

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
НАВЧАННЯ З ПІДКРІПЛЕННЯМ ДЛЯ МЕДИЧНОГО ДІАГНОСТУВАННЯ
RESEARCH AND SOFTWARE IMPLEMENTATION OF REINFORCEMENT
LEARNING METHODS FOR MEDICAL DIAGNOSTICS

Виконав: студент 2 курсу, групи КНТ-120м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Луковка О.О.

(прізвище та ініціали)

Керівник Олійник А.О.

(прізвище та ініціали)

Рецензент Скрупський С.Ю.

(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 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Луковки Олексія Олександровича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів навчання з підкріпленням для медичного діагностування. Research and Software Implementation of Reinforcement Learning Methods for Medical Diagnostics

керівник проєкту (роботи) Олійник Андрій Олександрович, д.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “05” листопада 2021 року № 435

2. Строк подання студентом проєкту (роботи) 5 грудня 2021 року

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблемної області та постановка завдань дослідження. 2. Матеріали та методи. 3. Розробка архітектури програми. 4. Основні рішення щодо реалізації компонентів системи. 5. Експлуатація, тестування та експериментальне дослідження програми.

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	Олійник А.О., професор		
Нормоконтроль	Липовець М.В., асистент		

7. Дата видачі завдання “4” вересня 2021 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.	9-10 тиждень	Розділ 5
7	Оформлення пояснювальної записки та документів до неї.	11-13 тиждень	Додатки
8	Нормоконтроль та рецензування.	14-15 тижні	
9	Захист роботи.	16 тиждень	

Студент(ка)

(підпис) Луковка О.О.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Олійник А.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
100 с., 31 рис., 5 табл., 4 дод., 22 джерела.

МЕДИЧНЕ ДІАГНОСТУВАННЯ, МАШИННЕ НАВЧАННЯ,
НАВЧАННЯ З ПІДКРІПЛЕННЯМ, ШТУЧНІ НЕЙРОННІ МЕРЕЖІ,
PYTHON, TENSORFLOW.

Об'єкт дослідження – процес використання методів навчання з підкріпленням для вирішення задач медичного діагностування.

Предмет дослідження – методи навчання з підкріпленням для синтезу нейромоделей медичного діагностування.

Мета роботи – дослідити, проаналізувати та реалізувати методи навчання з підкріпленням для синтезу нейромоделей медичного діагностування.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, технології навчання з підкріпленням, мова програмування Python, фреймворк TensorFlow, середовище розробки Anaconda, персональний комп'ютер з процесором Intel Core i7-10700K під управлінням операційної системи Microsoft Windows 10.

Результати. Створено програмний застосунок, що реалізує метод навчання з підкріпленням: дуельні подвійні глибокі Q-мережі, для синтезу діагностичних нейромоделей.

Висновки. Розроблено програмний застосунок для синтезу моделей на основі штучних нейронних мереж для автоматизації процесів медичного діагностування.

Галузь використання – медичні заклади та досліджувальні центри.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 100 pages, 31 figures, 5 tables, 4 appendixes, 22 sources.

MEDICAL DIAGNOSTICS, MACHINE LEARNING, REINFORCEMENT LEARNING, ARTIFICIAL NEURAL NETWORKS, PYTHON, TENSORFLOW.

The object of research is the process of using reinforcement learning methods to solve medical diagnostic problems.

The subject of the research is reinforcement learning methods for the synthesis of neuromodels in medical diagnostics.

The aim of the work is to investigate, analyze and implement reinforcement learning methods for the synthesis of neuromodels in medical diagnostics.

Materials, methods and technical tools: structural and Object-Oriented Programming, reinforcement learning technologies, Python programming language, TensorFlow framework, Anaconda development environment, personal computer with an Intel Core i7-10700k processor running the Microsoft Windows 10 operating system.

Results. A software application has been created that implements the reinforcement learning method: dueling double Deep Q-networks, for the synthesis of diagnostic neuromodels.

Conclusions. Software for synthesizing models based on artificial neural networks for automating medical diagnostic processes has been developed.

The field of use is medical institutions or research centers.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз проблемної області та постановка завдань дослідження.....	11
1.1 Медичне діагностування.....	11
1.2 Застосування машинного навчання та штучного інтелекту для медичного діагностування	16
1.3 Використання методів МН для вирішення проблем медичного діагностування.....	21
1.4 Дуельні Подвійні Глибокі Q-Мережі.....	27
1.5 Постановка завдання	28
1.6 Висновки за розділом 1	28
2 Матеріали та методи.....	29
2.1 Процес прийняття рішень Маркова і Q-learning.....	29
2.2 ДПГQ-М із використанням пріоритетної передачі досвіду	33
2.3 Аналіз чутливості запропонованого методу	36
2.4 Висновки за розділом 2	41
3 Розробка архітектури програми	42
3.1 Опис основних вимог до програми.....	42
3.2 Вибір мови програмування	43
3.3 Вибір фреймворку.....	45
3.4 Вибір середовища розробки.....	47
3.5 Проектування структури програмної системи.....	49
3.6 Вимоги до технічних засобів	50
3.7 Висновки за розділом 3	50
4 Основні рішення щодо реалізації компонентів системи	51
4.1 Модульна структура ПЗ	51
4.2 Модуль завантаження та обробки даних.....	52
4.3 Модуль синтезу ШНМ.....	53

4.4	Модуль збереження мережі	55
4.5	Модуль розгортання моделі.....	56
4.6	Висновки за розділом 4	57
5	Експлуатація, тестування та експериментальне дослідження програми....	58
5.1	Вхідні дані для тестування.....	58
5.2	Метрики оцінювання	59
5.3	Параметри та оптимізація	60
5.4	Результати тестування	60
5.5	Аналіз отриманих результатів	65
	Висновки	68
	Перелік джерел посилання	70
	Додаток А Технічне завдання	73
	Додаток Б Опис програми	78
	Додаток В Текст програми	82
	Додаток Д Слайди презентації.....	91

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CNN	– згорткові нейронні мережі;
DNN	– глибокі нейронні мережі;
DQN	– метод глибоких Q-мереж;
GRU	– вентильний рекурентний вузол;
IoT	– Інтернет речей;
LSTM	– довга короткочасна пам'ять;
MNN	– багат шарові нейронні мережі;
PG	– policy gradient;
RNN	– рекурентні нейронні мережі;
WDR	– зважена подвійна надійність;
ГНП	– глибоке навчання з підкріпленням;
ДПГQ-M	– дуельні подвійні глибокі Q-мережі;
ЕКГ	– електрокардіограма;
MN	– машинне навчання;
MPT	– магнітно-резонансної томографія;
ПЗ	– програмний застосунок;
ПК	– персональний комп'ютер;
ППД	– пріоритетна передача досвіду;
ПРМ	– прийняття рішень Маркова;
ТР	– тимчасова різниця
ШІ	– штучний інтелект;
ШІНМ	– штучні нейронні мережі.

ВСТУП

Розвиток системи охорони здоров'я і вихід України на європейський рівень в медицині відбувається досить повільно, чому сприяє цілий ряд проблем.

Основна проблема – це брак фінансування, що неминуче відбивається на рівні медичної техніки та лабораторного обладнання, використовуваних технологій і методів діагностики. Крім цього, низький рівень оплати праці призводить до відходу кваліфікованих фахівців в інші сфери діяльності. З іншого боку, обмежене фінансування впливає на якість підготовки молодих лікарів, розвиток медичної науки та післядипломної освіти.

Актуальною проблемою охорони здоров'я є тенденція до збільшення ризику різних захворювань, до підвищення рівня смертності, що пов'язано з погіршується екологією, низьким рівнем життя, несвоєчасною діагностикою, а також недостатньою профілактикою різних порушень здоров'я.

Використання в медичних установах сучасних інформаційних комп'ютерних технологій дозволяє не тільки підвищити ефективність процесів обробки медичної інформації, але і відкриває великі перспективи у вирішенні багатьох проблем сучасної медицини. До теперішнього моменту створення і використання медичних систем підтримки прийняття рішень виділилося в окремий напрям наукових досліджень, який об'єднує результати, отримані для інтелектуальних систем, зокрема експертних систем, підходів Data Mining, а також сучасні концепції інформаційних технологій.

Одне з основних завдань, вирішенню якої присвячено велике число публікацій [1]-[8], пов'язана зі створенням систем для медичного діагностування.

Обробці медичної інформації та подальшому прийняттю рішень присвячено безліч робіт. Незважаючи на це, ще досить багато питань в даній області залишилося відкритими, що зазначають, зокрема, сучасні автори.

Більш того, вкрай важливо засвідчити, що на сьогоднішній день медицина, що орієнтувалася раніше, в основному, на лікування гострих захворювань, тепер більше уваги приділяє недугою хронічним, багато з яких не так давно і хворобами не вважалися. Лікарі стикаються з необхідністю лікувати ожиріння, депресії, хвороби похилого віку. Діабет, серцева недостатність, аутоімунні розлади все частіше діагностуються поза фазою загострення, на самих ранніх стадіях, причому мова все частіше йде не тільки про підтримуючої терапії, але про можливість повністю вилікувати, виправити ці системні збої організму. Розвивається превентивна медицина, що дозволяє розпізнати схильність до певних типів захворювань ще до їх прояву і вжити заходів. Швидко зростають обсяги медичних даних, і ми починаємо розуміти, що від швидкості і якості їх аналізу залежать наше здоров'я і якість життя. І що все це – робота для спеціалізованих медичних систем підтримки прийняття рішень.

Метою дослідження є аналіз сучасних практик використання комп'ютерних технологій (машинне навчання, штучний інтелект, нейронні мережі, тощо) задля якісного покращення та автоматизації важливого етапу забезпечення високого рівня медичного сервісу – медичного діагностування.

1 АНАЛІЗ ПРОБЛЕМНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

Перший розділ буде присвячено детальному аналізу проблемної області пов'язаної із медичним діагностуванням та сучасних викликів цієї проблемної області. Під час роботи буде проведено дослідження основних термінів таких, як медичне діагностування, машинне навчання та навчання з підкріпленням.

1.1 Медичне діагностування

1.1.1 Визначення області медичного діагностування

Лікування хвороби починається з її розпізнавання, точного встановлення діагнозу. Процес визначення захворювань називається медичною діагностикою, а встановлення точного діагнозу та загальне вирішення про стан пацієнтка – медичним діагностуванням [1], [4]. Це всебічний огляд пацієнта, виявлення вогнищ патології та її ступеня. Для всіх відомих проблем зі здоров'ям є свій набір медичних досліджень, які допоможуть точно знайти проблему. Постановка діагнозу має вагоме значення в медицині, оскільки вона визначає методи майбутнього лікування. Клінічне обстеження включає три розділи [2]:

- семіотика – набір симптомів захворювання. Це невід'ємна частина розпізнавання хвороб, по ним доктор робить перші припущення про діагноз, визначає наступні види досліджень;

- діагностичне обстеження – включає ряд аналізів, оглядів, тестування. За допомогою таких заходів медики встановлюють вид хвороби, іноді її причину і способи лікування;

- постановка діагнозу – заключний етап медичного обстеження. На основі даних, отриманих при аналізах і оглядах, роблять остаточні висновки про стан пацієнта.

Дослідження на субклітинному рівні проводять при лабораторній діагностиці, сюди відноситься біохімічний аналіз крові, аналіз сечі, біопсія та інші. Щоб перевірити стан і функціонування окремої системи або органу, застосовують інструментальний метод. Сучасні способи вивчення людського тіла, такі як комп'ютерна томографія, дозволяють детально розглянути структуру і анатомію організму швидко і без болю. Для деяких видів діагностування дуже важливо, щоб пацієнт був спокійний і розслаблений, іноді потрібна особлива підготовка [3].

У наш час у медичної науки є величезні можливості, щоб детально вивчити будову і роботу органів і систем людського тіла, швидко і точно діагностувати будь-які порушення або хвороби. Деякі діагностичні методи дозволяють виявити тільки певні патології. Однак більшість досліджень універсально і застосовується лікарями різних спеціальностей [3].

Традиційно діагностичні методи поділяють на лабораторну діагностику, що відображає зміни в клітинному і хімічному складі біорідин та інших біоматеріалів, і інструментальну діагностику, що дозволяє наочно визначити те, що відбувається в конкретному органі.

Інструментальні методи діагностики, в свою чергу поділяються на [1], [4]:

- структурну діагностику, що виявляє зміни в будові органів і тканин, це рентген, ультразвукова діагностика, ендоскопія і т.д.
- функціональну діагностику, що досліджує роботу органів і систем по їх електричним, звуковим, механічним та іншим проявам.

Популярні інструментальні методи визначення захворювань:

- ультразвукове дослідження створює зображення внутрішніх органів на екрані завдяки відображенню від них ультразвукових хвиль;
- рентгенографія отримує зображення органів і систем за допомогою рентгенівського випромінювання (кісткові структури, шлунково-кишковий тракт, серце і великі судини);

- рентгеноскопія – це безперервна рентгенівська зйомка, що відображає биття серця, дихальні рухи легенів, перистальтику кишечника і т.д.;
- флюорографія, як вид рентгенологічного дослідження зазвичай застосовується для виявлення порушень в органах грудної клітини;
- комп'ютерна томографія є також різновидом рентгенологічного дослідження, що відрізняється високою роздільною здатністю і точністю (головний мозок і інші внутрішні органи);
- магнітно-резонансна томографія або ядерно-магнітний резонанс – діагностичне обстеження, що застосовує потужне магнітне поле для отримання точних зображень органів;
- радіонуклідний метод використовується для дослідження кровопостачання певних органів і виявлення пухлинних захворювань і метастазів, шляхом введення в вену спеціальних речовин-індикаторів;
- електрокардіографія фіксує на папері посилені електричні імпульси серця;
- ехокардіографія – це ультразвукове дослідження серця;
- електроенцефалографія реєструє електричну активність головного мозку;
- ендоскопічне дослідження обстежує порожнисті органи і порожнини за допомогою гнучкого волоконно-оптичного інструменту (ендоскопа).

1.1.2 Проблеми медичного діагностування

Наукове пізнання передбачає виявлення нових фактів, закономірностей або нового способу пояснення явищ. Медичне діагностування не відкриває принципово щось нове, а є розпізнавання конкретного патологічного процесу у певного пацієнта [2], [6]. Діагноз – це розпізнавання хвороби на основі клінічних і лабораторних даних. Це

складний пізнавальний специфічний процес, який розпадається на ряд етапів: обстеження хворого, аналіз отриманих даних і створення синтетичної картини захворювання у даного хворого, побудова власне діагнозу, перевірка істинності діагнозу і його уточнення в ході лікування, прогноз захворювання і його результатів [4].

У той же час діагностичний процес має деякі специфічні риси, що відрізняють його від інших видів пізнавальної діяльності. Перш за все, діагностика є процес розпізнавання, який пов'язаний не з відкриттям нової, невідомої раніше сутності, а з підведенням одиничного під загальне шляхом розпізнавання цього спільного в окремому. У своєму діагностичному міркуванні лікар йде від відомих загальних понять (різні захворювання як нозологічні одиниці) до чогось невідомого одиничного (хвороба у даного індивіда). Діагноз є констатація нозологічної одиниці (певної форми патологічного процесу) і її диференціація від інших нозологічних одиниць. При цьому необхідно враховувати, що конкретний прояв загального в кожному окремому випадку своєрідно, що перетворює діагностичний процес в творчий акт лікаря [5].

Інша особливість процесу медичної діагностики полягає в діалектичному переплетенні об'єктивних і суб'єктивних факторів пізнавального процесу. Суб'єктивне важко розпізнавати і формалізувати. Кожна хвороба має свої особливості у кожного суб'єкта. Якщо діагностику автомобіля можна доручити приладів, то діагностику хвороби повинен здійснити персонально лікар. Тільки фахівець може оцінити складність і своєрідність прояви хвороби [5].

Діагностика є судження, яке ґрунтується на певних законах формальної логіки (закон тотожності, протиріччя, виключеного третього і достатньої підстави). Ці закони логіки є обов'язковими для всіх розумових операцій, вони мають нормативний характер. Тому якість діагностики залежить не тільки від того, як лікар обстежує хворого, але і від здатності лікарів мислити логічно [5], [6].

В умовах комп'ютеризації медичної сфери значно підвищуються діагностичні можливості лікаря за рахунок появи потужних діагностичних приладів. Має місце автоматизація діагностичного процесу і лікування, створюються інформаційні моделі хвороб і експертні системи. Використання експертних систем дозволяють лікарю керуватися в своїй практиці досвідом найбільших фахівців. Використання комп'ютерів в медицині має багатообіцяючу перспективу, оскільки машини мають ряд переваг перед людиною. Але це не означає, що діагностика буде перекладена на плечі комп'ютера або діагностичних систем. Останні будуть використовуватися в якості поради лікаря [6].

Іншими словами, інформатизація медицини має як позитивні, так і негативні наслідки. Як результат інформатизації медичної сфери поліпшується якість діагностики, виключається суб'єктивізм і однобічність в медичній діяльності. Але це може вести і до стандартизації і дегуманізації медицини. Тільки в підручнику клінічна картина хвороби має класичний опис, а в дійсності можемо зустріти різні особливості які комп'ютер не в змозі враховувати. Комп'ютер і технічні засоби, будучи джерелом інформації про хворого, віддаляють лікаря від хворого. Лікар вважає за краще контактувати більше з технікою, ніж з живою людиною. Одночасно зростає можливість появи діагностичних помилок. З великої кількості прийомів і методів діагностики та лікування треба вибрати оптимальні і все буде залежати від професійної підготовки лікаря і його ставлення до свого обов'язку. Інформатизація медицини сприяє зростанню компетентності та професіоналізму лікарів, зростанню частки творчої праці [6].

Створення експертних систем дає можливість використовувати знання і досвід найбільших фахівців (які створили ці експертні системи) [7]. І все ж експертні системи обмежені рівнем знань і кваліфікації відповідного фахівця, дають нам ймовірні висновки про можливі хвороби. Тому основну роботу з аргументації та диференціації хвороби повинен здійснити особисто лікар (а не машина). Комп'ютеризація та інформатизація медичної сфери не може

повністю замінити класичні методи діагностики. Лікар повинен володіти як класичними, так і новими методами діагностики та лікування. Для цього потрібна більш широка професійна підготовка лікарів, яка передбачає високу філософську культуру, логіко-методологічну озброєність [8].

1.2 Застосування машинного навчання та штучного інтелекту для медичного діагностування

1.2.1 Машинне навчання та штучний інтелект

На сьогоднішній день накопичені і систематизовані найрізноманітніші підходи і математичні алгоритми для побудови систем штучного інтелекту (ШІ), такі як байєсівські методи, логістична регресія, метод опорних векторів, вирішальні дерева, ансамблі алгоритмів, тощо [5], [7], [8].

