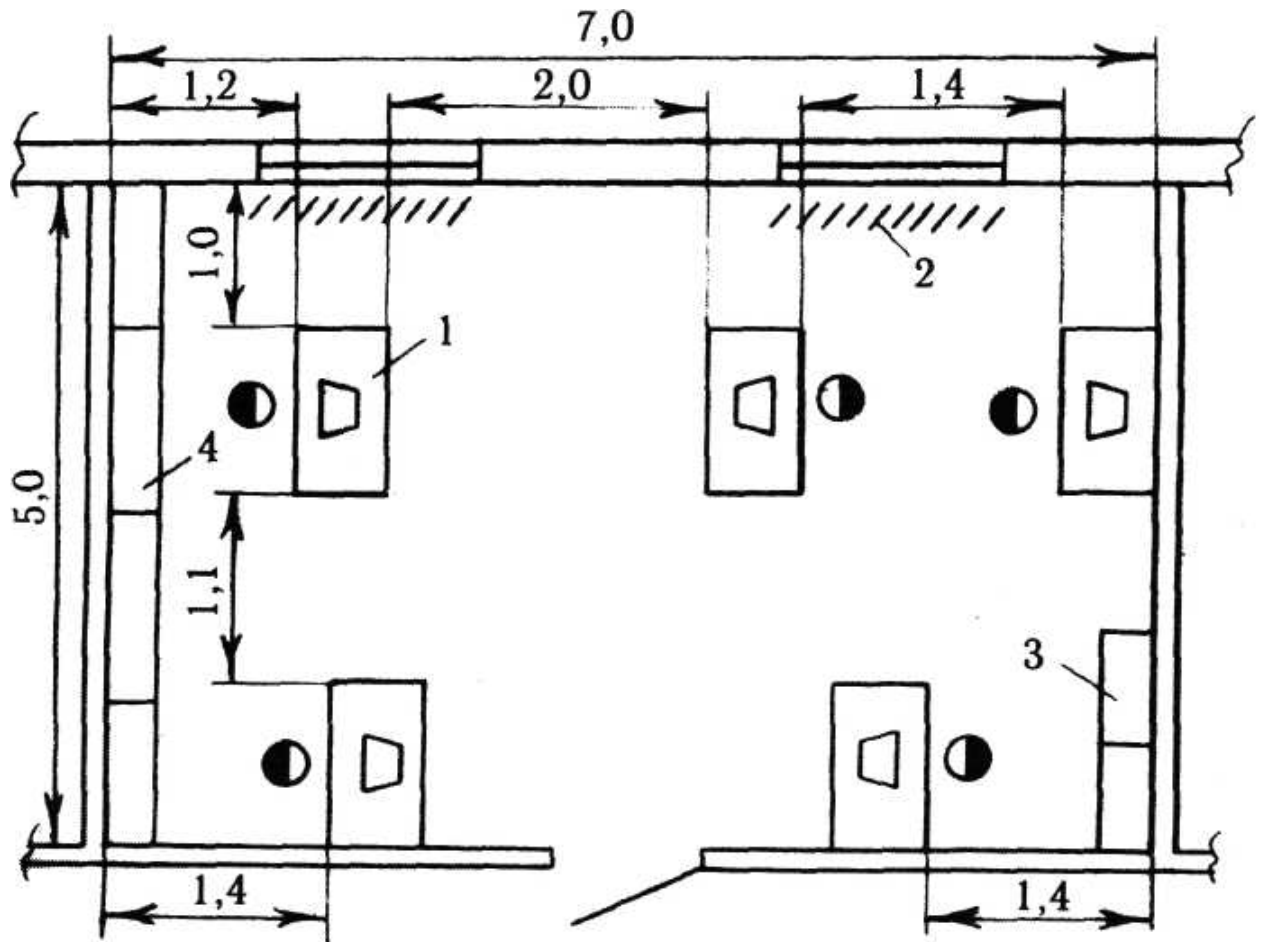


Рисунок 7.1 – План офісного приміщення з комп'ютеризованими робочими місцями, де: 1 - комп'ютеризоване робоче місце з ВДТ; 2 - сонцезахисні жалюзі; 3 - шафи для зберігання дискет та програмного забезпечення; 4 - шафи для зберігання документації та фахової літератури

7.4 Заходи безпеки у надзвичайних ситуаціях

7.4.1 Заходи з пожежної безпеки

Комплекс протипожежних заходів для приміщення офісу розроблено згідно вимог НАПБ А.01.001-2014 «Правила пожежної безпеки в Україні» [72].

Виходячи з аналізу речовин та матеріалів, які використовуються при роботі у приміщенні, відповідно до ДСТУ EN 2:2014 «Класифікація пожеж

(EN 2:1992, EN 2:1992/A1:2004, IDT)» [73] у приміщенні офісу обладнаному ПК з ВДТ можлива пожежа класу А (пожежа, що супроводжується горінням твердих матеріалів) та Е (горіння електроустановок, що перебувають під напругою до 1000 В).

Згідно до ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» [74], приміщення офісу належить до категорії «Д».

Так, як приміщення офісу обладнане ПК з ВДТ належить до категорії «Д» з пожежної небезпеки, то згідно вимог ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» [75] воно має II ступінь вогнестійкості.

У разі виникнення пожежі у приміщенні офісу, для евакуації персоналу, відповідно до вимог ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» [75] передбачено виходи, по обидві сторони приміщення, з одного боку вхідні двері, а з іншого – вікно (вихід на пожежні сходи). Згідно п. 2.29 (табл. 2) СНиП 2.09.02-85* «Производственные здания» [76], відстань від найбільш віддаленого робочого місця до найближчого евакуаційного виходу не обмежується.

Для запобігання пожеж у приміщенні офісу на силовому обладнанні, силових та освітлювальних колах встановлено захисні пристрої, що вимикають джерело живлення від ділянки електричного кола, у якій виникло коротке замикання.

Згідно вимог ДБН В.2.5-56:2014 «Системи протипожежного захисту» [77], у приміщенні офісу обладнаному ПК з ВДТ встановлена система пожежної й охоронної сигналізації «Сигнал-ВК6». Вона забезпечує виявлення теплових і димових ознак пожежі і місця виникнення пожежі з точністю до місця розміщення датчика.

Визначення типу та розрахунок кількості первинних засобів пожежегасіння проводиться згідно ДБН В.1.1.7-2016 «Захист від пожежі. Пожежна безпека об'єктів будівництва» [78]. Для офісу площею 35 м²,

передбачено два вуглекислотні вогнегасники типу ВВК-3,5 (з розрахунку один вогнегасник з величиною заряду вогнегасної речовини 3 кг і більше, на 20 м² площі приміщення). Додатково, на кожному поверсі будівлі, у якій розміщене приміщення обладнане ПК з ВДТ, передбачено два переносних порошкових вогнегасника – ВП-5. Відстань між вогнегасниками та місцями можливих загорянь не перевищує 10 м.

7.4.2 Заходи з цивільної безпеки

Розглянемо питання укриття населення у захисних спорудах цивільного захисту [79].

Захисні споруди цивільної оборони призначаються для захисту в мирний час населення, який переходить від наслідків аварій, катастроф та стихійного лиха, які загрожують масовому ураженню людей, а також у воєнний час - від сучасної зброї масового ураження. В мирний час захисні споруди використовуються для господарчих потреб.

За захисними властивостями споруди поділяються на найпростіші укриття (відкриті та перекриті щілини), сховища і протирадіаційні укриття.

Сховища відповідно діляться на класи, а протирадіаційні укриття на групи.

За призначенням захисні споруди будують для захисту населення та для розміщення органів управління (пунктів управління, вузлів зв'язку) і медичних установ.

За місцем розташування споруди бувають:

- вбудовані (за техніко-економічними, експлуатаційними показниками мають перевагу);
- ті що окремо розташовані (допускаються в тих випадках, коли за об'єктивними причинами будівництво вбудованих унеможлиблюється).

За термінами будівництва є такі, що побудовані завчасно, а також такі, що швидко зводяться (з/б плити з висотою ґрунту не менше 1,2 м зверху).

Захисні споруди за місткістю умовно поділяють на споруди малої місткості (до 150 чол.), середньої місткості (151 – 600 чол.) і великої місткості (600 чол. та більше).

