Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

«Затверджую»
в.о. завідуючого кафедри
комп'ютерних систем та робототехніки
_____ к. ф.-м. н., доцент Максим Хруслов
«___» грудня 2024 р.

Пояснювальна записка

до кваліфікаційної роботи
магістра

на тему «**МОДЕЛЬ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ У МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ**»

Спеціальність 123 – Комп'ютерна інженерія.
Галузь знань: 12 – Інформаційні технології.
Освітня програма «Комп'ютерна інженерія».

Захищено на засіданні

Екзаменаційної комісії № 42

протокол № __ від __.12.2024 р.

Оцінка _____ / _____

Голова Екзаменаційної комісії

_____ **СКОБ Ю. О.**

Виконала:
Студенка групи КІ–61
ВОЛИНЕЦЬ Катерина Андріївна

Керівник: к.т.н., доцент кафедри комп'ютерних систем та робототехніки
СТРІЛЕЦЬ Вікторія Євгенівна

Рецензент: к.т.н., доцент, в.о. зав. кафедри теоретичної та прикладної інформатики
МЕНЯЙЛОВ Євген Сергійович

Харків – 2024

АНОТАЦІЯ

Пояснювальна записка до магістерської кваліфікаційної роботи складається зі вступу, трьох розділів, висновків, списку використаних джерел і 4 додатків. Загальний обсяг роботи складає 81 сторінка, із яких 55 сторінок основної частини з 20 рисунками, 9 таблицями, списку використаних джерел із 40 найменувань та 4 додатками.

У даній роботі проведено дослідження і розробку алгоритмів обробки текстових даних у медичних інформаційних системах. Було створено прототип програмного забезпечення для підтримки медичних рішень.

Мета кваліфікаційної роботи – підвищення ефективності обробки медичних запитів та покращення взаємодії між пацієнтами та медичним персоналом через розробку та впровадження моделі автоматизованої класифікації текстових повідомлень у медичних інформаційних системах, створеної на основі методів обробки природної мови (NLP).

Об'єкт дослідження – процес автоматизованої класифікації текстових повідомлень і запитів у медичних інформаційних системах.

Предмет дослідження – алгоритми, математичні моделі та програмні методи обробки текстових запитів у медичних інформаційних системах.

Проблема, яка вирішується в кваліфікаційній роботі, полягає у створенні ефективної моделі, яка дозволяє автоматично обробляти текстові медичні запити і повідомлення, резюмувати їх і пропонувати лікарям можливі діагнози з урахуванням наявних даних.

Область застосування – медичні інформаційні системи (MIC), які використовуються для автоматизації процесів обробки текстової інформації, діагностики та підтримки прийняття рішень лікарями.

***Ключові слова*:** медичні інформаційні системи, резюмування тексту, медичні запити, машинне навчання, автоматизація, Python, обробка текстових даних.

ABSTRACT

An explanatory note to the master`s attestation work is created in the introduction, three sections, conclusions, a list of sources used and four additional substances.

The total volume of work is 89 pages, of which 57 pages of the main part with 30 figures, 9 table, 40 names of the list of used sources and 4 additions.In this work, research and development of the software model for automating the processing of medical queries have been conducted.

The purpose of the qualification work is to develop a software model for automating the processing of medical requests to provide doctors with the ability to receive and analyze possible diagnoses of patients based on their symptoms.

Object of research – automation processes for analyzing medical queries and diagnostic data to improve the quality of patient care.

Subject of research – algorithms, mathematical models, and software methods for processing text queries in medical information systems.

The problem solved in the qualification work is to create an effective model that enables the automatic processing of text medical queries, summarizing them, and offering possible diagnoses to doctors based on the available data.

Field of application – medical information systems (MIS) used to automate the processes of text information processing, diagnostics, and decision support for physicians.

Keywords: medical information systems, attention mechanism, automation, Python, text summarization, medical queries, data processing, diagnostics.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І УМОВНИХ ПОЗНАЧЕНЬ

ASP.NET	англ. Active Server Pages for .NET, укр. активні серверні сторінки для платформи .NET;
BERT	англ. Bidirectional Encoder Representations from Transformers, укр. двонаправлені подання з трансформерів;
BLEU	англ. Bilingual Evaluation Understudy, укр. двомовна оцінка моделі;
GC	англ. Garbage Collector, укр. збирач сміття;
GPT	англ. Generative Pre-trained Transformer, укр. генеративний попередньо навчений трансформер;
GRU	англ. Gated Recurrent Unit, укр. рекурентна нейронна мережа
HTTP	англ. Hypertext Transfer Protocol, укр. протокол передачі гіпертексту;
LSTM	англ. Long Short-Term Memory, укр. довготривала короткочасна пам’ять;
ML	англ. Machine Learning, укр. машинне навчання;
MPA	англ. Multi-Page Application, укр. багатосторінковий додаток;
MVC	англ. Model-View-Controller, укр. модель-вид-контролер;
NER	англ. Named Entity Recognition, укр. розпізнавання іменованих сутностей;
NLP	англ. Natural Language Processing, укр. обробка природної мови;
ORM	англ. Object-Relational Mapping, укр. об’єктно-реляційне відображення;
ROUGE	англ. Recall-Oriented Understudy for Gisting Evaluation, укр. метрика орієнтована на згадування;
RNN	англ. Recurrent Neural Network, укр. рекурентна нейронна мережа;
Seq2Seq	англ. Sequence to Sequence, укр. послідовність до послідовності;
SQL	англ. Structured Query Language, укр. мова структурованих запитів;
БД	база даних;
ДП	державне підприємство;

ЕМК	електронна медична картка;
ІС	інформаційна система;
ЛПЗ	лікувально-профілактичний заклад;
МІС	медична інформаційна система;
МОЗ	міністерство охорони здоров’я;
МПКС	медична приладо-комп'ютерна система;
СЕМК	системи електронної медичної картки;
ЦБД	центральна база даних

ВСТУП

Інформаційні процеси (пошук, збирання, зберігання, передавання, опрацювання, використання, захист інформації) присутні у всіх областях медицини і галузі охорони здоров’я. З кожним роком збільшується кількість інформаційних запитів від пацієнтів, медичних працівників і систем, що вимагає впровадження автоматизованих рішень для обробки таких даних. Одним із важливих завдань є розробка технологій для класифікації текстових повідомлень, які циркулюють між різними учасниками процесу лікування, такими як лікарі, медичні установи та пацієнти.

Актуальність роботи. Основне завдання дослідження полягає в необхідності створення та впровадження автоматизованої моделі класифікації повідомлень, що дозволить спростити та прискорити процес обробки медичних запитів. У сфері охорони здоров’я важливо, щоб усі запити, повідомлення та звернення класифікувалися точно та швидко, що допоможе уникнути затримок у наданні медичної допомоги та покращить взаємодію між пацієнтами й медичним персоналом.

Подібна модель дозволить автоматично визначати пріоритетність повідомлень, розподіляти їх за відповідними категоріями (наприклад, запити на отримання медичної консультації, результати обстежень чи адміністративні питання), а також підвищить швидкість реагування на термінові запити. Це дозволить медичним працівникам сконцентруватися на більш важливих аспектах лікування, зменшуючи навантаження на адміністративний персонал.

Розробка автоматизованої системи класифікації повідомлень у медичній сфері має потенціал значно підвищити загальну ефективність медичних установ, оптимізуючи роботу із запитами пацієнтів, забезпечуючи зручність та оперативність обробки даних і підтримуючи безперебійний обмін інформацією між всіма учасниками медичного процесу.

Метою дослідження є підвищення ефективності обробки медичних запитів та покращення взаємодії між пацієнтами та медичним персоналом через

розробку та впровадження моделі автоматизованої класифікації текстових повідомлень у медичних інформаційних системах, створеної на основі методів обробки природної мови.

Об’єкт дослідження – процес автоматизованої класифікації текстових повідомлень і запитів у медичних інформаційних системах.

Методи дослідження: методи штучного інтелекту, аналізу даних, методи машинного навчання, методи проектування програмних засобів, методи бізнес-аналізу, методи тестування програмних засобів.

Предмет дослідження – алгоритми, математичні моделі та програмні методи обробки текстових запитів у медичних інформаційних системах.

Завдання дослідження:

1. Проаналізувати існуючі підходи та методи обробки текстових даних у медичній сфері.

2. Підготувати набір даних для навчання моделі класифікації медичних повідомлень.

3. Розробити модель автоматизованої класифікації медичних текстових повідомлень.

4. Навчити модель на основі підготовлених даних та провести її оптимізацію для забезпечення високої точності.

5. Протестувати модель на реальних медичних запитаннях, оцінити її ефективність та якість.

РОЗДІЛ 1.
АНАЛІЗ МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1 Класифікація медичних інформаційних систем

Люди щодня стикаються з великим обсягом інформації. Отримана інформація стосується як особистого життя, так і професійної діяльності людей. Вся продукована інформація має оброблятись, зберігатись а також передаватись з високою ефективністю для функціонування організацій. З цією метою створюються різноманітні інформаційні системи, що збільшують продуктивність у всіх сферах сучасного світу.

Інформаційна система (ІС) – сукупність організаційних та технічних засобів для збереження й обробки інформації з метою забезпечення потреб користувача.

За ДСТУ 2392-94: інформаційна система – комунікаційна система, що забезпечує збирання, пошук, оброблення та пересилання інформації [1]. Законодавство України визначає інформаційну систему, як організаційно-технічну систему, в якій реалізована технологія обробки інформації з використанням технічних і програмних засобів [2].

Ключовим поняттям інформатизації системи охорони здоров'я є інформаційна система. Тому розглянемо класифікацію медичних інформаційних систем за призначенням.

МІС – це програмно-технічний комплекс, що готує й забезпечує процеси збирання, зберігання разом із обробкою інформації в медицині й галузі охорони здоров'я. Розглянемо їх докладніше.

Класифікація МІС ґрунтується на ієрархічному підході, що відповідає багаторівневій структурі охорони здоров’я [3]:

а) МІС базового рівня, основна ціль яких – комп’ютерна підтримка роботи лікарів різних спеціальностей. По виконуваних задачах виділяють:

– інформаційно-довідкові комплекси (призначення-пошук медичної інформації по запиту працівника);

– консультативно-діагностичні мережі (діагностика патологічних станів, прогноз а також рекомендації по методах лікування),

– приладо-комп'ютерні структури (інформаційна підтримка й автоматизація діагностичного та лікувального процесу, здійснюваних при безпосередньому контакті з організмом хворого);

– автоматизовані робочі місці спеціалістів (автоматизація всього технологічного процесу лікаря);

b) МІС рівня лікувально-профілактичних закладів (ЛПЗ) – об'єднують всі інформаційні потоки ЛПЗ в єдину систему й автоматизують різні види діяльності закладу;

c) МІС територіального рівня представлені:

– ІС територіального органу охорони здоров'я,

– МІС для вирішення медико-технологічних задач, що забезпечують інформаційною підтримкою діяльність медробітників спеціалізованих служб,

– комп'ютерні телекомунікаційні медичні мережі, що забезпечують створення єдиного інформаційного простору на рівні регіону;

d)державний рівень

Інформаційні системи лікувально-профілактичних закладів є найбільш затребуваними. Вони складаються із декількох блоків:

– інформаційні структури консультативних центрів забезпечують функціонування відповідних підрозділів й інформаційну підтримку лікарів під час консультацій, діагностики та прийнятті рішень у невідкладних станах;

– банки інформації медичних служб вміщають дані про якісний і кількісний склад робітників організації й інші необхідні дані;

– реєстраційно-статистична підсистема реєструє всі події й факти, що відбуваються на лікувальному закладі. Дані підсистеми зменшують повсякденну працю робітників, допомагають у оперативному керуванні, забезпечують отримання всіх статистичних даних, проводять фінансовий і економічний аналіз та організовують спільну роботу різних служб. Ці дії дозволяють зменшити й часові, а також фінансові трати, здійснені через помилки персоналу;

– лабораторна інформаційна система керує даними, що поступають із різних джерел й об’єднує їх в базу даних клініко-діагностичної лабораторії. Ця система забезпечує скорочення невиробничих витрат лабораторії, відмову від рукописних журналів, відстеження та оцінку якості досліджень.

– персоніфіковані регістри вміщують інформацію про прикріплене населення на основі сформованої історії хвороби або амбулаторної карти;

– скринінгові мережі забезпечують проведення долікарського профілактичного огляду населення та виявлення груп ризику.

Важливим різновидом спеціалізованих МІС являються медичні приладо-комп’ютерні системи (МПКС).

Використання комп’ютера в поєднанні з вимірювальною і керуючою технікою всередині медицини дозволило створити нові засоби для забезпечення автоматизованого збору інформації про стан хворого, її обробка в реальному часі та керування станом пацієнтів. Цей процес призвів до створення МПКС, які покращили інструментальні методи дослідження і інтенсивну терапію. Основною відмінністю цього класу МІС є робота в умовах безпосереднього контакту з об’єктом дослідження в реальному класі. Вони представлені складними програмно-апаратними комплексами. Для роботи МПКС, окрім обчислюваної техніки, необхідні спеціальні медичні прилади, обладнання, телетехніка, засоби зв’язку.

Типовими представниками МПКС є медичні системи для моніторингу під час операцій, комплекси комп’ютерного аналізу томографії, ультразвукової діагностики, радіографії, структури автоматизованого аналізу даних мікробіологічних і вірусологічних досліджень.

1.2 Медичні інформаційні системи в Україні

До недавнього часу інформаційні системи в Україні перебували у стагнації через брак фінансування, матеріалів та відсутність широкого доступу до Інтернету. У 2016 році було розпочато реформу охорони здоров’я. Згідно з концепцією реформи фінансування охорони здоров’я, було замінено застарілу

модель [4]. Це було пов’язано з неможливістю закупівлі медичних послуг без постійного використання даних про медичні параметри надання послуг на рівні закладу, а також індивідуальної історії хвороби пацієнта. Дані необхідні для фінансового планування, укладання договорів та моніторингу.

У 2018 році було розпочато співпрацю між державою та приватним сектором для розвитку eHealth. Приватні МІС зобов'язані пройти сертифікацію державного підприємства (ДП) «Електронне здоров’я» для підключення до центральної бази даних eHealth. Регулярно здійснюються перевірки та оновлення цих систем для підтримання їхньої відповідності вимогам безпеки та функціоналу. Якщо структури не відповідають вимогам після тестування, їх можуть відключити від бази даних.

Система електронної охорони здоров’я має надавати такі послуги:

– управління центральною базою даних та технічна підтримка;

– забезпечення безперервної роботи центральної бази даних;

– забезпечення розвитку, оновлення та підтримки програмного забезпечення бази даних (БД);

– приймання рішень про відключення та призупинення доступу МІС до БД;

– вжиття заходів щодо захисту інформації, яка міститься в БД [5].

Оскільки eHealth функціонує на національному рівні та не може виконувати функції госпітальної інформаційної мережі, в Україні існує низка МІС приватного сектору. Для того, щоб підключитися до бази даних eHealth, вони повинні бути сертифіковані ДП «Електронне здоров’я». ДП «Електронне здоров’я» постійно оновлює функціонал на рівні бази даних eHealth та повторно тестує МІС. Це означає, що МІС може бути надалі від’єднана від центральної бази даних (ЦБД) eHealth за результатами повторного тестування [6].

Компанії «Медсистем», CIET, Ukrmed software, TherDep та Medexpert пропонують свої МІС. Більшість цих систем засновані на клієнт-серверній архітектурі та забезпечують формування стандартних звітів та форм для

міністерства охорони здоров’я (МОЗ). Характеристики систем наведені в таблиці 1.1.

Таблиця 1.1

Зведені характеристики систем

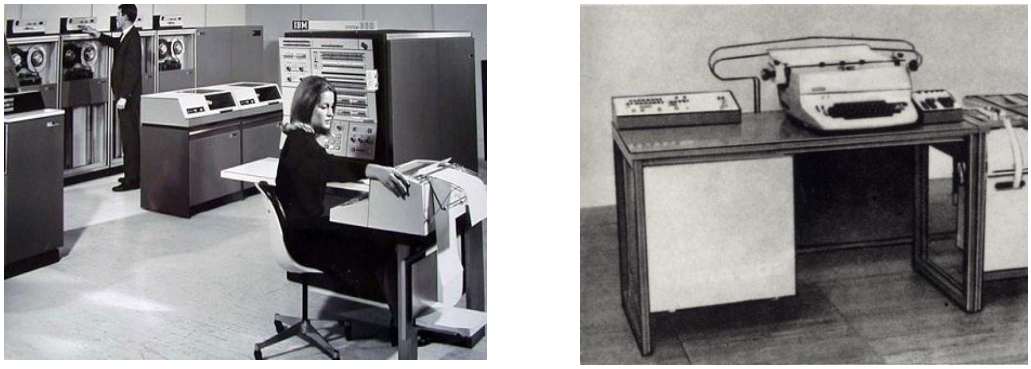
Характеристика	Назва МІС				
	Archi Med	Доктор Елекс	TherDep	Каштан	Укрмедсофт
Клієнт-серверна архітектура	+	+	+	+	+
Рівень вимого до апаратних засобів	Середній	Середній	Низький	Високий	Високий
Здатність співпрацювати з медичним обладнанням	+	+	–	–	–
Наявність механізмів захисту інформації	+	+	–	–	–
Наявність вебінтерфейсу	+	+	–	–	–
Під’єднання додаткових модулів	+	+	+	+	+
Механізми статистичного оброблення даних	+	+	+	+	+
Використання баз даних	+	+	+	+	+
Можливість конфігурації системи	+	+	+	+	+

Однак існують проблеми, які перешкоджають поширенню та розвитку МІС в Україні. Консерватизм лікарів, які все ще заповнюють інформацію на папері та копіюють її в МІС. Відсутність сучасного обладнання та незадовільний стан наявного обладнання. Інтернет недостатньо поширений в країні. Лікарні майже примусово переходять на МІС, а працівникам не роз’яснюються їхні функції та переваги.