Останнім часом ряд експертів приходить до висновку, що більшість сучасних і дійсно вдалих реалізацій – це рішення, побудовані на технології глибоких нейронних мереж (DNN) і глибокого машинного навчання [5], [7], [8].

Штучні нейронні мережі (ШНМ) засновані на спробі відтворити примітивну модель нервових систем в біологічних організмах. У живих істот нейрон – це електрично збудлива клітина, яка обробляє, зберігає і передає інформацію за допомогою електричних і хімічних сигналів через синаптичні зв'язки. Нейрон має складну будову і вузьку спеціалізацію. З'єднуючись один з одним для передачі сигналів за допомогою синапсів, нейрони створюють біологічні нейронні мережі. У головному мозку людини налічується в середньому близько 65 мільярдів нейронів і 100 трильйонів синапсів. По суті це і є базовий механізм навчання і мозкової діяльності всіх живих істот, тобто-їх інтелект. Наприклад, в класичному досвіді Павлова кожен раз безпосередньо перед годуванням собаки дзвонив дзвіночок, і собака швидко навчилася пов'язувати дзвінок дзвіночка з їжею. З фізіологічної точки зору результатом досвіду в її мозку стало встановлення синаптичних зв'язків між

ділянками кори головного мозку, відповідальними за слух, і ділянками, відповідальними за управління слинними залозами. В результаті при порушенні кори звуком дзвіночка у собаки починалося слиновиділення. Так собака навчилася реагувати на надходять із зовнішнього світу сигнали (дані) і робити «правильний» висновок [5], [7], [8].

Саме здатність біологічних нервових систем навчатися і виправляти свої помилки лягла в основу досліджень в області штучного інтелекту. Їх початковим завданням була спроба штучно відтворити низькорівневу структуру мозку – тобто створити комп'ютерний штучний мозок. В результаті була запропонована концепція штучного нейрона – математичної функції, яка перетворює кілька вхідних фактів в один вихідний, призначаючи для них ваги впливу. Кожен штучний нейрон може взяти зважену суму вхідних сигналів і в разі, якщо сумарний вхід буде перевищувати певний пороговий рівень, передати двійковий сигнал далі [5], [7], [8].

Штучні нейрони об'єднують в мережі-поєднуючи виходи одних нейронів з входами інших. З'єднані і взаємодіючі між собою штучні нейрони створюють ШНМ – певну математичну модель, яка може бути реалізована на програмному або апаратному забезпеченні. ШНМ – це просто програма чорний ящик, яка отримує на вхід дані і видає відповіді. Являючись побудованою з дуже великого числа простих елементів, ШНМ здатна вирішувати надзвичайно складні завдання [5], [7], [8].

В даний час існує безліч моделей реалізації ШНМ. Є класичні одношарові ШНМ, вони застосовуються для вирішення простих завдань. Одношарова нейронна мережа ідентична в математичному сенсі звичайному поліному, ваговій функції, традиційно застосовуваної в експертних моделях. Число змінних в поліномі дорівнює числу входів мережі, а коефіцієнти перед змінними рівні ваговим коефіцієнтам синапсів [5], [7], [8].

Є математичні моделі, в яких вихід однієї ШНМ направляється на вхід іншої, і створюються каскади зв'язків, так звані багатошарові нейронні

мережі (MNN) і один з найбільш потужних її варіантів – згорткові нейронні мережі (CNN) [5], [7], [8].

Іншою цікавою різновидом нейромереж є нейронні мережі зі зворотним зв'язком (RNN), коли вихід з шару мережі подається назад на один з входів. У таких платформ є ефект пам'яті і вони здатні відстежувати динаміку змін входних факторів. Простий приклад: посмішка – людина починає посміхатися з ледве помітних рухів мимічних м'язів очей і обличчя, перш ніж явно покаже свої емоції. RNN дозволяє виявити такий рух ще на ранніх фазах, що буває корисно для прогнозування поведінки живого об'єкта в часі за допомогою аналізу серії зображень або конструюванні послідовного потоку мови на природній мові [5], [7], [8].

Машинне навчання (МН) – це процес машинного аналізу підготовлених статистичних даних для пошуку закономірностей і створення на їх основі потрібних алгоритмів (налаштування параметрів нейронної мережі), які потім будуть використовуватися для прогнозів [5], [7], [8].

Створені на етапі МН методи дозволять надалі комп'ютерному ШІ зробити коректні висновки на підставі наданих йому даних.

Розрізняють 3 основних підходи до МН [5], [7], [8]:

- навчання з учителем;
- навчання з підкріпленням;
- навчання без вчителя.

1.2.2 Навчання з підкріпленням та його переваги

Навчання з підкріпленням – спосіб МН, при якому виконується автоматичне навчання процесу прийняття рішень у часі. Це завдання набуло широкого поширення в багатьох наукових та інженерних областях [9]-[12].

Навчання з підкріпленням – підхід, який спочатку включає додатковий вимір (найчастіше це час) в процес навчання, що робить його значно ближче до людського сприйняття штучного інтелекту [9]-[12].

У навчанні безпілотних автомобілів у машини немає завдання запам'ятати детальну карту міста, країни, континенту, всі вулиці і повороти. Але вона обов'язково повинна зрозуміти шаблони повторюваних ситуацій і узагальнити їх [9]-[12].

Перша мета агента в навчанні з підкріпленням – мінімізувати помилки. Машина вчиться аналізувати інформацію перед кожним наступним ходом. Наприклад, безпілотний автомобіль під час навчання вчиться вчасно реагувати на сигнал світлофора, зупинитися перед пішоходом на переході, пропустити швидко рухається автомобіль або спецтранспорт справа. Щоб досягти кращого результату, машина навчається у віртуальній моделі міста з випадковими пішоходами та іншими учасниками дорожнього руху [9]-[12].

Друга мета агента – отримати від виконання завдання максимальну вигоду. Сама вигода при цьому повинна бути запрограмована заздалегідь: максимально швидкий час проходження маршруту, оптимальне витрачання ресурсів підприємства, обслуговування якомога більшої кількості відвідувачів [9]-[12].

Порівнюємо навчання із підкріпленням з іншими стратегіями МН. Результати порівняння представимо у вигляді табл. 1.1.

Таблиця 1.1 – Порівняння підходів МН

	Навчання з учителем	Навчання без учителя	Навчання з підкріпленням
Залежність від вхідних даних	Високий рівень	Низький рівень	Середній рівень
Використовувані методи	Високий рівень	Високий рівень	Високий рівень
Коло практичних завдань	Середній рівень	Середній рівень	Високий рівень (в останній час)
Подальший розвиток	Перспективна галузь (за умови використання підходів IoT)	Перспективна галузь	Перспективна галузь

Порівняння підходів до МН відбувалося за наступними критеріями [9]-[12]:

- залежність від вхідних даних – наскільки методи підходу МН залежать від вхідних даних задачі;
- використовувані методи – коло можливих методів, які можуть бути застосовані під час роботи;
- коло практичних завдань – оцінка перспективних завдань, що можуть бути вирішені з використанням конкретного підходу МН;
- подальший розвиток – опосередкована оцінка перспектив розвитку підходу МН на основі сучасних практик та досліджень.

Так в результаті порівняння можемо засвідчити наступні результати. Усі методи навчання з учителем сильно залежать від вхідних даних, що є їхнім слабким місцем, адже не завжди нам доступний повний набір вхідних даних про об'єкт діагностування або ж ми не завжди можемо бути впевнені у високому рівні якості таких даних. З іншого боку, навчання з учителем найбільш знаний підхід МН та встиг накопичити багато методів, що можуть бути використані, а з урахуванням того, що з розвитком технологій Інтернету речей (IoT) коло практичних завдань таких методів значно розширилося, що забезпечує постійний розвиток підходу. Методи, що належать підходу навчання без учителя взагалі не пов'язані із вхідними даними. Коло їх практичних завдань значно ширше, адже достатньо часто ми не маємо вичерпної інформації про об'єкт діагностування або ж середовище. На сьогоднішній день цей підхід також накопичив достатньо багато методів. Навчання з підкріпленням найбільш новий підхід у МН. Цей підхід ще не встиг накопичити так багато методів, проте є найбільш перспективним напрямком, адже багато задач і з галузі IoT, і з галузі робототехніки вимагають оперувати постійно оновлюючимися даними про середовище в якому взаємодіють агенти.

1.3 Використання методів МН для вирішення проблем медичного діагностування

Напрямок медицини та охорони здоров'я вже сьогодні вважається одним із стратегічних і перспективних з точки зору ефективного впровадження МН. Використання МН може масово підвищити точність діагностики, полегшити життя пацієнтам з різними захворюваннями, підвищити швидкість розробки і випуску нових ліків і т. д. [7],[8], [9]-[12].

У найближчому майбутньому в медицині та охороні здоров'я будуть автоматизуватися за допомогою ІІІ наступні завдання.

1. Автоматизовані методи діагностики, наприклад, аналіз рентгенологічних або магнітно-резонансної томографії (МРТ) знімків на предмет автоматичного виявлення патології, мікроскопічний аналіз біологічного матеріалу, автоматичне кодування електрокардіограми (ЕКГ), електроенцефалограм і т. д. Зберігання великої кількості розшифрованих результатів діагностичного обстеження в електронному вигляді, коли є не тільки самі дані, але і формалізований висновок по них, дозволяють створювати дійсно надійні і цінні програмні продукти, здатні якщо не замінити лікаря, то надати йому ефективну допомогу. Наприклад, самостійно виявляти і звертати увагу на рутинну патологію, скорочувати час і вартість обстеження, впроваджувати аутсорсинг і Дистанційну діагностику [7],[8], [9]-[12].

2. Системи розпізнавання мови і розуміння природної мови можуть надати істотну допомогу як лікарю, так і пацієнту. Починаючи від вже звичайної розшифровки мови і перетворенні її в текст в якості більш просунутого інтерфейсу спілкування з медичними інформаційними системами, звернення в Call-центр або голосового помічника – і далі до таких ідей, як автоматичний Мовний переклад при вступі іноземця, синтез мови при прочитанні записів з міс, робот-реєстратор в приймальному відділенні

лікарні або реєстратурі поліклініки, здатний відповідати на прості питання і маршрутизувати пацієнтів і т. д. [7],[8], [9-12].

3. Системи аналізу великих даних і передбачення подій також є цілком вирішуваними вже зараз завданнями для ШІ, які можуть дати істотний ефект. Наприклад, оперативний аналіз змін захворюваності дозволяє швидко передбачити зміну оборотності пацієнтів в медичній організації або потреба в лікарських препаратах, попередити епідемію або дати точний прогноз погіршення здоров'я, що часом може врятувати життя пацієнтові [7],[8], [9]-[12].

4. Системи автоматичної класифікації та звірки інформації допомагають зв'язати інформацію про пацієнта, що знаходиться в різних формах в різних інформаційних системах. Наприклад, побудувати інтегральну електронну медичну карту з окремих епізодів, описаних з різною детальністю і без чіткого або суперечливого структурування інформації. Перспективною є технологія машинного аналізу вмісту контенту соціальних мереж, інтернет-порталів з метою швидкого отримання соціологічної, демографічної, маркетингової інформації про якість роботи системи охорони здоров'я та окремих лікувальних установ [7],[8], [9-12].

5. Автоматичні чат-боти для підтримки пацієнтів можуть надати суттєву допомогу у підвищенні прихильності пацієнтів до здорового способу життя та призначеного лікування. Вже зараз чат-боти можуть навчитися відповідати на рутинні питання, підказувати тактику поведінки пацієнтів в простих ситуаціях, з'єднувати пацієнта з потрібним лікарем в телемедицині, давати рекомендації по дієті і т.д. такий розвиток охорони здоров'я в бік самообслуговування і більшої залученості пацієнтів в охорону власного здоров'я без візиту до лікаря може заощадити істотні фінансові ресурси [7],[8], [9]-[12].

6. Розвиток робототехніки та мехатроніки. Всім відомий робот-хірург Da Vinci – це лише перший крок у бік якщо не заміни лікаря на машину, то як мінімум підвищення якості роботи медичних співробітників. Інтеграція

робототехніки з ШІ розглядається зараз як один з перспективних напрямків розвитку, здатний перекласти на машини рутинні маніпуляції – в тому числі і в медицині [7],[8], [9]-[12].

1.3.1 Перспективи використання методів навчання з підкріпленням у медицині

Хоча кілька областей медицини почали певною мірою експериментувати з навчанням з підкріпленням, цей підхід показав свої найбільш помітні успіхи в реалізації динамічних режимів лікування для пацієнтів з тривалими захворюваннями або станами. Він також досяг рівня функціональності в області автоматизованої медичної діагностики, планування та розподілу ресурсів охорони здоров'я, пошуку та розробки ліків та управління охороною здоров'я [9]-[13].

Динамічні режими лікування. Найбільш поширеним застосуванням навчання з підкріпленням в галузі охорони здоров'я в реальному світі є створення і постійне налаштування динамічних режимів лікування для пацієнтів з більш тривалими станами. Динамічні режими лікування – це послідовність правил, що регулюють рішення в галузі охорони здоров'я, включаючи тип лікування, дозування ліків і терміни призначення, адаптовані до окремого пацієнта на основі його історії хвороби і стану з плином часу. Клінічні спостереження та оцінки пацієнтів надають вхідні дані, а алгоритм виводить варіанти лікування для забезпечення найбільш бажаного стану навколишнього середовища пацієнта. Навчання з підкріпленням використовується для автоматизації прийняття рішень в рамках цих поточних режимів лікування. Це вже допомогло розробити динамічні режими лікування для хронічних захворювань, включаючи рак і вірус імунodefіциту людини, а також могло б поліпшити надання невідкладної допомоги з використанням великих даних, зібраних у відділеннях інтенсивної терапії [9]-[13].

Автоматизоване медичне діагностування. Медичні діагнози – це, по суті, вправа по зіставленню інформації про пацієнта (такий як історія хвороби і поточні симптоми) з правильним профілем захворювання. Хоча це може здатися відносно простим, з клінічної точки зору це може бути неймовірно складним завданням, що представляє величезне навантаження (як за часом, так і за необхідною когнітивною енергією) для зайнятих лікарів [9-13].

Методи МН для поліпшення діагностики важливі для індустрії охорони здоров'я та її пацієнтів. Але методи навчання з підкріпленням особливо перспективні, тому що більшість рішень для діагностики МН вимагають великої кількості анотованих даних в навчальних цілях. Агенти ж навчання з підкріпленням, навпаки, вимагають менших обсягів позначених даних [9-13].

Планування та розподіл ресурсів охорони здоров'я. Система охорони здоров'я більш-менш схожа на будь-який інший бізнес, що надає послуги, де пацієнти є клієнтами, а медичні ресурси-постачальниками послуг. Завдяки добре задокументованій придатності навчання з підкріпленням для управління бізнес-процесами, воно може допомогти лікарням і клінікам управляти повсякденними операціями, аналізуючи і розробляючи оптимальний розподіл ресурсів і планування людських ресурсів на основі сезонних тенденцій, поточного персоналу і рівнів стаціонарного обслуговування та інших точок даних [9]-[13].

Відкриття, розробка та поширення ліків. Традиційне відкриття ліків має багато недоліків, і найбільш руйнівним з них є те, що його процес, заснований на людях, методом проб і помилок, занадто витратний за часом і витратами. Це відбувається навіть при використанні сучасних методів, таких як комп'ютерні моделі та моделювання для аналізу поведінки молекул і атомів. Однак, незважаючи на весь цей час і гроші, показники успіху як і раніше відносно низькі, і трохи менше 10% з'єднань проходять випробування фази I. З цих причин методи навчання з підкріпленням все частіше

застосовуються для розробки ліків саме для автоматизації та поліпшення гіпотез розробки ліків та вибору сполук [9]-[13].

Деякі розробники ліків експериментують з передовими методами машинного навчання в поєднанні з квантовими обчисленнями, обчислювальна потужність яких може допомогти дослідникам порівнювати молекули більшого масштабу, ніж в даний час можливо з використанням класичних комп'ютерів.

Управління охороною здоров'я. Навчання з підкріпленням також використовувався для розробки адаптивних і персоналізованих втручань для поточного управління здоров'ям, включаючи режими фізичних вправ і контролю ваги для пацієнтів з ожирінням або діабетом. ШІ вже зарекомендував себе як цінний інструмент для заохочення залученості пацієнтів і прихильності програмам управління охороною здоров'я [9]-[13].

1.3.2 Проблеми використання підходу навчання з підкріпленням в медицині

Незважаючи на свій майбутній успіх, методи навчання з підкріпленням в медицині все ще стикається з кількома значними, але переборними проблемами, перш ніж він коли-небудь побачить великомасштабне клінічне впровадження. Перенесення агента навчання з підкріпленням з навчального або модельованого середовища в реальне може бути складним, по-перше, і оскільки єдиний зворотний зв'язок, який агент може зрозуміти, – це винагороди та штрафи, оновлення або налаштування алгоритму можуть виявитися проблематичними [9]-[13].

Є також приклади, коли агенти навчання з підкріпленням навчаються грати у свої системи, включаючи це відео OpenAI, в якому агент з'ясував, як збирати нагороди, не закінчуючи гонку [9]-[13].

Але є й інші проблеми, в тому числі: брак даних. Незважаючи на те, що агенти навчання з підкріпленням найкраще навчаються на роботі (так би

мовити), дослідник глибокого навчання Ісаак Годфрід зазначає, що в більшості випадків використання реальних пацієнтів для навчання алгоритмам навчання з підкріпленням не є самим етичним підходом. Це означає, що вони повинні тренуватися в імітованих середовищах на основі історичних даних спостережень за конкретними методами лікування, які часто важко отримати з різних причин, включаючи міркування особистої медичної інформації [9]-[13].

Часткова спостережуваність. У той час як агенти навчання з підкріпленням часто можуть враховувати повний стан модельованих середовищ, людське тіло набагато складніше і швидше, ніж навіть найбільш докладні моделі або набори історичних даних. Ця часткова спостережуваність у вигляді показників артеріального тиску, температури та інших показників означає, що агенти в клінічних умовах часто не мають повного уявлення про стан навколишнього середовища [9]-[13].

Формулювання і конфігурація винагороди. Хоча агенти для охорони здоров'я розробляються з урахуванням довгострокового успіху, це легше сказати, ніж зробити, при розробці політики винагороди, яка врівноважує довгострокові вигоди з короткостроковими (але іноді оманливими) поліпшеннями [9]-[13].