Найпростіші укриття – це споруди, що зменшують радіуси ураження людей світловим випромінюванням, проникаючою радіацією, ударною хвилею та послабляють дію радіоактивних випромінювань. У системі захисту населення особливе значення мають найпростіші укриття типу щілин. Це найбільш масові захисні споруди, що можуть бути збудовані населенням у найкоротший терміни. Щілини будують відкритими і перекритими. Відкрита щілина зменшує ймовірність ураження ударною хвилею (у 1,2 – 2 рази), світловим випромінюванням і проникаючою радіацією. Перекрита щілина захищає: від світлового випромінювання – повністю, від ударної хвилі у 1,5 – 3 рази, від проникаючої радіації і радіоактивного випромінювання – у 200-300 разів, а також надійно захищає від осколкових і кулькових бомб, від запалювальних засобів.

Відкрита щілина - зигзагоподібна траншея з кількох прямолінійних ділянок довжиною до 15 м, глибина її – 1,8 – 2,0 м., шириною: зверху – 1,1 – 1,3 м., на дні – 0,8 м. Будівництво щілини починається з розмітки і трасування, тобто визначення її плану на місцевості. Копають спочатку на ширину дна.

В міру заглиблення поступово підрівнюють крутизну, доводячи до потрібних параметрів. Стінки (крутизну) щілини укріплюють дошками, жердинами, очеретом, іншими наявними матеріалами. Коли є час і в разі потреби, щілину перекривають колодками, шпалами або малогабаритними плитами. Зверху покриття влаштовують шар гідроізоляції з толю, руберойду, хлорвінілової плівки або утрамбовують шар глини і насипають шар ґрунту товщиною 50-60 см. У перекритій щілині роблять вхід з однієї або з двох

сторін з дверима і тамбуром. Для вентиляції встановлюють витяжну коробку. Нормальна місткість щілини - 10 - 15 чоловік.

Сховища є найбільш надійним захистом від усіх уражальних чинників: високих температур і шкідливих газів у зонах пожеж, вибухонебезпечних, радіоактивних і сильнодіючих отруйних речовин, обвалів і уламків зруйнованих будинків і споруд тощо, а також засобів масового ураження і звичайних засобів ураження. Воно обладнане комплексом інженерних споруд, що забезпечують необхідні умови життєдіяльності протягом певного часу.

За місцем знаходження сховища бувають збудованими (у підвалах будинків) і відокремленими (поза будинками). Їх споруджують заздалегідь, у мирний час, але можуть будувати і в період загрози нападу або під час воєнних дій (швидко зведені).

За місткістю розрізняють малі сховища (150-300 чол.), середні (300-600 чол.) і великі (понад 600 чол.).

Сховища мають фільтровентиляційні установки промислового виготовлення, фільтровентиляційні установки очищають зовнішнє повітря, розподіляють його по відсіках і створюють у захисному приміщенні надлишковий тиск, що перешкоджає проникненню зараженого повітря через тріщини і щілини. В усіх сховищах передбачаються два режими вентиляції: чистої, коли зовнішнє повітря очищається від пилу, фільтровентиляції, коли воно проходить крізь поглинальні фільтри, де очищається від радіоактивного пилу, отруйних речовин, бактеріологічних засобів.

Система водопостачання забезпечує людей водою для життя і гігієнічних потреб від зовнішньої водопровідної мережі. На той випадок, коли водопровід перестане діяти, передбачено аварійний запас води або її джерело. Кожна захисна споруда має систему каналізації для відводу фекальних стоків. Санвузол влаштовують у приміщенні, ізольованому перегородками від секцій сховища, обов'язково з витяжкою. Система опалення сховища працює від опалювальної мережі будинку, під яким

воно знаходиться. Освітлюється сховище від міської (об'єктної) електромережі, в аварійних випадках – від автономної електростанції, а якщо її немає – від акумуляторів або ліхтарями.

Запас продуктів харчування робиться з розрахунку не менше ніж на дві доби для кожної людини у сховищі. Медичне обслуговування здійснюють санітарні пости і медпункти об'єктів народного господарства.

За призначенням у сховищах передбачаються основні та допоміжні приміщення. До основних приміщень належать: приміщення для осіб, що укриваються, пункти управління, медичний пункт.

До допоміжних приміщень належать: тамбур – шлюзи (тамбури); фільтровентиляційні приміщення, санітарні вузли, електрощитові, приміщення для зберігання продовольства та ємностей запасу питної води, приміщення артезіанської свердловини, дренажні станції перекачування, балонні приміщення, приміщення для дозиметричного контролю, роздягальні.

Протирадіаційними укриттями називаються негерметичні захисні споруди, що забезпечують захист людей в умовах надзвичайних ситуацій. До протирадіаційних укриттів можна віднести не тільки спеціально побудовані споруди, а й будівлі господарського призначення (погреби, підпілля, овочесховища), пристосовані під укриття, і звичайні житлові будівлі.

Захисні властивості укриттів визначаються коефіцієнтом послаблення радіації, що залежить від товщини огорожувальних конструкцій, властивостей матеріалу, з якого виготовлені конструкції, а також від енергії гамма-випромінювання. Наприклад підвали дерев'яних будинків послаблюють радіацію в 7-12 разів, а кам'яних – у 200-300 разів.

У протирадіаційному укритті, розрахованому на 50 чоловік і більше, повинно бути не менше двох виходів розміром 80 x 180 см., причому бажано, щоб вони були розташовані в протилежних кінцях укриття під кутом 90° один до одного. Для підсилення захисних властивостей у приміщенні забивають вікна і зайві двері, насипають шар ґрунту на перекриття і роблять,

якщо треба ґрунтову підсипку ззовні біля стін, що виступають вище поверхні землі. Для герметизації приміщень ретельно замурують тріщини, щілини, отвори у стінах і стелі, біля вікон і дверей, припасовують двері, оббивають їх повстю, ущільнюють дверні рами валиком з повсті або з іншої м'якої тканини. Укриття, що вміщує до 30 чоловік, провітрюється природною вентиляцією через припливний і витяжний короби. Для створення тяги витяжний короб встановлюють на 1,5-2 м. вище припливного. На зовнішньому вигляді вентиляційного короба роблять дашок, а в припливному коробі – щільно підігнані засуви.

У пристосованих під укриття приміщеннях встановлюють бачки з водою з розрахунку 3-4 л. на одну людину на добу, а в туалеті виносну тару або влаштовують люфт-клозет з вигрібною ямою. Крім того, в укритті встановлюють нари (лавки) для відпочинку, стелажі для продуктів харчування. Освітлення – від електромережі або переносними електричними ліхтарями.

ВИСНОВКИ

У проведеному дослідженні методів класифікації процесів за даними багатовимірних сигналів було досліджено методи виділення ділянок активності з одновимірного сигналу, розроблено метод знаходження ділянок найбільшої активності багатовимірного сигналу, для подальшого їх використання для навчання та тестування НМ класифікатора, розроблено метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних.

Було проведено експерименти на основі реальних даних. Вирішувалась практична задача класифікації пацієнтів за даними багатовимірних ЕМГ сигналів з груп їх м'язів. За допомогою розробленої програмної бібліотеки була підтверджена ефективність розроблених методів виділення періодів активності з багатовимірного ЕМГ сигналу та знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу. Також була підтверджена ефективність застосування глибокої згорткової НМ для класифікації таких багатовимірних сигналів.

Програмна бібліотека для попередньої обробки даних та проведення експериментів з НМ моделями була створена на мові програмування Python у середовищі розробки PyCharm з використанням допоміжних бібліотек, таких як Pandas, NumPy, TensorFlow та Matplotlib.

Розроблені та застосовані у дослідженні методи попередньої обробки даних та побудови НМ моделей можуть будувати застосовані для вирішення задач у яких необхідно класифікувати стан об'єкту за його багатовимірними сигналами діагностування.