1.3 Досвід впровадження МІС у світі

Останніми роками розробка та впровадження інформаційних систем у медичних закладах отримала значний поштовх у всьому світі. У сімдесятих роках минулого століття лікарняні МІС були широко розповсюджені, але дуже

дорогі та тому використовувалися лише у великих, завантажених лікарнях. Ці структури працювали на мейнфреймах, тобто великих комп’ютерах або мінікомп’ютерах [7] (рис. 1.1).



а) б)

Рисунок 1.1 – Госпітальні ІС: а) майн фрейм; б) мінікомп’ютер

З поширенням персональних комп’ютерів і комунікаційних технологій з використанням локальних, глобальних і бездротових мереж та Інтернету, охорона здоров’я стала сучасною інтегрованою інформаційною системою, що поєднує окремі модулі. У червні 2019 р. Міністерство охорони здоров'я України оприлюднило для громадського обговорення проект Концепції інформатизації охорони здоров'я. Основою стратегії визнано тактику орієнтованості на пацієнта, що означає безперервне накопичення та зберігання даних з прив'язкою до облікового запису пацієнта в його електронній медичній картці, надання пацієнту, як суб’єкту персональних даних, можливості керувати власними медичними даними й доступом до них.

Організація або комісія, відповідальна за інтеграцію всіх національних інформаційних систем в межах ЄС і надання громадянам єдиної електронної медичної картки на всій території ЄС, впровадила єдину програму електронної охорони здоров’я (eHealth). Європейський Союз активно інвестує в програми електронної охорони здоров'я, сприяючи стандартизації, страховому покриттю та обробці медичної інформації пацієнтів за допомогою інформаційних технологій. Загальний бюджет програми EU4Health на 2021-2027 роки становить

5,3 млрд євро, що є безпрецедентною фінансовою підтримкою ЄС у сфері охорони здоров'я [8]. Крім того, Європейська комісія схвалила участь України в кількох проєктах, підтвердивши понад 4,6 млн євро загального фінансування на їх реалізацію [9]. Ці інвестиції сприяють розвитку електронної охорони здоров'я, підвищуючи ефективність медичних послуг та забезпечуючи доступність якісної медичної допомоги для громадян.

Дослідження показують, що у 2009 році 48,2% лікарів у США використовували системи електронних медичних записів (ЕМК) [10], а у 2012 році – 72% лікарів, причому 22% лікарів використовували електронні медичні записи з базовими функціями, що відповідають федеральним вимогам, порівняно з 22% у 2009 році майже 40% [11]. З того часу темпи зростання використання ЕМК знизилися, і у 2018 році їх використовували 86% лікарів [12].

Переваги використання медичних інформаційних систем в лікарні залежать від виду МІС і успішності її впровадження в конкретному закладі. Згідно зі звітом про використання ЕМК лікарями за 2023 рік, опублікованим Національним центром медичної статистики, лікарі вбачають такі переваги використання електронних медичних карток [13]:

– покращення якості медичних послуг (74%);
– можливість віддаленого доступу до інформації (74%); та
– попередження про важливі результати аналізів (52%); та
– попередження про можливі помилки при виписуванні рецептів (43%);
– заохочення до вживання запобіжних заходів (40%);
– зменшення кількості лабораторних досліджень (30%);
– спрощення взаємодії з пацієнтом (25%).

Важливим аспектом проєктування МІС є забезпечення інтероперабельності при інтеграції різних рівнів систем в єдину національну МІС в майбутньому, а також зниження витрат при заміні частин мережі на більш сучасні. З цією метою існує низка організацій, які займаються стандартизацією інформаційних технологій у сфері охорони здоров’я. Деякі з них перераховані нижче:

– CCOW – Робоча група з об’єктів клінічного контексту;

– CCR – Continuity of Care Record (Безперервність медичної допомоги);

– CEN – Європейський комітет зі стандартизації;

– DICOM – Протокол цифрових зображень і комунікацій в медицині;

– EDIFACT – електронний обмін даними для адміністрування, торгівлі та транспорту;

– HL7 – Стандарт обміну медичною інформацією, версія 2 та версія 3;

– HL7 CDA – HL7 Clinical Document Architecture;

– IEEE – Інститут інженерів з електротехніки та електроніки Америки;

– IHE – Інститут підприємств охорони здоров’я;

– ISO/TC215 – Технічний комітет 215;

– OpenEHR – Електронна медична картка;

– SNOMED – Систематична номенклатура медицини.

Також активно розробляються міжнародні стандарти для зберігання, обміну та швидкого доступу до медичних даних. Стандарт ISO 13606-1:2008 «Медична інформатика. Обмін електронними медичними записами. Частина 1: Еталонна модель визначає еталонну модель передачі електронних медичних записів. Вимоги до інформаційної сумісності персональних комп’ютеризованих медичних записів також визначені в стандарті ISO 21549:2004 Медична інформатика. Частина 1: Загальна структура; Частина 2: Загальні об’єкти; Частина 3: Обмежені клінічні дані; Частина 4: Розширені клінічні дані; FDIS Частина 5: Ідентифікаційні дані; FDIS Частина 6: Адміністративні дані; ISO 21549-7:2007 Дані про лікарські засоби. МІС повинні забезпечувати безпечний обмін чутливою інформацією з використанням сучасних телекомунікацій.

1.4 Економічні переваги МІС

Для визначення економічної ефективності впровадження інформаційних технологій в лікарнях варто скористатися дослідженнями закордонних країн, які вже накопичили багаторічний досвід комп’ютеризації охорони здоров’я. Наразі найпоширенішою медичною інформаційною системою є електронна медична

картка (ЕМК). У європейських країнах та США ця система майже повністю замінила традиційні паперові медичні картки.

Для амбулаторно-поліклінічних закладів зазначають такі переваги електронних медичних карток:

– економія витрат на вилучення карток;
– економія на лабораторних дослідженнях;
– економія витрат на фармацевтичні препарати;
– економія на рентгенології у стаціонарних відділеннях;
– покращення роботи з документацією для медсестер;
– переваги електронних медичних записів;
– економія на лабораторних дослідженнях;
– економія на фармацевтичних витратах;
– скорочення тривалості перебування в лікарні.

Нижче наведено більш детальний аналіз переваг за кожним пунктом.

1.4.1 Скорочення дій з медичними картами

Амбулаторний сектор

Економічні вигоди реалізуються в амбулаторному секторі, оскільки паперова документація більше не потрібна, що усуває потребу в спеціальному персоналі для ведення цих записів. Заощадження від цього оцінюється в понад 60%.

Виписка паперової картки займає приблизно чотири хвилини, а кожен лікар виписує в 1,6 раза більше пацієнтів на день. Якщо нормальне навантаження становить 15 пацієнтів на день, то один лікар витрачає 384 робочі години на рік. Це може коштувати лікарням 5530 доларів США на рік, або 1,7 мільярда доларів США в національному масштабі.

Стаціонарний сектор

У стаціонарному секторі ЕМК, які дозволяють медичному персоналу отримати доступ до записів пацієнтів, заощаджують час, що витрачається на

документацію та збір інформації. Вони також можуть зменшити витрати, пов’язані з паперовими формами, і запобігти пропуску процедур.

Час, що витрачається на паперову роботу, можна скоротити: системи електронної медичної картки (СЕМК) може зменшити кількість карткових транзакцій на 60-70%, а кількість персоналу, залученого до цих транзакцій, – на 50%. Заощаджений час можна використати по-різному:

– зменшити кількість медичного персоналу;

– пролікувати більше пацієнтів.

Коли у відділеннях інтенсивної терапії використовували ЕМК, медсестри витрачали на 52 хвилини менше часу на паперову роботу. У різних дослідженнях економія часу медсестер завдяки впровадженню електронних медичних карток оцінюється в межах від 10% до 20% [14-15].

Прогнозоване збільшення потреби в медсестрах і заміна лікарів медсестрами у зв’язку зі зростанням населення, старінням і розвитком медицини, як очікується, дозволить щорічно економити 7,1 мільярда доларів [16].

1.4.2 Економія на лікарських препаратах

Комп’ютеризація виписування рецептів та впровадження модулів підтримки прийняття медичних рішень дозволяють лікарям користуватися електронними базами даних про лікарські засоби, їх комбінації, протипоказання тощо. Це дає можливість лікарям підбирати лікування відповідно до медичних стандартів, враховуючи ціну ліків, розумні комбінації та оптимальну тривалість застосування. За різними оцінками, система альтернативного вибору ліків може знизити витрати на ліки на 15%. Враховуючи щорічні витрати на ліки в стаціонарному секторі США, це означає економію в розмірі 5,7 млрд доларів. В амбулаторному секторі економія оцінюється в 12,7 млрд доларів США.

1.4.3 Економія на лабораторних та амбулаторних дослідженнях

Якщо СЕМК оснащені модулем призначення процедур і тестів, а також модулем підтримки прийняття медичних рішень, можна зменшити кількість

непотрібних і дублюючих тестів. Це відбувається тому, що система дозволить лікарям оскаржити всі проведені аналізи та запропонувати найкращу схему їх проведення з урахуванням застосованих препаратів, переходу на іншу фазу лікування та інших факторів. Економія коштів у розмірі 22% в амбулаторному секторі та 11,8% у стаціонарному секторі може бути досягнута, наприклад, внаслідок структурування призначень на аналізи. Загалом, за оцінками, економія на лабораторних дослідженнях може скласти 3 млрд доларів в стаціонарному секторі та 2,2 млрд доларів в амбулаторному секторі.

Економія витрат на рентген в амбулаторному секторі може становити 14%, що дорівнює 3,6 млрд доларів.

1.4.4 Скорочення термінів госпіталізації

Під час госпіталізації пацієнта медичний персонал часто стикається з різними втратами часу, які можуть суттєво вплинути на ефективність лікування. Однією з основних проблем є затримки, пов’язані з пошуком необхідної медичної документації. Такі втрати можна мінімізувати за допомогою системи медичних записів, яка може скоротити тривалість перебування пацієнта в лікарні. За різними оцінками, час госпіталізації можна скоротити на 10-20%. Таке скорочення часу госпіталізації може призвести до економії 36,7 млрд доларів.

1.4.5 Зіставлення витрат та результатів впровадження МІС

Витрати на впровадження СЕМК складаються з двох основних компонентів: капітальних витрат та витрат на обслуговування. Капітальні витрати включають витрати на закупівлю програмного забезпечення, створення необхідної локальної інфраструктури, а також оплату праці фахівців, що відповідають за впровадження і модернізацію мережі. Ці витрати, як правило, розраховані на кілька років, протягом яких відбувається повноцінна інтеграція структури у робочі процеси медичних закладів. Витрати на обслуговування зазвичай оцінюються як відсоток від капітальних витрат і включають технічну

підтримку, оновлення програмного забезпечення та інші регулярні витрати, пов’язані з підтримкою функціонування системи.

Загалом, потенційна економія майже вдвічі перевищує середній показник. Загальні економічні вигоди обох секторів у США підсумовані в таблиці 1.2.

Таблиця 1.2 дозволяє оцінити ефекти від впровадження інформаційних технологій у закладах охорони здоров'я. Враховуючи, що стаціонарний сектор витрачає 6,7 млрд. доларів щорічно, економічна вигода в 31,3 млрд. доларів є дуже значною, що означає рентабельність інвестицій на рівні 367% [17].

Таблиця 1.2

Сумарні вигоди від впровадження СЕМК в США

№	Види переваг	Оцінки економії, %	Характеристика
1	Час роботи з документами	11	Заробітна плата персоналу по відділенням за рік
2	Лікарські препарати	15	Вартість лікарських препаратів за рік
3	Лабораторні дослідження	11,8	Сумарні річні витрати лабораторій
4	Радіологічні дослідження	14	Сумарні витрати відділень, що ведуть радіологічні дослідження
5	Термін госпіталізації	15	Вартість ліжко-місць, помножена на кількість ліжко-днів за рік

1.5 Постановка задачі дослідження

Підсумовуючи викладене в попередніх розділах, можна стверджувати, що метою моєї дипломної роботи є розробка автоматизованої моделі резюмування текстових медичних запитів для полегшення роботи медичного персоналу та забезпечення швидкого доступу до ключової інформації.

Для досягнення поставленої мети дипломної роботи необхідно вирішити такі задачі:

– дослідити сучасні підходи до автоматизації обробки текстових даних у медичних інформаційних системах (МІС), включаючи можливі архітектури;

– підготувати навчальні дані для моделі, включаючи медичні запити та відповідні анотації з використанням реальних даних;

– спроектувати архітектуру моделі, орієнтовану на автоматизоване резюмування текстових медичних запитів;

– розробити програмне забезпечення для навчання та оптимізації моделі, забезпечуючи її інтеграцію в МІС;

– протестувати моделі на ефективність та точність резюмування;

– підготувати рекомендації для впровадження моделі в роботу медичних закладів, включаючи можливості подальшого розвитку системи.

Це дозволить створити ефективний інструмент для автоматизованого аналізу текстових даних, що оптимізує процес взаємодії між лікарями та пацієнтами.

Висновки за розділом 1

У цьому розділі визначено законодавчі положення про медичні інформаційні структури. Описано стан розвитку медичних інформаційних технологій в Україні та досвід впровадження медичних інформаційних технологій у світі.

Також було проаналізовано в чому полягають вигоди від впровадження інформаційних систем, зокрема СЕМК, в амбулаторному та стаціонарному секторах лікарні.

РОЗДІЛ 2.
ЗАДАЧІ МАШИННОГО НАВЧАННЯ В МІС

2.1 Задачі машинного навчання в медичних інформаційних системах

Машинне навчання (англ. Machine learning, ML) значно впливає на розвиток МІС, допомагаючи автоматизувати процеси, поліпшити якість обслуговування пацієнтів і сприяти інноваціям у медичній галузі [18]. В медицині найчастіше використовуються методи машинного навчання, пов’язані з розпізнаванням та обробкою зображень. А це: робота з рентгенівськими знімками молочної залози на предмет наявності злоякісних утворень, серцево-судинної системи з метою виявлення дефектів на шляху проходження крові через судини, легень для діагностування легеневих захворювань тощо. Розглянемо детально кілька ключових завдань машинного навчання, які можна вирішити у МІС.

Класифікація запитань у медичній інформаційній системі є надзвичайно важливим завданням, оскільки пацієнти та медичні працівники задають широкий спектр запитань, що стосуються різних лікарських аспектів, таких як діагнози, лікування, причини хвороб, побічні ефекти ліків тощо [19]. Щоб автоматично сортувати ці запитання за категоріями, можна використовувати алгоритми класифікації з підкріпленим навчанням.

Виявлення фокусу запитання. У МІС важливо розуміти, на чому зосереджене запитання, щоб надати відповідь або направити користувача до відповідного розділу системи. Фокус може бути на хворобі, лікуванні, ліках або обстеженні. Виявлення фокусу запитання можна автоматизувати за допомогою моделей, що спеціалізуються на розпізнаванні іменованих сутностей (англ. NER, named entity recognatin).

Пошук відповідей. Пошук релевантної відповіді на задане запитання є однією з найважливіших задач у лікарських ІС.

Резюмування запитань полягає в автоматичному скороченні текстів із збереженням ключової інформації [20]. У контексті МІС це завдання може бути

використано для спрощення та скорочення лікарських запитань, особливо коли користувачі або пацієнти ставлять занадто складні або довгі запитання. Автоматичне резюмування допомагає зробити запитання більш доступними для швидкого розуміння лікарями або медичними системами.

Визначення парафраз – це задача, в якій визначається, чи є два запитання або тексти семантично подібними [21]. У МІС це може бути корисно для групування схожих запитів або для автоматичного пошуку відповідей на подібні запитання. Враховуючи, що пацієнти можуть задавати одні й ті самі питання різними способами, система, здатна розпізнавати парафрази, дозволяє уникнути дублювання та підвищує ефективність обробки запитів.

NER *(англ. Named Entity Recognition, розпізнавання іменованих сутностей)* полягає у витягуванні важливих медичних термінів з тексту, таких як назви хвороб, симптомів, ліків тощо [22]. В МІС це завдання дозволяє автоматично структурувати медичну інформацію та робити її більш доступною для подальшого аналізу та обробки.

Витяг медичної інформації. Задача витягу медичної інформації полягає в автоматичному виявленні та вилученні ключових даних з текстів, наприклад, з медичних записів, запитань пацієнтів або наукових статей [23]. Це завдання є важливим для структуризації даних, створення аналітичних звітів та полегшення пошуку відповідей в МІС.

Отже, застосування МН у МІС відкриває великі можливості для автоматизації процесів, поліпшення ефективності обробки даних і підвищення якості обслуговування пацієнтів. Перераховані завдання дозволяють створювати більш ефективні системи для підтримки процесів. В рамках кваліфікаційної роботи детально розглянуто задачу резюмування запитань, оскільки вона є важливою для спрощення роботи з довгими текстовими запитами та полегшення аналізу інформації медичними фахівцями.

2.2 Задача обробки природної мови в МІС

Задача обробки природної мови (англ. NLP, natural language processing) у рамках автоматизованої класифікації повідомлень у лікарських ІС є надзвичайно важливою для ефективної роботи з великими обсягами текстової інформації. Одним з таких завдань є резюмування запитань, яке полягає в автоматичному скороченні тексту при збереженні ключової інформації. В контексті МІС це завдання стає особливо актуальним, коли пацієнти або лікарі подають складні або довгі запитання.

Мета резюмування – зробити такі запитання більш доступними та зрозумілими для медичних працівників або систем, які обробляють ці запитання. Це також дозволяє знизити навантаження на медичний персонал, який може швидше ознайомитися зі стиснутими версіями запитань та прийняти правильні рішення.

Автоматизація процесу резюмування текстів у МІС має такі ***переваги***:

– **Ефективність.** Ручна обробка великої кількості текстових запитів вимагає значних ресурсів. Завдяки автоматизованим системам резюмування лікарі можуть швидко ознайомитися з суттю запитання, не втрачаючи часу на вивчення детальних повідомлень. Це особливо важливо, коли мова йде про невідкладні медичні випадки або в умовах великої кількості запитів.

– **Спростування медичних процесів.** Автоматичне резюмування спрощує роботу медичних працівників, оскільки дозволяє отримати стиснуту версію складних медичних питань. Це прискорює прийняття рішень та покращує якість обслуговування пацієнтів.