Хоча до того, як використання навчання з підкріпленням в охороні здоров'я наблизиться до досконалості, залишається ще багато проблем, останнім часом був досягнутий прогрес у кількох напрямках, особливо в більш ефективній і точній розробці динамічних режимів лікування для хронічних та інших станів. Оскільки політика винагороди та надання більшої кількості даних для створення середовища для агентів, безсумнівно, ми побачимо подальші поліпшення-і, в кінцевому рахунку, кращі результати для пацієнтів та більш ефективні операції для медичних працівників [9]-[13].

1.4 Дуельні Подвійні Глибокі Q-Мережі

ШІ змінює підхід до охорони здоров'я та біомедичних досліджень. Навчання з підкріпленням і методи глибокого навчання, зокрема, надають способи безпосередньо допомогти лікарям приймати більш ефективні рішення за допомогою чітких рекомендацій щодо лікування [13]. Недавні застосування глибокого навчання з підкріпленням (ГНП) для підтримки клінічних рішень включають оцінку стратегій ведення сепсису, контроль механічної вентиляції і вибір терапії вірусу імунодефіциту людини. Однак якість цих стратегій, пропонуваних ГНП, важко визначити: стратегії лікування, як правило, вивчаються на основі ретроспективних даних без доступу до не спостережуваних контрфактичних даних, що відображає те, що сталося б, якби клініцисти слідували стратегії ГНП. На відміну від стратегій ГНП для Atari8 або інших ігор, які можна оцінити, просто використовуючи їх для гри, тестування стратегії ГНП в галузі охорони здоров'я за допомогою рандомізованого дослідження може бути непомірно дорогим і неетичним [13], [14].

Таким чином, вкрай важливо, щоб було знайдено способи оцінки стратегій, заснованих на ГНП, до початку експериментів. Однією з конкретних осей оцінки є надійність. Методи ГНП пропонують безліч варіантів, і якщо політика обробки вихідних даних дуже чутлива до будь-якого вибору, це може означати, що [13], [14]:

- або дійсно важливо зробити правильний вибір;
- або слід бути обережніше, щодо довіри до політики виведення.

Тому що здається невеликим і, можливо, незначним зміна може мати великий вплив на результати. Навпаки, якщо політика виведення стійка до рішень аналізу, то будь-які помилки, швидше за все, викликані традиційними джерелами упередженості в спостережних дослідженнях (такими як невідстежуване змішання), які можуть бути з більшою готовністю розглянуті

експертами в предметній області, що оцінюють достовірність і застосовність результатів ГНП [13], [14].

Саме тому зусередимося застосуємо підхід ГНП, що поєднує переваги звичайного навчання з підкріпленням та складних топологій ШНМ: Дуельних Подвійних Глибоких Q-Мереж (ДПГQ-М) [13], [14].

1.5 Постановка завдання

Маючи набір даних $DataSet = \{x_1, x_2, \dots, x_n, y\}$, що складається із характеристик хворого $\{x_1, x_2, \dots, x_n\}$ та діагностичного рішення $\{y\}$, використовуючи дві окремих ШНМ з визначеними метапараметрами: $NN^1 = \{Neurons_i^1, Neurons_h^1, Neurons_o^1, Connections_n^1, Weights_n^1\}$ та $NN^2 = \{Neurons_i^2, Neurons_h^2, Neurons_o^2, Connections_n^2, Weights_n^2\}$, де NN^1 є визначальним рішенням, а NN^2 регулює нагороду під час навчання отримати рішення NN^1 , що характеризується найбільшою точністю роботи: $Error_{NN^1} \rightarrow 0$.

1.6 Висновки за розділом 1

Під час роботи над першим розділом було розглянуто проблемну область медичного діагностування. В результаті проведеного аналізу було з'ясовано, що одним з найбільш актуальних підходів до синтезу моделей для медичного діагностування є використання методів МН, а саме ГНП. Використання таких підходів дозволить підвищити якість фіналізуючих рішень та усунути недоліки використання класичних методів МН.

2 МАТЕРІАЛИ ТА МЕТОДИ

В першому розділі було проаналізовано сучасне становище у медичного діагностування. Було проаналізовано сучасні практики застосування МН та засобів ШІ для вирішення актуальних проблем медичного діагностування. Попередній аналіз довів, що МН, а саме підходи ГНП є перспективною основою для підвищення точності методів медичного діагностування.

Тому актуальним є дослідження та реалізація нових методів ГНП, що поєднують переваги звичайного навчання з підкріпленням та глибокого навчання. Проте для цього слід визначити, які саме методи та засади будуть використовуватися.

Другий розділ буде присвячено дослідженню методу ДПГQ-М, (сучасного методу ГНП, який призвів до багатьох успіхів у масштабуванні навчання з підкріпленням для вирішення складних послідовних завдань прийняття рішень), його базовій реалізації, можливих модифікацій та особливостям застосування для медичного діагностування.

Розділ буде організовано наступним чином: на початку, буде розглянуто роботу в області ГНП із введенням спеціалізованих позначень і термінології; потім буде детально описано компоненти базової реалізації пропонованого підходу використання ДПГQ-М. Для окремих компонентів, які досліджуються в аналізі чутливості, буде обговорено чому різні специфікації можуть бути кращими і як слід оцінювати чутливість до альтернативних специфікацій.

2.1 Процес прийняття рішень Маркова і Q-learning

Процес прийняття рішень Маркова (ПРМ) використовується для моделювання середовища і траєкторій пацієнта, який складається з [9-12]:

- набору станів S , плюс розподілу початкових станів $p(s_0)$;

- набору дій A ;
- динаміки переходів $T(s_{t+1} | s_t, a_t)$, яка відображає пару дій стану в момент часу t на розподіл станів в момент часу $t + 1$;
- функція негайної/миттєвої винагороди $r_t = R(s_t, a_t, s_{t+1})$;
- коефіцієнт дисконтування $\gamma \in [0, 1]$, де нижчі значення роблять більший акцент на негайній винагороді.

Кожне впровадження політики накопичує винагороди від навколишнього середовища, в результаті чого повертається $R = \sum_{t=0}^{T-1} \gamma^t r_t + 1$.

Мета ГНП – знайти оптимальну політику π^* , яка забезпечує максимальну очікувану віддачу від усіх станів. $\pi^* = \arg \max_{\pi} E[R | \pi]$. Щоб знайти π^* , одним з алгоритмів навчання з підкріпленням є Q-learning. Основна ідея оцінки політики полягає в тому, щоб використовувати вивчення тимчасової різниці (TR) протягом ітерації політики, щоб мінімізувати помилку TR [9]-[12].

$$Q^{\pi}(s, a) \leftarrow Q^{\pi} + \alpha(r + \gamma + Q^{\pi}(s', a') - Q^{\pi}(s, a)), \quad (2.1)$$

де Q – це абстрактне значення задане на початку стану s ;

s – стан агента;

a – дія агента;

α – можливий вихід, отриманий агентом після дії a у стані s ;

r – винагорода для агента після виконання дії та досягнення нового стану;

γ – коефіцієнт дисконтування;

π – політика поведінки агента.

Більш формально Q-learning спрямоване на апроксимацію оптимальної функції значення дії з урахуванням спостережуваного стану s і дії a в момент часу t . Майбутня винагорода ГНП дисконтується на кожному часовому кроці t з постійним коефіцієнтом [9]-[12].

$$Q^*(s, a) = E[r_{t+\gamma} \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \dots | s_t, a_t = a, \pi], \quad (2.2)$$

де Q – це абстрактне значення задане на початку стану s ;

s – стан агента;

a – дія агента;

r – винагорода для агента після виконання дії та досягнення нового стану;

γ – коефіцієнт дисконтування;

t – момент поточного часу;

π – політика поведінки агента.

2.1.1 Метод Policy Gradient

Найбільш поширений і досліджений метод навчання з підкріпленням – Policy Gradient (PG). Популярність цього методу пояснюється теоретично підкріпленими правилами оптимізації очікуваного нагородження [9]-[12]:

- чітка політика;
- прозорі правила.

У загальному випадку метод PG буде полягати у виконанні 4 основних кроків [9]-[12]:

1) прогнати N сценаріїв – τ_i зі стратегією (політикою) $\pi_\theta(a | s)$. При цьому τ_i – це деякий сценарій, тобто послідовність із станів агента (s_i) і дій виконаних у цих станах (a_i): $\tau = (s_1, a_1; s_2, a_2; s_3, a_3; \dots, s_n, a_n)$, а поведінка агента (подальші стани і дії) визначаються його стохастичною стратегією: $\pi_\theta(a_n | s_n)$;

2) розрахувати середнє арифметичне

$$\nabla_\theta J(\theta) \leftarrow \frac{1}{N} \sum_{i=1}^N \left(\sum_{t=1}^{T^i} \nabla_\theta \log \pi_\theta(a_t^i | s_t^i) \right) \left(\sum_{t=1}^{T^i} r(a_t^i | s_t^i) \right), \quad \text{де } J(\theta) \text{ – це функція}$$

максимізованого математичного очікування суми отриманих вигравів агента

θ , а $\nabla_{\theta} J(\theta)$ – це градієнт цієї функції. Тоді, $r(a_t^i | s_t^i)$ – це винагорода, отримана від дії a_t^i в стані s_t^i на кроці T , на якому стався перехід в термінальний стан;

$$3) \theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta);$$

4) якщо не зійшлися до екстремуму, повторити з пункту 1.

2.1.2 Метод Q-learning

Q-learning це метод, який вивчає значення з таблиці, яка вимірює, наскільки добре буде виконати певну дію в будь-якому стані (можна виміряти це за допомогою простого скалярного значення, тому чим більше значення, тим краще дію). Значення, що зберігаються в цій таблиці, називаються Q-values [9]-[12]. Це оцінки суми майбутніх нагород. Іншими словами, вони оцінюють, скільки ще винагороди ми можемо отримати до кінця гри, перебуваючи в стані (s_i) і виконуючи дію (a_i). Цей метод дозволяє на кожному кроці отримувати більше інформації про середовище. Ця інформація використовується для оновлення значень у таблиці [9]-[12].

Основная концепция Q-learning, базируется на уравнении Беллмана:

$$Q(s, a) = r + \gamma \max_{a'} Q(s', a'), \quad (2.3)$$

Рівняння стверджує, що значення Q для певної пари стан-дія має бути нагородою, отриманою при переході в новий стан (шляхом виконання цієї дії), доданої до значення найкращої дії в наступному стані. А для вирішення конфлікту, коли працює гіпотеза, що отримання нагороди прямо зараз більш цінно, ніж отримання нагороди в майбутньому, використовується це число від 0 до 1 (зазвичай від 0,9 до 0,99), яке множиться на нагороду в майбутньому, знецінюючи майбутні нагороди [9]-[12].

2.2 ДПГQ-М із використанням пріоритетної передачі досвіду

ДПГQ-М це найбільш сучасний підхід до синтезу нейромоделей на основі принципів навчання з підкріпленням. Для апроксимації оптимальної функції значення дії (a_i) ми можемо використовувати глибоку Q-мережу: $Q(s, a, \theta)$ з параметром θ . Для оцінки цієї мережі, оптимізуємо наступної послідовності випадання функцій на ітерації:

$$L_i(\theta_i) = E_{s,a,r,s'} \left[\left(y_i^{DQN} - Q(s, a, \theta_i) \right)^2 \right]; \quad y_i^{DQN} = r + \gamma \max_{a'} Q(s', a', \theta'),$$

оновлення параметрів градієнта спуску така, що

$$\nabla_{\theta_i} L_i(\theta_i) = E_{s,a,r,s'} \left[y_i^{DQN} - Q(s, a, \theta_i) \nabla_{\theta_i} Q(s, a, \theta_i) \right] \quad [13]-[15].$$

ДПГQ-М – це особливий сучасний метод ГНМ, що складається з окремих архітектур – дуелянтів, які поділяють потоки винагород та політик у глибоких Q-мережах, щоб визначити значення наступного стану. Пріоритетне відтворення досвіду, тобто вибірка міні-пакетів досвіду, які мають великий очікуваний вплив на навчання, ще більше підвищує ефективність [13]-[15].

Спочатку слід описати базову реалізацію дуельного ДПГQ-М з використанням пріоритетної передачі досвіду (ППД). Така реалізація поєднує в собі елементи з перспективних напрямків вивчення ГНП. При описі базової лінії реалізації також слід проілюструвати множину компонентів, що беруть участь у налагодженні специфікації ДПГQ-М. Під час аналізу чутливості методу вони будуть систематично змінюватися для налаштування порівняння з базовою моделлю виконання [13]-[15].

2.2.1 Дискретизація часу

На початку роботи слід розділити вхідні дані дата сету $DataSet = \{x_1, x_2, \dots, x_n, y\}$ на одногодинні осередки. Щоб уникнути

неправильного коригування коваріативних значень (коваріацій). Це необхідно саме для медичних даних, адже достатньо часто такі дані мають певне дублювання через повторне вимірювання та оновлення вибірок вже після прийняття лікувальних заходів на пацієнтах. Таким чином виконується процедура повторної прив'язки часу. Якщо дія обробки відбулася в межах існуючого тимчасового інтервалу, вимірювання коваріації, виконані після дії обробки в цьому інтервалі, були перенесені в наступний інтервал, і час дії обробки стало новою кінцевою точкою тимчасового інтервалу [13]-[15].

Однак, слід зазначити, що повторне включення часу рідко зустрічається в літературі з ГНП, але такі дії необхідні, щоб уникнути коригування змінних після лікування.

2.2.2 Стиснення вхідних даних

В роботі [14] запропоновано кодування медичної інформації про пацієнтів з використанням рекурентного автокодування на основі Довгої короткочасної пам'яті (LSTM), що представляє сукупну історію для кожного пацієнта. LSTM може підсумувати послідовні дані через структуру кодера в вектор фіксованої довжини, а потім відновлювати їх у вихідну послідовну форму через структуру декодера. Узагальнена інформація може бути використана для представлення характеристик часових рядів. Моделі LSTM-RNN можуть запобігти зниженню або вибуху градієнта і зазвичай використовуються для захоплення довгострокових структур послідовності [13]-[15].

2.2.3 Визначення дії та дискретизація лікування

Використовуючи пропозиції [14] та [15], слід зосередитися на внутрішніх процесах та показниках пацієнтів в якості дії для ПРМ. Будемо обчислювати погодинну ставку лікування як дію і підсумовуємо ставку, коли

трапляються події лікування, що перекривають одна одну або однакового типу. Погодинна ставка кожної процедури ділиться на 5 осередків, визначених квантилями відповідно до поточної практики лікаря. Відповідно, для медичного втручання визначено простір дій 5 на 5. Дія (0,0) означає, що лікування не проводиться; в той час як індекс (4,4) являє собою верхні квантильні показники внутрішніх процесів.

2.2.4 Формулювання винагороди

Продовжуючи напрацювання [14] при визначенні винагороди в момент часу t як зміни негативних логарифмічних коефіцієнтів 30-денного погіршення стану між t і $t+1$ відповідно до прогностичної моделі для 30-денного погіршення стану. Ймовірність погіршення стану будемо оцінювати за допомогою 2-шарової ШНМ з 50 і 30 прихованими блоками з регуляризацією L1 з урахуванням повторного вбудовування стислої історії у відповідний час. Нехай $f(o)$ – ймовірність погіршення стану при даних спостереженнях в поточний момент часу o , а $f(o')$ – ймовірність погіршення стану при даних спостереженнях на наступному часовому кроці. Тоді винагорода визначається, як:

$$r(o, a, o') = -\log \frac{f(o')}{1 - f(o')} + \log \frac{f(o)}{1 - f(o)}. \quad (2.4)$$

2.2.5 Топологія ДПГQ-М

Переглядаючи концепції [15], було вирішено, що фінальна дуельна ДПГQ-М з архітектурою ППД має два прихованих шару розміром 128, з використанням пакетної нормалізації після кожної, функцій активації Leaky-ReLU з витоком даних, поділу на потоки переваг і створення цінності однакового розміру і проекції на простір дій шляхом об'єднання цих двох

потоків. Архітектура ДПГQ-M поділяє функцію цінності V на цінність основного фізіологічного стану пацієнта, що називається потоком створення цінності – *Value*, і цінність даного лікування, званого потоком переваг – *Advantage*.

2.3 Аналіз чутливості запропонованого методу

У цьому розділі буде описано, як змінюється базова реалізація ДПГQ-M, описана в попередньому підрозділі під час аналізу чутливості при зміні метабараметрів. Для кожного компонента, який буде варіюватися, будуть вказуватися розглянуті альтернативи, пояснення, чому вибір може бути важливим, а також пояснення, чому кожна альтернатива може вважатися розумною. Мета цього аналізу чутливості не полягає в тому, щоб визначити, які варіанти є найкращими, оскільки для такого визначення немає ніякої достовірної інформації. Скоріше, це означає розуміння надійності вивченої політики лікування щодо апріорі прийнятих альтернатив. Зокрема, досліджується вплив на засвоєні стратегії: включення історії лікування в визначення стану; зміна розміру тимчасового інтервалу; зміна специфікації винагороди; вказівка різних моделей повторюваного впровадження [13]-[15].

2.3.1 Включення історії попереднього лікування

Одне з рішень полягає в тому, чи слід включати історію попередніх процедур в представлення стану пацієнта. Кілька попередніх робіт [13]-[15] цього не робили, але, враховуючи припущення Маркова, на якому базується метод глибоких Q-мереж (DQN), це тотожно припущенню, що минулі методи лікування не можуть вплинути на майбутні результати за допомогою шляхів, які не проходять через виміряні коваріати, включені в стан. Це припущення, як правило, буде помилковим. Наприклад, вазопресори (судинозвужуючі препарати) мають потенційно серйозні довгострокові серцево-судинні

побічні ефекти, але додатковий ризик після введення великої кількості вазопресорів не буде відображений у короткострокових змінах у вимірних коваріатах пацієнтів. Таким чином, під час аналізу чутливості будуть порівнюватися вивчені політики з реалізацій DQN із зведеннями стану, які включають і виключають історію лікування. У цьому аналізі буде розглядатися сукупна доза всього попереднього лікування до $t-1$ в якості показника історії лікування при t [13]-[15].

2.3.2 Тривалість тимчасових бункерів

При застосуванні методу дискретизації часу при навчанні з підкріпленням до даних з виконуваними діями і вимірами, записаними у вигляді часового ряду, неминуче виникає рішення про реалізацію, яке полягає в тому, як розбити час на дискретні фрагменти, в яких можна визначити стан пацієнта. З дискретними даними коротші часові комірки будуть кращі з двох причин. По-перше, стан на кожному тимчасовому кроці відображає стан пацієнта ближче до того моменту, коли було прийнято рішення про лікувальну дію на цьому тимчасовому кроці. Це покращує коригування плутанини, оскільки стани більшою мірою відображають інформацію, яка фактично вплинула на рішення про лікування. По-друге, більша кількість тимчасових кроків дозволяє розробляти більш гнучкі і динамічні стратегії навчання, які більшою мірою реагують на зміни в стані пацієнта. Однак при обмежених даних здатність вивчати більш гнучкі стратегії з більш короткими часовими інтервалами може бути згубною, приводячи до нестабільності передбачуваної оптимальної політики [13]-[15].