Результати роботи опубліковано в 5 друкованих працях, в тому числі 1 стаття у науковому фаховому виданні України, що індексується наукометричною базою Web of Science та 4 тези доповідей на наукових конференціях.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Ukil, A. Intelligent systems and signal processing in power engineering / A. Ukil. – Berlin: Springer, 2007. – 372 p.
2. Kay, S. M. Fundamentals of statistical signal processing / S. M. Kay. – New Jersey: Prentice-Hall, 1998. – 560 p.
3. Hamilton, J. Fundamentals of statistical signal processing / J. Hamilton. – New Jersey: Princenton University Press, 1994. – 799 p.
4. Tverdohleb, J. Wavelet analysis of complex nonstationary oscillatory signals / [J. Tverdohleb, I. Limarev, V. Logominov et al.] // 4-th IEEE International Scientific-Practical Conference “Problems of Infocommunications. Science and Technology”. – 2017. – P. 17-20.
5. Лимарев І.О. Довготермінове прогнозування стану трансмісії вертольоту на основі рекурентних нейронних мереж / І.О. Лимарев // Радіoeлектроніка і молодь в ХХІ ст.: матеріали ХХІІ Міжнародного молодіжного форуму. – 2018. – Т. 5. – С. 223-224.
6. Lymarev I. O. Methods of large-scale signals transformation for diagnosis in neural network models / [I. O. Lymarev, S. A. Subbotin, A. A. Oliinyk et al.] // Radio Electronics, Computer Science, Control. – 2018. – № 4. – P. 63–71.
7. Limarev I. Diagnostic Signal Nonstationarity Reduction to Predict the Helicopter Transmission State on the Basis of Intelligent Information Technologies / [I. Lymarev, S. Subbotin, A. Oliinyk et al.] // Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019). – 2019. – P. 510–522.
8. Лимарев, И.О. Методы преобразования сигналов большой размерности в диагностических моделях управляемых данными / [И.О. Лимарев, С.А. Субботин, А.А. Олейник и др.] // Нейроинформатика, ее приложения и анализ данных: Материалы XXVI Всероссийского семинара. – 2018. – С. 90–95.

9. Gogolou, A. The training set quality measures for neural network learning / A. Gogolou // IEEE transactions on visualization and computer graphics. – 2018. – Vol. 25, № 1. – P. 523–533.
10. Deep learning for sensor-based activity recognition: A survey / [J. Wang, Y. Chen, S. Hao et al.] // Pattern Recognition Letters. – 2019. – Vol. 119. – P. 3–11.
11. Florian M. Knowledge-driven Biometric Authentication in Virtual Reality / M. Florian, H. Fawaz, M. Khamis // Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems. – 2020. – P. 1–10.
12. Nwe, T. L. Convolutional neural network with multi-task learning scheme for acoustic scene classification / T. L. Nwe, T. H. Dat, B. Ma // Asia-Pacific Signal and Information Processing Association Annual Summit and Conference. – 2017. – P. 1347–1350.
13. Susto, A. Time-Series Classification Methods: Review and Applications to Power Systems Data / A. Susto, A. Cenedese, T. Matteo // Big Data Application in Power Systems. – 2018. – P. 179–220.
14. Dau, H. The UCR time series archive / [H. Dau, A. Bagnall, K. Kamgar et al.] // IEEE/CAA Journal of Automatica Sinica. – 2019. – P. 1293–1305.
15. Bagnall, A. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances / [H. Dau, A. Bagnall, K. Kamgar et al.] // Data Mining and Knowledge Discovery. – 2019. – Vol. 33, № 3. – P. 606–660.
16. LeCun, Y. Gradient-based learning applied to document recognition / [Y. LeCun, L. Bottou, Y. Bengio et al.] // Proceedings of the IEEE. – 1998. – Vol. 88, № 11. – P. 2278–2324.
17. Lines, J. Time series classification with ensembles of elastic distance measures / J. LeCun, A. Bagnall // Data Mining and Knowledge Discovery. – 1998. – Vol. 29, № 3. – P. 565–592.
18. Marteau, P. Time Warp Edit Distance with Stiffness Adjustment for

Time Series Matching / P. Marteau // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2009. – Vol. 31, № 2. – P. 306–318.

19. Stefan, A. The Move-Split-Merge Metric for Time Series / A. Stefan, V. Athitsos, G. Das // IEEE Transactions on Knowledge and Data Engineering. – 2013. – Vol. 25, № 6. – P. 1425–1438.

20. Bagnall, A. Time-series classification with COTE: The collective of transformation-based ensembles / [A. Bagnall, J. Lines, J. Hillset al.] // International Conference on Data Engineering. – 2016. – Vol. 27, № 9. – P. 1548–1549.

21. Hills, J. Classification of time series by shapelet transformation / [J. Hills, J. Lines, E. Baranauskas et al.] // Classification of time series by shapelet transformation. – 2014. – Vol. 28, № 4. – P. 851–881.

22. Bostrom, A. Binary Shapelet Transform for Multiclass Time Series Classification / A. Bostrom, A. Bagnall // Big Data Analytics and Knowledge Discovery. – 2015. – P. 257–269.

23. Lines, J. HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification / J. Lines, S. Taylor, A. Bagnall // IEEE International Conference on Data Mining. – 2016. – P. 1041–1046.

24. Schäfer, P. The BOSS is concerned with time series classification in the presence of noise / P. Schäfer // Data Mining and Knowledge Discovery. – 2015. – Vol. 29, № 6. – P. 1505–1530.

25. Kate, J. Using dynamic time warping distances as features for improved time series classification / J. Kate // Data Mining and Knowledge Discovery. – 2016. – Vol. 30, № 2. – P. 283–312.

26. Houtao, D. A time series forest for classification and feature extraction / [D. Houtao, G. Rungerb, E. Tuv et al.] // Information Sciences. – 2013. – Vol. 239. – P. 142–153.

27. Baydogan, M. A Bag-of-Features Framework to Classify Time Series / [M. Baydogan, G. Rungerb, E. Tuv et al.] // IEEE Transactions on Pattern

Analysis and Machine Intelligence. – 2013. – Vol. 35., № 11. – P. 2796–2802.

28. Lines, J. Time series classification with ensembles of elastic distance measures / J. Lines, A. Bagnall // Data Mining and Knowledge Discovery. – 2015. – Vol. 29., № 3. – P. 565–592.

29. Nweke, H. Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges / [H. Nweke, W. Ying, M. Ali et al.] // Expert Systems with Applications. – 2018. – Vol. 105. – P. 233–261.

30. Krizhevsky, A. ImageNet Classification with Deep Convolutional Neural Networks / A. Krizhevsky, I. Sutskever, G. Hinton // Advances in Neural Information Processing Systems. – 2012. – Vol. 25. – P. 1097–1105.

31. LeCun, Y. Deep learning / Y. LeCun, Y. Bengio, G. Hinton // Nature. – 2015. – Vol. 521. – P. 436–444.

32. Ioffe, S. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift / S. Ioffe, C. Szegedy // International Conference on Machine Learning. – 2015. – Vol. 37. – P. 448–456.

33. Instance Normalization: The Missing Ingredient for Fast Stylization [Electronic resource]. – Access mode: <https://arxiv.org/abs/1711.10925>.

34. Geng, Y. Cost-Sensitive Convolution based Neural Networks for Imbalanced Time-Series Classification / Y. Geng, X. Luo // Intelligent Data Analysis. – 2015. – Vol. 37., № 2. – P. 357–370.

35. Wang, Z. Time series classification from scratch with deep neural networks: A strong baseline / Z. Wang, W. Yan, T. Oates // International Joint Conference on Neural Networks. – 2017. – P. 1578–1585.

36. Zhao, B. Convolutional neural networks for time series classification / [B. Zhao, H. Lu, S. Chen et al.] // Systems Engineering and Electronics. – 2017. – Vol. 28., № 1 – P. 162–169.

37. Strodthoff, N. Detecting and interpreting myocardial infarction using fully convolutional neural networks / N. Strodthoff, C. Strodthoff // Physiological Measurement. – 2019. – Vol. 40., № 1 – P. 162–170.