– **Адаптація до великих обсягів даних.** У сучасних МІС система резюмування дозволяє обробляти великі обсяги запитів без втрати ефективності. Це робить її важливим інструментом задля підтримки масштабованих систем охорони здоров’я, де обробляються тисячі запитань щодня.

– **Персоналізація.** За допомогою моделей, що враховують специфіку запитань, система може адаптувати резюме під конкретного користувача, надаючи інформацію в зручному та зрозумілому форматі.

Попри перелічені переваги, існують й певні ***недоліки***, які можуть обмежувати застосування автоматичного резюмування в медичних інформаційних системах: [24]

– **Втрата важливої інформації.** Автоматичне скорочення тексту може призвести до втрати ключових деталей, що є критичними в інтересах медичних рішень. Наприклад, важлива інформація про алергії або особливості попереднього лікування може бути втрачена під час скорочення.

– **Складність розуміння медичного контексту.** Моделі NLP можуть не завжди правильно інтерпретувати медичні терміни або їх взаємозв'язок. Різні контексти можуть значно впливати на правильність резюмування, і в деяких випадках неправильна інтерпретація може призвести до серйозних помилок.

– **Потреба у великих і якісних даних.** Для навчання моделей автоматичного резюмування потрібні великі обсяги даних, які повинні бути ретельно анотовані. У медичних інформаційних системах це може бути складно через чутливість даних і необхідність забезпечення конфіденційності.

Таким чином, автоматизація резюмування запитань у МІС є потужним інструментом для оптимізації медичних процесів, але вимагає ретельного підходу до налаштування моделей та забезпечення якісних даних для навчання.

2.3 Моделі класифікації повідомлень

Для резюмування медичних запитань застосовуються сучасні методи NLP, серед яких найбільш популярними є моделі послідовність-послідовність (Seq2seq, англ. Sequence to Sequence) [25]. Ці моделі дозволяють ефективно перетворювати довгі та складні питання на короткі й інформативні резюме.

Seq2Seq-моделі (рис. 2.1) мають кілька різновидів, які відрізняються архітектурою та методами покращення. Ці варіації були розроблені для вирішення специфічних проблем, таких як обробка довгих послідовностей, покращення збереження інформації або підвищення точності моделі.

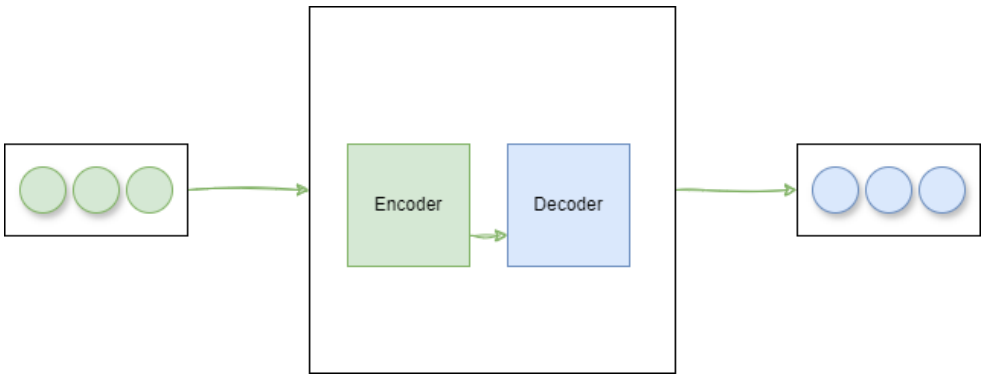


Рисунок 2.1 – Модель Seq2seq

Основні різновиди Seq2Seq-моделей:

1. *Класичні Seq2Seq-моделі* на основі рекурентних нейронних мереж (RNN, англ. Recurrent Neural Network).

Це базова версія Seq2Seq-моделі, яка використовує рекурентні нейронні мережі як енкодер і декодер (рис. 2.2). Вони ефективно працюють з послідовними даними, такими як текст, але мають обмеження у випадку довгих послідовностей через "затухання градієнтів" і втрату пам'яті.

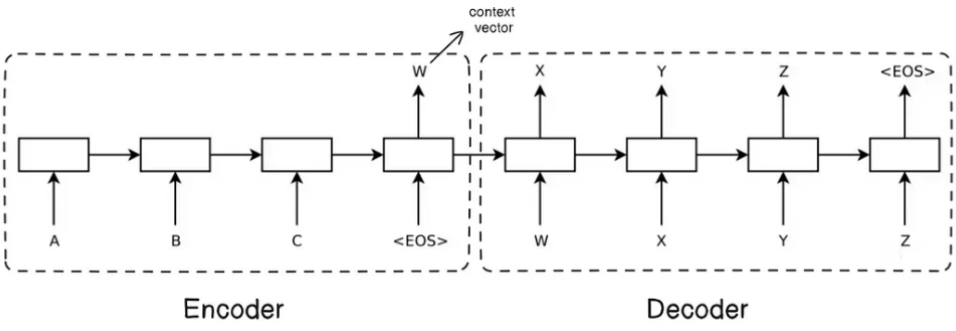


Рисунок 2.2 – Архітектура базової Seq2Seq-моделі [20]

Базова Seq2Seq-модель має такі основні складові:

– вхідна послідовність – це послідовність токенів (наприклад, слів у реченні) подається в енкодер;

– енкодер, який обробляє надану послідовність, створює і виводить контекстний вектор;

– контекстний вектор – це вектор, який фіксує інформацію з усієї вхідної послідовності;

– декодер генерує вихідну послідовність токен за токеном, опрацьовуючи контекстний вектор;

– вихідна послідовність – згенерована декодером послідовність, яка є виходом моделі.

2. *Seq2Seq з механізмом уваги* (Attention).

Seq2Seq з механізмом уваги (рис. 2.3), дозволяє декодеру звертати увагу на різні частини вхідної послідовності під час генерації вихідного тексту [26]. Це вирішує проблему втрати інформації при роботі з довгими послідовностями.

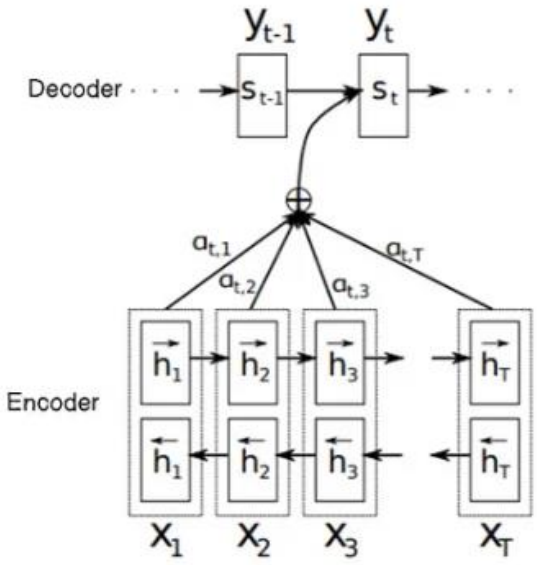


Рисунок 2.3 – Архітектура Seq2Seq моделі з механізмом уваги [21]

Seq2Seq-модель відрізняється від базової змінами в енкодері, декодері та додаванням механізму уваги. А саме:

– енкодер обробляє послідовність і виводить послідовність прихованих станів;

– шар уваги: на кожному кроці декодування механізм уваги обчислює вектор контексту як зважену суму прихованих станів кодера. Вагові коефіцієнти

визначаються на основі відповідності кожного прихованого стану поточному етапу декодування;

– декодер використовує контекстний вектор з шару уваги для генерації кожного токену вихідної послідовності.

3. *Seq2Seq на основі LSTM* (*англ. Long Short-Term Memory)*

Seq2Seq на основі LSTM дозволяє зберігати важливу інформацію протягом довгого часу [27].

4. *Seq2Seq на основі GRU (англ. Gated Recurrent Unit)*

GRU – це спрощена версія LSTM, яка працює швидше та використовує менше параметрів [28]. GRU також добре вирішує проблему затухання градієнтів, зберігаючи інформацію протягом тривалих послідовностей.

5. *Seq2Seq на основі трансформерів*.

Seq2Seq на основі трансформерів є одним із найбільш ефективних різновидів Seq2Seq-моделей, які використовуються для NLP [29]. Вони не залежать від послідовного аналізу слів, як це роблять RNN, тому можуть паралельно обробляти весь текст. Трансформери використовують механізм самоуваги (self-attention), що дозволяє кожному слову у вхідній послідовності бути пов'язаним з усіма іншими словами.

6. *Bidirectional Seq2Seq* (двонаправлена Seq2Seq).

У цій моделі енкодер обробляє послідовність у двох напрямках: зліва направо і справа наліво. Це допомагає захопити контекст з обох боків кожного слова, покращуючи якість резюмування або перекладу.

7. *Pre-trained Seq2Seq-моделі* з Transfer Learning.

Попередньо треновані моделі, такі як BERT (англ. Bidirectional Encoder Representations from Transformers), GPT (англ. Generative pre-trained transformer) або T5, можна використовувати для завдань Seq2Seq завдяки перенесенню навчання (transfer learning) [30-31]. Ці моделі спочатку навчаються на великих загальних корпусах текстів, а потім адаптуються до конкретних задач, таких як резюмування або переклад.

Seq2Seq-моделі мають безліч варіацій, кожна з яких має свої переваги та недоліки. Вибір конкретної моделі залежить від поставленого завдання, обсягу даних та ресурсів для навчання. Для задачі резюмування медичних запитань використано Seq2Seq з механізмом уваги (Attention). Ця модель вирішує проблему втрати інформації під час обробки довгих текстів, що є важливим у медичних запитах, де кожна деталь може мати значення. Крім того, механізм уваги дозволяє декодеру звертати увагу на важливі частини вхідної послідовності, що значно покращує якість резюмування.

2.4 Seq2Seq-модель з механізмом уваги для резюмування

Seq2Seq-модель з механізмом уваги є потужним інструментом для вирішення задач автоматизованого резюмування, особливо коли йдеться про роботу з довгими текстовими послідовностями, такими як лікарські запити. Ця модель здатна забезпечувати точне і релевантне скорочення тексту, зберігаючи основну інформацію, необхідну для розуміння суті запиту. Механізм уваги значно покращує роботу Seq2Seq, дозволяючи моделі звертати увагу на важливі частини тексту під час генерації резюме.

Компоненти Seq2Seq-моделі з механізмом уваги:

1. *Енкодер* (Encoder) відповідає за обробку вхідної послідовності та перетворення її у векторне представлення. Це векторне представлення узагальнює всю інформацію з вхідного тексту, яку потім використовує декодер для генерації скороченого тексту. У класичних моделях використовується RNN або його поліпшені версії, такі як LSTM або GRU. У моделях з механізмом уваги енкодер також створює матрицю з векторів для кожного слова у вхідній послідовності. Для реалізації використовується LSTM, для кращого зберігання інформації про контекст у довгих послідовностях [32], що є важливим при роботі з медичними текстами (рис. 2.4).

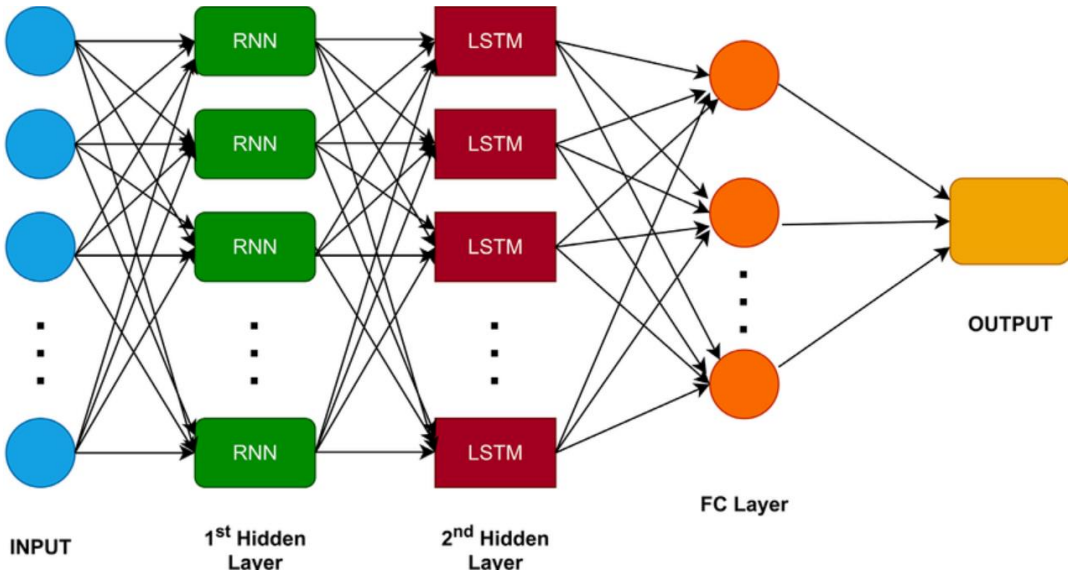


Рисунок 2.4 – Гібридна архітектура базової моделі RNN-LSTM [33]

2. *Механізм уваги* (Attention Mechanism) є основною відмінністю від класичної Seq2Seq-моделі. Замість того, щоб використовувати один єдиний вектор, який узагальнює весь вхідний текст, механізм уваги дозволяє моделі динамічно вибирати, які частини вхідної послідовності важливі для кожного кроку генерації вихідної послідовності. Це означає, що на кожному етапі декодер має доступ до всієї вхідної послідовності, але з різними "вагами" для кожної частини.

Під час генерації кожного слова у вихідній послідовності механізм уваги обчислює ваги для кожного слова у вхідному тексті, показуючи, наскільки важливим є це слово для поточного кроку декодування. Потім ці ваги використовуються для зваженого середнього по векторних представленнях вхідних слів, щоб отримати "контекстний вектор", який передається декодеру.

Варіанти механізму уваги:

– Глобальна увага (Global Attention) дозволяє моделі аналізувати всю вхідну послідовність на кожному етапі декодування [34]. Модель аналізує кожне слово вхідного тексту і обирає найбільш релевантні частини для кожного вихідного слова.

– локальна увага (Local Attention). Цей варіант фокусується тільки на частині вхідної послідовності, обмежуючи область уваги на певному діапазоні

слів. Це може бути корисним для дуже довгих текстів, де не всі слова важливі для кожного вихідного слова.

На рис. 2.5 показаний приклад моделі з механізмом уваги [20]. Контекстний вектор c_i залежить від послідовності анотацій $(h_1,\ldots, h_T)$, $T = 5$, на які енкодер відображає вхідне речення. Кожна анотація h_i містить інформацію про всю вхідну послідовність із акцентомДля реалізації використовується LSTM, для кращого зберігання інформації про контекст у довгих послідовностях на частинах, які оточують i-е слово вхідної послідовності.

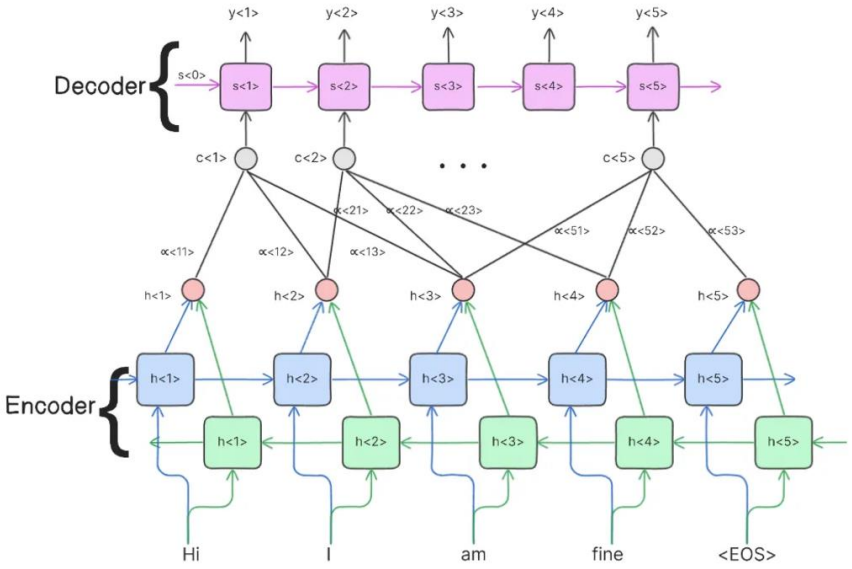


Рисунок 2.5 – Приклад Seq2Seq-моделі з механізмом уваги [21]

Контекстний вектор c_iобчислюється як зважена сума анотацій h_i:

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j. \quad (2.1)$$

Вага α_{ij} кожної анотації h_i обчислюється за формулою:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T} \exp(e_{ik})}, \quad (2.2)$$

де $e_{ij} = a(s_{i-1}, h_j)$ – модель вирівнювання, яка оцінює, наскільки добре збігаються вхідні дані навколо позиції j і вихідні дані в позиції i. Оцінка базується на прихованому стані RNN s_{i-1} та j-й анотації h_j вхідного речення. Модель вирівнювання a подають як нейронну мережу прямого зв’язку, яка навчається спільно з усіма іншими компонентами запропонованої моделі.

У кваліфікаційній роботі буде використано глобальну увагу (Global Attention), яка дозволяє аналізувати всю послідовність вхідного тексту на кожному етапі декодування.

3. *Декодер* (Decoder) приймає як векторне представлення вхідного тексту, так і контекстний вектор, отриманий через механізм уваги. Використовуючи цю інформацію, він генерує скорочену версію тексту. Як і в енкодері, зазвичай використовуються RNN або його варіанти, такі як LSTM або GRU. Кожен крок декодера виробляє слово резюме, яке потім використовується для прогнозування наступного слова.

До переваг Seq2Seq-моделі з увагою відносять:

1. Поліпшення обробки довгих текстів.

Класичні Seq2Seq-моделі на основі рекурентних нейронних мереж (RNN) мають обмеження у випадку довгих послідовностей через ‘затухання градієнтів і втрату пам'яті. [35]. Механізм уваги вирішує цю проблему, дозволяючи моделі звертати увагу на важливі частини тексту в будь-який момент, що значно покращує якість резюмування довгих текстів.

2. Збереження важливої інформації.

Завдяки тому, що декодер може звертатися до різних частин вхідного тексту, механізм уваги допомагає уникнути втрати ключових деталей, що є особливо важливим у задачах резюмування медичних запитань, де кожне слово може мати значення.