2.3.3 Горизонт винагород

Ключовим рішенням є визначення функції винагороди. В ідеалі функція винагороди підсумовує всю довгострокову корисність перебування,

оскільки це те, що дійсно має бути оптимізовано. Однак з міркувань практичності дослідники часто вибирають короткострокові винагороди, вимірювані на кожному часовому кроці. Коли винагорода є короткостроковою, з'являється більше сигналу в тому, що легше оцінити зв'язки між винагородою і діями. Якщо метод ГНП дійсно надійний, вивчені політики будуть в цілому схожі, незалежно від того, чи було вирішено оптимізувати справжню винагороду у вигляді відсотків або більш короткі терміни. Щоб дослідити вплив використання довгострокової винагороди на короткострокову винагороду, також слід порівнювати функції винагороди, які представляли собою зважену суміш довгострокової та середньострокової інформації про результати для ряду ваг [13]-[15].

Визначимо винагороду функції корисності винагороди U наступним чином. Нехай M – найгірший можливий бал погіршення стану пацієнта. Нехай Y буде помічений ступінь погіршення в кінці перебування. Нехай $S=1$, якщо пацієнт прожив більше 1 року після надходження, 0 в іншому випадку. Нехай H – таймер, що запустився після надходження. Нехай C – константа, яка контролює відносну вагу, присвоєну балу погіршення стану в кінці перебування і виживання. Визначемо $r'(C)$ як [13]-[15]:

$$\text{if } H \geq 24 \cdot 365 \text{ then } U = \log\left(1 + \frac{M - Y}{C}\right), \text{ else } U = \log\left(\frac{H}{24 \cdot 365} + 1\right). \quad (2.5)$$

Для великих значень C час виживання – це все, що має значення. При низьких значеннях C стан пацієнта в кінці перебування має велике значення для пацієнтів, які виживають більше 1 року. Для всіх значень C винагорода є середньо- і довгостроковими (оскільки вони засновані на стані пацієнта в кінці перебування або пізніше, на відміну від наступного тимчасового інтервалу), але різні рівні C відображають різну суб'єктивну пріоритетність результатів лікування пацієнтів. Порівнюватися будуть вивчені політики

DQN з використанням короткострокової винагороди (2.4) і довгострокової винагороди (2.5) для різних значень C [13]-[15].

2.3.4 Вибір моделі кодування

Інше питання полягає в тому, як узагальнити історію хвороби пацієнта. У DQNs важливо, щоб інформація, що міститься в стані S_t в кожен момент часу t , задовольняла декільком умовам. По-перше, S_t повинен задовольняти припущенню «послідовної взаємозамінності», яке було б виконано, якби S_t містив всю відповідну інформацію про змінні, що впливають на рішення про лікування в момент часу t і пов'язаних з майбутніми винагородами, тобто S_t повинен містити достатню інформацію для коригування на змішання. Якщо послідовна взаємозамінність не вдається, то оцінки впливу дій на майбутні винагороди будуть упередженими, і, отже, оцінка оптимальної стратегії лікування також буде упередженою [13]-[15].

Для вивчення оптимальної стратегії лікування також важливо, щоб S_t містив відповідну інформацію про змінні, які є модифікаторами ефекту. Модифікатор ефекту – це змінна з властивістю, згідно з якою умовний середній вплив дії на майбутні винагороди залежить від значення змінної. Хороші правила лікування призначають лікування на основі значення модифікаторів ефекту. Модифікатори ефекту можуть бути або не бути плутанинами, які необхідно включити в модель, щоб уникнути зміщення, але можуть не бути хорошими вхідними даними для правил прийняття рішень про лікування [13]-[15].

Нарешті, DQN роблять дуже сильне марковське припущення про стани S_t – повинен бути визначений як достатньо багатий, щоб це припущення Маркова могло приблизно виконуватися. Таким чином, щоб забезпечити реалістичні довгострокові тимчасові залежності, стану в кожен

момент часу повинні містити докладні зведення історії хвороби пацієнта. Без апріорного знання того, які саме аспекти історії хвороби пацієнта слід зберегти для коригування з урахуванням плутанини, модифікації ефекту моделі і задоволення припущення Маркова, розумною стратегією є визначення станів пацієнта як вкладень, що генеруються [13]-[15].

Однак вкладення RNN не оптимізовані для збереження типів інформації, спеціально необхідних для утримання в станах DQN. Різні варіанти методу вбудовування чорного ящика можуть генерувати стани, які в різному ступені задовольняють вимогам DQN, і створювати різні вивчені політики без принципового способу вибору між ними [13]-[15].

При аналізі чутливості будуть порівнюватися дві загальні моделі кодування LSTM і вентильний рекурентний вузол (GRU). Топології виконують зіставні функції по цілому ряду задач. Кожна складається з двох прихованих шарів по 128 прихованих одиниць і навчається міні-партіями по 128 одиниць і оптимізатором Adam протягом 200 епох або до конвергенції. Немає підстав апріорі вважати, що будь-який з варіантів призведе до більш відповідних вкладень, ніж інший, і сенс їх порівняння полягає в тому, щоб визначити, чи є рішення важливим [13]-[15].

2.3.5 Випадкові перезапуски

Нарешті, рішення ДПГQ-М є невипуклою оптимізацією, і тому для пошуку хорошого локального оптимуму часто використовуються випадкові перезапуски. Як повідомляється [15] широкий спектр результатів може бути отриманий за допомогою одного і того ж алгоритму ГНП в залежності від випадкової ініціалізації ваг мережі. Щоб простежити вплив ініціалізації випадкової ваги в наборі даних, слід повторно підганяти базову модель, використовуючи різні початкові значення. У той час як в якості рішення зазвичай просто використовується кращий з випадкових перезапусків, висока варіативність випадкових перезапусків може означати, що відтворення

результату буде більш складним, оскільки проблема має безліч різноманітних локальних оптимумів [13]-[15].

2.4 Висновки за розділом 2

У другому розділі проаналізовано базову реалізацію ДПГQ-M та модифікації мета параметрів для визначення рівня чутливості.

Так ДПГQ-M було обрано для синтезу моделей в медичному діагностуванні. Такий підхід реалізує усі переваги ГНП. Проте для аналізу чутливості методу було обрано мета параметри, що під час виконання можливо змінювати. Це необхідно для побудови найбільш адаптивної моделі.

Так, дискретизація часу у вхідних даних рідко зустрічається в матеріалах присвячених ГНП, але такий механізм необхідний, щоб уникнути коригування змінних після лікування. Було спроектовано механізм визначення винагород під час синтезу – горизонт винагород. Тепер винагороди є середньо- і довгостроковими (оскільки вони засновані на стані пацієнта в кінці перебування на лікуванні або пізніше, на відміну від наступного тимчасового інтервалу). Такий підхід допомагає визначити різну суб'єктивну пріоритетність результатів лікування пацієнтів. Щоб простежити вплив ініціалізації випадкових ваг в наборі даних, базова модель буде повторно підганятися, використовуючи різні початкові значення. У той час як в якості рішення зазвичай просто використовується кращий з випадкових перезапусків, висока варіативність випадкових перезапусків може означати, що відтворення результату буде більш складним, оскільки проблема має безліч різноманітних локальних оптимумів.

3 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

Другий розділ було присвячено дослідженню методу ДПГQ-М: його базовій реалізації, можливим модифікаціям та особливостям застосування для медичного діагностування. На початку було розглянуто роботу в області ГНП, потім було детально описано компоненти базової реалізації пропонуваного підходу використання ДПГQ-М. Для окремих компонентів, які досліджуються в аналізі чутливості, було обговорено різні специфікації, що можуть бути кращими і як слід дозволять оцінювати чутливість до альтернативних специфікацій.

Третій розділ буде присвячено висвітленню питань з проектування архітектури та реалізації програмного застосунку (ПЗ) для дослідження запропонованого методу та можливих модифікацій. Особливу увагу буде присвячено вибору технічних засобів: мови програмування, фреймворку МН, тощо.

3.1 Опис основних вимог до програми

Об'єктом розробки є ПЗ для реалізації методу ДПГQ-М та можливих модифікацій для аналізу чутливості методу. Мета роботи – розробити програмну систему, яка в повній мірі дозволить застосувати та дослідити роботу методу, проаналізувати вплив модифікацій на чутливість методу.

Необхідно реалізувати 3-основні модуля для подібної системи. Перший модуль представляє клієнтську частину, де від користувача приймаються вхідні дані – навчальна вибірка зібрана від пацієнтів. Другий модуль представляє сервер для обробки даних – на цьому рівні буде використовуватися фреймворк МН. Третій модуль забезпечує представлення отриманих результатів та збереження синтезованої моделі для подальшого використання.

Таким чином серед функціональних вимог виділяються:

- завантаження вхідних даних (для навчання та тестування);
- обробка даних;
- синтез моделі;
- дослідження якісних результатів;
- збереження синтезованої моделі.

Для забезпечення надійності програми, вона має базуватися на комп'ютері. На протязі усього сеансу роботи застосунок не повинен збоїти або зависати. Після оновлення інформаційної бази або функціоналу, застосунок має перезапускатися.

Більш того до інформаційної бази, яку використовує застосунок, висуваються наступні умови:

- вносити зміни до бази може тільки адміністратор;
- оновлення бази слід виконувати дотримуючись правил зберігання.

3.2 Вибір мови програмування

Python – це високорівневий, універсальний і дуже популярна мова програмування. Мова програмування Python (остання версія Python 3) використовується в веб-розробці, додатках машинного навчання, а також у всіх передових технологіях в індустрії програмного забезпечення. Мова програмування Python дуже добре підходить для початківців, а також для досвідчених програмістів з іншими мовами програмування, такими як C++ і Java [16]-[18].

Python в даний час є найбільш широко використовуваним багатоцільовим мовою програмування високого рівня.

Python дозволяє програмувати в об'єктно-орієнтованій і процедурній парадигмах [16]-[18].

Програми на Python зазвичай менше, ніж на інших мовах програмування, таких як Java. Програмістам доводиться друкувати відносно менше, а вимоги мови до відступів роблять їх постійно читаються.

Мова Python використовується майже всіма компаніями – технологічними гігантами, такими як Google, Amazon, Facebook, Instagram, Dropbox, Uber, і т. д. [16]-[18].

Найбільшою силою Python є величезна колекція стандартної бібліотеки, яку можна використовувати для наступних цілей [16]-[18]:

- машинне навчання;
- додатки з графічним інтерфейсом (такі як Kivy, Tkinter, PyQt і т. д.);
- веб-фреймворки, такі як Django (використовуються YouTube, Instagram, Dropbox);
- обробка зображень (наприклад, OpenCV,);
- очищення веб-сторінок (наприклад, Scrappy, BeautifulSoup);
- тестові платформи;
- мультимедіа;
- наукові обчислення;
- обробка тексту та багато іншого.

Проведемо порівняння мов програмування для реалізації ПЗ, результати порівняння представимо в табличній формі (табл. 3.1). У якості критеріїв будемо використовувати наступні [16]-[18]:

- швидкість роботи – умовна швидкість роботи ПЗ, розроблених на основі мови програмування;
- фреймворки для методів МН – наявність, швидкість оновлення та підтримка фреймворків для реалізації методів МН;
- спільнота – обсяг активних розробників, що використовують мову програмування та кількість інформаційних ресурсів.

Таблиця 3.1 – Порівняння мов програмування

Критерій	Python	Java	Scala	R
Швидкість роботи	$\pm$	+	$\pm$	$\pm$
Бібліотеки	++	$\pm$	$\pm$	+
Фреймворки для методів МН	++	+	+	+
Спільнота	++	+	$\pm$	+

За результатами порівняння можна зробити висновок, що найбільш оптимальним та простим буде використання Python у якості мови програмування для розробки ПЗ. Так застосунки розроблені на Python не відрізняються найшвидшою роботою, проте наявність безлічі відкритих бібліотек для МН та відповідних фреймворків значно полегшує та пришвидшує роботу з розробки.

3.3 Вибір фреймворку

TensorFlow – це фреймворк для МН від Google, який призначений для проектування, створення та вивчення моделей глибокого навчання. Глибоке навчання-це область машинного навчання, алгоритми в якій були натхненні структурою і роботою мозку. Ви можете використовувати TensorFlow, щоб виробляти чисельні обчислення. Саме по собі це не здається специфічним, проте ці обчислення проводяться за допомогою data-flow графів. У цих графах вершини являють собою математичні операції, в той час як ребра являють собою дані, які зазвичай представляються у вигляді багатовимірних масивів або тензорів, які повідомляються між цими ребрами [19].

Назва «TensorFlow» походить від обчислень, які нейромережа виробляє з багатовимірними даними і тензорами. Буквально – потік тензорів. На даний момент це все, що потрібно знати про тензори, але ми повернемося до них трохи пізніше [19].

TensorFlow добре підходить для автоматизованої анотації зображень в таких системах як Deep Dream. Також з 26 жовтня 2015 року Google використовує систему RankBrain для збільшення релевантності ранжирування пошукової видачі Google. RankBrain заснований на TensorFlow [19].

TensorFlow дозволяє проводити навчання генеративно-змагальних мереж.

Інтеграція TensorFlow з Python забезпечується дистрибутивом Anaconda [19, 20].

Проведемо порівняння фреймворків МН для реалізації ПЗ, результати порівняння представимо в табличній формі (табл. 3.2). У якості критеріїв будемо використовувати наступні:

- відкритість – відкритість сирцевого програмного коду фреймворку;
- платформа – можливість використання ПЗ розробленого за допомогою фреймворку на різних операційних та апаратних платформах;
- топології ШНМ – підтримка сучасних та складних топологій ШНМ;
- паралельні обчислення – можливість використання паралельних обчислювальних ресурсів комп’ютерних систем для синтезу, навчання та тестування ШНМ;
- підтримка – розвинена спільнота та підтримка фреймворка.

Таблиця 3.2 – Порівняння мов програмування

Критерій	TensorFlow (Python)	Caffe (C++)	Microsoft Cognitive Toolkit (C++)	Theano (Python)
Відкритість	+	+	+	+
Платформа	++	++	+	++
Топології ШНМ	++	++	±	±
Паралельні обчислення	++	–	–	++
Підтримка	++	+	±	+

За результатами порівняння можна зробити висновок, що найбільш оптимальним та простим буде використання TensorFlow у якості фреймворку МН. Так застосунки розроблені з використанням TensorFlow є кроссплатформними. Більш того сам фреймворк підтримує найсучасніші та найскладніші топології ШНМ, а активна спільнота та підтримка полегшать та значно пришвидшують розробку.

3.4 Вибір середовища розробки

Anaconda – це дистрибутиви Python і R. він надає все необхідне для вирішення завдань з аналізу та обробки даних (з застосовністю до Python) [20].

Anaconda – це набір бінарних систем, що включає в себе Scipy, Numpy, Pandas і їх залежності [20]:

- Scipy – це пакет статистичного аналізу;
- Numpy – це пакет числових обчислень;
- Pandas – рівень абстракції даних для об'єднання і перетворення даних;
- Anaconda корисна тим, що об'єднує все це в єдину систему.

Двійкова система Anaconda – це інсталятор, який збирає всі пакети з залежностями всередині системи [20].

Коли починають працювати із Python, зазвичай встановлюють найновішу версію Python з останніми версіями бібліотек (пакетів), які потрібні або з якими хочеться поекспериментувати [20].

Коли ви досягнете ази Python і завантажте програми Python з GitHub, Kaggle або інших джерел. Цим додаткам можуть знадобитися інші версії бібліотек (пакетів) Python, ніж ті, які в даний час використовуєте (минулі версії пакетів або минулі версії Python) [20].

В цьому випадку Вам необхідно налаштувати різні середовища.

Крім цієї ситуації, є й інші варіанти використання, коли можуть виявитися корисними додаткові середовища [20]:

- є застосунок, який колись працював чудово. Але тепер намагалися запустити його, і воно не працює. Можливо, один з пакетів більше не сумісний з іншими частинами програми (через так звані критичні зміни). Можливе рішення полягає в тому, щоб налаштувати нове середовище для вашої програми, яка містить версію Python і пакети, повністю сумісні з застосунком;

- розробники співпрацюють з кимось ще і хочете переконатися, що застосунок працює на комп'ютері члена команди, або навпаки;

- розробники доставляють застосунок своєму клієнту і знову хочуть переконатися, що він працює на комп'ютері вашого клієнта;

- середовище складається з певної версії Python і деяких пакетів. Отже, якщо потрібно розробляти або використовувати застосунки з різними вимогами до Python або версіями пакетів, необхідно налаштувати різні середовища.

У таблиці 3.3 наведено порівняння найбільш популярних середовищ розробки для Python. У якості критеріїв будемо використовувати наступні:

- наявність безкоштовної версії – чи вільно розповсюджується середовище розробки;

- підтримка TensorFlow – чи є підтримка TensorFlow за замовчанням;

- кроссплатформність – чи є можливість працювати з середовищем розробки на різних програмних та апаратних платформах;

- швидкість роботи – загальна оцінка реагування та роботи середовища;

- підтримка та оновлення – якість та регулярність оновлення середовища розробки.

Як видно з таблиці зв'язка Anaconda найбільш оптимальний варіант середовища розробки для подальшої роботи. Додатково слід зазначити, що

саме Anaconda підлаштована під наукові обчислення, а отже легко інтегрує оновлені версії бібліотек необхідних для подальшої роботи.

Таблиця 3.3 – Порівняння середовищ розробки для Python

Критерії порівняння	Середовища розробки для Python			
	Anaconda	PyCharm	Atom	PhpStorm
Наявність безкоштовної версії	++	++	++	±
Підтримка TensorFlow	++	+	+	±
Кроссплатформність	++	++	++	–
Швидкість роботи	+	+	++	±
Підтримка та оновлення	++	±	±	++

3.5 Проектування структури програмної системи

При проектуванні структури майбутньої програми було виділено 4 рівня майбутнього ПЗ, що продемонстровано на рис. 3.1.