38. Lin, S. GCRNN: Group-Constrained Convolutional Recurrent Neural Network / S. Lin, G. Runger // IEEE Transactions on Neural Networks and Learning Systems. – 2018. – Vol. 99. – P. 1–10.
39. Serrà, J. Towards a universal neural network encoder for time series / J. Serrà, S. Pascual, A. Karatzoglou // Artificial Intelligence Research and Development: Current Challenges, New Trends and Applications. – 2018. – Vol. 308 – P. 120.
40. Bekey, G. A. Pattern recognition of multiple EMG signals applied to the description of human gait / [G. A. Bekey, C. W. Chang, J. Perry et al.] // Proceedings of the IEEE. – 1985. – Vol. 65. – P. 674–681.
41. Shiavi, R. G. Representing and clustering electromyographic gait patterns with multivariate techniques / R. G. Shiavi, P. Griffin // Medical and Biological Engineering and Computing. – 1981. – Vol. 19. – P. 605–611.
42. Hodges, P. W. A comparison of computer-based methods for the determination of onset of muscle contraction using electromyography / P. W. Hodges, B. H. Bui // Electroencephalography and Clinical Neurophysiology, Electromyography and Motor Control. – 1996. – Vol. 101. – P. 511-519.
43. Bonato, P. A statistical method for the measurement of muscle activation intervals from surface myoelectric signal during gait / P. Bonato, T. D'Alessio, M. Knaflitz // IEEE Transactions on Biomedical Engineering. – 1998. – Vol. 45. – P. 287–299.
44. Lidiérth, M. The A computer based method for automated measurement of the periods of muscular activity from an EMG and its application to locomotor EMGs / M. Lidiérth // Electroencephalography and Clinical Neurophysiology. – 1986. – Vol. 64. – P. 378–380.
45. Abbink, J. H. Detection of onset and termination of muscle activity in surface electromyograms/ J. H. Abbink, A. van der Bilt, H. W. van der Glas // Journal of Oral Rehabilitation. – 1998. – Vol. 25. – P. 365–369.
46. Solnik, S. Teager-Kaiser energy operator signal conditioning improves EMG onset detection/ [S. Solnik, P. Rider, K. Steinweg et al.] //

European Journal of Applied Physiology. – 2010. – Vol. 101. – P. 489-498.

47. Silva, H. Towards improving the usability of electromyographic interfaces/ [H. Silva, R. Scherer, J. Sousa et al.] // Journal of Oral Rehabilitation. – 2012. – P. 1-2.

48. Londral, A. A wireless user-computer interface to explore various sources of biosignals and visual biofeedback for severe motor impairment / [A. Londral, H. Silva, N. Nunes et al.] // Journal of Accessibility and Design for All. – 2013. – Vol. 3., № 2 – P. 118–134.

49. PyCharm [Electronic resource]. – Access mode: <https://www.jetbrains.com/pycharm/>.

50. Pandas [Electronic resource]. – Access mode: <https://www.pandas.pydata.org/>.

51. NumPy [Electronic resource]. – Access mode: <https://www.numpy.org/>.

52. TensorFlow [Electronic resource]. – Access mode: <https://www.tensorflow.org/>.

53. Matplotlib [Electronic resource]. – Access mode: <https://www.matplotlib.org/>.

54. pip [Electronic resource]. – Access mode: <https://www.pip.pypa.io/en/stable/>.

55. Остапенко, В.В. Методичні вказівки по економічному обґрунтуванню дипломних проєктів для студентів спеціальностей 7.080403 «Програмне забезпечення автоматизованих систем», 7.080402 «Інформаційні технології проєктування» і 7.091501 «Комп'ютерні та інтелектуальні системи і мережі» всіх форм навчання [Текст] / В.В. Остапенко, Е.М. Касьян, Є.М. Сердюк. – Запоріжжя : ЗНТУ, 2009. – 22 с.

56. Система стандартов безопасности труда. Изделия электротехнические. Общие требования безопасности : ГОСТ 12.2.007.0-75*. – [Введен с 1978-01-01]. – М. : Стандартинформ, 2008. – 12 с.

57. Правила улаштування електроустановок - 2017. – [Чинні від

2017-01-01]. – К. : Міненерговугілля України, 2017. – 617 с.

58. Система стандартів безпеки праці. Електробезпека. Термины и определения : ГОСТ 12.1.009-76. – [Введен с 1977-01-01]. – М. : ИПК Издательство стандартов, 1999. – 4 с.

59. Нормативно-правовий акт з охорони праці. Правила устрою електроустановок. Електрооборудование специальных установок: НПАОП 40.1-1.32-01. – [Чинний від 2002-01-01]. – К.: Міністерства праці та соціальної політики, 2001. – 78 с.

60. Система стандартів безпеки праці. Електробезпека. Защитное заземление, зануление : ГОСТ 12.1.030-81. – [Введен с 1982-07-01]. – М. : Стандартиформ, 2008. – 10 с.

61. Нормативно-правовий акт з охорони праці. Правила безпечної експлуатації електроустановок : НПАОП 40.1-1.01-97. – [Чинний від 1998-01-01]. – К.: Держнагляд охорони праці, 1998. – 78 с.

62. Нормативно-правовий акт з охорони праці. Правила безпечної експлуатації електроустановок споживачів : НПАОП 40.1-1.21-98. – [Чинний від 1999-01-01]. – К.: Держнагляд охорони праці, 1998. – 56 с.

63. Державні будівельні норми України. Природне і штучне освітлення : ДБН В2.5-28:2018. – [Чинний від 2019-03-01]. – К.: Мінрегіон України, 2018. – 137 с.

64. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин: ДСанПіН 3.3.2.007-98. – [Чинні від 1999-01-01]. – К.: Міністерство охорони здоров'я, 1999. – 4 с.

65. Державні санітарні норми. Санітарні норми виробничого шуму, ультразвуку та інфразвуку : ДСН 3.3.6.037-99. – [Чинні від 1999-01-01]. – К.: Міністерство охорони здоров'я, 1999. – 34 с.

66. Державні санітарні норми. Санітарні норми мікроклімату виробничих приміщень : ДСН 3.3.6-042-99. – [Чинні від 1999-01-01]. – К.: Міністерство охорони здоров'я, 1999. – 27 с.

67. Система стандартов безопасности труда. Общие санитарно-гигиенические требования к воздуху рабочей зоны : ГОСТ 12.1.005-88. – [Введен с 1989-01-01]. – М. : Стандартинформ, 2008. – 95 с.

68. Гігієнічні норми. Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень : ГН2152-80. – [Чинні від 1999-09-01]. – К. : Міністерство праці та соціальної політики України, 1999. – 7 с.

69. Ергономічні вимоги до роботи з відеотерміналами в офісі. Частина 1. Загальні положення : ДСТУ ISO 9241-1:2003. – [Чинний від 2003-01-01]. – К.: Держспоживстандарт України, 2011. – 22 с.

70. Нормативно-правовий акт з охорони праці. Правила охорони праці під час експлуатації електронно-обчислювальних машин : НПАОП 0.00-1.28-10. – [Чинний від 2010-05-05]. – К.: Держнаглядохоронпраці, 2010. – 38 с.

71. Нормативно-правовий акт з питань пожежної безпеки. Нормы определения категорий помещений, зданий и наружных установок по взрывопожарной и пожарной безопасности : НАПБ Б.03.002-2007. – [Чинний від 2008-01-01]. – К.: Держнаглядохоронпраці, 2008. – 93 с.

72. Нормативно-правовий акт з питань пожежної безпеки. Правила пожежної безпеки в Україні : НАПБ А.01.001-2014. – [Чинний від 2015-06-01]. – К.: Міністерство внутрішніх справ України, 2015. – 86 с.

73. Класифікація пожеж (EN 2:1992, EN 2:1992/A1:2004, IDT) : ДСТУ EN 2:2014. – [Чинний від 2016-01-01]. – К.: Держспоживстандарт України, 2015. – 39 с.

74. Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою : ДСТУ Б В.1.1-36:2016. – [Чинний від 2017-01-01]. – К.: Мінрегіон України, 2016. – 34 с.

75. Державні будівельні норми України. Пожежна безпека об'єктів будівництва. Загальні вимоги» : ДБН В.1.1-7:2016. – [Чинний від 2017-01-01]. – К.: Мінрегіон України, 2017. – 47 с.

76. Строительные нормы и правила. Производственные здания : СНиП 2.09.02-85*. – [Введен с 1987-01-01]. – М.: ИПК Издательство стандартов, 1986. – 46 с.