3. Гнучкість моделі.

Механізм уваги робить модель більш адаптивною до різних типів текстів і завдань. Модель може легко налаштовуватися на завдання з різною довжиною вхідних і вихідних послідовностей, що важливо при роботі з питаннями.

Серед недоліків Seq2Seq-моделі з увагою зазначають:

1. Високі обчислювальні витрати.

Механізм уваги вимагає значних обчислювальних ресурсів, оскільки для кожного вихідного слова необхідно аналізувати всю вхідну послідовність. Це може бути проблемою при роботі з великими даними або в реальному часі.

2. Потреба у великих обсягах даних.

Як і всі нейронні мережі, Seq2Seq-моделі з увагою потребують великих анотованих наборів даних для ефективного навчання. Це може бути складно забезпечити, особливо у випадку з медичними даними, які часто є конфіденційними і обмеженими.

Seq2Seq-модель з механізмом уваги є потужним інструментом для автоматизованого резюмування, особливо при роботі з довгими текстами, такими як медичні запити. Механізм уваги дозволяє моделі звертати увагу на важливі частини тексту, що покращує точність резюмування та зберігає ключову інформацію. Однак для досягнення оптимальних результатів важливо враховувати обчислювальні витрати та потребу у великих даних, а також можливість покращення моделі через використання трансформерів та попередньо натренованих моделей.

2.5 Проектування Seq2seq-моделі з механізмом уваги для резюмування медичних запитань

2.5.1 Навчальні дані

Для побудови ефективної моделі Seq2Seq з механізмом уваги, одним із перших кроків є підготовка даних. У даному проєкті було обрано датасет з TREC-2017 LiveQA Medical Task з репозиторію GitHub [36]. Цей датасет містить питання користувачів, які стосуються медичних тем, а також відповідні парафрази та резюме.

TREC-2017-LiveQA-Medical-Train-1.xml та TREC-2017-LiveQA-Medical-Train-2.xml кожне запитання може мати кілька підзапитань та відповідей. Містить анотації фокусів та типів запитань, що буде використовуватися для навчання моделі (табл. 2.1).

Таблиця 2.1

Приклад структури датасету TrainingDataset

Question ID	Subject	Message	Focus	Type	Answer
68	Coronary Artery Spasms	I ask this as a General Question. Can a Pacemaker help prevent these spasms?	Coronary Artery Spasms	Treatment	The goal of treatment is to control chest pain and prevent a heart attack. A medicine called nitroglycerin can relieve an episode of pain.
69	EYELID DROOP	WANT TO KNOW IF EYELID DROOP WILL BE GO AWAY	EYELID DROOP	Treatment	The main goals of ptosis surgery are elevation of the upper eyelid to permit normal visual development.

TrainingDataset (таблиця 2.1) містить такі поля:

– Question ID: ідентифікатор запитання;

– Subject: тема або фокус запитання, який коротко відображає основну тему, до якої належить запитання (наприклад, "Coronary Artery Spasms", "EYELID DROOP").

– Message: текст самого запитання, що описує конкретну медичну проблему або ситуацію, яку потрібно вирішити.

– Focus: основний медичний фокус запитання (наприклад, хвороба, симптом, або препарат), що допомагає моделі краще класифікувати та резюмувати запитання.

– Type: тип запитання, що визначає характер запиту (наприклад, "Treatment" – запит про лікування).

– Answer: відповідь на запитання, яка надає коротку медичну інформацію або пояснення на запит пацієнта.

Для аналізу текстових листувань між лікарями та пацієнтами, а також для передбачення захворювань, було використано тестові дані з китайського датасету DialMed [37]. Цей датасет є високоякісною базою даних, створеною для завдання рекомендації ліків на основі медичних діалогів. В датасеті DialMed

(табл. 2.2) зібрано11 996 медичних діалогів, а також16 поширених захворювань, 3 медичні відділення, 70 відповідних поширених ліків [38].

Таблиця 2.2

Приклад структури датасету test.txt

label	disease	dialog	original_dialog
黄连素	急性胃肠炎	患者：从星期一喝完酒了，之后就一直拉肚子就是那**种肚子疼...** | 医生：**你好，拉肚子一天几次？** | 患者：这几天都是两到三次 | 医生：大便是水样的吗？	患者：从星期一喝完酒了，之后就一直拉肚子就是那**种肚子疼...** | 医生：**你好，拉肚子一天几次？** | 患者：这几天都是两到三次 | 医生：大便是水样的吗？
曲美布汀, 益生菌	其他疾病或无**确**诊	患者：找上吃法老是拉肚子 | 医生：**你好，吃什么**东西拉肚子呢？拉的水样便还是稀便？肚子痛吗？ | 患者：喝热奶茶疼的很厉害，稀便，肚子痛 | 医生：中午吃饭后肚子痛拉肚子吗？哪个区域痛得最厉害？绞痛还是涨痛呢？	患者：找上吃法老是拉肚子 | 医生：**你好，吃什么**东西拉肚子呢？拉的水样便还是稀便？肚子痛吗？ | 患者：喝热奶茶疼的很厉害，稀便，肚子痛 | 医生：中午吃饭后肚子痛拉肚子吗？哪个区域痛得最厉害？绞痛还是涨痛呢？

Test.txt (таблиця 2.2) містить такі поля:

– label: використовується для передбачення або класифікації медичних рекомендацій;

– disease: вказує, який діагноз був поставлений, і може бути використано для навчання або аналізу точності діагнозу;

– dialog: основна текстова інформація, яка може використовуватися для побудови моделей аналізу тексту, витягу сутностей або предикативних систем;

– original_dialog: початкові дані для порівняння та перевірки точності обробки тексту (наприклад, вилучення назв медикаментів або симптомів).

В поточній реаліцації використовується лише disease та dialog.

2.5.2 Попередня обробка даних

Для побудови моделі необхідно правильно підготувати дані. Для цього необхідно провести:

– *токенізацію*: всі тексти (запитання та відповіді) розбиваються на слова (токени). Це дозволить моделі працювати з текстом як з послідовністю слів.

– *перетворення в словникові індекси*: кожне слово у тексті перетворюється на унікальний індекс, відповідно до словника, який було побудовано на основі всіх доступних даних.

– *визначення довжини послідовності*: запитання будуть обмежені за довжиною, щоб уникнути надмірного споживання пам’яті під час навчання. Усі тексти будуть урізані або доповнені до певної довжини.

2.5.3. Налаштування параметрів навчання

Для навчання моделі необхідно вибрати *метод навчання* (оптимізатор). Серед існуючих рекомендованим для глибоких мереж є *Adam* – один із найбільш популярних оптимізаторів для навчання нейронних мереж, оскільки він забезпечує швидке і стабільне збіження [39]. Adam поєднує в собі переваги двох інших методів – AdaGrad та RMSProp, що робить його ефективним для різних типів даних.

Використовуватиметься *функція втрат Cross-Entropy Loss*, яка є стандартною для задач класифікації та генерації тексту [19]. Ця функція оцінює, наскільки добре прогнозоване моделью слово відповідає очікуваному результату.

2.5.4 Поділ даних на тренувальні, валідаційні та тестові набори

Тренувальний набір. Використовується для навчання моделі, зокрема файли TREC-2017-LiveQA-Medical-Train-1.xml, TREC-2017-LiveQA-Medical-Train-2.xml та перекладений за допомогою бібліотеки GoogleTranslator із пакета deep_translator. test.txt**.**

*Валідаційний набір***.** Використовується частина DialMed для перевірки якості навчання моделі під час тренувального процесу.

Тестовий набір. Після завершення навчання модель буде оцінюватися на основі TREC-2017-LiveQA-Medical-Train-1.xml та TREC-2017-LiveQA-Medical-Train-2.xml**.**

2.5.5. Оцінка якості моделі

Для оцінки точності моделі будуть використовуватись кілька метрик:

ROUGE (англ. Recall-Oriented Understudy for Gisting Evaluation). Це стандартна метрика для оцінки якості автоматизованого резюмування. Вона вимірює, наскільки добре згенероване резюме збігається з еталонними текстами.

BLEU (англ. Bilingual Evaluation Understudy). BLEU є однією з основних метрик для оцінки якості перекладу та генерації тексту [40]. Вона порівнює кожне слово в згенерованому тексті з відповідними словами у контрольному тексті.

Висновки за розділом 2

У даному розділі детально розглянуто задачу автоматизованого резюмування запитань у медичних інформаційних системах є критично важливою для покращення якості та ефективності обробки текстової інформації. Використання сучасних технологій, таких як seq2seq-моделі, дозволяє значно спростити роботу медичного персоналу, зменшити навантаження на лікарів і забезпечити швидкий доступ до ключової інформації. Проте важливо враховувати недоліки таких систем, особливо ризики втрати важливої медичної інформації та складність інтерпретації медичних термінів, щоб забезпечити максимальну точність і безпеку обробки даних.

РОЗДІЛ 3.
ПРОГРАМНА РЕАЛІЗАЦІЯ МОДЕЛІ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ

Розробка програмного забезпечення для медичної галузі потребує інтеграції широкого спектру функцій, що забезпечують точність обробки даних, конфіденційність інформації та зручність використання для різних груп користувачів. У рамках цього бакалаврського проєкту було створено веборієнтовану систему на платформі .NET 6, зокрема з використанням фреймворку Active Server Pages for .NET (англ. ASP.NET) і Object-Relational Mapping (англ. ORM-системи) Entity Framework. Розробка здійснювалася в інтегрованому середовищі Visual Studio 2022, що забезпечило стабільний і продуктивний робочий процес.

Вебдодаток охоплює основні функціональні можливості для роботи з даними медичних працівників та пацієнтів. Реалізовано обробку інформації про пацієнтів, лікарів, призначення, результати аналізів та медичні показники. Додаток розроблений як багатосторінковий додаток (англ. MPA), що містить інтерфейси для трьох основних ролей: адміністратора, лікаря та пацієнта. Для забезпечення високого рівня безпеки впроваджено механізм авторизації з розмежуванням прав доступу залежно від ролі користувача.

Використання .NET 8 стало ключовим у розробці проєкту. У порівнянні з попередніми версіями, такими як .NET 6, новий фреймворк пропонує значні покращення в продуктивності, гнучкості та зручності розробки. Одним із важливих оновлень є вдосконалення механізмів асинхронного програмування, що сприяє більш ефективному використанню ресурсів. Завдяки оптимізованим алгоритмам обробки запитів і покращенню роботи збирача сміття Garbage Collector (англ. GC), .NET 8 дозволяє зменшити затримки під час виконання додатків і підвищити їх стабільність.

Також у .NET 8 інтегровано розширені можливості для роботи з Entity Framework, що полегшує створення та підтримку складних баз даних. Завдяки цьому ORM-система забезпечує високу продуктивність, підтримку транзакцій і

простоту виконання міграцій бази даних. Інструменти Visual Studio 2022, у свою чергу, забезпечили зручний і інтерактивний процес розробки, включно з можливістю відлагодження, управління проєктом і швидкого внесення змін.

Загалом, використання сучасних технологій, таких як .NET 8 і Visual Studio 2022, дозволило створити продуктивну, гнучку та масштабовану систему, яка відповідає високим вимогам до програмного забезпечення для медичної галузі.

3.1 Аналіз вимог до програмного продукту

Аналіз вимог є критично важливим етапом у розробці будь-якого програмного продукту. Цей процес має на меті виявити основні проблеми, які програмний продукт повинен вирішувати, та визначити ключові можливості, що забезпечать його функціональність і ефективність. Він включає розуміння контексту використання, формулювання функціональних і нефункціональних вимог, вивчення системних дизайнів та інтеграційних процесів, а також ідентифікацію можливих ризиків і обмежень.

Для створюваного програмного продукту можна виділити такі вимоги:

1) програма повинна зберігати повідомлення та прогнозовані діагнози пацієнтів у базі даних;

2) додаток повинен забезпечувати функціональність для обміну повідомленнями між фіхівцями та пацієнтами;

3) користувачі повинні мати змогу вводити, зберігати та переглядати повідомлення;

4) застосунок повинен відображати статуси повідомлень;

5) час відправлення та отримання повідомлень повинен бути чітко зазначений;

6) програма повинна інтегрувати механізм аналізу повідомлень пацієнтів для визначення типу звернення, доступний для ролі "лікар";

7) додаток повинен включати функцію прогнозування діагнозів на основі змісту повідомлень пацієнтів, що дозволяє швидко ідентифікувати потенційні захворювання.

Загалом, аналіз вимог забезпечив всебічне розуміння призначення, функціональності та обмежень програмного продукту.

3.2 Логіка та структура роботи системи

Платформа базується на багатошаровій архітектурі, яка забезпечує розподіл відповідальності між рівнями представлення, логіки та збереження даних. Рівень представлення включає вебінтерфейс, який дозволяє медичним працівникам і пацієнтам зручно взаємодіяти із системою.

Логічний рівень побудований на основі ASP.NET Model-View-Controller (англ. MVC) виконує обробку бізнес-логіки, включаючи аналіз текстів і прогнозування, тоді як рівень збереження даних відповідає за зберігання повідомлень, діагнозів у базі даних Structured Query Language (англ. SQL) Server.

На етапі даталогічного проектування розробляється логічна структура бази даних, яка забезпечує організоване та ефективне зберігання даних у відповідності до функціональних і нефункціональних вимог системи. Цей механізм створюється на основі інфологічної моделі з урахуванням особливостей вибраної системи управління базами даних (СУБД).

Даталогічна модель передбачає використання первинних і зовнішніх ключів для забезпечення зв’язків між таблицями. Для вебдодатку було розроблено таблицю dbo.ChatMessages, яка відповідає за зберігання даних обміну повідомленнями між медичними працівниками та пацієнтами. Таблиця побудована з урахуванням принципів нормалізації, що забезпечує ефективне управління даними.

Таблиця 3.1

Структура таблиці dbo.ChatMessages

Ім’я стовпця	Тип даних	Призначення	Чи є ключем
id	int	Унікальний ідентифікатор повідомлення	Primary key
PatientId	int	Ідентифікатор пацієнта	Foreign key
DoctorId	int	Ідентифікатор лікаря	Foreign key
Message	nvarchar(max)	Текст повідомлення	–
IsFromPatient	bit	Позначка, чи повідомлення надійшло від пацієнта	–
Timestamp	datetime	Час і дата відправлення повідомлення	–
IsReadByDoctor	Bit	Статус прочитання повідомлення лікарем	–
IsReadByPatient	bit	Статус прочитання повідомлення пацієнтом	–
Disease	nvarchar(255)	Захворювання, пов’язане із повідомленням (опціональне поле)	–

Таблиця підтримує двосторонній обмін повідомленнями між фахівцями та пацієнтами із фіксацією часу, статусу та автора.

3.3 Прогнозування діагнозу пацієнта

Прогнозування діагнозу є однією з ключових функцій системи, яка забезпечує лікарів автоматизованими інструментами для швидкої ідентифікації потенційних захворювань на основі текстових повідомлень пацієнтів.

Додаток використовує архітектуру машинного навчання для аналізу повідомлень хворих та визначення можливих діагнозів. У процесі роботи враховуються такі захворювання:

– Acute gastroenteritis – гострий гастроентерит;

– Pneumonia – пневмонія;

– Enteritis – ентерит;

– Cirrhosis – цироз;

– Esophagitis – езофагіт;

– Upper respiratory tract infection – інфекція верхніх дихальних шляхів;

– Rhinitis – риніт;

– Viral hepatitis – вірусний гепатит;

– Duodenitis – дуоденіт;

– Dermatitis – дерматит;

– Gastritis – гастрит;

– Allergic dermatitis – алергічний дерматит;

– Hepatitis – гепатит;

– Fatty liver – жировий гепатоз;

– Seborrheic dermatitis – себорейний дерматит.

– Other diseases or undiagnosed – інші захворювання або недіагностовані стани;

– Allergic rhinitis – алергічний риніт.

Загальна кількість унікальних захворювань: 17.

Коли лікар на своєму інтерфейсі отримує повідомлення від хворого та натискає кнопку "Аналізувати", відбувається наступна послідовність дій:

1) *Відправлення запиту на сервер*. Форма з методом POST надсилає запит до дії AnalyzeDisease на сервері, передаючи ідентифікатор пацієнта як параметр (рис. 3.1).

```
</form>
<!-- Кнопка "Аналізувати" -->
<form method="post" asp-action="AnalyzeDisease" asp-route-patientId="@patient?.Id" style="display: inline;">
    <button type="submit" class="btn btn-primary" style="height: 50px; font-size: 16px;">Аналізувати</button>
</form>
```

Рисунок 3.1 – Логіка передачі запиту від інтерфейсу до сервера

2) *Обробка запиту на сервері*. Серверна частина, реалізована на платформі ASP.NET з використанням .NET 8 та Entity Framework, приймає запит.

Вебдодаток ідентифікує лікаря на основі електронної пошти, отриманої з контексту безпеки, та визначає його DoctorId через запит до бази даних (рис. 3.2).