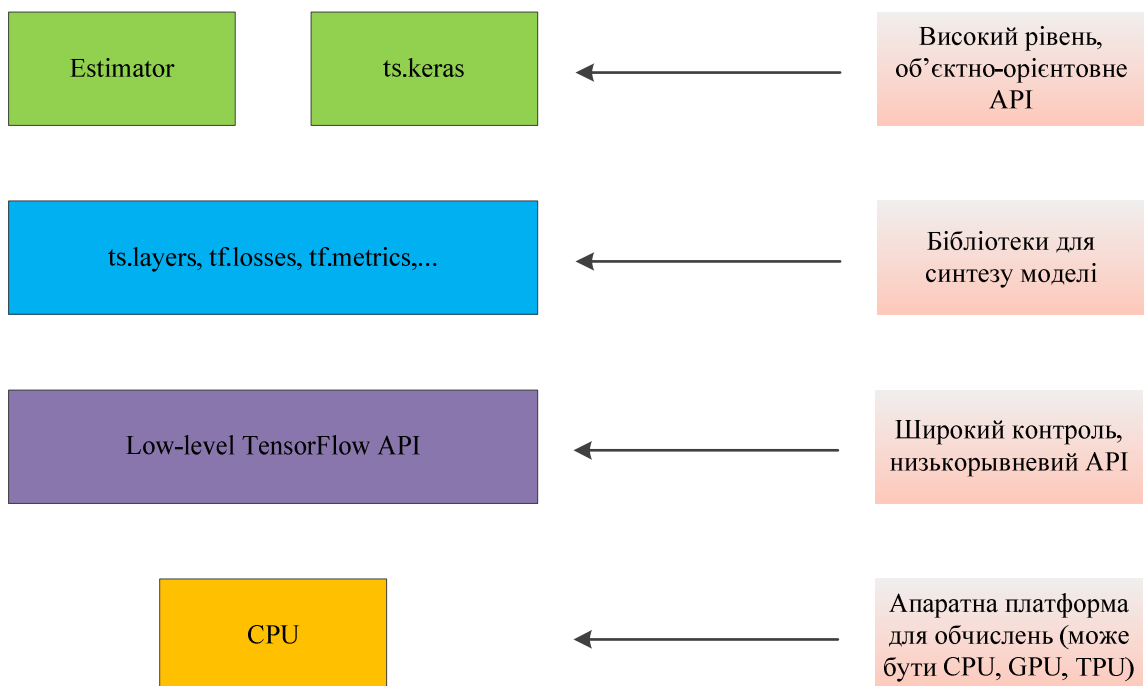


Рисунок 3.1 – Рівні розроблюваного ПЗ

Розробку ж попередньо описаних модулів можна аргументувати способом розподілення основного функціоналу ПЗ за таким принципом

гарантує формування функціонально завершених блоків програми. Такий варіант розподілення представлено на рис. 3.2.

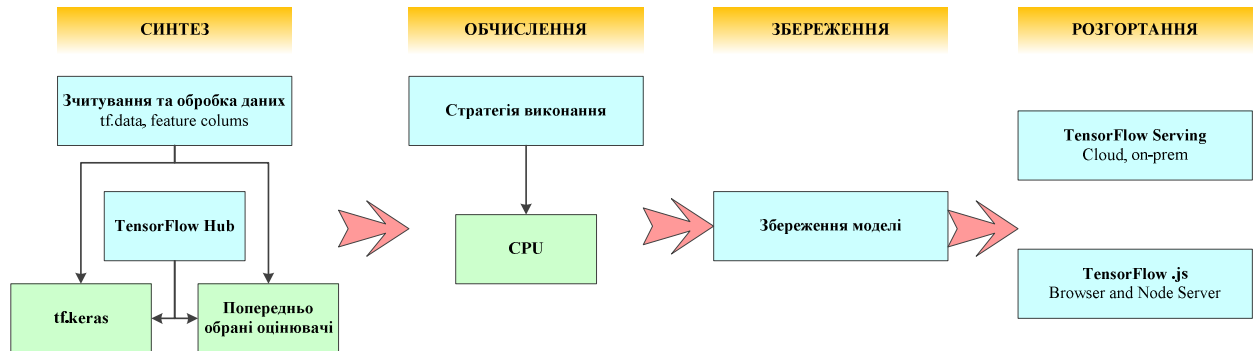


Рисунок 3.2 – Модулі розроблюваного ПЗ

3.6 Вимоги до технічних засобів

Для нормальної розробки програми, повинні виконуватися наступні умови до апаратної частини:

- наявність персонального комп'ютера (ПК), на якому буде працювати застосунок;
- процесор із тактовою частотою 3,0 GHz та більше;
- оперативна пам'ять 4,0 ГБ та більше;
- вільний простір на жорсткому диску не менше, ніж 2 ГБ (для збереження вибірок).

Вимоги до програмної частини:

- встановлений дистрибутив Python.

3.7 Висновки за розділом 3

Під час роботи над розділом було спроектовано архітектуру майбутнього ПЗ. Були спроектовані рівні та модулі майбутньої програми. Було проаналізовано найбільш популярні мови програмування та після ретельного порівняння перевага була віддана Python, а в якості фреймворку МН було обрано TensorFlow.

4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

У третьому розділі було досліджено та вирішено ряд питань з проектування архітектури та реалізації ПЗ для дослідження запропонованого методу та можливих модифікацій. Особливу увагу було присвячено вибору технічних засобів: мови програмування та фреймворку МН.

Четвертий розділ буде присвячено висвітленню основних рішень щодо реалізації ПЗ. Будуть проаналізовані розроблені модулі та загальна архітектура ПЗ.

4.1 Модульна структура ПЗ

Модульна структура ПЗ має передумову: розділена за модулями структура полегшує розуміння і спрощує процес модифікації проекту за потреби.

Для реалізації запропонованого методу та дослідження його чутливості в залежності від мета параметрів були розроблені модулі, які за функціоналом повторюють раніше спроектовані модулі ПЗ. Однак, в середині вони мають більшу роздробленість за функціоналом. Серед розроблених модулів можна виділити:

- модуль завантаження та обробки даних;
- модуль синтезу ШНМ;
- модуль збереження моделі;
- модуль розгортання моделі.

Розглянемо розроблені модулі детальніше.

4.2 Модуль завантаження та обробки даних

Модуль завантаження та обробки вхідних даних розроблено для завантаження вибірок та обробки даних отриманих від користувача. На діаграмі класів модулю, що наведено на рис. 4.1, наведено два основних класів та їх зв'язок із зовнішнім пакетом обробки даних, що підключається у вигляді бібліотеки.

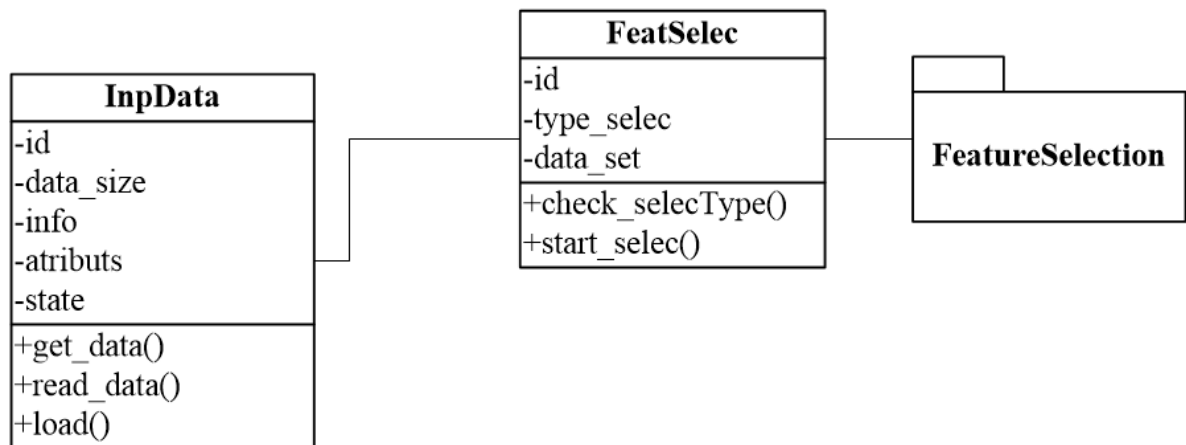


Рисунок 4.1 – Класи модулю завантаження та обробки даних

Клас «InpData» – клас, що відповідає за завантаження та зчитування вхідних даних. Серед властивостей класу можна виділити:

- `id` – службовий ідентифікатор вибірки даних;
- `data_size` – загальний обсяг вибірки даних;
- `info` – додаткова інформація про вибірку даних надана користувачем;

- `atributs` – незалежні параметри вибірки;
- `state` – залежний параметр вибірки.

Серед методів класу слід виділити:

- `get_data` – отримання ідентифікаторів вибірки даних;
- `read_data` – зчитування та буферизація даних;
- `load` – завантаження даних вибірки для подальшої обробки.

Клас «FeatSelec» – клас, що відповідає за препроцесинг вхідних даних:

- id – службовий ідентифікатор;
- type_selec – вибір типу обробки даних;
- data_set – ідентифікатор вибірки даних.

Серед методів класу слід виділити:

- check_selecType – перевірка вибору та запуск процесу обробки даних;
- start_selec – запуск процесу обробки даних.

4.3 Модуль синтезу ШНМ

Модуль синтезу ШНМ відповідає за реалізацію процесу синтезу ШНМ. На діаграмі класів модулю, що наведено на рис. 4.2, наведено основний класи модулю.

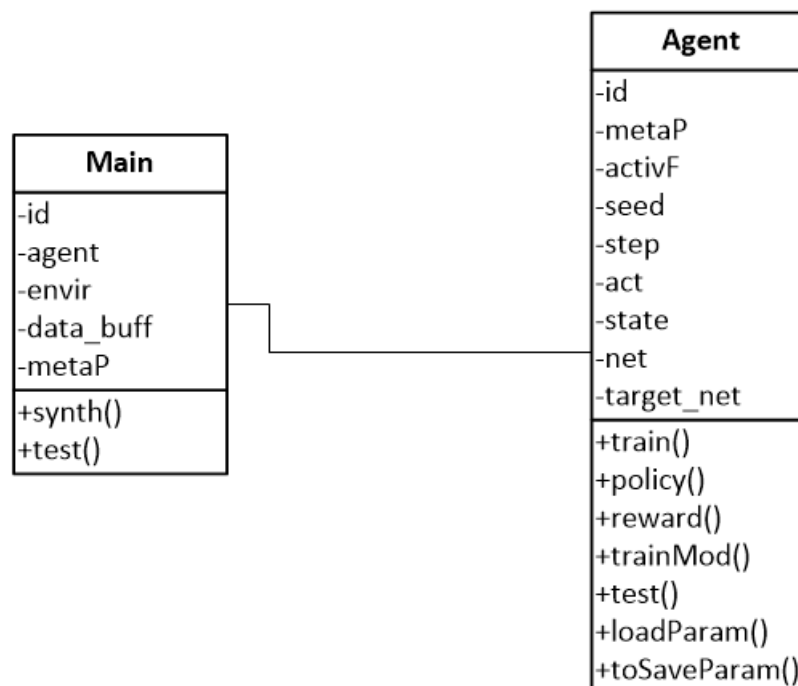


Рисунок 4.2 – Основні класи модулю

Клас «Main» – клас, що відповідає за запуск процесу навчання із заданими метапараметрами. Серед властивостей класу можна виділити:

- id – внутрішній ідентифікатор;
- agent – визначення властивостей мережі агента;
- envir – визначення властивостей мережі для дослідження середовища;
- data_buff – буфер для збереження вибірки даних
- metaP – налаштовані метапараметри для навчання.

Серед методів класу слід виділити:

- synth – запуск процесу синтезу;
- test – запуск процесу тестування.

Клас «Agent» – клас, що відповідає за запуск процесу навчання із заданими метапараметрами. Серед властивостей класу можна виділити:

- id – внутрішній ідентифікатор;
- metaP – масив для збереження метапараметрів навчання;
- activF – визначення функції активації;
- seed – розмір зерна при навчанні;
- step – розмір кроку при навчанні;
- act – збереження дії агента під час навчання;
- state – збереження стану агента під час навчання;
- net – визначення масиву параметрів мережі.

Серед методів класу слід виділити:

- train – загальний процес запуску навчання мережі;
- policy – визначення стратегії навчання на основі попередніх дій та станів;
- reward – визначення нагороди (штрафу) під час навчання;
- trainMod – запуск процесу навчання мережі-агента;
- test – запуск процесу тестування результуючого рішення;
- loadParam – завантаження визначених метапараметрів;

– toSaveParam – буферизація результуючого агента для збереження моделі.

4.4 Модуль збереження мережі

Модуль збереження мережі – це окремий модуль для налаштування властивостей запису результуючої мережі. На діаграмі класів модулю, що наведено на рис. 4.3, представлено єдиний та основний клас модулю.

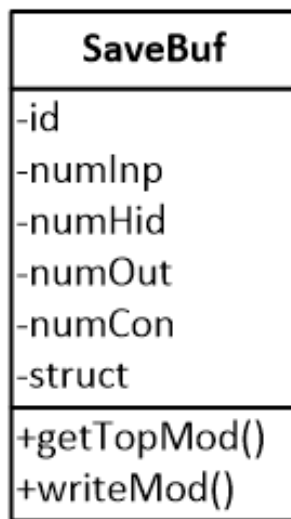


Рисунок 4.3 – Основний клас модулю

Клас «SaveBuf» – клас, що відповідає за збереження результуючої моделі. Серед властивостей класу можна виділити:

- id – внутрішній ідентифікатор;
- numInp – кількість нейронів вхідного шару;
- numHid – кількість нейронів прихованого шару;
- numOut – кількість нейронів вихідного шару;
- numCon – кількість міжнейронних зв’язків;
- struct – параметрів мережі.

Серед методів класу слід виділити:

- `getTopMod()` – отримування та зчитування структури мережі та її параметрів;
- `writeMod()` – запис структури та параметрів.

4.5 Модуль розгортання моделі

Модуль розгортання моделі – це окремий модуль для розгортання попередньо синтезованих моделей та їх дослідження. На діаграмі класів модулю, що наведено на рис. 4.4, представлено взаємодію класу модуля із пакетами фреймворку TensorFlow.

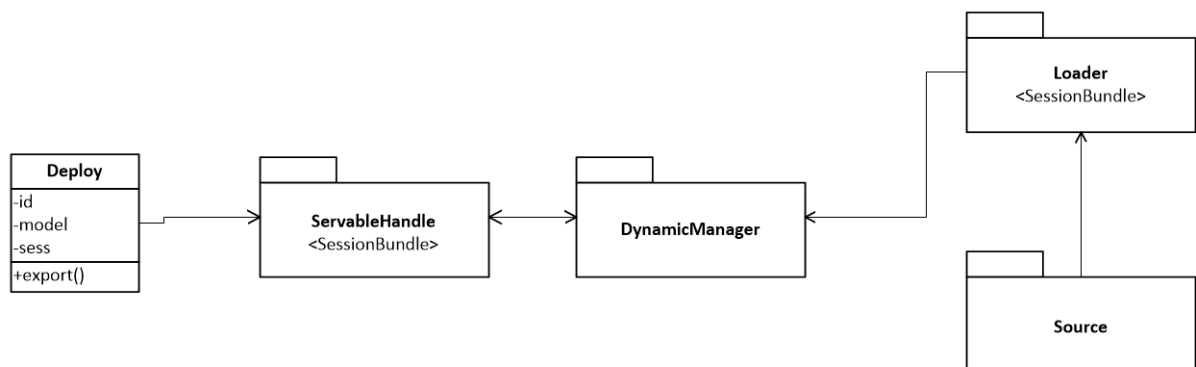


Рисунок 4.4 – Діаграма класів модулю

Tensorflow Serving – система для розгортання TensorFlow-моделей з такими можливостями як:

- автоматичний патчинг;
- гаряча заміна моделей і версіонування;
- можливість обробки паралельних запитів.

Додатковим плюсом є можливість перегнати модель з Keras в Tensorflow-модель і розгортати через Serving (якщо звичайно в Keras використовується TensorFlow бекенд).

Основною частиною Tensorflow Serving є сервер моделей.

Розглянемо схему роботи сервера моделей. Після запуску сервера моделей завантажує модель з шляху, зазначеному при запуску, і починає

слухати вказаний порт. Сервер спілкується з клієнтами через виклики віддалених процедур, використовуючи бібліотеку gRPC. Це дозволяє створити клієнтську програму на будь-якій мові, що підтримує gRPC.

Як вже було згадано раніше Tensorflow Serving підтримує гарячу заміну моделей. Сервер моделей постійно сканує вказаний при запуску шлях на наявності нових моделей і при знаходження нової версії автоматично завантажує цю версію. Це дозволяє викладати нові версії моделей без необхідності зупинки сервера моделей.

Таким чином, Tensorflow Serving має достатній функціонал для повноцінної роботи в production. Тому використання таких підходів, як створення власної обгортки над моделлю, виглядає невиправдано, тому що Tensorflow Serving пропонує ті ж можливості і навіть більше без необхідності писати і підтримувати самописні рішення.

4.6 Висновки за розділом 4

Розроблено ПЗ для реалізації синтезу діагностичних моделей на основі методу ДПГQ-М із зміною метапараметрів для дослідження чутливості моделей. Обрано механізм реалізації відповідних методів у вигляді програмних модулів та класів. Наведена архітектура ПЗ продемонструвала задовільні результати масштабування та розгортання на ПК.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

У попередніх розділах було розглянуто питання розроблення нового методу на основі стратегій ГНП. Запропонований метод ДПГQ-М має низку метапараметрів, що можуть налаштовуватися за різними сценаріями.

Цей розділ буде присвячено опису розробленого ПЗ. Буде описан метрики для оцінки якості роботи методу та модифікацій, детально означено сферу застосування, та наведено алгоритм користування. Також у цьому розділі буде розглянуто тестування та проаналізовано отримані результати.

5.1 Вхідні дані для тестування

Для тестування будуть використані дані з банку даних Medical Information Mart for Intensive Care (MIMIC-III v1.4) [21], [22]. Набір даних містить усіх пацієнтів з MIMIC-III, що відповідають критеріям Sepsis-3 за 2008-2012 рр. [21], [22]. Він включає 7956 пацієнтів з 649661 записами про клінічні події. У цьому аналізі будуть витягатися і збиратися статичні характеристики (наприклад, демографічні), історії минулого лікування і зведення погодинного спостереження (середнє, максимальне і мінімальне протягом години) всіх лабораторних показників протягом перших 72 годин перебування пацієнтів у відділенні інтенсивної терапії. Для внутрішньовенної рідини будуть витягатися зазвичай використовувані внутрішньовенні рідини для реанімації (колоїди і кристалоїди). Детальні характеристики наведені в табл. 5.1. Всі виміряні значення були стандартизовані. Набір даних був розділений з використанням 80 % для навчання і перевірки та 20% для тестування [21], [22].