77. Державні будівельні норми України. Системи протипожежного захисту : ДБН В.2.5-56:2014. – [Чинний від 2015-07-01]. – К.: Мінрегіон України, 2014. – 19 с.

78. Державні будівельні норми України. Захист від пожежі. Пожежна безпека об'єктів будівництва : ДБН В.1.1.7-2016. – [Чинний від 2017-01-01]. – К.: Мінрегіон України, 2014. – 39 с.

79. Укриття населення в захисних спорудах цивільного захисту [Електронний ресурс]. – 2016. – Режим доступу: http://www.zprozrda.gov.ua/documents/civilniy-zahist/pamyatka/ukrittya_naselennya_v_zahisnih_sporudah.pdf/.

ДОДАТОК А
Текст програми

Б.1 Текст файла main.py

```
import time
import os
import numpy as np
import sklearn
import tensorflow as tf
import tensorflow.keras as keras
from sklearn.model_selection import train_test_split

def fit_classifier():

    X = np.load ('X_norm.npy')
    Y = np.load ('Y.npy')
    x_train, x_test, y_train, y_test = train_test_split (X, Y, test_size = 0.33, shuffle =
True, random_state = 1)
    y_true_samples = y_test
    number_classes = 1
    input_shape = x_train.shape[1:]
    nn_classifier = Classifier_MDCNN(out_dir, input_shape, number_classes)
    nn_classifier.fit(x_train, y_train, x_test, y_test, y_true_samples)

def create_dir(dir_path):

    if os.path.exists(dir_path):
        return None
    else:
        try:
            os.makedirs(dir_path)
        except:
```

```

    return None
    return dir_path

```

```

class Classifier_MDCNN:

```

```

    def __init__(self, out_directory, input_shape, number_classes, verbose=True,
build=True):

```

```

        self.out_directory = out_directory

```

```

        if build:

```

```

            self.model = self.build_model(input_shape, number_classes)

```

```

            if verbose:

```

```

                self.model.summary()

```

```

            self.verbose = verbose

```

```

            self.model.save_weights(self.out_directory + 'nn_model_init.hdf5')

```

```

    def build_model(self, input_shape, number_classes):

```

```

        inp_layer = keras.layers.Input(input_shape)

```

```

        conv1_layer = keras.layers.Conv1D(filters = 64, kernel_size = 8, padding =
'same')(inp_layer)

```

```

        conv1_layer = keras.layers.BatchNormalization()(conv1_layer)

```

```

        conv1_layer = keras.layers.Activation(activation='relu')(conv1_layer)

```

```

        conv2_layer = keras.layers.Conv1D(filters = 128, kernel_size = 5, padding =
'same')(conv1_layer)

```

```

        conv2_layer = keras.layers.BatchNormalization () (conv2_layer)

```

```

        conv2_layer = keras.layers.Activation ('relu') (conv2_layer)

```

```

        conv3_layer = keras.layers.Conv1D (filters = 256, kernel_size = 3, padding =
'same') (conv2_layer)

```

```

        conv3_layer = keras.layers.BatchNormalization () (conv3_layer)

```

```

conv3_layer = keras.layers.Activation ('relu') (conv3_layer)
ga_pulling_layer = keras.layers.GlobalAveragePooling1D() (conv3_layer)
out_layer = keras.layers.Dense (number_classes, activation = 'sigmoid')
(ga_pulling_layer)

nn_model = keras.models.Model (inputs = inp_layer, outputs = out_layer)
nn_model.compile (loss = 'binary_crossentropy',
optimizer=keras.optimizers.Adam(), metrics=['accuracy'])

reduce_learning_rate = keras.callbacks.ReduceLROnPlateau (monitor = 'loss',
factor = 0.5, patience = 50, min_lr = 0.0001)

file = self.out_directory + 'best_model.hdf5'

model_chp = keras.callbacks.ModelCheckpoint (filepath = file, monitor =
'loss', save_best_only = True)

self.callbacks = [reduce_learning_rate, model_chp]

return nn_model

def fit (self, x_train, y_train, x_val, y_val, y_true_samples):

    batch_s = 8
    n_epoch = 200
    start = time.time ()

    history = self.model.fit (x_train, y_train, batch_size = batch_s, epochs =
n_epoch, verbose = self.verbose, validation_data = (x_val, y_val), callbacks =
self.callbacks)

    dur = time.time() - start

    self.model.save (self.out_directory + 'last_nn_model.hdf5')

    model = keras.models.load_model (self.out_directory + 'best_nn_model.hdf5')

    y_prediction = model.predict (x_val)

    y_prediction = np.argmax (y_prediction, axis=1)

    save_log (self.out_directory, history, y_prediction, y_true_samles, dur)

```

```
def predict (self, x_test, y_true, x_train, y_train, y_test, df_metrics=True):
```

```
    model_path = self.out_directory + 'best_model.hdf5'
```

```
    model = keras.models.load_model (model_path)
```

```
    y_predict = model.predict (x_test)
```

```
    if df_metrics:
```

```
        y_predict = np.argmax (y_predict, axis=1)
```

```
        df_metrics = calc_metrics (y_true, y_predict, 0.0)
```

```
        return df_metrics
```

```
    else:
```

```
        return y_predict
```

```
def create_classifier (input_shape, number_classes, out_dir, verbose=False):
```

```
    return Classifier_MDCNN (out_dir, input_shape, number_classes, verbose)
```

```
root_dir = '/home/nesc/Study/md-tsc'
```

```
classifier_name = 'mcnn'
```

```
tmp_out_dir = root_dir + '/results/' + classifier_name + '/'
```

```
dataset_name = 'EMG'
```

```
out_dir = tmp_out_dir + dataset_name + '/'
```

```
create_dir(out_dir)
```

```
fit_classifier()
```

```
create_dir(out_dir)
```

Б.2 Текст файлы hedges_bui.py

```
def hedges_bui(sig=None, rest_sig=None, w_size=None, threshold_val=None):
```

```
    if isinstance(rest_sig, np.ndarray) or isinstance(rest_sig, list):
```

```
        if len(rest_sig) >= 2:
```

```

rest_sig = np.array(rest_sig)
if len(rest_sig) == 2:
    rest_signal = sig[rest_sig[0]:rest_sig[1]]
else:
    rest_signal = rest_sig
rest_sig_zero_mean = rest_signal - np.mean(rest_signal)
statistics = st.signal_stats(signal = rest_sig_zero_mean)
mean_rest = statistics['mean']
std_dev_rest = statistics['std_dev']
elif isinstance(rest_sig, dict):
    mean_rest = rest_sig['mean']
    std_dev_rest = rest_sig['std_dev']
sig_zero_mean = sig - np.mean(sig)
abs_sig = np.abs(sig_zero_mean)
mvgav_val = np.convolve(abs_sig, np.ones((w_size,)) / w_size, mode='valid')
t_sig = (1 / std_dev_rest) * (mvgav_val - mean_rest)
length = len(sig)
start = np.nonzero(t_sig >= threshold_val)[0]
stop = np.nonzero(t_sig < threshold_val)[0]
onsets = np.union1d(np.intersect1d(start - 1, stop),
                    np.intersect1d(start + 1, stop))
onsets += int(w_size / 2)
if np.any(onsets):
    if onsets[-1] >= length:
        onsets[-1] = length - 1

return (onsets, t_sig)

```

Б.3 Текст файла bonato.py

```

def bonato (sig=None, rest_sig=None, threshold_val=None,
active_state_dur=None, samples_fail=None, f_size=None):
    if isinstance(rest_sig, np.ndarray) or isinstance(rest_sig, list):
        if len(rest_sig) >= 2:
            rest_sig = np.array(rest_sig)
            if len(rest_sig) == 2:
                rest_signal = sig[rest_sig[0]:rest_sig[1]]
            else:
                rest_signal = rest_sig
            rest_zero_mean = rest_signal - np.mean(rest_signal)
            statistics = st.signal_stats(signal=rest_zero_mean)
            var_rest = statistics['var']
        elif isinstance(rest_sig, dict):
            var_rest = rest_sig['var']
    sig_zero_mean = sig - np.mean(sig)
    tf_l = []
    onset_time_l = []
    offset_time_l = []
    alarm_time = 0
    state_duration = 0
    j = 0
    n = 0
    onset_val = False
    alarm_val = False
    for k in range(1, len(sig_zero_mean), 2):
        tf = (1 / var_rest) * (sig_zero_mean[k - 1] ** 2 + sig_zero_mean[k] ** 2)
        tf_l.append(tf)
        if onset_val is True:

```

```

if alarm_val is False:
    if tf < threshold_val:
        alarm_time = k // 2
        alarm_val = True
    else:
        if tf < threshold_val:
            state_duration += 1
            if j > 0
                n += 1
            if n == samples_fail:
                n = 0
                j = 0
            if state_duration == active_state_dur:
                offset_time_l.append(alarm_time)
                onset_val = False
                alarm_val = False
                n = 0
                j = 0
                state_duration = 0
            else
                j += 1
            if j > f_size:
                alarm_val = False
                n = 0
                j = 0
                state_duration = 0
        else:
            if alarm_val is False:
                if tf >= threshold_val:
                    alarm_time = k // 2

```

```

        alarm_val = True
    else
        if tf >= threshold_val:
            state_duration += 1
            if j > 0:
                n += 1
                if n == samples_fail:
                    n = 0
                    j = 0
            if state_duration == active_state_dur:
                onset_time_l.append(alarm_time)
                onset_val = True
                alarm_val = False
                n = 0
                j = 0
                state_duration = 0
            else:
                j += 1
        if j > f_size:
            alarm_val = False
            n = 0
            j = 0
            state_duration = 0

    onsets = np.union1d(onset_time_list, offset_time_list)
    onsets *= 2

    return (onsets, tf_list)

```

Б.4 Текст файла `lidiert.py`

```
def lidiert(sig=None, rest_sig=None, w_size=None, threshold_value=None,
active_state_dur=None, f_size=None):
    if isinstance(rest_sig, np.ndarray) or isinstance(rest_sig, list):
        if len(rest_sig) >= 2:
            rest_sig = np.array(rest_sig)
            if len(rest_sig) == 2:
                rest_signal = sig[rest_sig[0]:rest_sig[1]]
            else:
                rest_signal = rest_sig
            rest_zero_mean = rest_signal - np.mean(rest_signal)
            statistics = st.signal_stats(signal=rest_zero_mean)
            mean_rest = statistics['mean']
            std_dev_rest = statistics['std_dev']
        elif isinstance(rest_sig, dict):
            mean_rest = rest_sig['mean']
            std_dev_rest = rest_sig['std_dev']
    sig_zero_mean = sig - np.mean(sig)
    f_sig = np.abs(sig_zero_mean)
    mvgav_sig = np.convolve(f_sig, np.ones((w_size,)) / w_size, mode='valid')
    tf_sig = (1 / std_dev_rest) * (mvgav_sig - mean_rest)
    onset_time_list = []
    offset_time_list = []
    alarm_t = 0
    state_dur = 0
    j = 0
    onset_val = False
    alarm_val = False
    for k in range(0, len(tf_sig)):
```

```

if onset_val is True:
    if alarm_val is False:
        if tf_sig[k] < threshold_value:
            alarm_t = k
            alarm_val = True
    else:
        if tf_sig[k] < threshold_value:
            state_dur += 1
            if j > 0:
                j = 0
            if state_dur == active_state_dur:
                offset_time_list.append(alarm_t)
                onset_val = False
                alarm_val = False
                j = 0
                state_dur = 0
        else:
            j += 1
            if j > f_size:
                alarm_val = False
                j = 0
                state_dur = 0
    else:
        if alarm_val is False:
            if tf_sig[k] >= threshold_value:
                alarm_t = k
                alarm_val = True
        else:
            if tf_sig[k] >= threshold_value:
                state_dur += 1

```

```

        if j > 0:
            j = 0
        if state_dur == active_state_dur:
            onset_time_list.append(alarm_t)
            onset_val = True
            alarm_val = False
            j = 0
            state_dur = 0
    else:
        j += 1
        if j > f_size:
            alarm_val = False
            j = 0
            state_dur = 0
onsets = np.union1d(onset_time_list,
                    offset_time_list)
onsets += int(w_size / 2)

return (onsets, tf_sig)

```

Б.5 Текст файла **abbink.py**

```

def abbink(sig=None, rest_sig=None, w_size=None, alarm_s=None,
threshold_val=None, trans_threshold=None):

```

```

    if isinstance(rest_sig, np.ndarray) or isinstance(rest_sig, list):
        if len(rest_sig) >= 2:
            rest_sig = np.array(rest_sig)
            if len(rest_sig) == 2:
                rest_signal = sig[rest_sig[0]:rest_sig[1]]

```

```

else:
    rest_signal = rest_sig
    rest_zero_mean = rest_signal - np.mean(rest_signal)
    statistics = st.signal_stats(signal=rest_zero_mean)
    mean_rest = statistics['mean']
    std_dev_rest = statistics['std_dev']
elif isinstance(rest_sig, dict):
    mean_rest = rest_sig['mean']
    std_dev_rest = rest_sig['std_dev']
signal_zero_mean = sig - np.mean(sig)
fwlo = np.abs(signal_zero_mean)
mvgav = np.convolve(fwlo, np.ones((w_size,)) / w_size, mode='valid')
tf = (1 / std_dev_rest) * (mvgav - mean_rest)
t_sig = t_sig.tolist()
onset_time_l = []
offset_time_l = []
alarm_t = 0
onset_s = False
alarm_s = False
for k in range(0, len(tf)):
    if onset_s is True:
        if alarm_s is False:
            if t_sig[k] < threshold_val:
                alarm_t = k
                alarm_s = True
    else:
        if alarm_t > alarm_s and k == (alarm_t + alarm_s + 1):
            transition_ind = []
            for j in range(alarm_s, alarm_t):

```

```

        low_l = [t_sig[j - alarm_s + a] for a in range(1, alarm_s + 1)]
        low = sum(i < trans_threshold for i in low_l)
        high_l = [t_sig[j + b] for b in range(1, alarm_s + 1)]
        high = sum(i > trans_threshold for i in high_l)
        transition_ind.append(low + high)
        offset_time_l = np.where(transition_ind ==
np.amin(transition_ind))[0].tolist()

        onset_s = False
        alarm_s = False
    else:
        if alarm_s is False:
            if t_sig[k] >= threshold_val:
                alarm_t = k
                alarm_s = True
        else:
            if alarm_t > alarm_s and k == (alarm_t + alarm_s + 1):
                transition_ind = []
                for j in range(alarm_s, alarm_t):

                    low_l = [t_sig[j - alarm_s + a] for a in range(1, alarm_s + 1)]
                    low = sum(i < trans_threshold for i in low_l)
                    high_l = [t_sig[j + b] for b in range(1, alarm_s + 1)]
                    high = sum(i > trans_threshold for i in high_l)
                    transition_ind.append(low + high)
                onset_time_l = np.where(transition_ind ==
np.amax(transition_ind))[0].tolist()

                onset_s = True
                alarm_s = False
    onsets = np.union1d(onset_time_l,

```

```

        offset_time_l)
    onsets += int(w_size / 2)

    return (onsets, t_sig)

```

Б.6 Текст файла solnik.py

```

def solnik(sig=None, rest_sig=None, threshold_val=None,
active_state_dur=None):

    if isinstance(rest_sig, np.ndarray) or isinstance(rest_sig, list):
        if len(rest_sig) >= 2:
            rest_sig = np.array(rest_sig)
            if len(rest_sig) == 2:
                rest_signal = sig[rest_sig[0]:rest_sig[1]]
            else:
                rest_signal = rest_sig
            rest_zero_mean = rest_signal - np.mean(rest_signal)
            statistics = st.signal_stats(signal=rest_zero_mean)
            mean_rest = statistics['mean']
            std_dev_rest = statistics['std_dev']
        elif isinstance(rest_sig, dict):
            mean_rest = rest_sig['mean']
            std_dev_rest = rest_sig['std_dev']
        sig_zero_mean = sig - np.mean(sig)
        threshold_val = mean_rest + threshold_val * std_dev_rest
        tf_l = []
        onset_time_l = []
        offset_time_l = []
        alarm_t = 0