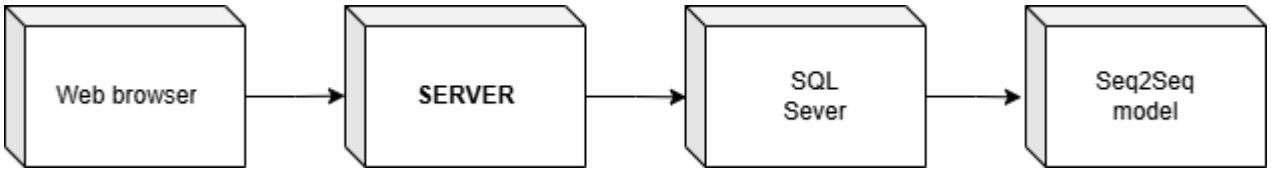


Рисунок 3.2 – Процес обробки запиту на сервері ASP.NET MVC

3) *Отримання повідомлень пацієнта.* Використовуючи patientId, та doctorId система витягує всі повідомлення, надіслані пацієнтом та лікарем, з таблиці ChatMessages у базі даних (рис. 3.3).

```
0 references
public IActionResult AnalyzeDisease(int patientId)
{
    using (var db = new BloodParametersContext())
    {
        string? email = User.Claims
            .FirstOrDefault(x => x.Type == ClaimsIdentity.DefaultNameClaimType)?.Value;

        if (email == null)
        {
            return RedirectToAction("Patients");
        }

        int? doctorId = db.Users
            .Where(x => x.Email == email)
            .Select(x => x.DoctorId).FirstOrDefault();

        if (doctorId == null)
        {
            return RedirectToAction("Patients");
        }
        var messages = db.ChatMessages
            .Where(m => m.PatientId == patientId && m.DoctorId == doctorId)
            .OrderBy(m => m.Timestamp)
            .ToList();
        int idFirstMessage = 0;
        int idLastMessage = 0;
```

Рисунок 3.3 – Обробка та передача повідомлень базі даних

4) *Інтеграція з Python-скриптом.* Для аналізу повідомлень сервер викликає зовнішній Python-скрипт, який реалізує алгоритм машинного навчання для визначення можливих захворювань на основі тексту повідомлень. Виклик здійснюється через Hypertext Transfer Protocol (англ. HTTP) (рис. 3.4).

```
string pythonExe = "python";
string scriptPath = Path.Combine(Environment.CurrentDirectory, "test3.py");

var processStartInfo = new System.Diagnostics.ProcessStartInfo
{
    FileName = pythonExe,
    Arguments = $"{scriptPath} \"{string.Join("|", conversation)}\"",
    RedirectStandardOutput = true,
    RedirectStandardError = true,
    UseShellExecute = false,
    CreateNoWindow = true
};

using (var process = System.Diagnostics.Process.Start(processStartInfo))
{
    string output = process.StandardOutput.ReadToEnd();
    string error = process.StandardError.ReadToEnd();
    process.WaitForExit();

    if (process.ExitCode != 0)
    {
        Console.WriteLine($"Error executing Python script: {error}");
        return View("Error");
    }
```

Рисунок 3.4 – Виклик Python-скрипта для прогнозування діагнозу

5) *Обробка результатів аналізу*. Python-скрипт повертає результати аналізу у вигляді ймовірного діагнозу. Сервер отримує інформацію та зберігає їх у базі даних для подальшого використання.

6) *Відображення результатів лікарю*. Після завершення аналізу сервер відправляю лікарю в чаті хворого його можливий діагноз.

3.4 Класифікація повідомлень

Процес класифікації повідомлень реалізується під час завантаження сторінки чату лікаря з пацієнтом. Коли лікар переходить до чату, автоматично надсилається GET-запит до серверної частини, яка аналізує всі повідомлення, надіслані заявником. На основі аналізу повідомлень визначається тип звернення, який може належати до однієї з таких категорій:

- treatment – пов’язане з лікуванням;
- cause – виявлення причини симптомів;
- prognosis – прогноз перебігу захворювання;
- diagnosis – встановлення діагнозу;
- organization – організаційні питання;
- symptom – опис симптомів;

- information – запит інформації;
- susceptibility – сприйнятливість до певних умов або захворювань.

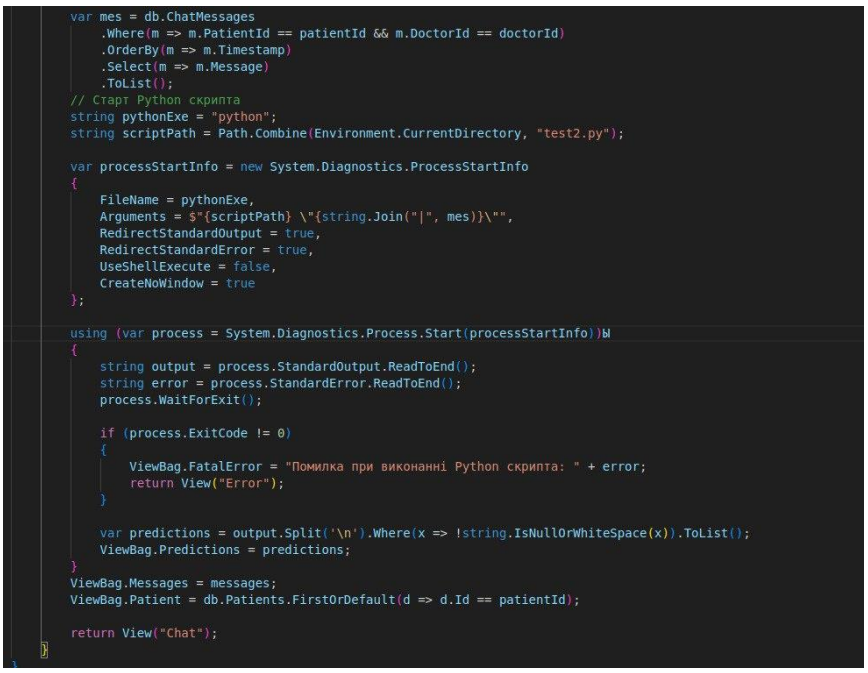
Послідовність роботи:

1) *Обробка GET-запиту:* після відкриття чату серверна частина виконує запит до бази даних для отримання всіх повідомлень, пов’язаних із заданим patientId та doctorId (рис. 3.5).

```
[HttpGet]
0 references
public IActionResult Chat(int patientId)
{
    using (var db = new BloodParametersContext())
    {
        string? email = User.Claims
            .FirstOrDefault(x => x.Type == ClaimsIdentity.DefaultNameClaimType)?.Value;

        if (email == null)
        {
            ViewBag.FatalError = "Немає запису пацієнта, пов'язаного з вашим акаунтом";
            return RedirectToAction("Patients");
        }

        int? doctorId = db.Users
            .Where(x => x.Email == email)
            .Select(x => x.DoctorId).FirstOrDefault();

        if (doctorId == null)
        {
            ViewBag.FatalError = "Немає запису пацієнта, пов'язаного з вашим акаунтом";
            return RedirectToAction("Patients");
        }
        var messages = db.ChatMessages
        .Where(m => m.PatientId == patientId && m.DoctorId == doctorId)
        .OrderBy(m => m.Timestamp)
        .ToList();
```

Рисунок 3.5 – Отримання повідомлень із бази даних

2) *Підготовка даних для аналізу:* зібрані запити проходять попередню обробку. Текст кожного повідомлення структурується у вигляді послідовності для передачі до Python-скрипта (рис. 3.6).



```
    var mes = db.ChatMessages
        .Where(m => m.PatientId == patientId && m.DoctorId == doctorId)
        .OrderBy(m => m.Timestamp)
        .Select(m => m.Message)
        .ToList();
    // Старт Python скрипта
    string pythonExe = "python";
    string scriptPath = Path.Combine(Environment.CurrentDirectory, "test2.py");

    var processStartInfo = new System.Diagnostics.ProcessStartInfo
    {
        FileName = pythonExe,
        Arguments = $"{scriptPath} \"{string.Join("|", mes)}\"",
        RedirectStandardOutput = true,
        RedirectStandardError = true,
        UseShellExecute = false,
        CreateNoWindow = true
    };

    using (var process = System.Diagnostics.Process.Start(processStartInfo))
    {
        string output = process.StandardOutput.ReadToEnd();
        string error = process.StandardError.ReadToEnd();
        process.WaitForExit();

        if (process.ExitCode != 0)
        {
            ViewBag.FatalError = "Помилка при виконанні Python скрипта: " + error;
            return View("Error");
        }

        var predictions = output.Split('\n').Where(x => !string.IsNullOrWhiteSpace(x)).ToList();
        ViewBag.Predictions = predictions;
    }
    ViewBag.Messages = messages;
    ViewBag.Patient = db.Patients.FirstOrDefault(d => d.Id == patientId);

    return View("Chat");
}
```

Рисунок 3.6 – Процес класифікації повідомлень

3) *Інтеграція з Python-скриптом:* сервер викликає Python-скрипт через HTTP-запит. Цей скрипт використовує попередньо навчену модель для класифікації тексту повідомлень пацієнта та визначення типу звернення.

4) *Отримання результатів:* модель повертає результат класифікації у форматі текстового рядка з типом звернення. Сервер обробляє відповідь і зберігає результати в базі даних.

5) *Відображення класифікації:* на інтерфейсі лікаря поряд із кожним листом пацієнта відображається його тип звернення. Це дозволяє лікарю швидко орієнтуватися в суті повідомлень пацієнта.

3.5 Інтерфейс користувача

Інтерфейс користувача розробленого вебдодатку забезпечує зручну взаємодію між лікарями та пацієнтами, дозволяючи швидко обмінюватися повідомленнями, переглядати медичні записи та запускати автоматизовані процеси, такі як визначення діагнозу.

Стилізація повідомлень (рис. 3.7)

1. Повідомлення лікаря:
    - Синій фон: використовується для візуального розмежування.

- Відображається статус запитів:
  - *Переглянуто*: якщо вхідні дані прочитані пацієнтом.
  - *Доставлено*: якщо вхідні дані ще не було прочитане.
- Відображається час відправлення повідомлення.

2. Повідомлення пацієнта:
   - Світло-блакитний фон для чіткого візуального розмежування.
   - Відображається час відправлення запиту.
3. Поле введення повідомлення
   - Розташування: нижче списку повідомлень.
   - Функціональність:
     - Поле введення тексту.
     - Кнопка "Відправити" для надсилання.

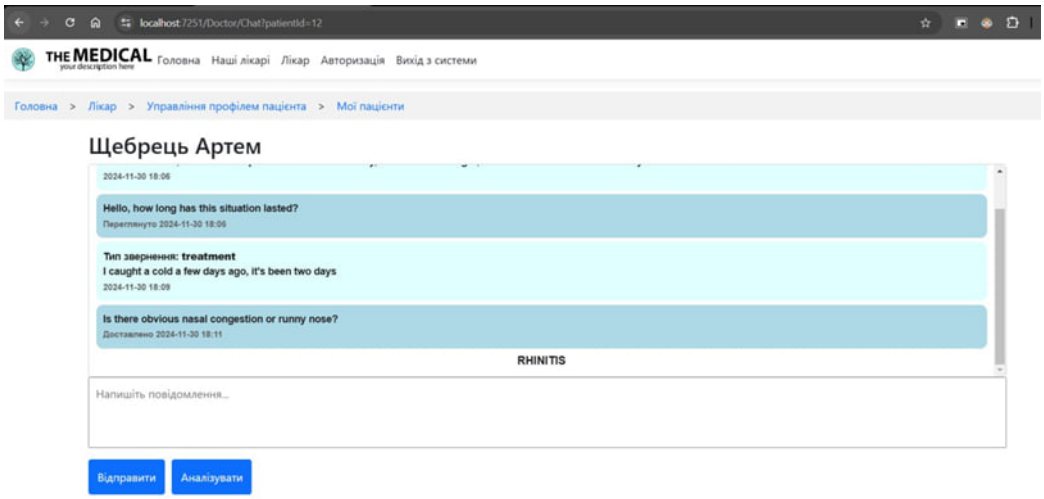


Рисунок 3.7 – Поле введення для користувача

4. Автоматичне прокручування
   - Реалізовано функцію автоматичного прокручування до останнього повідомлення у наступних випадках:
     - Під час завантаження сторінки чату.
     - При надходженні нового запиту.

5. Кнопка "Аналізувати" (рис. 3.8)

- Функція: Лікар може запустити аналіз скарг пацієнта на основі всіх повідомлень.
- Відображення результатів: прогнозований діагноз відображається під відповідними повідомленням у вигляді виділеного тексту.

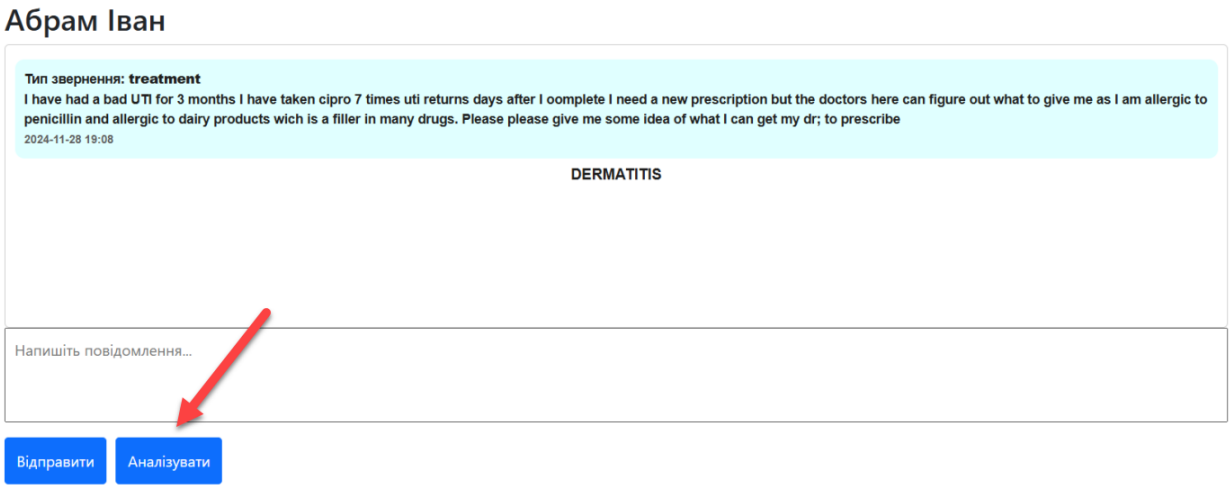


Рисунок 3.8 – Кнопка "Аналізувати" для лікаря

6. Взаємодія з пацієнтами

1. Список пацієнтів (рис. 3.9):
   - Відображається інформація про пацієнта.
   - Реалізовано кнопки для швидкого доступу до:
     - Аналізів.
     - Медичних показників.
     - Призначень.
     - Чату з пацієнтом.
   - Відображається кількість непрочитаних повідомлень.

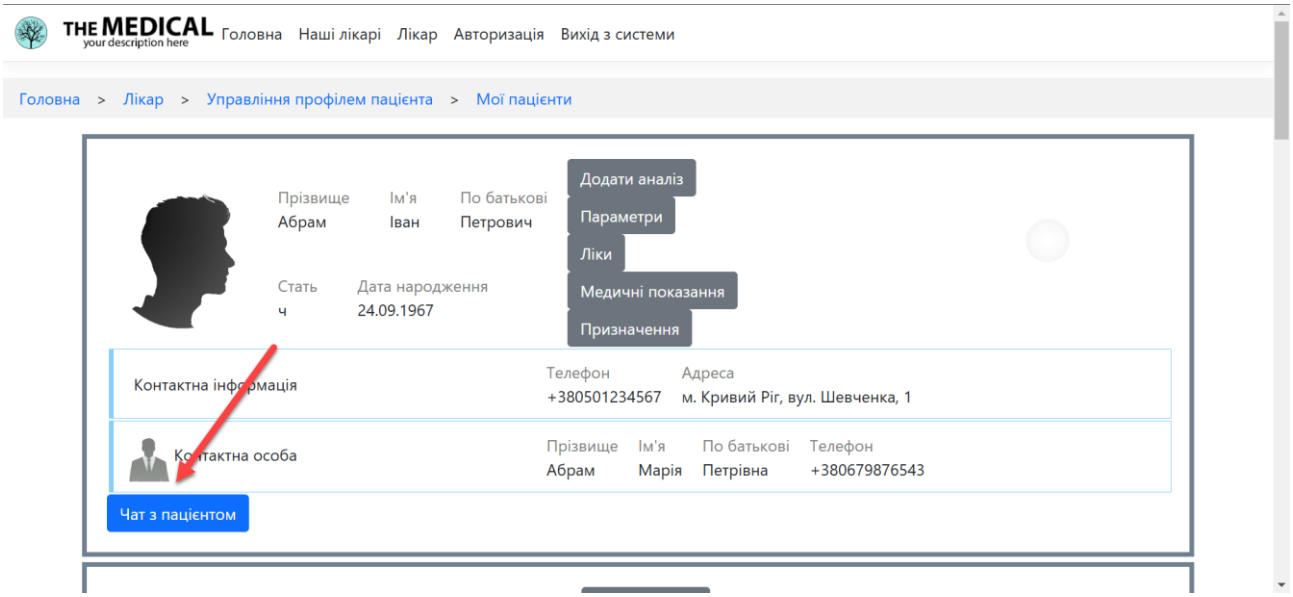


Рисунок 3.9 – Список пацієнтів із кнопками доступу

2. Відображення статусу повідомлень (рис. 3.10):
   - *Переглянуто*: запит було прочитано.
   - *Доставлено*: запит ще не було прочитано.
   - Реалізовано позначку **"Нові повідомлення"** для виділення непрочитаних повідомлень у чаті.

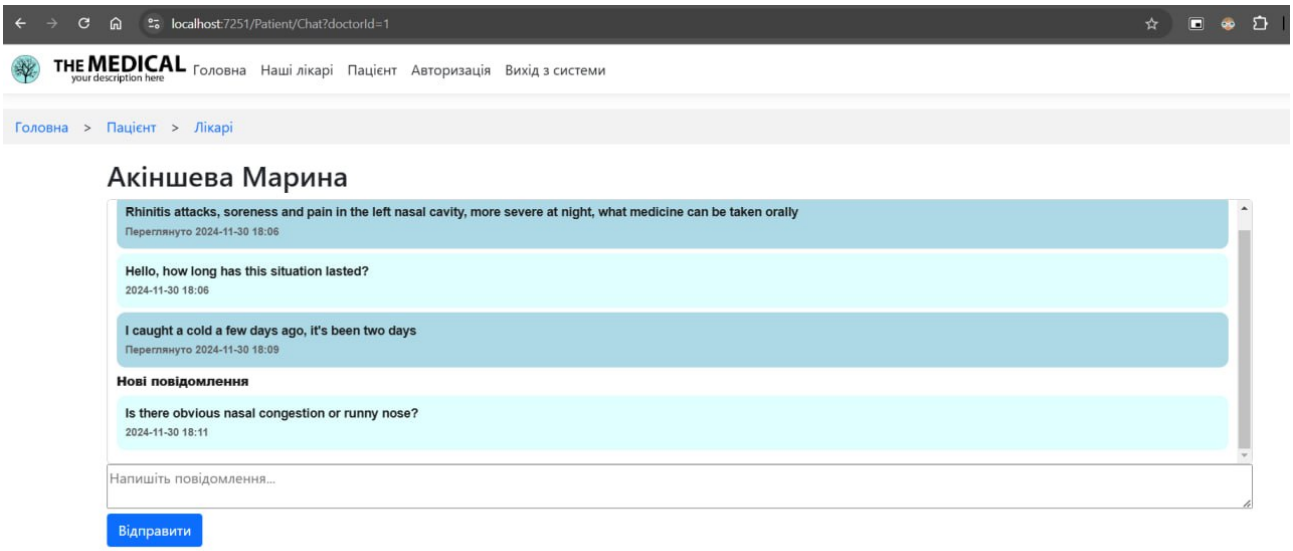


Рисунок 3.10 – Відображення статусу повідомлень

7. Відображення класифікації (рис. 3.11)

- На інтерфейсі лікаря поряд із кожним запитом пацієнта відображається **тип звернення** (наприклад, "Діагностика", "Лікування").
- Це дозволяє лікарю швидко орієнтуватися в суті скарг пацієнта.