Таблиця 5.1 – Детальна інформація про атрибути, включаючи життєво важливі показники, лабораторні показники та історію лікування

Демографічні ознаки	Вік, вага, зріст, етнічна приналежність
Життєві показники \ Лабораторні показники	Шкала ком Глазго, частота серцевих скорочень, температура, частота дихання, діастолічний артеріальний тиск, систолічний артеріальний тиск, середній артеріальний тиск, калій, натрій, хлорид, магній, кальцій, аніонний розрив, гемоглобін, гематокрит, тромбоцити, смуги, SpO ₂ , FiO ₂ , PaO ₂ , TotalCO ₂ , pCO ₂ , глюкоза, креатинін та інші
Лікування	Внутрішньовенна ін'єкція та застосування вазопресорів

5.2 Метрики оцінювання

Попередні роботи використовували позаполітичні оцінки, такі як зважена подвійна надійність (WDR) для оцінки якості запропонованої політики. Однак ці оцінки можуть мати високу дисперсію, а також зміщення. Замість цього пропонується порівнювати політики на основі розподілу рекомендованих ними дій. Якщо ці розподіли сильно відрізняються один від одного або від лікарів-клініцистів, то це може бути причиною для скептицизму.

Зокрема, для кожного моменту часу в історії хвороби пацієнта в наборі тестів буде обчислюватися рекомендована дія ДПГQ-M для дуелі. Розподіл дій для стратегії – це просто частота, з якою кожна з 25 комбінацій доз вазопресора і внутрішньовенної ін'єкції рекомендується для всіх людей в наборі тестів.

Надалі будуть використовуватися теплові карти, подібні до рис. 5.1, для відображення розподілу дії, де вісь y – представляє дозування внутрішньовенної ін'єкції (IV Fluid), а вісь x – являє дозування вазопресора (Vasopressor). Щільність кольору являє частоту, з якою лікувальна дія застосовується у всіх випадках в наборі тестів. Ці розподіли дій є сукупними,

оскільки підсумовуються дані по всіх часових точках і пацієнтам. Порівняння розподілу сукупних дій між різними експериментальними установками може дати уявлення про те, як різняться політики.

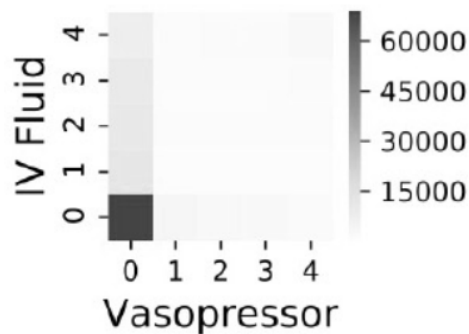


Рисунок 5.1 – Приклад теплової карти

5.3 Параметри та оптимізація

ДПГQ-M будуть навчатися на 100000 кроків (за винятком експерименту горизонт винагороди, де виконується рання зупинка на 15000 кроків, щоб запобігти надмірному налаштуванню) з розміром партії 30. Буде проведено 5 перезапусків для кожної експериментальної установки (за винятком експерименту випадкового перезапуску, де розглядаються відмінності між окремими перезапусками). З політик, отриманих в результаті 5 перезапусків, буде обиратися політика з найбільшим значенням, оціненим за допомогою зваженого подвійно надійного методу оцінки. Для моделей, навчених з довгостроковою винагородою, $r'(C)$, де оцінка WDR нездійсненна, обиралася політика з 5 перезапусків на основі значення Q .

5.4 Результати тестування

Під час тестування базові сценарії синтезу ДПГQ-M було змінено, як зазначалося вище. Отримані результати порівнювалися та вивчалися відповідно до їх розподілу. Надалі синтез ДПГQ-M скорочувався та

модифікувався, наприклад, ДПГQ-M, навчені за допомогою LSTM протягом 1-ї години, називається DQN-LSTM-1 година.

5.4.1 Вплив історії лікування

Тут порівнюються стратегії лікування, отримані за допомогою ДПГQ-M, які включають і не включають історію лікування в представлені стану пацієнта (рис. 5.2). DQN-LSTM-1hr, навчений без історії лікування, рекомендує ненульові дози вазопресорів в усі моменти часу. В цьому випадку часто рекомендуються високі дози кожного виду лікування порівняно з лікарями та агентом, який пройшов навчання, з історією лікування, включеною до визначення штату. Виключення інформації про минуле лікування збільшує частоту середньої рекомендованої дози вазопресору і внутрішньовенної ін'єкції в 1,6-1,8 рази та 1,7-3,1 рази відповідно в наборі тестів.

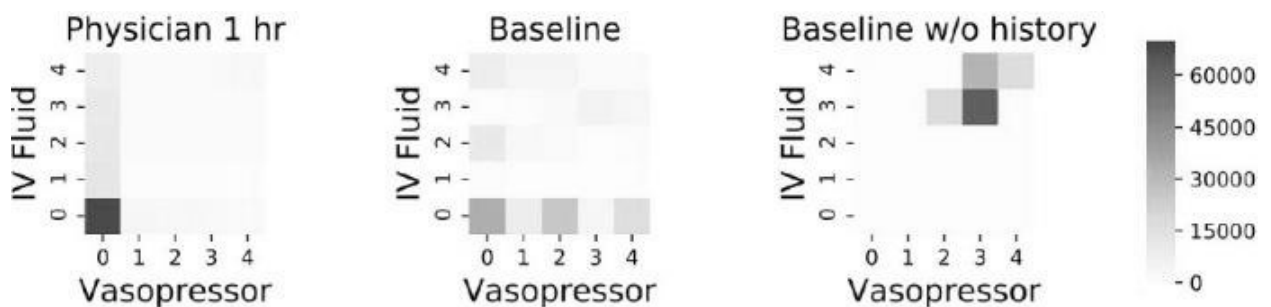


Рисунок 5.2 – Розподіл дій ДПГQ-M, навчених з/без інформації про історію лікування

Зверніть увагу, що агент, навчений без історії лікування, наполегливо призначає високу (3-й квартиль, 4-й квартиль) дозування вазопресора і внутрішньовенної ін'єкції. Лікування низькими дозами рідко призначається пацієнтам. Таким чином приходимо до висновку, що виключення історії лікування призводить до агресивної політики лікування.

5.4.2 Тривалість тимчасових бункерів

На рис. 5.3 показано, що, хоча різні способи сегментації часу не впливають на розподіл дій лікаря (за визначенням), вони мають великий вплив на розподіл дій ДПГQ-М. Проміжки по 4-години ДПГQ-М збільшили частоту ненульових доз вазопресорів на 40% і знизили загальне використання тільки внутрішньовенної ін'єкції, що є менш агресивною дією, на 34% в порівнянні з базовими налаштуваннями.

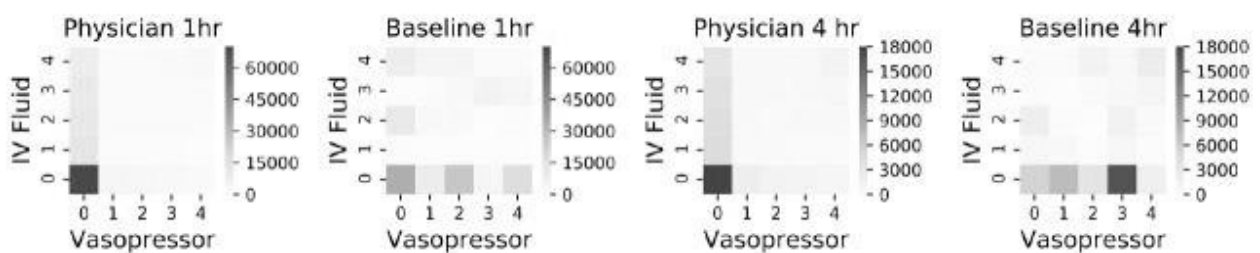


Рисунок 5.3 – Порівняння розподілу дій по 4-годинним проміжкам часу ДПГQ-М і 1-годинному проміжку часу

Більш тривалі тимчасові осередки призводять до більш агресивної політики.

5.4.3 Горизонт винагород

На рис. 5.4 ми бачимо, що довгострокові цілі призвели до більш агресивної політики – зокрема, до більш частих високих доз внутрішньовенної ін'єкції, ніж короткострокові. Базова винагорода за всі значення C (для $C=100$ важко візуально побачити на тепловій карті, але агент вводив максимальну дозу рідини в 40% випадків). Політики також значно відрізняються залежно від рівня акценту на середньострокові та довгострокові результати, визначені значеннями C . Було помітно більш високі відмінності між випадковими перезапусками в налаштуваннях довгострокової винагороди, ніж у налаштуваннях короткострокового

базового рівня. Це може вказувати на те, що оптимізація є більш складною і нестабільною для отримання довгострокових винагород.

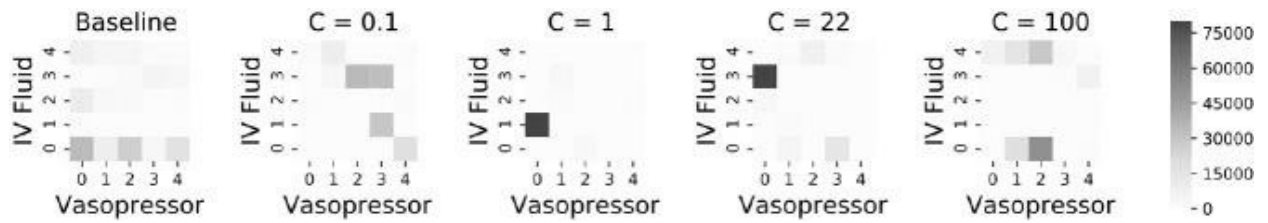


Рисунок 5.4 – Порівняння навчених ДПГQ-M з короткостроковим і довгостроковим / проміжним винагородою

Довгострокові цілі призводять до більш агресивної та менш стабільної політики

5.4.4 Модель кодування

Результати порівняння використання LSTM і GRU можна знайти на рис. 5.5. Можна спостерігати, що як базова реалізація (LSTM), так і реалізація GRU рекомендували ненульові дози вазопресорів значно частіше, ніж лікарі. Проте реалізація GPU була більш агресивною, рекомендуючи ненульові дози внутрішньовенної ін'єкції значно частіше, ніж лікарі та DQN-LSTM.

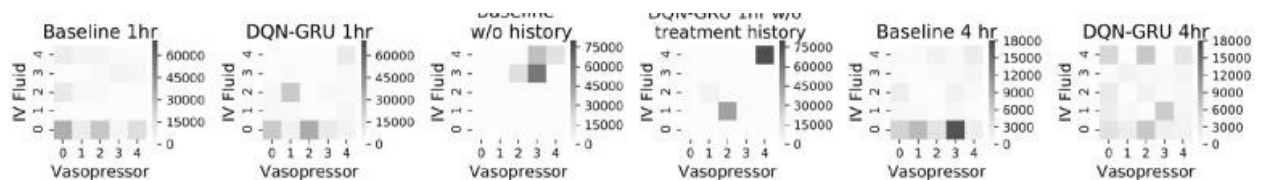


Рисунок 5.5 – Впровадження GRU

Вибір архітектури кодування також взаємодіє з іншими налаштуваннями аналізу, ефекти яких різняться в залежності від архітектури вбудовування.

Виключення історії попереднього лікування має ще більш екстремальний ефект, коли вбудовування використовують архітектуру DQN-GRU, при цьому максимальне дозування обох методів лікування надається більшу частину часу.

Різна часова сегментація 4-годинних часових інтервалів призвела до більш частих високих доз вазопресорів при базовій реалізації LSTM, але більш частим високим дозам внутрішньовенних ін'єкцій при реалізації GRU.

Неможливо застосувати клінічні, фізіологічні або статистичні знання для визначення того, яка архітектура вбудовування є більш придатною, оскільки вони концептуально дуже схожі. Таким чином, зміна, що впливає з вибору вбудовування, є джерелом занепокоєння.

Загалом відстежується висока чутливість до архітектури.

5.4.5 Випадкові перезапуски

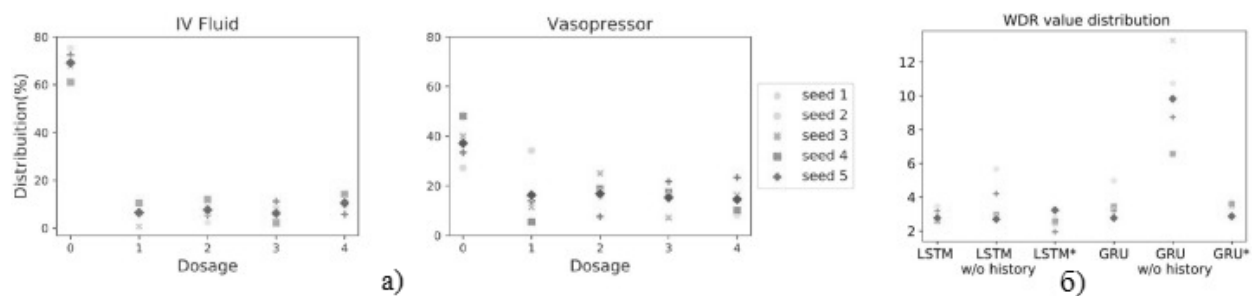
Остаточний аналіз чутливості розглядав відмінності між перезапусками методи – оцінювання чутливість до того, де метод був ініціалізований. Хоча суттєві відмінності в розподілі дій між ДПГQ-М та стратегією лікаря залишалися незмінними для всіх насін'я у нашій базовій моделі, все ще спостерігалися значні відмінності в результируючих розподілах дій, особливо для вазопресорів (рис. 5.6, а). На рисунку 5.6, б видно, що, незважаючи на ці відмінності, оціночні значення цих політик аналогічні; це показує, що різниця не в тому, що оптимізація іноді призводила до поганого оптимуму, а в тому, що існує безліч оптимумів з аналогічним оціночною якістю, які призводять до якісно різних політиків. Це ще одна причина для занепокоєння, оскільки це говорить про те, що агент не може визначити, чи є будь-яка з цих абсолютно різних політик краще, ніж інші.

Загалом, політики ГНП мають багато локальних оптимумів.

5.5 Аналіз отриманих результатів

Сучасні підходи до навчання з використанням стратегій ГНП в значній мірі недостатньо випробувані у відповідних досвідчених медичних установах на складних та динамічних завданнях. Було представлено аналіз чутливості, який досліджує, як агент ДПГQ-M буде реагувати на альтернативні специфікації, включаючи:

- включення попередньої історії лікування пацієнта;
- дискретизація часових інтервалів;
- налаштування топології ШНМ для кодування;
- специфікації винагороди;
- випадкові перезапуски методу.



а) розподіл лікування за випадковими перезапусками у вихідному стані. У той час як дисперсія внутрішньовенних ін'єкцій невелика, розподіл вазопресора варіюється в різних насінні; б) порівняння значень оцінки WDR за випадковим насінням в кожному налаштуванні.

Рисунок 5.6 – Порівняння результатів випадкових перезапусків

Було продемонстровано, що вибір між однаково апріорно виправданими налаштуваннями реалізації може мати великий клінічно значущий вплив на визначення політик (стратегій) ГНП. Враховуючи цю відсутність надійності, результати окремих реалізацій слід сприймати скептично.

Більш надійні результати були отримані при врахуванні історичних даних про попереднє лікування пацієнта. Виключення історії лікування виправдано тільки відповідно до неправдоподібних припущень Маркова про те, що минулі методи лікування впливають на майбутні результати тільки через вимірювані проміжні змінні. Якщо це припущення не виправдається і виникнуть кумулятивні небезпеки від занадто інтенсивного лікування (наприклад, набряк легенів від перевантаження рідиною і серцево-судинні побічні ефекти від вазопресорів в нашому додатку), то минуле лікування вплине як на відповідь (лікування з меншою ймовірністю буде корисним при надмірному лікуванні в минулому), так і на поточне рішення про лікування (лікування з меншою ймовірністю буде призначено при великому лікуванні в минулому, і результати, ймовірно, будуть гірше при великому лікуванні в минулому). Таким чином, відсутність історії попереднього лікування зробить надмірне лікування більш корисним, ніж воно є насправді. Дійсно, поведінка, яка спостерігалася в агентах ДПГQ-M, відповідала поведінці, яку передбачала теорія. Агенти, навчені без історії лікування, включені в їх стан, рекомендували вазопресори або внутрішньовенні ін'єкції на кожному етапі, що є явно шкідливою стратегією. Проте всі три попередні впровадження ГНП при сепсисі виключали історію лікування з визначень стану.

Хоча досі не можливо дати остаточних статистичних або фізіологічних пояснень більшості результатів ГНП, що спостерігаються в аналізі чутливості, тут обговорюється можливі джерела нестабільності ГНП. Однією з причин, як виявилось, була необґрунтована політика, що впливає з екстраполяції за рамки рішень про лікування, що спостерігаються в даних в рамках нинішньої практики. Наприклад, в аналізі підгруп за шкалою погіршення було видно, що агенти ГНП рекомендують явно шкідливі дії, рідко зустрічаються в даних, тобто не дають високих доз пацієнтам з високим ризиком і часто дають високі дози пацієнтам з низьким ризиком. Крім того, той факт, що різні ініціалізації знайшли рішення з аналогічними оціночними значеннями Q , але якісно різними розподілами дій, говорить про

те, що проблема недостатньо обмежена. Тобто потрібні більш ефективні способи включення знань про те, які функції важливі і які дії розумні для обмеження засвоєної політики.

Так ще передбачається довгий шлях від поточного стану медичних застосунків ГНП до клінічно дієвих ідей або рекомендацій щодо лікування. В даний час дослідники ІІІ застосовують алгоритми ГНП до клінічних проблем і стверджують, що політика, яку вони вивчають, значно покращить результати в галузі охорони здоров'я порівняно з поточною практикою. Початок дослідження чутливості методів ГНП є невеликим, але важливим кроком на шляху до клінічно дієвої політики ГНП.

Майбутніми напрямками роботи є одразу декілька стратегічних напрямків. По-перше, широкий спектр параметрів аналізу може бути зіставлений у великих і фізіологічно достовірних імітаційних експериментах, в яких буде доступна базова істинність отриманих в результаті вивчених стратегій. Це може пролити світло на деякі експлуатаційні характеристики і передовий досвід. Наприклад, в цій структурі можна було б оцінити альтернативні підходи до запобігання екстраполяції моделей занадто далеко від поточної практики, як це було запропоновано вище.