```

```

state_dur = 0
onset_val = False
alarm_val = False
for k in range(1, len(sig_zero_mean) - 1):
    tf_sig = sig_zero_mean[k] ** 2 - sig_zero_mean[k + 1] * sig_zero_mean[k -
1]
    tf_sig = np.abs(tf_sig)
    tf_l.append(tf_sig)
    if onset_val is True:
        if alarm_val is False
            if tf_sig < threshold_val:
                alarm_t = k
                alarm_val = True
    else:
        if tf_sig < threshold_val:
            state_dur += 1
            if state_dur == active_state_dur:
                offset_time_l.append(alarm_t)
                onset_val = False
                alarm_val = False
                state_dur = 0
    else:
        if alarm_val is False:
            if tf_sig >= threshold_val
                alarm_t = k
                alarm_val = True
    else
        if tf_sig >= threshold_val:
            state_dur += 1
            if state_dur == active_state_dur:

```

```

        onset_time_l.append(alarm_t)
        onset_val = True
        alarm_val = False
        state_dur = 0

onsets = np.union1d(onset_time_l, offset_time_l)

return (onsets, tf_l)

```

Б.7 Текст файла `silva.py`

```

def silva(sig=None, w_size=None, threshold_s=None, threshold_val=None):

    sig_zero_mean = sig - np.mean(sig)
    f_sig = np.abs(sig_zero_mean)
    tf_mvgav_sig = np.convolve(f_sig, np.ones((w_size,)) / w_size, mode='valid')
    threshold_mvgav_sig = np.convolve(f_sig, np.ones((threshold_s,)) / threshold_s,
mode='valid')
    onset_time_l = []
    offset_time_l = []
    onset_s = False
    for k in range(0, len(threshold_mvgav_sig)):
        if onset_s is True:
            if tf_mvgav_sig[k] < threshold_mvgav_sig[k] and tf_mvgav_sig[k] <
threshold_val:
                offset_time_l.append(k)
                onset_s = False
        else:
            if tf_mvgav_sig[k] >= threshold_mvgav_sig[k] and tf_mvgav_sig[k] >=
threshold_val:
                onset_time_l.append(k)

```

```

        onset_s = True
    onsets = np.union1d(onset_time_l, offset_time_l)
    onsets += int(w_size / 2)

    return (onsets, tf_mvgev_sig)

```

Б.8 Текст файла **londral.py**

```

def londral(sig=None, rest_sig=None, w_size=None, threshold_val=None,
active_state_dur=None):

```

```

    if isinstance(rest_sig, np.ndarray) or isinstance(rest_sig, list):
        if len(rest_sig) >= 2:
            rest_sig = np.array(rest_sig)
            if len(rest_sig) == 2:
                rest_signal = sig[rest_sig[0]:rest_sig[1]]
            else:
                rest_signal = rest_sig
            rest_zero_mean = rest_signal - np.mean(rest_signal)
            statistics = st.signal_stats(signal=rest_zero_mean)
            mean_rest = statistics['mean']
            std_dev_rest = statistics['std_dev']
        elif isinstance(rest_sig, dict):
            mean_rest = rest_sig['mean']
            std_dev_rest = rest_sig['std_dev']
        sig = sig - np.mean(sig)
        threshold_val = mean_rest + threshold_val * std_dev_rest

    def londral_f(signal=None):

```

```

    tf = (1 / w_size) * (sum(j ** 2 for j in signal) - (1 / w_size) * (sum(signal) **
2))

```

```

    return tf

```

```

_, tf = st.windower(
    signal=sig,
    size=w_size, step=1,
    fcn=londral_f,
    kernel='rectangular',
)
onset_time_l = []
offset_time_l = []
alarm_t = 0
state_dur = 0
onset_s = False
alarm_s = False
for k in range(0, len(tf)):
    if onset_s is True:
        if alarm_s is False:
            if tf[k] < threshold_val:
                alarm_t = k
                alarm_s = True
        else:
            if tf[k] < threshold_val:
                state_dur += 1
                if state_dur == active_state_dur:
                    offset_time_l.append(alarm_t)
                    onset_s = False
                    alarm_s = False

```

```

        state_dur = 0
    else:
        if alarm_s is False:
            if tf[k] >= threshold_val:
                alarm_t = k
                alarm_s = True
            else:
                if tf[k] >= threshold_val:
                    state_dur += 1
                    if state_dur == active_state_dur:
                        onset_time_l.append(alarm_t)
                        onset_s = True
                        alarm_s = False
                        state_dur = 0
    onsets = np.union1d(onset_time_l, offset_time_l)
    onsets += int(w_size / 2)

    return (onsets, tf)

```

ДОДАТОК Б
Слайди презентації

Дослідження, розробка та програмна реалізація методів аналізу та класифікації багатовимірних сигналів

Автор: ст. гр. КНТ-219м
Лимарєв Ігор Олегович

Керівник: зав. каф. ПЗ, д.т.н., професор
Субботін Сергій Олександрович

2020 р.

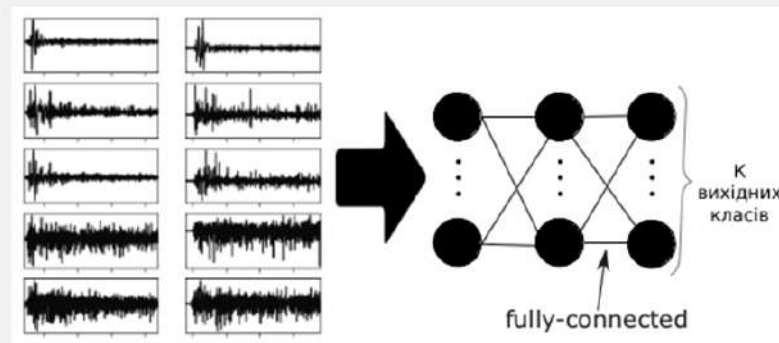
Рисунок Б.1 – Титульний слайд

- **Мета роботи** – створення методів програмної системи для автоматизації класифікації багатовимірних сигналів.
- **Об’єкт дослідження** – процес класифікації сигналів в системах діагностування.
- **Предмет дослідження** – методи та програмні засоби класифікації багатовимірних сигналів для діагностування.

2

Рисунок Б.2 – Мета, об’єкт, предмет і задачі дослідження

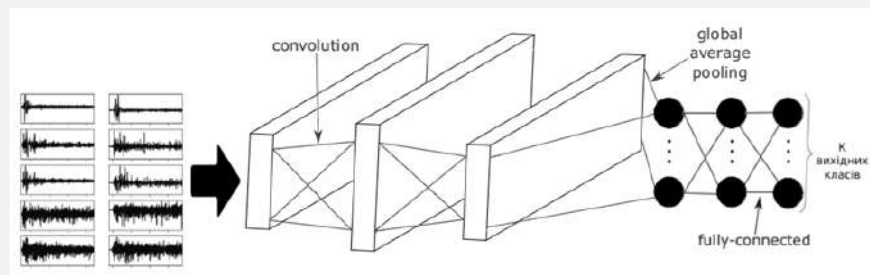
Аналіз предметної області



Багатошаровий перцептрон

3

Рисунок Б.3 – Аналіз предметної області: багатошаровий перцептрон

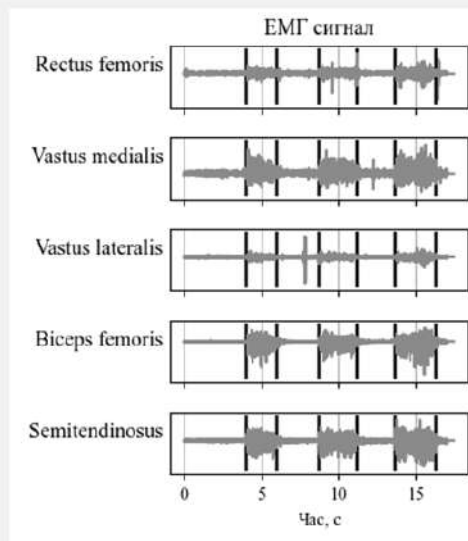


Згорткова нейронна мережа

4

Рисунок Б.4 – Аналіз предметної області: згорткова нейронна мережа

Виділення ділянок активності ЕМГ сигналу



5

Рисунок Б.5 – Виділення ділянок активності ЕМГ сигналу

Метод виділення періодів активності з багатовимірного ЕМГ сигналу

Етап ініціалізації: $\{x_1(t), x_2(t), \dots, x_n(t)\}$

Етап масштабування: $f(t) = \frac{x(t) - \overline{rest}}{\sigma(rest)}$

Етап об'єднання: $X(t) = \begin{bmatrix} f_1(t) \\ f_2(t) \\ \dots \\ f_n(t) \end{bmatrix}$