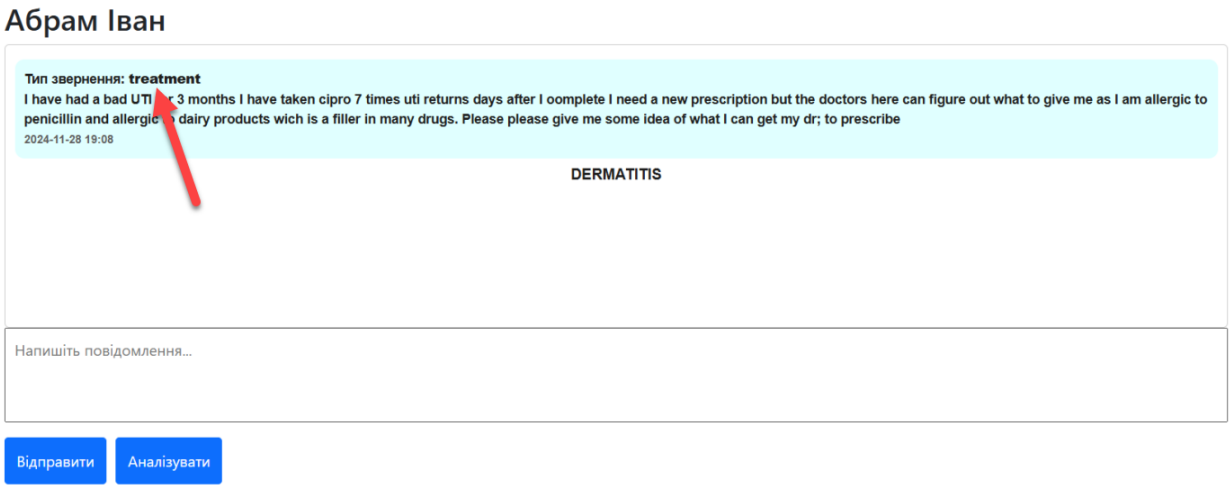


Рисунок 3.11 – Відображення типу звернення в чаті

8. Функціонал для ролей

1. Лікар:
   - Може відправляти вхідні повідомлення своїм пацієнтам.
   - Відображаються кількість непрочитаних повідомлень від кожного пацієнта.
   - Має можливість переглядати історію повідомлень та прогнозувати діагнози.
2. Пацієнт:
   - Може відправити запити будь-якому лікарю (рис. 3.12).
   - Має доступ до історії своїх повідомлень.
   - Відображається статус повідомлень із аналогічною логікою.

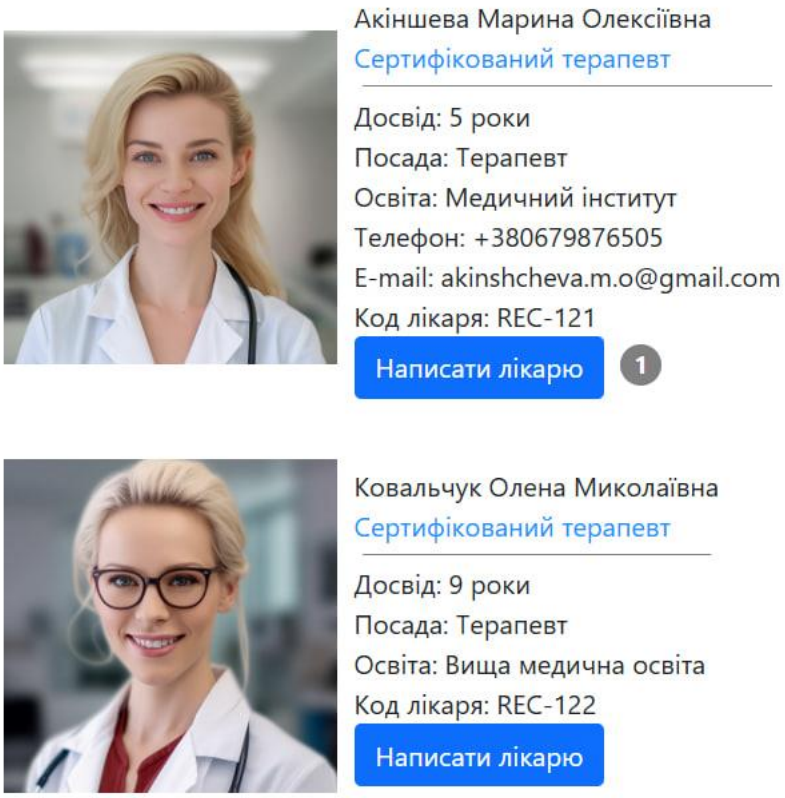


Рисунок 3.12 – Інтерфейс пацієнта з можливістю написати повідомлення

9. Розширення функціоналу. Реалізовано механізм підрахунку нових повідомлень (вони не включаються в поточну вибірку для навчання нейронної мережі). У майбутньому планується допрацювати систему, щоб нові повідомлення автоматично додавалися до навчальних даних.

Даний функціонал забезпечує зручну взаємодію між лікарем і пацієнтом, спрощуючи роботу з повідомленнями, статусами та аналізами.

3.6 Тестування

3.6.1. Планування тестування

Тестування програмної системи є важливим етапом розробки, оскільки воно дозволяє перевірити відповідність функціональності системи її вимогам, забезпечити стабільність роботи та виявити можливі помилки. Для тестування розробленої системи було обрано функціональне тестування, яке охоплює такі аспекти:

- Перевірка роботи основних функцій: збереження повідомлень, оцінка можливих діагнозів, класифікація звернень.
- Оцінка інтерфейсу користувача: перевірка відображення статусів повідомлень, коректного взаємодії лікаря і пацієнта.
- Інтеграція компонентів: перевірка взаємодії між серверною частиною ASP.NET, базою даних SQL Server та Python-скриптами.
- Продуктивність: тестування швидкості обробки повідомлень і визначення діагнозів.

3.6.2. Виконання тестів та аналіз результатів

Для перевірки функціональності та точності системи було проведено кілька етапів тестування:

Тестування збереження повідомлень

Перевірено роботу таблиці dbo.ChatMessages. Повідомлення, надіслані пацієнтом і лікарем, зберігаються коректно із зазначенням часу, статусу та автора. Усі дані відповідають структурі таблиці 3.2.

Тестування класифікації звернень

На етапі завантаження чату лікаря автоматично визначався тип звернення (наприклад, діагноз, лікування, симптом). Усі результати класифікації були перевірені на точність і відповідність введеним даним.

Таблиця 3.2

Результати класифікації повідомлень

Тип звернення	Кількість повідомлень	Прогнозовано коректно (%)	Відхилення (%)
Лікування	225	87.0	13.0
Причина	222	93.2	6.8
Прогноз	138	85.6	14.4
Діагностика	138	88.4	11.6

Тестування прогнозування діагнозів

Було виконано функціональне тестування модуля визначення захворювань. У таблиці 3.3 наведено порівняння отриманих результатів із референсними даними.

Таблиця 3.3

Результати прогнозування діагнозів на основі 5 епохи

Захворювання	Кількість повідомлень	Прогнозовано коректно (%)	Відхилення (%)
Гострий гастроентерит	50	92.0	8.0
Пневмонія	40	89.5	10.5
Риніт	60	95.0	5.0
Інші захворювання	70	88.0	12.0

3.7 Аналіз отриманих результатів

У даному розділі здійснено аналіз результатів роботи моделі на кожному етапі навчання. Було розглянуто зміни основних метрик, таких як Loss, BLEU, Precision, Recall, F1-Score, та ROUGE, що дозволяє оцінити прогрес навчання моделі та її здатність до узагальнення (табл. 3.4).

Таблиця 3.4

Результати класифікації повідомлень

type	precision	recall	F1-score	Support
Cause	0.98	0.91	0.94	44
Diagnosis	0.81	0.93	0.87	28
Information	0.55	0.35	0.43	17
Organization	0.95	1.00	0.98	20
Prognosis	0.86	0.86	0.86	28
Susceptibility	1.00	1.0	1.0	17
Symptom	0.89	0.94	0.92	18
Treatment	0.81	0.87	0.84	45

Аналіз результатів класифікації показує, що модель найкраще справляється з визначенням категорій, таких як Susceptibility та Organization, які

мають F1-Score, що наближається до 1.0. Категорії Information та Treatment мають нижчі показники F1-Score (рис. 3.13), що може свідчити про необхідність додаткових даних або удосконалення моделі для цих типів звернень.

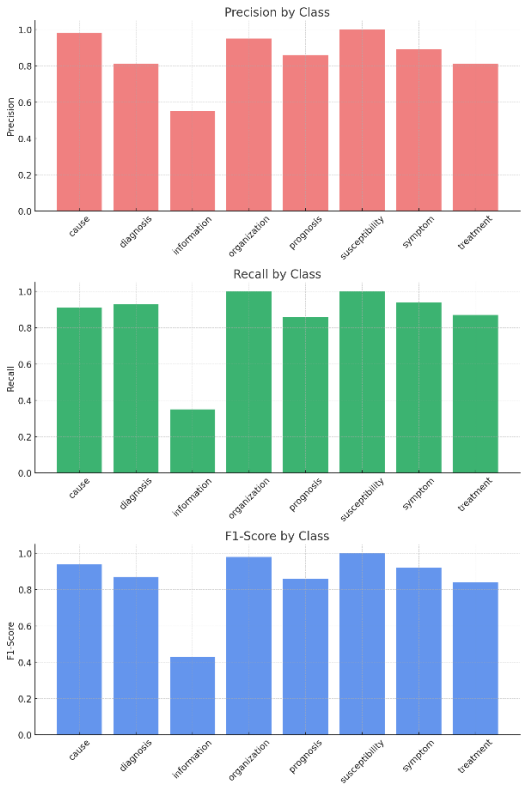


Рисунок 3.13 – F1-Оцінка за класами для класифікації повідомлень

Графік відображає порівняння F1-Score між різними класами, надаючи візуальне уявлення про ефективність роботи моделі щодо кожного типу звернення. Це дозволяє швидко оцінити, в яких категоріях модель працює краще, а в яких необхідно вдосконалити.

Результати навчання моделі протягом 10 епох (табл. 3.5) демонструють поступове зниження Loss, що свідчить про стабільне навчання. Метрики Precision, Recall, F1-Score, а також ROUGE-1, ROUGE-2 і ROUGE-L зростають протягом навчання, що вказує на підвищення якості роботи моделі. Особливо помітний прогрес після третьої епохи, що свідчить про ефективність налаштувань моделі.

Таблиця 3.5

Результати прогнозування діагнозів

Epoch	Loss	BLEU	Precision	Recall	F1-Score	ROUGE-1 (F)	ROUGE-2 (F)	ROUGE-L (F)
1	1.2189	0.2360	0.5275	0.5013	0.5085	0.5496	0.2722	0.5496
2	0.7438	0.1648	0.7114	0.6764	0.6863	0.6635	0.2831	0.6635
3	0.4335	0.2403	0.8290	0.8078	0.8128	0.7712	0.3583	0.7712
4	0.1755	0.1850	0.9287	0.9199	0.9220	0.8363	0.3857	0.8363
5	0.0403	0.2518	0.9858	0.9833	0.9839	0.8897	0.4354	0.8897
6	0.0031	0.2842	1.0000	0.9996	0.9997	0.9288	0.4681	0.9288
7	0.0003	0.3130	1.0000	1.0000	1.0000	0.9364	0.4800	0.9364
8	0.0001	0.3346	1.0000	1.0000	1.0000	0.9446	0.4865	0.9446
9	0.0000	0.3480	1.0000	1.0000	1.0000	0.9509	0.4910	0.9509
10	0.0000	0.3575	1.0000	1.0000	1.0000	0.9543	0.4949	0.9543

Графік на рис. 3.14 демонструє зростання метрик на кожній ітерації навчання, підтверджуючи поліпшення точності моделі. Це особливо важливо для підтвердження її придатності для реального використання.

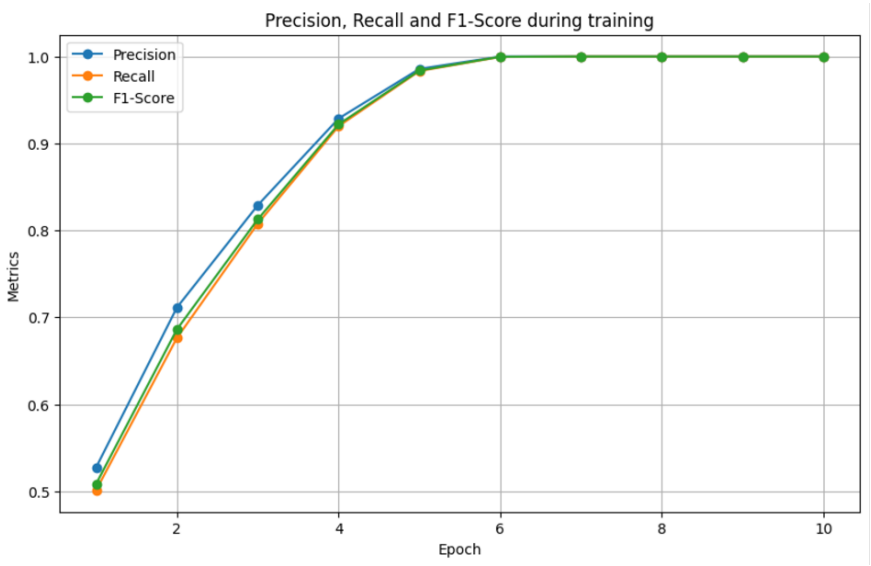


Рисунок 3.14 – Відображення підвищення метрик Precision, Recall та F1-Score

Отримані результати демонструють, що модель є ефективною для виконання поставлених задач, зокрема для прогнозування текстів та класифікації повідомлень.

Висновки за розділом 3

У цьому розділі розроблено веборієнтовану систему для медичної галузі, яка відповідає сучасним вимогам до обробки та збереження даних. Використання платформи .NET 8, ASP.NET MVC та Entity Framework забезпечило високу продуктивність і гнучкість системи, дозволяючи інтегрувати функції обміну повідомленнями, класифікації звернень і прогнозування діагнозів. Завдяки багатошаровій архітектурі досягнуто розподілу відповідальностей між користувацьким інтерфейсом, бізнес-логікою та базою даних, що підвищує масштабованість і підтримуваність додатку.

Особливу увагу приділено графічному дизайну користувача, який забезпечує зручність взаємодії для лікарів та пацієнтів. Візуальне розмежування повідомлень, функція автоматичного прокручування, відображення статусів і класифікації звернень сприяють ефективності роботи медичного персоналу. Реалізовано інтерактивні елементи, такі як кнопка "Аналізувати", яка запускає автоматичний аналіз повідомлень і дозволяє швидко визначати ймовірні діагнози.

Проведене тестування підтвердило стабільність і точність роботи системи. Модуль класифікації звернень показав високі результати, а функціонал прогнозування діагнозів продемонстрував ефективність інтеграції машинного навчання. Загалом, створений додаток є надійним інструментом для автоматизації ключових процесів у медичній практиці, покращуючи якість обслуговування пацієнтів та ефективність роботи лікарів.

ВИСНОВКИ

У цій роботі виконано комплексний аналіз проблем, пов’язаних із автоматизацією обробки медичних даних, забезпеченням ефективної взаємодії між лікарями та пацієнтами, а також прогнозуванням діагнозів на основі текстових повідомлень. Проведено огляд існуючих методів та технологій, які застосовуються в сучасних медичних інформаційних системах, включаючи підходи до класифікації текстів та аналізу медичних записів.

Розроблена система інтегрує багатосторінковий додаток (MPA) на платформі .NET 8 із використанням фреймворку ASP.NET і ORM-системи Entity Framework. Запропоновано архітектурне рішення, що включає розподіл рівнів представлення, бізнес-логіки та збереження даних. Основним результатом є функціональний вебдодаток, який забезпечує:

- збереження та обробку даних пацієнтів, лікарів, медичних показників і результатів аналізів;
- двосторонній обмін повідомленнями між лікарями та пацієнтами з можливістю класифікації повідомлень за типами звернень;
- прогнозування діагнозів пацієнтів із використанням моделей машинного навчання.

Було запропоновано алгоритм класифікації повідомлень, який передбачає:

- автоматичне визначення типу звернення під час завантаження сторінки чату на основі тексту повідомлень;
- інтеграцію Python-скриптів для виконання аналізу за допомогою попередньо навчених моделей;
- збереження результатів класифікації у базі даних із подальшим відображенням на інтерфейсі лікаря.

Система також підтримує функціонал прогнозування діагнозів, що включає інтеграцію з зовнішніми скриптами та обробку результатів через API. Проведено детальний аналіз отриманих результатів із використанням метрик

BLEU, ROUGE, Precision, Recall та F1-Score, які підтвердили високу точність системи.

Окрім того, було виконано функціональне тестування розробленого додатку, що підтвердило його стабільність, зручність у використанні та відповідність поставленим вимогам. Виконано візуалізацію ключових етапів роботи системи та створено блок-схеми процесів, включаючи архітектуру системи та взаємодію компонентів.