Крім того, коли політики чутливі до алгоритмічного вибору, можна шукати області широкої згоди між політиками, рекомендованими в різних налаштуваннях. Ці області угоди будуть представляти собою фрагменти політики, тобто рекомендації, застосовні тільки до конкретних контекстів. Потім ці фрагменти стратегії можуть бути ретельно оцінені експертами з предмету на предмет правдоподібності, а їх наслідки можуть бути оцінені епідеміологами з використанням більш стабільних методів оцінки ефекту лікування. Пошук підходів, що дозволяють зробити ГНП надійною та перевіреною людиною, допоможе нам належним чином використовувати інформацію в медичних записах для поліпшення медичної допомоги.

ВИСНОВКИ

Під час виконання дипломної кваліфікаційної роботи магістра, було запропоновано новий ГНП для синтезу діагностичних нейромделей для вирішення прикладних задач медичного діагностування – ДПГQ-M. Також було розроблено ПЗ, що реалізує роботу запропонованого методу.

Для цього були поставлені і вирішені такі задачі:

- ознайомлення та аналіз проблемної області медичного діагностування;
- розгляд мови програмування Python та фреймворку МН – TensorFlow;
- розроблення ПЗ для реалізації та тестування запропонованого методу;
- аналіз чутливості результуючих рішень від налаштування метепараметрів методу.

Розроблений програмний комплекс, реалізує основні функції, а саме:

- завантаження та зчитування історичних даних про пацієнтів;
- синтез нейромделей для медичного діагностування;
- налаштування та тонке налаштування метепараметрів методу синтезу;
- представлення результатів;
- розгортання попередньо синтезованих нейромделей для їх дослідження.

Запропонований метод використовує найсучасніші засади ГНП для підвищення результатів точності діагностуючих моделей.

Розроблений програмний комплекс має практичну спрямованість та готовий до використання.

Під час розробки програми та аналізу розробленого програмного комплексу з'явилися ідеї з удосконалення, такі як: використання паралельних потужностей обчислювальних систем задля збільшення кількості незалежних

агентів під час синтезу ШНМ та зіставлення спектр метапараметрів методу із даними у великих, фізіологічно достовірних імітаційних експериментах.

Нові можливості зможуть розширити коло вирішення практичних задач методом, адже тонке налаштування параметрів та збільшення кількості агентів зможе в значній мірі покрити стохастичний простір пошуку рішення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Diadiun T. V. The study of consumer aspects of medical products for diagnosis of pregnancy [Text] / T. V. Diadiun, I. I. Baranova, A. O. Lytovchenko // Вісник фармації. – 2015. – № 3. – С. 60-62.
2. Batyuk L. Monitoring system of biophysical properties of the red blood cells of patients for medical diagnosis purposes [Text] / L. Batyuk, N. Kizilova // Системи обробки інформації. – 2020. – Вип. 3. – С. 13-20.
3. Matveeva N. A. Artificial neural networks in medical diagnosis [Текст] / N. A. Matveeva // Системні технології. – 2021. – Вип. 2. – С. 33-41.
4. Соколова М. П. Проблеми виявлення і реєстрації професійних захворювань в Україні: автореф. дис... канд. мед. наук: 14.02.01 / М. П. Соколова ; Держ. установа "Ін-т медицини праці АМН України". – К., 2009. – 20 с.
5. Мірошніченко Н. С. Медичне діагностування на основі нейромережних технологій [Текст] / Н. С. Мірошніченко, Г. І. Перова // Прикладная радиоэлектроника. – 2017. – Т. 16, № 1-2. – С. 46-50.
6. Sineglazov V. M. Analysis of using Software Packages for Imaging in Medical Diagnostics [Text] / V. M. Sineglazov, D. S. Raduchych // Electronics and control systems. – 2016. – № 4. – С. 114-118.
7. Kravchenko O. Development of medical diagnostic decision support systems and their economic efficiency [Text] / O. Kravchenko // Technology audit and production reserves. – 2018. – № 2(2). – С. 4-10.
8. Perova I. Fast Medical Diagnostics using Autoassociative Neuro-fuzzy Memory [Text] / I. Perova, Ye. Bodyanskiy // Computing. – 2017. – Vol. 16, Issue 1. – С. 34-40.
9. Лещинська О. Л. Інтелектуальні методи синтезу семантичних моделей програм: автореф. дис... канд. техн. наук: 05.13.23 / О. Л. Лещинська ; Харк. нац. ун-т радіоелектрон. – Х., 2009. – 19 с.

10. Півошенко В. В. Аналіз та експериментальне дослідження методу безмодельного навчання з підкріпленням [Текст] / В. В. Півошенко, М. С. Кулик, Ю. Ю. Іванов, А. С. Васюра // Вісник Вінницького політехнічного інституту. – 2019. – № 3. – С. 40-49.
11. Осауленко В. М. Двовимірна модель навчання в спайкових нейронних мережах з гомеостазом та навчанням з підкріпленням [Текст] / В. М. Осауленко // Системні дослідження та інформаційні технології. – 2017. – № 2. – С. 130-140.
12. Бочкарьов О. Ю. Структурна адаптація процесів збирання даних в автономних розподілених системах із використанням методів навчання з підкріпленням [Текст] / О. Ю. Бочкарьов // Computer systems and networks. – 2020. – Vol. 2, № 1. – С. 13-26.
13. Is Deep Reinforcement Learning Ready for Practical Applications in Healthcare? A Sensitivity Analysis of Duel-DDQN for Hemodynamic Management in Sepsis Patients [Text] / [M.Y. Lu, Z. Shahn, D. Sow, F. Doshi-Velez, Li-wei H. Lehman] // Proceedings of the AMIA Annual Symposium, 14-18 November 2020, Virtual Rockvill. – Rockvill: AMIA 2020 – Pp. 773-782
14. Continuous State-Space Models for Optimal Sepsis Treatment – a Deep Reinforcement Learning Approach [Text] / [A. Raghu, M. Komorowski, I. Ahmed, L. Celi, P. Szolovits, M. Ghassemi] // MLHC. – Michigan: MLHC, 2017. – 17 p.
15. Improving Sepsis Treatment Strategies by Combining Deep and Kernel-Based Reinforcement Learning [Text] / [X. Peng, Y. Ding, D. Wihl, O. Gottesman, M. Komorowski, L.-W. H Lehman, A. Ross, A. Faisal, F. Doshi-Velez] // Proceedings of the AMIA Annual Symposium, 3-7 November 2018, San Francisco, USA. – San Francisco: AMIA 2018 – Pp. 887-896.
16. Програмування числових методів мовою Python [Текст] : навч. посіб. для студентів ВНЗ / [А. Ю. Дорошенко та ін.] ; за ред. чл.-кор. НАН України А. В. Анісімова ; Київ. нац. ун-т ім. Тараса Шевченка. – Київ : Київський університет, 2013. – 463 с.

17. Обвінцев О. В. Об'єктно-орієнтоване програмування. Курс на основі Python [Текст] : матеріали лекцій : навч. посіб. / Обвінцев О. В. ; Київ. нац. ун-т ім. Тараса Шевченка. – Київ : Основа, 2017. – 324 с.
18. Кухар М. А. Аналіз можливостей мови програмування Python для роботи з просторовими даними [Текст] / М. А. Кухар // ScienceRise. – 2019. – № 4. – С. 40-46.
19. Мельник В. М. Ідентифікація емоцій людини за допомогою нейронної мережі на основі Keras та TensorFlow [Текст] / В. М. Мельник, К. В. Мельник, Б. В. Шульга // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2019. – № 36. – С. 109-122.
20. Anaconda [Electronic resource]. – Access mode: <https://www.anaconda.com/products/individual>
21. MIMIC-III Clinical Database [Electronic resource]. – Access mode: <https://physionet.org/content/mimiciii/1.4/>
22. MIMIC-III, a freely accessible critical care database [Text] / [A. E. W. Johnson, T. J. Pollard, L. Shen, L.-W. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, R. G. Mark] // Proceedings of the AMIA Annual Symposium, 12-16 November 2016, Chicago, USA. – Chicago: AMIA 2016 – Pp. 887-896.

ДОДАТОК А
Технічне завдання

A.1 Вступ

A.1.2 Призначення та область застосування

Розроблений ПЗ може використовуватися для реалізації стратегій ГНП в медичному діагностуванні. Така система може використовуватися у медичних закладах та науково-дослідницьких установах.

Програма представляє графічний інтерфейс користувача для зручної роботи із реалізованими функціями.

A.2 Підстави для розробки

Підставою для розробки є завдання на магістерську роботу, яке було визначене за темою «Дослідження та програмна реалізація методів навчання з підкріпленням для медичного діагностування» і затверджене наказом № 435 від 05 листопада 2021 р. за Національним університетом «Запорізька політехніка».

A.3 Призначення розробки

Розроблений ПЗ може використовуватися для медичного діагностування.

Розроблений застосунок має реалізовувати основні функції, а саме:

- завантаження та зчитування історичних даних про пацієнтів;
- синтез нейромоделей для медичного діагностування;
- налаштування та тонке налаштування метапараметрів методу синтезу;
- представлення результатів;
- розгортання попередньо синтезованих нейромоделей для їх дослідження.

А.4 Вимоги до програми

А.4.1 Вимоги до функціональних характеристик

ПЗ повинне забезпечувати виконання нижче перерахованих функцій:

- завантаження та зчитування історичних даних про пацієнтів;
- синтез нейромоделей для медичного діагностування;
- налаштування та тонке налаштування метапараметрів методу синтезу;
- представлення результатів;
- розгортання попередньо синтезованих нейромоделей для їх дослідження.

А.4.2 Вимоги до надійності

Для забезпечення надійності програми, вона має базуватися на комп'ютері. На протязі усього сеансу роботи додаток не повинен збоїти або зависати. Після оновлення інформаційної бази або функціоналу, додаток треба перезапустити.

А.4.3 Вимоги до технічних та програмних засобів

Для нормальної розробки програми, повинні виконуватися наступні умови до апаратної частини:

- наявність ПК, на якому буде працювати застосунок;
- процесор із тактовою частотою 3,0 GHz та більше;
- оперативна пам'ять 4,0 ГБ та більше;
- вільний простір на жорсткому диску не менше, ніж 2 ГБ (для збереження вибірок).

Вимоги до програмної частини:

- встановлений дистрибутив Python.

A.5 Вимоги до програмної документації

Склад програмної документації повинен включати в себе:

- технічне завдання;
- опис проектування;
- опис тестування;
- опис програми.

A.6 Стадії і етапи розробки

A.6.1 Стадії розробки

Розробка повинна бути проведена в три стадії:

- розробка технічного завдання;
- робоче проектування;
- впровадження.

A.6.2 Етапи розробки

На стадії розробки технічного завдання повинен бути виконаний етап розробки, погодження та затвердження справжнього технічного завдання.

На стадії робочого проектування повинні бути виконані перераховані нижче етапи робіт:

- розробка програми;
- розробка програмної документації;
- випробування програми.

На стадії впровадження повинен бути виконаний етап розробки підготовка і передача програми.

A.6.3 Зміст робіт по етапах

На етапі розробки технічного завдання повинні бути виконані перераховані нижче роботи:

- постановка задачі;
- визначення і уточнення вимог до технічних засобів;
- визначення вимог до програми;
- визначення стадій, етапів і термінів розробки програми та документації на неї;
- узгодження і затвердження технічного завдання.

На етапі розробки програми повинна бути виконана робота з програмування (кодування) і налагодженні програми.

На етапі розробки програмної документації повинна бути виконана розробка програмних документів відповідно до вимог до складу документації.

На етапі випробувань програми повинні бути виконані перераховані нижче види робіт:

- розробка, узгодження та затвердження та методики випробувань;
- проведення приймально-здавальних випробувань;
- коректування програми і програмної документації за результатами випробувань.

ДОДАТОК Б
Опис програми

Б.1 Загальні свідчення

ПЗ розроблено за допомогою мови програмування Python, середовища розробки Anaconda та фреймворку для МН – TensorFlow. Тобто для реалізації основної бізнес-логіки застосунку використовується мова програмування Python.

Б.2 Функціональне призначення

ПЗ повинне забезпечувати виконання нижче перерахованих функцій:

- завантаження та зчитування історичних даних про пацієнтів;
- синтез нейромоделей для медичного діагностування;
- налаштування та тонке налаштування метапараметрів методу синтезу;
- представлення результатів;
- розгортання попередньо синтезованих нейромоделей для їх дослідження.

Б.3 Опис логічної структури

Для представлення рівнів ПЗ можна використовувати структурну діаграму представлену на рис. Б.1. При проектуванні структури майбутньої програми було виділено 4 рівня ПЗ.

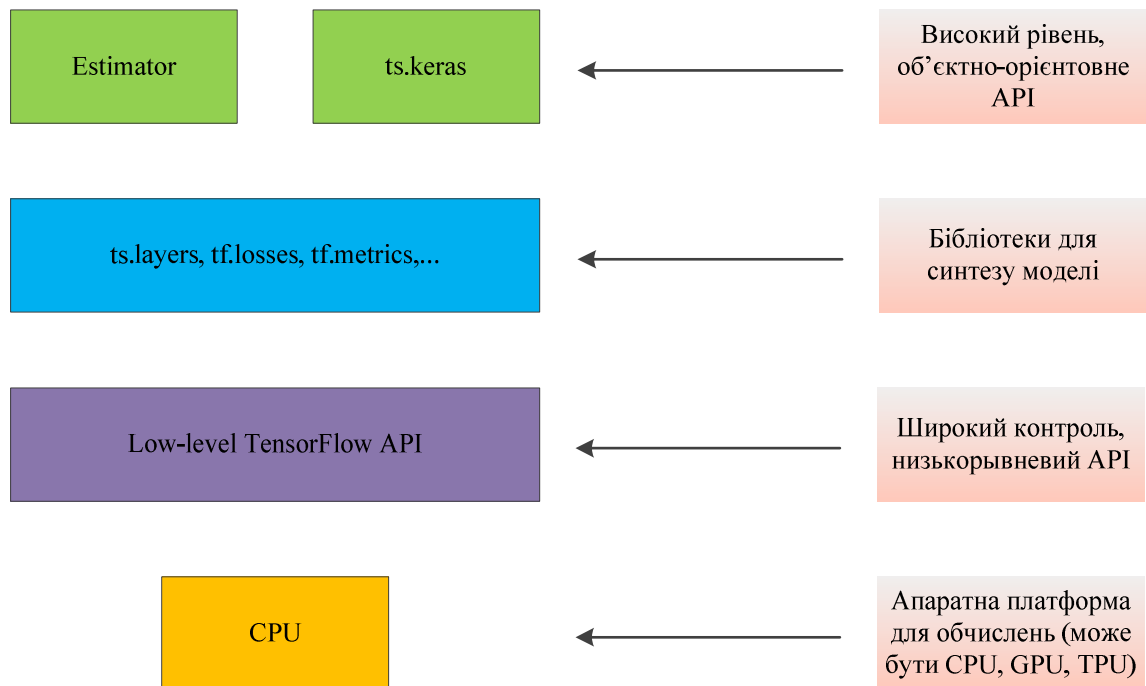


Рисунок Б.1 – Структурна схема рівнів розроблюваного ПЗ

Розробку ж попередньо описаних модулів можна аргументувати способом розподілення основного функціоналу ПЗ за таким принципом гарантує формування функціонально завершених блоків програми. Такий варіант розподілення представлено на рис. Б.2.

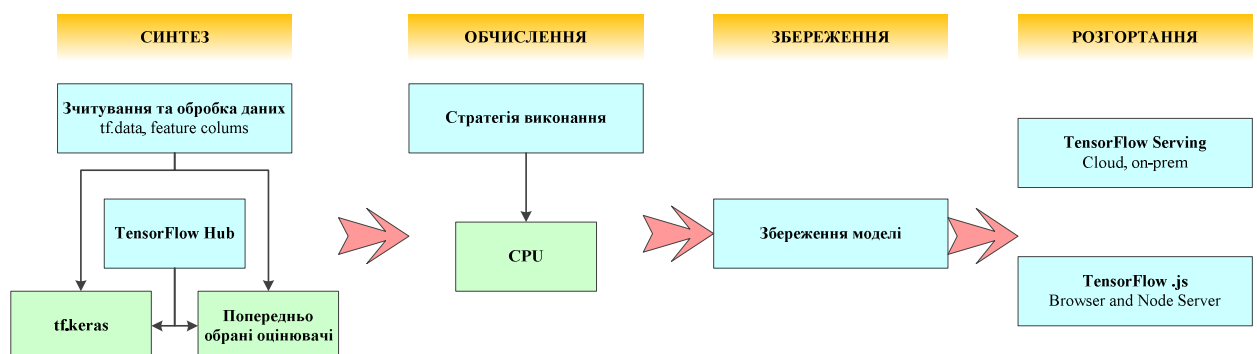


Рисунок Б.2 – Модулі розроблюваного ПЗ

Модуль завантаження та обробки вхідних даних розроблено для завантаження вибірок та обробки даних отриманих від користувача.

Модуль синтезу ШНМ відповідає за реалізацію процесу синтезу ШНМ.

Модуль збереження мережі – це окремий модуль для налаштування властивостей запису результуючої мережі.

Модуль розгортання моделі – це окремий модуль для розгортання попередньо синтезованих моделей та їх дослідження.

Б.4 Використані технічні засоби

Для нормальної розробки програми, повинні виконуватися наступні умови до апаратної частини:

- наявність ПК, на якому буде працювати застосунок;
- процесор із тактовою частотою 3,0 GHz та більше;
- оперативна пам'ять 4,0 ГБ та більше;
- вільний простір на жорсткому диску не менше, ніж 2 ГБ (для збереження вибірок).

Вимоги до програмної частини:

- встановлений дистрибутив Python.