Етап усереднення: $s(t) = \frac{\sum_{i=1}^n X_i}{n}$

Етап виділення періодів активності: $onsets = \{(t_{1s}, t_{1p}), (t_{2s}, t_{2p}), \dots, (t_{ms}, t_{mp})\}$

6

Рисунок Б.6 – Методи виділення періодів активності з багатовимірного ЕМГ сигналу

Метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних

Етап ініціалізації: $L = \{onsets_1, onsets_2, \dots, onsets_c\}$

Етап усереднення екземплярів: $I_{onset} = \frac{\sum_{i=1}^m |t_{is} - t_{if}|}{m}$

Етап усереднення набору даних: $D = \frac{\sum_{c=1}^c I_c}{c}$

Етап перерахунку початкових наборів пар точок: Отримати набори пар точок з однаковою оптимальною довжиною проміжку активації D для усього набору даних.

7

Рисунок Б.7 – Методи знаходження оптимальної довжини активації багатовимірного ЕМГ сигналу для навчально набору даних

Схема структури програмної бібліотеки



8

Рисунок Б.8 – Схема структури програмної бібліотеки

Схема функціонування програмної бібліотеки



9

Рисунок Б.9 – Схема функціонування програмної бібліотеки

Вибір мови програмування

Можливості	Скриптові мови програмування						
	VBScript	JavaScript	Python	Tcl	Ruby	PHP	Perl
Імперативна парадигма	+	+	+	+	+	+	+
Об'єктно-орієнтована парадигма	+	+	+	+	+	+	+
Статична типізація	+	-	-	-	-	-	-
Динамічна типізація	+	+	+	+	+	+	+
Неявне приведення типів	+	+	-	+	-	+	+
Підтримка алгебраїчних типів даних	-	+	+	+	+	+	+
«Прибирання сміття» при керуванні пам'ятю	+	+	+	+	+	+	+
Наявність бібліотек з відкритим кодом для роботи з багатовимірними матрицями	-	+	+	-	+	+	-
Наявність бібліотек з відкритим кодом для візуалізації даних	-	+	+	-	+	-	-

10

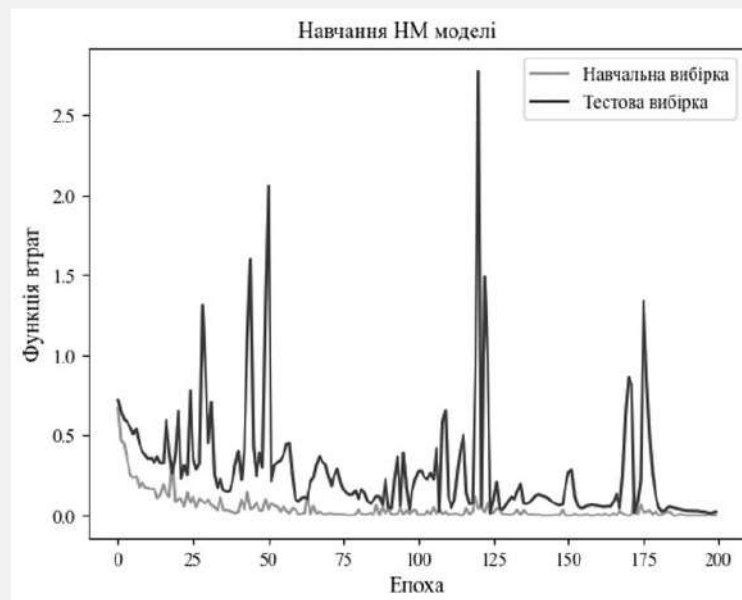
Рисунок Б.10 – Вибір мови програмування

Експериментальні результати

№ НМ моделі	К-ть згорткових шарів	Наявність шарів батч- нормалізації	Ассигасу	PPV	TPR
1	2	-	0.80	0.80	0.73
2	2	+	0.90	0.90	0.75
3	3	-	0.85	0.90	0.82
4	3	+	1.00	1.00	1.00
5	4	-	0.85	0.80	0.82
6	4	+	0.95	0.90	0.90
7	5	-	0.85	0.80	0.82
8	5	+	0.90	0.80	0.90

11

Рисунок Б.11 – Експериментальні результати



12

Рисунок Б.12 – Графік навчання НМ моделі

Техніко-економічні показники

Калькуляційні статті	Витрати, грн
Електропостачання	680
Разом	680
Заробітна плата основних виробничих працівників	76800
ЄСВ	23496
Амортизація	3000
Витрати на утримання та експлуатацію основних засобів, поточний ремонт	1500
Виробнича собівартість	98196
Адміністративні витрати	30000
Повна собівартість	128876

13

Рисунок Б.13 – Техніко-економічні показники

Висновки

Проведено дослідження методів класифікації процесів за даними багатовимірних сигналів.

Наукова новизна:

- вперше розроблено метод знаходження ділянок найбільшої активності багатовимірного ЕМГ сигналу, що базується на існуючих одновимірних методах;
- вперше розроблено метод знаходження оптимальної довжини ділянки активації багатовимірного ЕМГ сигналу для навчального набору даних.

Практична цінність:

- розроблено програмну бібліотеку для попередньої обробки даних багатовимірних сигналів та їх класифікації, у якій реалізовано розроблені методи.

14

Рисунок Б.14 – Висновки

Публікації та здобутки студента

- результати роботи опубліковано в 5 друкованих працях, в тому числі 1 стаття у науковому фаховому виданні України, що індексується наукометричною базою Web of Science та 4 тези доповідей на наукових конференціях.
- 1 місце у II турі Всеукраїнського конкурсу студентських наукових робіт з напрямку „Інформатика і кібернетика” у 2017-2018 н.р.
- 1 місце у II турі Всеукраїнського конкурсу студентських наукових робіт з напрямку „Комп’ютерні науки” у 2018-2019 н.р.
- диплом переможця Обласного конкурсу для обдарованої молоді у галузі фізико-математичних наук серед студентів у 2018 р.

15

Рисунок Б.15 – Публікації та здобутки студента

Перелік публікації

**Статті у міжнародних рецензованих виданнях,
індексованих наукометричними базами:**

- Lymarev I. O. Methods of large-scale signals transformation for diagnosis in neural network models / [I. O. Lymarev, S.A. Subbotin, A. A. Oliinyk et al.] // Radio Electronics, Computer Science, Control. – 2018. – № 4. – P. 63–71.
- Tverdohle, J. Wavelet analysis of complex nonstationary oscillatory signals / [J. Tverdohle, I. Limarev, V. Logominov et al.] // 4-th IEEE International Scientific-Practical Conference "Problems of Infocommunications. Science and Technology". – 2017. – P. 17-20.
- Limarev I. Diagnostic Signal Nonstationarity Reduction to Predict the Helicopter Transmission State on the Basis of Intelligent Information Technologies / [I. Lymarev, S. Subbotin, A. Oliinyk et al.] // Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019). – 2019. – P. 510–522.

Тези доповідей на національних та регіональних конференціях:

- Лимарев І.О. Довготермінове прогнозування стану трансмісії вертольоту на основі рекурентних нейронних мереж / І.О. Лимарев // Радіоелектроніка і молодь в ХХІ ст.: матеріали ХХІІ Міжнародного молодіжного форуму. – 2018. – Т. 5. – С. 223-224.
- Лимарев, И.О. Методы преобразования сигналов большой размерности в диагностических моделях управляемых данными / [И.О. Лимарев, С.А. Субботин, А.А. Олейник и др.] // Нейроинформатика, ее приложения и анализ данных: Материалы XXVI Всероссийского семинара. – 2018. – С. 90–95.

16

Рисунок Б.16 – Перелік публікації

Підтвердження здобутків



17

Рисунок Б.17 – Підтверження здобутків