Таким чином, розроблене програмне забезпечення демонструє значний потенціал для використання у медичних установах, сприяючи автоматизації рутинних процесів, покращенню якості обслуговування пацієнтів і зменшенню навантаження на лікарів.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Інформація та документація. Базові поняття. Терміни та визначення: ДСТУ 2392-94 – К.: Держстандарт України, №68 від 29 березня 1994 р. – IV, 9 с.

2 Закон України: за станом на 19.04.2014 / Верховна Рада України. – Офіц. вид. – К.: Відомості Верховної Ради України (ВВР), 1994. – № 31, ст. 286

3 Коваленко, В. М., Білошапка, Н. І. «Медична інформатика». Навчальний посібник. Київ: Медицина, 2015.

4 Розпорядження: за станом на 30.11.2016 р./ Кабінет Міністрів України. – Офіц. вид. – К.: Офіційний вісник України, 2017. – № 2, с. 175

5 Адміністратор Центральної бази даних eHealth України. URL: https://ehealth.gov.ua/ (дата звернення: 12.09.2024).

6 Підключені до eHealth Медичні Інформаційні Системи. URL: https://ehealth.gov.ua/pidklyucheni-do-ehealth-mis/ (дата звернення: 14.09.2024).

7 Мельникова Н. І. Моделювання експертних систем призначення лікування / Н.І. Мельникова // Радіоелектроніка, інформатика, управління. – 2012. – № 1. – С. 63–70.

8 Європейська комісія. Програма EU4Health на 2021–2027 роки. Офіційний сайт. URL: https://phc.org.ua/eu4health (дата звернення: 29.09.2024).

9 Кабінет Міністрів України. ЄС виділяє Україні понад 4,6 млн євро на проєкти у сфері охорони здоров'я. URL: https://eu-ua.kmu.gov.ua/news/yes-vydilyaye-ukrayini-ponad-4-6-mln-yevro-na-proyekty-u-sferi-ohorony-zdorov-ya (дата звернення: 29.09.2024).

10 Hsiao Chun-Ju, Hing Esther, Socey Thomas C., Cai Bill. Electronic Medical Record/Electronic Health Record Systems of Office-based Physicians: United States, 2009 and Preliminary 2010 State Estimates. National Center for Health Statistics, December 2010. URL: http://www.cdc.gov//nchs/data/hestat/emr_ehr_09/emr_ehr_09.pdf (дата звернення: 16.09.2024).

11 Jamoom E., Yang N., Hing E. Adoption of Certified Electronic Health Record Systems and Electronic Information Sharing in Physician Offices: United States, 2013 and 2014 // NCHS Data Brief. – 2015. – №236.

12 Patel V., DesRoches C., Charles D., Searcy T., Jha A. Progress and Challenges with the Implementation and Use of Electronic Health Records // Health Affairs. – 2019. – №38(7). – С. 1176–1184.

13 National Center for Health Statistics. Physician Adoption of Electronic Health Record Systems: United States, 2023. URL: https://www.cdc.gov/nchs/data/databriefs/dbxxx.pdf (дата звернення: 17.09.2024).

14 Neville R., Neville J., et al. Time Efficiency in Using Electronic Health Records in Intensive Care Units: A Comparative Study // Journal of Critical Care. – 2017. – Vol. 32. – С. 119–124.

15 Український науковий журнал. Використання електронних медичних карток для оптимізації роботи медичного персоналу // Наукові вісті НЛТУ України. – 2017. – №27(10).

16 VoxUkraine. Прогноз щодо економічної ефективності збільшення кількості медсестер і заміни лікарів медсестрами. URL: https://voxukraine.org/wp-content/uploads/2021/09/medsestrynstvo-web.pdf (дата звернення: 29.09.2024).

17 Patient safety. Wikipedia. URL: https://en.wikipedia.org/wiki/Patient_safety (дата звернення: 10.10.2024).

18 Russell, S., Norvig, P. *Artificial Intelligence: A Modern Approach*. – Prentice Hall, 2016. – 1152 p.

19 Goodfellow, I., Bengio, Y., Courville, A. *Deep Learning*. – MIT Press, 2016. – 775 p.

20 Benkhouya, A., Rahmani, H. *Synthesis of Abstractive Text Summarization*. – 2020. DOI: 10.13140/RG.2.2.25514.64964.

21 Ullah, I., Raza, B. *Software Defined Network Enabled Fog-to-Things Hybrid Deep Learning Driven Cyber Threat Detection System* // Security and Communication Networks. – 2021. – Vol. 2021. – P. 15. DOI: 10.1155/2021/6136670.

22 Li, J., Sun, L., He, L. *A Survey on Text Classification: From Traditional to Deep Learning* // ACM Transactions on Intelligent Systems and Technology. – 2022. – Vol. 13. – P. 1–41. DOI: 10.1145/3495162.

23 Qamar, U., Raza, M.S. *Deep Learning* // In: *Data Science Concepts and Techniques with Applications*. – Cham: Springer, 2023. – Pp. 217–270. DOI: 10.1007/978-3-031-17442-1_8.

24 Ghosh, S., Mishra, B. *Challenges in Automated Text Summarization* // IEEE Access. – 2020. – Vol. 8. – Pp. 24567–24579. DOI: 10.1109/ACCESS.2020.2965131.

25 Sutskever, I., Vinyals, O., Le, Q. V. *Sequence to Sequence Learning with Neural Networks* // Advances in Neural Information Processing Systems. – 2014. – Pp. 3104–3112.

26 Bahdanau, D., Cho, K., Bengio, Y. *Neural Machine Translation by Jointly Learning to Align and Translate* // ICLR. – 2015.

27 Hochreiter, S., Schmidhuber, J. *Long Short-Term Memory* // Neural Computation. – 1997.

28 Cho, K., van Merriënboer, B., Gulcehre, C., et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation* // EMNLP. – 2014.

29 Vaswani, A., Shazeer, N., Parmar, N., et al. *Attention Is All You Need* // Advances in Neural Information Processing Systems. – 2017.

30 Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* // NAACL-HLT. – 2019.

31 Brown, T., Mann, B., Ryder, N., et al. *Language Models are Few-Shot Learners* // Advances in Neural Information Processing Systems. – 2020.

32 Greff, K., Srivastava, R. K., Koutník, J., et al. *LSTM: A Search Space Odyssey* // IEEE Transactions on Neural Networks and Learning Systems. – 2017.

33 Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P., He, L. A Survey on Text Classification: From Traditional to Deep Learning // ACM Transactions on

Intelligent Systems and Technology. – 2022. – Vol. 13. – Pp. 1–41. – DOI: 10.1145/3495162.

34 Luong, M.-T., Pham, H., Manning, C. D. *Effective Approaches to Attention-based Neural Machine Translation* // EMNLP. – 2015.

35 Pascanu, R., Mikolov, T., Bengio, Y. *On the difficulty of training Recurrent Neural Networks* // Proceedings of the 30th International Conference on Machine Learning. – 2013.

36 TREC-2017 LiveQA: Medical Question Answering Task. URL: https://github.com/abachaa/LiveQA_MedicalTask_TREC2017/tree/master (дата звернення: 16.10.2024).

37 DialMed Dataset GitHub-репозиторій. URL: https://github.com/ (дата звернення: 20.11.2024).

38 He, Z., Han, Y., Ouyang, Z., Gao, W., Chen, H., Xu, G., Wu, J. DialMed: A Dataset for Dialogue-based Medication Recommendation // Proceedings of the 29th International Conference on Computational Linguistics. – 2022. – P. 721–733. – Gyeongju, Republic of Korea: International Committee on Computational Linguistics

39 Kingma, D. P., Ba, J. *Adam: A Method for Stochastic Optimization* // arXiv preprint arXiv:1412.6980. – 2014.

40 Papineni, K., Roukos, S., Ward, T., Zhu, W.-J. *BLEU: a Method for Automatic Evaluation of Machine Translation* // Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. – 2002.

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) **Магістр**
Галузь знань: 12 – Інформаційні технології
Спеціальність: 123 «Комп'ютерна інженерія»
Освітня програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
комп'ютерних систем та робототехніки
_____Максим ХРУСЛОВ
«12» вересня 2024 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Волинець Катерини Андріївни
(прізвище, ім'я, по батькові студента)

1.Тема роботи **Модель класифікації повідомлень у медичних інформаційних системах**

керівник роботи Стрілець Вікторія Євгенівна, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету ***№ 4101- 5/3657 від 12 листопада 2024 року***

2. Строк подання студентом роботи ***30 листопада 2024 року***

3. Перелік питань, які потрібно розробити

1. Аналіз сучасних методів обробки текстових даних та їх класифікації.
2. Розробка та налаштування моделей для автоматизованої класифікації медичних повідомлень.
3. Дослідження ефективності та якості моделей для класифікації медичних повідомлень.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Проведення літературного огляду з проблематики автоматизованої класифікації медичних повідомлень	5.09.2024 – 15.09.2024
2	Огляд моделей класифікації повідомлень у медичних інформаційних системах та алгоритми їх застосування	16.09.2024 – 1.10.2024
3	Підбір та аналіз даних для навчання моделей автоматизованої класифікації	01.10.2024 – 15.10.2024
4	Розробка та налаштування моделей для класифікації медичних запитів. Вибір відповідної архітектури.	16.10.2024 – 24.10.2024
5	Аналіз продуктивності моделей на тестових даних. Порівняння різних підходів та методів оцінки.	16.10.2024 – 17.10.2024
6	Розробка рекомендацій щодо застосування моделей автоматизованої класифікації в медичних інформаційних системах	17.10.2024 – 20.10.2024
7	Оформлення пояснювальної записки	01.10.2024 – 27.11.2024
8	Представлення кваліфікаційної роботи керівнику та рецензенту	28.11.2024 – 30.11.2024

5. Дата видачі завдання ***5 вересня 2024 року.***

Студент К.А.Волинець _____
ініціали, прізвище підпис

Керівник роботи В.Є. Стрілець ____ _____ ____
ініціали, прізвище підпис

Додаток Б

Затверджую

_____

«_____»_____2024 р.

Технічне завдання

на розробку програмного виробу «МОДЕЛЬ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ У МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ**»**

1.	Введення	1.1. Назва: МОДЕЛЬ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ У МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ 1.2. Галузь застосування: системи управління медичними закладами.
2.	Підстава для розробки	2.1. Навчальний план за спеціальністю 123 – Комп'ютерна інженерія. 2.2. Завдання на кваліфікаційну роботу магістра № ***4101-5/3657*** від ***12 листопада 2024 року*** (представити як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3.	Призначення розробки	3.1. Мета розробки: розробити автоматизовану модель для резюмування текстових медичних запитів. 3.2. Надає можливість автоматично обробляти текстові медичні запити, визначати їх основний зміст, створювати узагальнені резюме, що дозволяє скоротити час аналізу та прийняття рішень лікарями. 3.3. Вихідні дані розробки: медичні текстові дані з відкритих датасетів, які містять реальні запити пацієнтів і відповіді і типів інформації.
4.	Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: модель повинна забезпечувати точність резюмування не меншу за існуючі аналоги. 4.2. Вимоги до надійності: забезпечувати точність резюмування текстів, збереження ключової інформації та релевантність відповідей 4.3.Вимоги до умов експлуатації: немає 4.4. Вимоги до складу і параметрів технічних засобів: обчислювальне обладнання з 16 ГБ оперативної пам'яті. 4.5. Вимоги до інформаційної та програмної сумісності: сумісність з C#, Python, TensorFlow або PyTorch, а також з іншими бібліотеками для обробки природної мови. 4.6. Вимоги до маркування та упаковки: немає

<table>
<tr><td></td><td></td><td colspan="2">4.7. Вимоги до транспортування і зберігання: на цифрових носіях інформації або у вигляді хмарного програмного продукту.
4.8. Спеціальні вимоги: не передбачені.</td></tr>
<tr><td>5.</td><td>Вимоги до програмної документації</td><td colspan="2">Програмною документацією до виробу «Метод аналізу інформативності змінних стану при діагностиці систем з використанням інформаційних критеріїв» вважати:
1) Справжнє Технічне завдання на розробку виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).
2) Методику резюмування текстових медичних запитів (представити в розділах 2 та 3 пояснювальної записки до кваліфікаційної роботи).
3) Опис моделі та архітектури (представити в розділі 3 пояснювальної записки до кваліфікаційної роботи).</td></tr>
<tr><td>6.</td><td>Вимоги до техніко-економічних показників</td><td colspan="2">Програмною документацією до виробу «Метод аналізу інформативності змінних стану при діагностиці систем з використанням інформаційних критеріїв» вважати:
1) Справжнє Технічне завдання на розробку виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).
2) Методику розрахунку інформативності змінних стану (у вигляді розділу 3 пояснювальної записки до кваліфікаційної роботи).
3) Опис виробу (представити в розділі 3 пояснювальної записки до кваліфікаційної роботи)</td></tr>
<tr><td rowspan="6">7.</td><td rowspan="6">Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td>від 2 жовтня 2024
до 16 жовтня 2024</td><td>Аналіз практики предметної області.</td></tr>
<tr><td>від 2 жовтня 2024
до 2 листопада 2024</td><td>Аналіз існуючих моделей та алгоритмів для резюмування текстових даних.</td></tr>
<tr><td>від 3 листопада 2024
до 30 листопада 2024</td><td>Підготовка даних для навчання моделі (токенізація, нормалізація, анотація датасетів).</td></tr>
<tr><td>від 10 листопада 2024
до 17 листопада 2024</td><td>Розробка архітектури моделі Seq2Seq з механізмом уваги.</td></tr>
<tr><td>від 12 листопада 2024
до 21 листопада 2024</td><td>Навчання моделі на основі підготовленого датасету.</td></tr>
</table>

		від 1 вересня 2024 до 21 вересня 2024	Тестування та оптимізація моделі на реальних медичних запитаннях.
		від 22 вересня 2024 до 15 жовтня 2024	Оформлення результатів. Написання пояснювальної записки.
		від 16 листопада 2024 до 21 листопада 2024	Представлення кваліфікаційного проекту керівнику кваліфікаційної роботи та рецензенту.
8.	Порядок контролю і приймання програмного продукту (моделі)	**1.** Перевірку ходу розробки програми виконувати раз в 3 тижні. **2.** Захист розробленої моделі провести на засіданні Атестаційної комісії. Пояснювальну записку подати на паперових носіях в 1 примірнику і в електронному вигляді в 1 примірнику.	

Виконавець

студент групи КІ- 61

Волинець К. А. _____

Замовник

к. т. н., доцент кафедри КСР

Стрілець В. Є . _____

Додаток В

Програма і методика випробувань програмного виробу
«МОДЕЛЬ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ У МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ»

1 Об'єкт випробувань

Назва програмного виробу : «МОДЕЛЬ КЛАСИФІКАЦІЇ ТА ПРОГНОЗУВАННЯ У МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ»

Галузь застосування : Автоматизація медичних процесів та обробка даних у медичних інформаційних системах.

2. Мета випробувань

Перевірка відповідності заявлених функціональних можливостей системи (Додаток Б до пояснювальної записки до кваліфікаційної роботи).

3. Загальні положення

Підстави для проведення випробувань

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

Місце і тривалість випробувань

Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період переддипломної практики.

Обсяг випробувань

Приймальні випробування програмного вироби проводяться в обсязі відповідному цієї програми і методики випробувань.

Організації, які беруть участь у випробуваннях

Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу

Модель повинна задовольняти наступним вимогам:

- працювати на операційних системі Windows;

- передбачена точна класифікація повідомлень та автоматичне прогнозування діагнозів;
- обчислювальне обладнання з оперативною пам’яттю не менше 16 ГБ.
- система має бути інтегрована з іншими компонентами, включаючи бази даних SQL Server;
- підтримка мов програмування C#, Python та бібліотек TensorFlow або PyTorch.
- сумісність із сучасними бібліотеками для обробки природної мови, такими як SpaCy, NLTK, Hugging Face Transformers.
- обробка запитів повинна виконуватися швидко і стабільно;
- вимоги до маркування та упаковки (не висуваються);
- вимоги до транспортування і зберігання (не висуваються).
- спеціальні вимоги (не пред’являються).

5. Вимоги до програмної документації

Програмною документацією щодо розроблюваного програмного продукту вважати:

- дане технічне завдання на розробку програми (представити як Додаток Б до пояснювальної записки до кваліфікаційної роботи);
- програму і методику випробувань розробленої програми (представити як Додаток В до пояснювальної записки до кваліфікаційної роботи);
- рекомендацій щодо застосування створеної програмної стандартизації у проектах (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).
- текст програми (представити як Додаток Г до пояснювальної записки до кваліфікаційної роботи).

6. Засоби і порядок випробувань

6.1 Засоби випробувань

- Для проведення випробувань необхідний проект для розробки веб-додатку на мові програмування C# та Python з використання бібліотек TensorFlow або PyTorch.

6.2 Порядок проведення випробувань

Як правило, випробування проводяться в два етапи:

- ознайомчий (1-й етап);

- випробування програмного виробу (2-й етап).

Перелік перевірок, що проводяться на 1 етапі випробувань, включає в себе:

– Перевірку комплектності програмної документації.

– Перевірка комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в ТЗ документації.

– Перевірку комплектності складу технічних і програмних засобів.

– Методику проведення перевірок на 1 етапі випробувань.

Якість програмної документації перевіряється на відповідність вимогам стандартів ЕСПД.

Перелік перевірок, що проводяться на 2 етапі випробувань, включає в себе:

– перевірку відповідності технічних характеристик програми вимогам технічного завдання;

– перевірку ступеня виконання функціональних вимог до програми;

– методику проведення перевірок, що входять до переліку по 2 етапу випробувань.

Програма працює відповідно до умов експлуатації операційної системи Windows.

Для роботи необхідний компілятор мови програмування python, версії не нижчої ніж 8.0 та .NET 8

Порядок проведення випробувань:

1. Запуск програми здійснюється через веб-сервер, реалізований на мові програмування C# у середовищі ASP.NET;

2. Після запуску програми необхідно у браузері комп’ютеру перейти за посиланням: http://localhost:7251/;

3. У вікні браузера відкриється інтерфейс користувача для обміну повідомленнями між лікарем та пацієнтом;

4. Для проведення тестування потрібно авторизуватись в системі під певною роллю (лікаря чи пацієнта).