ДОДАТОК В
Текст програми

Сирцевий код розробленого застосунку:

class Main:

```

def __init__(self, envir, agent, args):

    self.id = args.id
    self.envir = envir
    self.agent = (args.consecutive_frames,) + agent
    #
    self.data_buff = 20000
    #
    if(len(agent) < 3):
        self.tau = 1e-2
    else:
        self.tau = 1.0

def synth(self, batch_size):
    q = self.agent.forcst(s)
    next_seed = self.agent.forcst(new_s)
    step = self.agent.target_forcst(new_s)

    for i in range(s.shape[0]):
        old_q = q[i, a[i]]
        if d[i]:
            q[i, a[i]] = r[i]
        else:
            nBestact = np.argmax(next_seed[i,:])
            q[i, a[i]] = r[i] + self.gamma * step[i, nBestact]
        if(self.id):
            self.buffer.update(idx[i], abs(old_q - q[i, a[i]]))
    self.agent.fit(s, q)

def test(self, env, args, sum_wrt):

    results = []

```

```
tqdm_e = tqdm(range(args.nb_episodes), desc='Score', leave=True,
unit=" episodes")
```

```
for e in tqdm_e:
```

```
    time, cumul_raeward, done = 0, 0, False
```

```
    old_stuet = env.reset()
```

```
    while not done:
```

```
        if args.render: env.render()
```

```
        a = self.policy_action(old_stuet)
```

```
        new_stuet, r, done, _ = env.step(a)
```

```
        self.memorize(old_stuet, a, r, done, new_stuet)
```

```
        old_stuet = new_stuet
```

```
        cumul_raeward += r
```

```
        time += 1
```

```
        if(self.buffer.size() > args.batch_size):
```

```
            self.synth(args.batch_size)
```

```
            self.agent.transfer_wigts()
```

```
    if(args.gather_stats):
```

```
        mean, stdev = gather_stats(self, env)
```

```
        results.append([e, mean, stdev])
```

```
score = TenFlSum('score', cumul_raeward)
```

```
sum_wrt.add_summary(score, global_step=e)
```

```
sum_wrt.flush()
```

```
tqdm_e.set_description("Score: " + str(cumul_raeward))
```

```
tqdm_e.refresh()
```

```
return results
```

```
def memorize(self, stuet, action, raeward, done, new_stuet):
```

```
    if(self.id):
```

```
        ag_vl = self.agent.forcst(stuet)
```

```

        ag_vl_t = self.agent.target_forcst(new_stuet)
        nBestact = np.argmax(self.agent.forcst(new_stuet))
        new_vl = raeward + self.gamma * ag_vl_t[0, nBestact]
        pt_error = abs(new_vl - ag_vl)[0]
    else:
        pt_error = 0
    self.buffer.memorize(stuet, action, raeward, done, new_stuet, pt_error)

```

```

def safe_wigts(self, path):
    path += '_LR_{}'.format(self.lr)
    if(self.id):
        path += '_PER'
    self.agent.save(path)

```

```

def load_wigts(self, path):
    self.agent.load_wigts(path)

```

```

class Agent:

```

```

    def __init__(self, stepivF, agent, args):

        self.metaP = args.metaP
        self.stepivF = stepivF
        self.agent = (args.consecutive_frames,) + agent
        #
        self.act = ReLU
        self.stuet = 0.95
        self.seed = 0.224
        self.step = 4216
        self.data_buff = 5575
        #
        if(len(agent) < 3):
            self.tau = 541
        else:
            self.tau = 1.0
        # Create stepor and critic networks

```

```

self.agent = Agent(self.agent, stepivF, self.act, self.tau, args.dueling)
# Memory Buffer for Experience Replay
self.buffer = MemoryBuffer(self.data_buff, args.metaP)

def policy(self, s):
    """ Apply an epsilon-greedy policy to pick next stepion
    """
    if random() <= self.seed:
        return randrange(self.stepivF)
    else:
        return np.argmax(self.agent.forcst(s)[0])

def trainMod(self, batch_size):
    """ Train Q-network on batch sampled from the buffer
    """
    # Sample experience from memory buffer (optionally with PER)
    s, a, r, d, new_s, idx = self.buffer.sample_batch(batch_size)

    # Apply Bellman Equation on batch samples to train our DDQN
    q = self.agent.forcst(s)
    next_seed = self.agent.forcst(new_s)
    step = self.agent.target_forcst(new_s)

    for i in range(s.shape[0]):
        old_q = q[i, a[i]]
        if d[i]:
            q[i, a[i]] = r[i]
        else:
            next_best_stepion = np.argmax(next_seed[i,:])
            q[i, a[i]] = r[i] + self.stuet * step[i, next_best_stepion]
        if(self.metaP):
            # Update PER Sum Tree
            self.buffer.update(idx[i], abs(old_q - q[i, a[i]]))
    # Train on batch
    self.agent.fit(s, q)
    # Decay seed

```

```

self.seed *= self.step

def train(self, env, args, sum_wrt):
    """ Main DDQN Training Algorithm
    """

    results = []
    tqdm_e = tqdm(range(args.nb_episodes), desc='Score', leave=True,
unit=" episodes")

    for e in tqdm_e:
        # Reset episode
        time, cumul_raeward, done = 0, 0, False
        old_stuet = env.reset()

        while not done:
            if args.render: env.render()
            a = self.policy(old_stuet)
            new_stuet, r, done, _ = env.step(a)

            self.loadParam(old_stuet, a, r, done, new_stuet)

            old_stuet = new_stuet
            cumul_raeward += r
            time += 1

            if(self.buffer.size() > args.batch_size):
                self.trainMod(args.batch_size)
                self.agent.transfer_wigts()

        if(args.gather_stats):
            mean, stdev = gather_stats(self, env)
            results.append([e, mean, stdev])

```

```

score = TenFlSum('score', cumul_raeward)
sum_wrt.add_summary(score, global_step=e)
sum_wrt.flush()

tqdm_e.set_description("Score: " + str(cumul_raeward))
tqdm_e.refresh()

```

```

return results

```

```

def loadParam(self, stuet, stepion, raeward, done, new_stuet):

```

```

    if(self.metaP):
        ag_vl = self.agent.forcst(stuet)
        ag_vl_t = self.agent.target_forcst(new_stuet)
        next_best_stepion = np.argmax(self.agent.forcst(new_stuet))
        new_vl = raeward + self.stuet * ag_vl_t[0, next_best_stepion]
        pt_error = abs(new_vl - ag_vl)[0]
    else:
        pt_error = 0
    self.buffer.loadParam(stuet, stepion, raeward, done, new_stuet,
pt_error)

```

```

def toSaveParam(self, path):
    path += '_act_{ }'.format(self.act)
    if(self.metaP):
        path += '_PER'
    self.agent.save(path)

```

```

def load_wigts(self, path):
    self.agent.load_wigts(path)
def __init__(self, stute_prsnt, act_prsnt, lr, tau, duelGener):
    self.stute_prsnt = stute_prsnt
    self.act_prsnt = act_prsnt
    self.tau = tau

```

```

self.duelGener = duelGener
self.Mod = self.network(duelGener)
self.Mod.compile(AgentFir(lr), 'mse')
self.target_Mod = self.network(duelGener)
self.target_Mod.compile(AgentFir(lr), 'mse')
self.target_Mod.set_weights(self.Mod.get_weights())

def huber_loss(self, y_true, y_pred):
    return K.mean(K.sqrt(1 + K.square(y_pred - y_true)) - 1, axis=-1)

def network(self, duelGener):
    inp = Input((self.stute_prsnt))

    if(len(self.stute_prsnt) > 2):
        inp = Input((self.stute_prsnt[1:]))
        x = net_blok(inp, 32, (2, 2), 8)
        x = net_blok(x, 128, (2, 2), 4)
        x = net_blok(x, 128, (2, 2), 3)
        x = Flatten()(x)
        x = Dense(256, activation='relu')(x)
    else:
        x = Flatten()(inp)
        x = Dense(128, activation='relu')(x)
        x = Dense(128, activation='relu')(x)

    if(duelGener):
        x = Dense(self.act_prsnt + 1, activation='linear')(x)
        x = Lambda(lambda i: K.expand_dims(i[:,0],-1) + i[:,1:] -
K.mean(i[:,1:], keepdims=True), output_shape=(self.act_prsnt,))(x)
    else:
        x = Dense(self.act_prsnt, activation='linear')(x)
    return Mod(inp, x)

def transfer_weights(self):

    W = self.Mod.get_weights()

```

```

tgt_W = self.target_Mod.get_weights()
for i in range(len(W)):
    tgt_W[i] = self.tau * W[i] + (1 - self.tau) * tgt_W[i]
self.target_Mod.set_weights(tgt_W)

def fit(self, inp, targ):

    self.Mod.fit(self.reshape(inp), targ, epochs=1, verbose=0)

def forecast(self, inp):

    return self.Mod.forecast(self.reshape(inp))

def target_forecast(self, inp):

    return self.target_Mod.forecast(self.reshape(inp))

def reshape(self, x):
    if len(x.shape) < 4 and len(self.stute_prsnt) > 2: return
np.expand_dims(x, axis=0)
    elif len(x.shape) < 3: return np.expand_dims(x, axis=0)
    else: return x

def save(self, path):
    if(self.duelGener):
        path += '_duelGener'
    self.Mod.save_weights(path + '.h5')

def load_topolog(self, path):
    self.Mod.load_topolog(path)

```

ДОДАТОК Д
Слайди презентації

Міністерство освіти та науки України
Національний Університет «Запорізька політехніка»



**Дослідження та програмна реалізація
методів навчання з підкріпленням для
медичного діагностування**

Луковка О.О.	Олійник А.О.
КНТ-120м	д.т.н., професор

[1] – <http://www.zntu.edu.ua/>
 [2] – <https://nuz.zntu.net/>

Рисунок Д.1 – Слайд №1

Об'єкт, предмет та мета роботи

Об'єкт дослідження – процес використання методів навчання з підкріпленням для вирішення задач медичного діагностування.

Предмет дослідження – методи навчання з підкріпленням для синтезу нейромоделей медичного діагностування.

Мета роботи – дослідити, проаналізувати та реалізувати методи навчання з підкріпленням для синтезу нейромоделей медичного діагностування.

2

Рисунок Д.2 – Слайд №2

Постановка завдання

Головна мета магістерської роботи полягає у дослідженні та програмній реалізації методів навчання з підкріпленням, для синтезу нейромоделей медичного діагностування, на основі методу Дуельних Подвійних Глибоких Q-Мереж (ДПГQ-М).

$$DataSet = \{x_1, x_2, \dots, x_n, y\} \left\{ \begin{array}{l} NN^1 = \{Neuron_1^1, Neuron_2^1, Neuron_n^1, Connection_1^1, Weight_n^1\} \\ NN^2 = \{Neuron_1^2, Neuron_2^2, Neuron_n^2, Connection_1^2, Weight_n^2\} \end{array} \right\} Error_{NN^1} \rightarrow 0$$

3

Рисунок Д.3 – Слайд №3

Актуальність роботи

Попередній аналіз довів, що методи машинного навчання (МН), а саме підходи глибокого навчання з підкріпленням (ГНП) є перспективною основою для підвищення точності методів медичного діагностування.

Тому актуальним є дослідження та реалізація нових методів ГНП, що поєднують переваги звичайного навчання з підкріпленням та глибокого навчання. Проте для цього слід визначити, які саме методи та засади будуть використовуватися.

4

Рисунок Д.4 – Слайд №4

Порівняння підходів МН для медичного діагностування

	Навчання з учителем	Навчання без учителя	Навчання з підкріпленням
Залежність від вхідних даних	Високий рівень	Низький рівень	Середній рівень
Використовувані методи	Високий рівень	Високий рівень	Високий рівень
Кола практичних завдань	Середній рівень	Середній рівень	Високий рівень (в останній час)
Подальший розвиток	Перспективна галузь (за умови використання підходів IoT)	Перспективна галузь	Перспективна галузь

5

Рисунок Д.5 – Слайд №5

Використання ДПГQ-М для медичного діагностування

 $Q(s, a, \theta)$

$$L_t(\theta_t) = E_{s,a,r,s'} \left[\left(y_t^{DQN} - Q(s, a, \theta_t) \right)^2 \right] \quad y_t^{DQN} = r + \gamma \max_{a'} Q(s', a', \theta)$$

$$\nabla_{\theta_t} L_t(\theta_t) = E_{s,a,r,s'} \left[y_t^{DQN} - Q(s, a, \theta_t) \nabla_{\theta_t} Q(s, a, \theta_t) \right]$$

Аналіз чутливості

- включення історії попереднього лікування;
- тривалість часових проміжків;
- горизонт винагород;
- вибір моделі кодування;
- випадкові перезавпуски.

6

Рисунок Д.6 – Слайд №6

Порівняння мов програмування

Критерій	Python	Java	Scala	R
Швидкість роботи	±	+	±	±
Бібліотеки	++	±	±	+
Фреймворки для методів МН	++	+	+	+
Спільнота	++	+	±	+

Рисунок Д.7 – Слайд №7

Порівняння фреймворків МН

Критерій	TensorFlow (Python)	Caffe (C++)	Microsoft Cognitive Toolkit (C++)	Theano (Python)
Відкритість	+	+	+	+
Платформа	++	++	+	++
Топології ШНМ	++	++	±	±
Паралельні обчислення	++	-	-	++
Підтримка	++	+	±	+

Рисунок Д.8 – Слайд №8

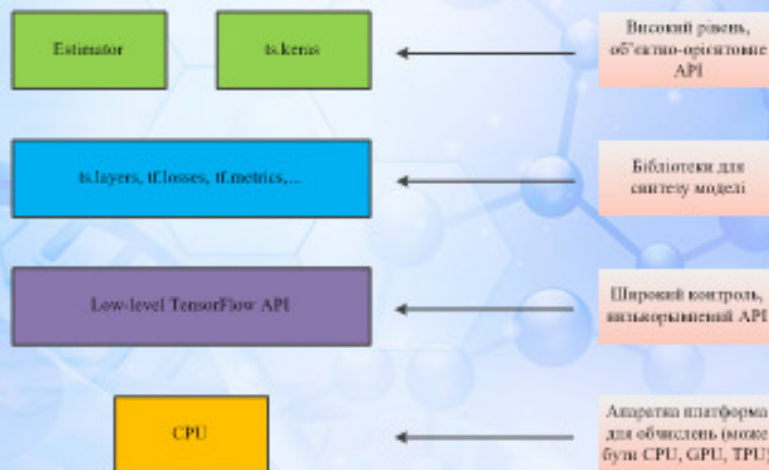
Порівняння середовищ розробки

Критерії порівняння	Середовища розробки для Python			
	Anaconda	PyCharm	Atom	PhpStorm
Наявність безкоштовної версії	++	++	++	±
Підтримка TensorFlow	++	+	+	±
Кроссплатформність	++	++	++	–
Швидкість роботи	+	+	++	±
Підтримка та оновлення	++	±	±	++

9

Рисунок Д.9 – Слайд №9

Проектування структури програмної системи



10

Рисунок Д.10 – Слайд №10



Рисунок Д.11 – Слайд №11



Рисунок Д.12 – Слайд №12

Вибірка даних для тестування

Демографічні ознаки	Вік, вага, зріст, етнічна приналежність
Життєві показники Лабораторні показники	Шкала ком Глазго, частота серцевих скорочень, температура, частота дихання, діастолічний артеріальний тиск, систолічний артеріальний тиск, середній артеріальний тиск, калій, натрій, хлорид, магній, кальцій, аніонний розрив, гемоглобін, гематокрит, тромбоцити, смуги, SpO_2 , FiO_2 , PaO_2 , $TotalCO_2$, pCO_2 , глюкоза, креатинін та інші
Лікування	Внутрішньовенна ін'єкція та застосування вазопресорів

13

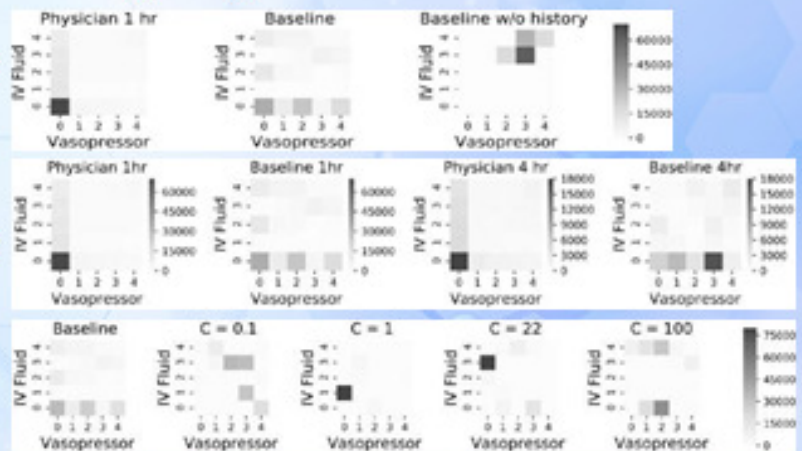
Рисунок Д.13 – Слайд №13

Аналіз результатів

Розподіл дій ДПГQ-M,
навчених з/без інформації
про історію лікування

Порівняння розподілу дій
по 4-годинним проміжкам
часу ДПГQ-M і 1-годинному
проміжку часу

Порівняння навчених
ДПГQ-M з
короткостроковим і
довгостроковим /
проміжним винагородою

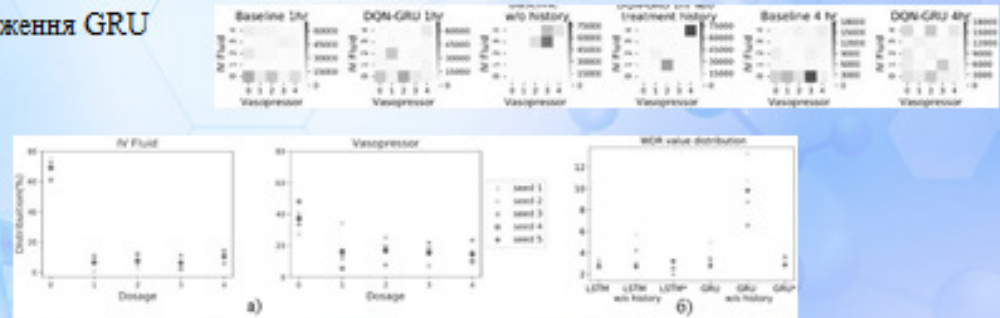


14

Рисунок Д.14 – Слайд №14

Аналіз результатів

Впровадження GRU



Порівняння результатів випадкових перезапусків: а) розподіл лікування за випадковими перезапусками у вихідному стані. У той час як дисперсія внутрішньовенних ін'єкцій невелика, розподіл вазопресора варіюється в різних насінні; б) порівняння значень оцінки WDR за випадковим насінням в кожному налаштуванні

15

Рисунок Д.15 – Слайд №15

Висновки

Під час виконання дипломної кваліфікаційної роботи магістра, було запропоновано новий ГНП для синтезу діагностичних нейромделей для вирішення прикладних задач медичного діагностування – ДПГQ-M. Також було розроблено ПЗ, що реалізує роботу запропонованого методу.

Запропонований метод використовує найсучасніші засади ГНП для підвищення результатів точності діагностуючих моделей. Розроблений програмний комплекс має практичну спрямованість та готовий до використання.

Під час розробки програми та аналізу розробленого програмного комплексу з'явилися ідеї з удосконалення, такі як: використання паралельних потужностей обчислювальних систем задля збільшення кількості незалежних агентів під час синтезу ШНМ та зіставлення спектр метапараметрів методу із даними у великих, фізіологічно достовірних імітаційних експериментах.

16

Рисунок Д.16 – Слайд №16

Дякую за увагу

[1]

[2]



[1] – <http://www.ntu.edu.ua/>
[2] – <http://niz.ntu.net/>

17

Рисунок Д.17 – Слайд №17