Для перевірки функціональності програми пропонується виконати такі тести.

Тест 1: Перевірка створення повідомлення

- **Передумови:** Чат між лікарем і пацієнтом ще не має повідомлень.
- **Дії:**
  1. Відкрити чат між лікарем і пацієнтом.
  2. Ввести текст повідомлення у поле для введення.
  3. Натиснути кнопку «Відправити».
- **Очікуваний результат:**

У базі даних таблиці ChatMessages створюється новий запис із зазначенням тексту повідомлення, часу відправлення, статусу «Доставлено».

До проведення тестування записів в таблиці немає (рис. В.1) та маємо порожній чат (рис. В.2).

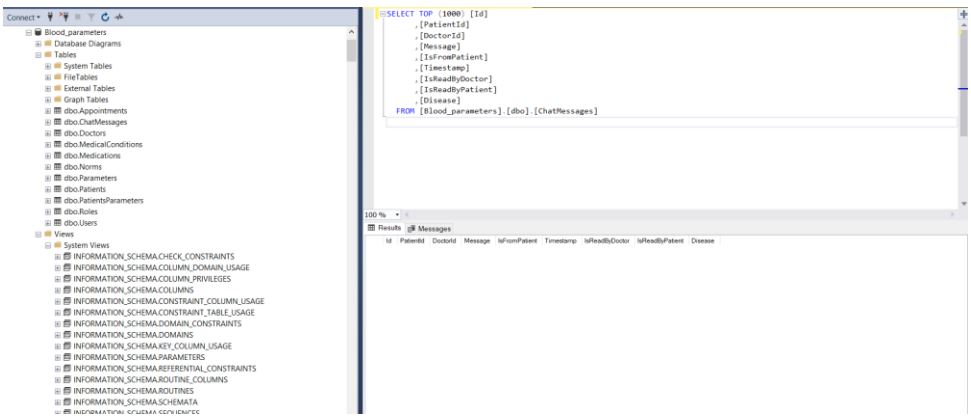


Рисунок В. 1 – Записів в таблиці не знайдено

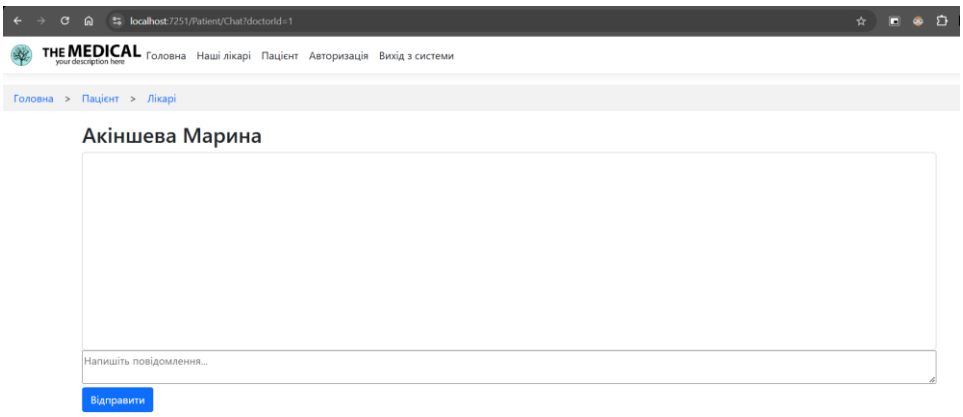


Рисунок В. 2 – Відображення чату з порожнім діалогом на сторінці пацієнта

Після проведення тестування таблиця ChatMessages є заповненою і повідомлення у чаті з’явилися.

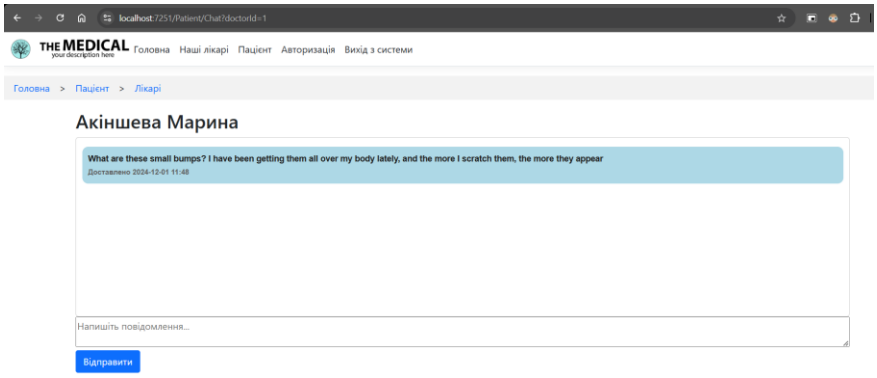


Рисунок В. 3 – Відображення відправленого повідомлення пацієнтом, але не переглянутого Лікарем

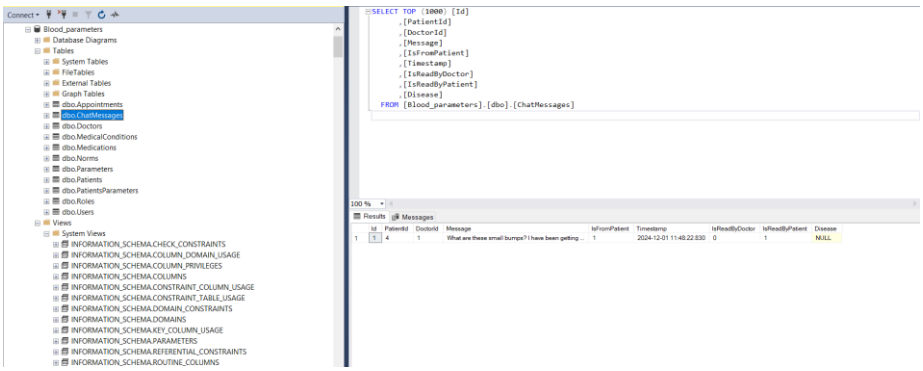


Рисунок В. 4 – Відображення запису в таблиці ChatMessages із зазначенням тексту повідомлення, часу відправлення

Фактичний результат: Повідомлення успішно створено, відображається в чаті, позначено як «Доставлено». Дані про повідомлення коректно збережені у базі даних.

Тест 2: Перевірка статусу повідомлення

- **Переумови: успішне виконання Тесту 1**
- **Дії:**

1. Перевірити, чи у лікаря відображається позначка про непрочитане повідомлення.
2. Відкрити повідомлення пацієнтом.
3. Ввести текст повідомлення у поле для введення.
4. Натиснути кнопку «Відправити».

- **Очікуваний результат:**

У базі даних таблиці ChatMessages створюється новий запис із зазначенням тексту повідомлення, часу відправлення, статусу «Доставлено» і зміна статусу в вже переглянутого повідомлення.

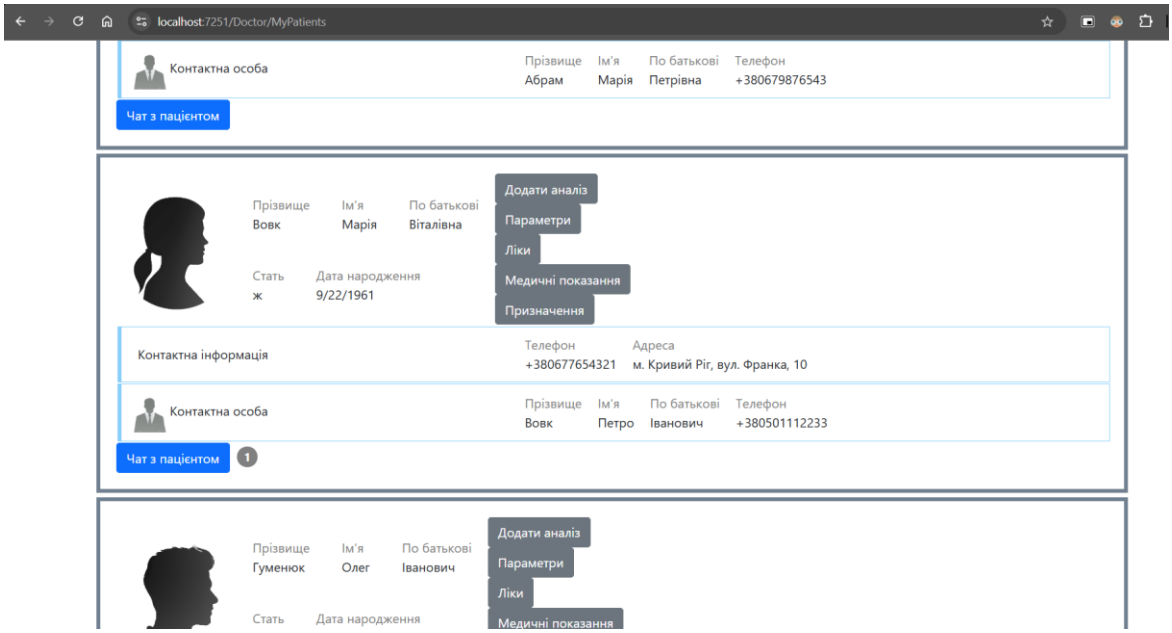


Рисунок В. 5 – Відображення позначки з кількістю нових повідомлень

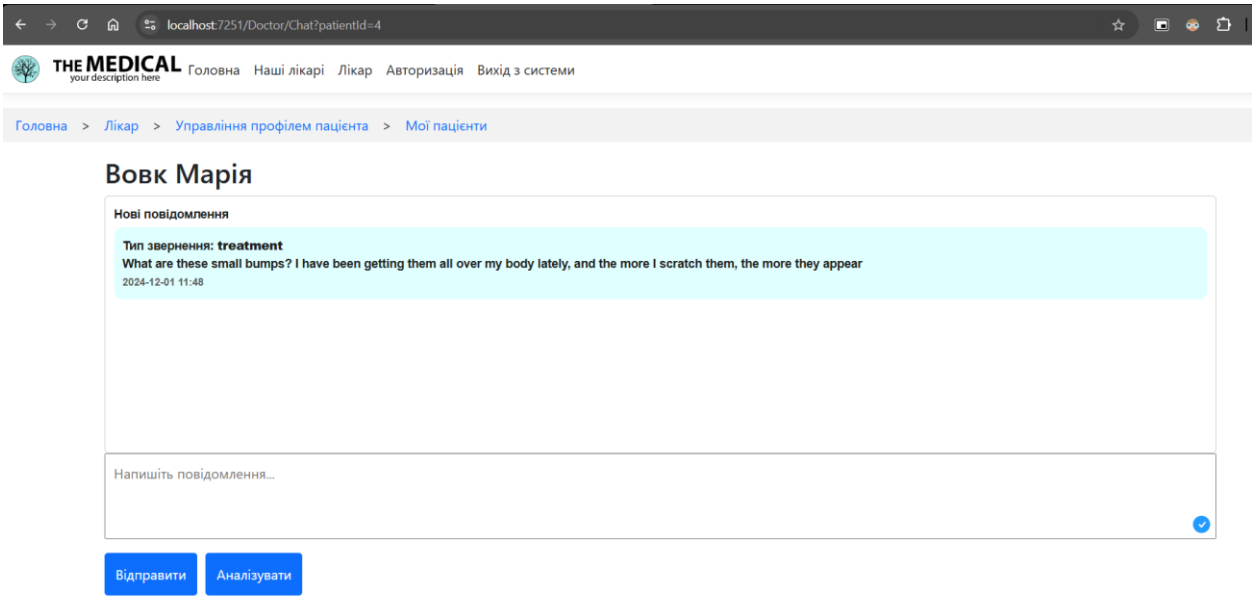


Рисунок В. 6 – Відображення чату з пацієнтом з повідомленням про нове повідомленя та класифікацією повідомлення

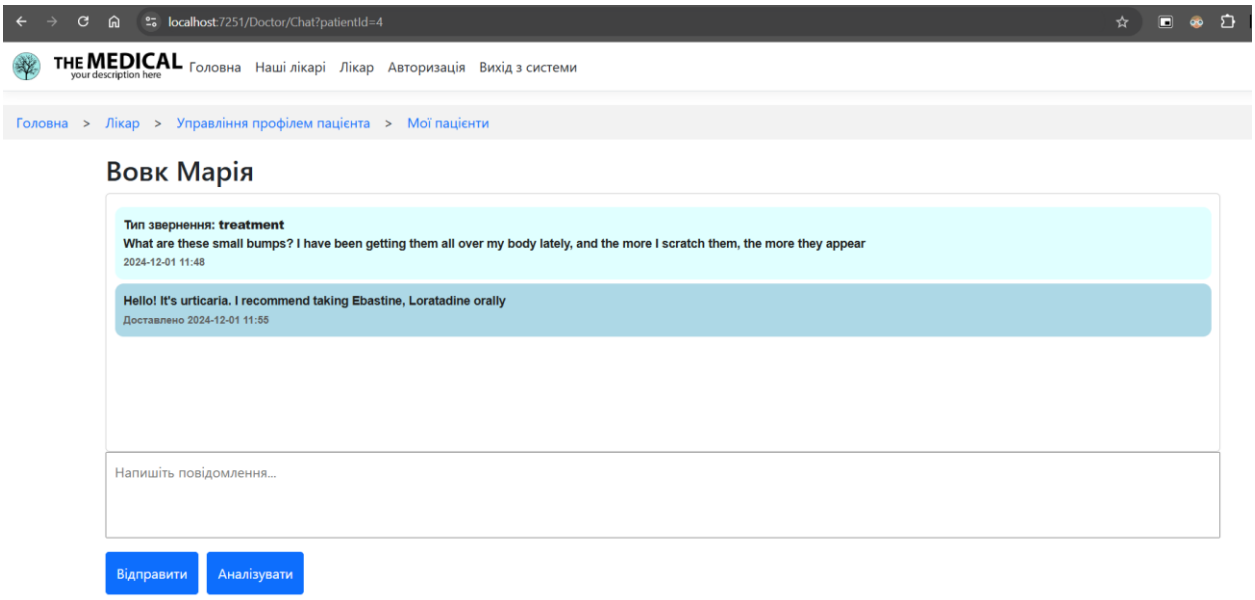


Рисунок В. 7 – Відображення відправленого повідомлення пацієнтом, але не переглянутого Пацієнтом

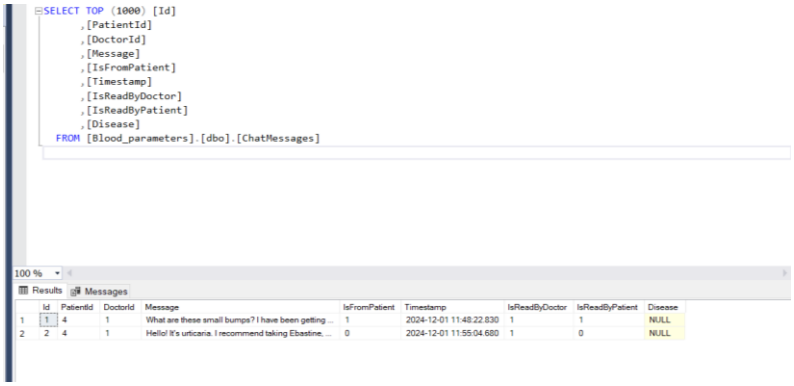


Рисунок В. 8 – Відображення записів в таблиці ChatMessages із зазначенням тексту повідомленнь, часу відправлення

Фактичний результат. Після відкриття повідомлення лікарем відображається класифікація повідомлення. Дані у базі даних оновлені відповідно.Повідомлення лікаря успішно створено, відображається в чаті, позначено як «Доставлено». Дані про повідомлення коректно збережені у базі даних.

Тест 3: Прогнозування діагнозу

Передумови: лікар отримав повідомлення від пацієнта

- **Дії:**

1. Лікар натискає кнопку «Аналізувати» у чаті з пацієнтом.
2. Система передає історію повідомлень на сервер для аналізу.

- **Очікуваний результат:**

1. Система повертає прогнозований діагноз (наприклад, "Allergic dermatitis").
2. Діагноз зберігається у базі даних та відображається в інтерфейсі лікаря.

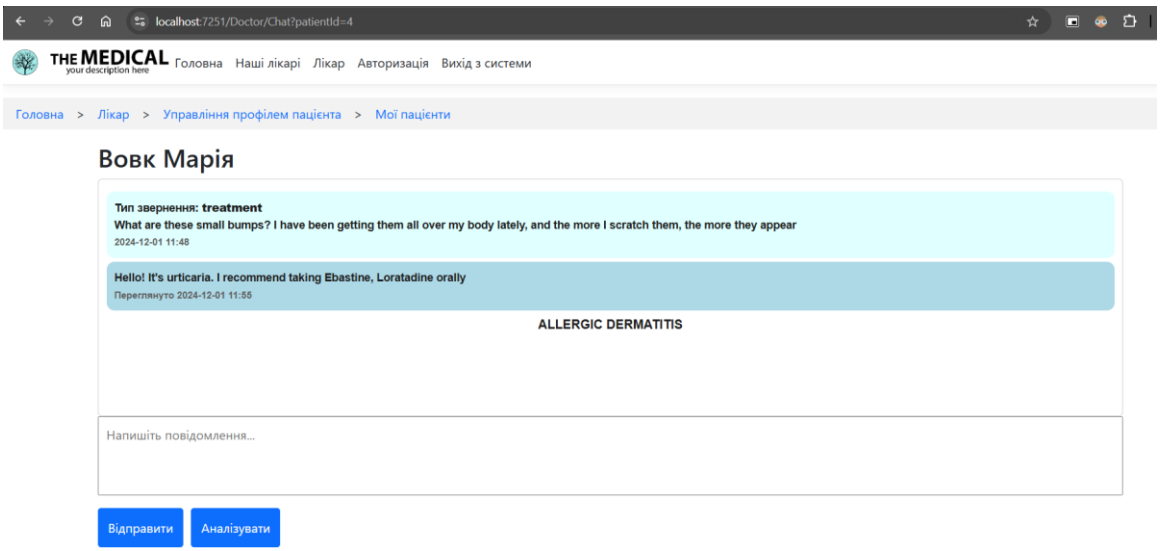


Рисунок В. 9 – Відображення можливого діагнозу пацієнта

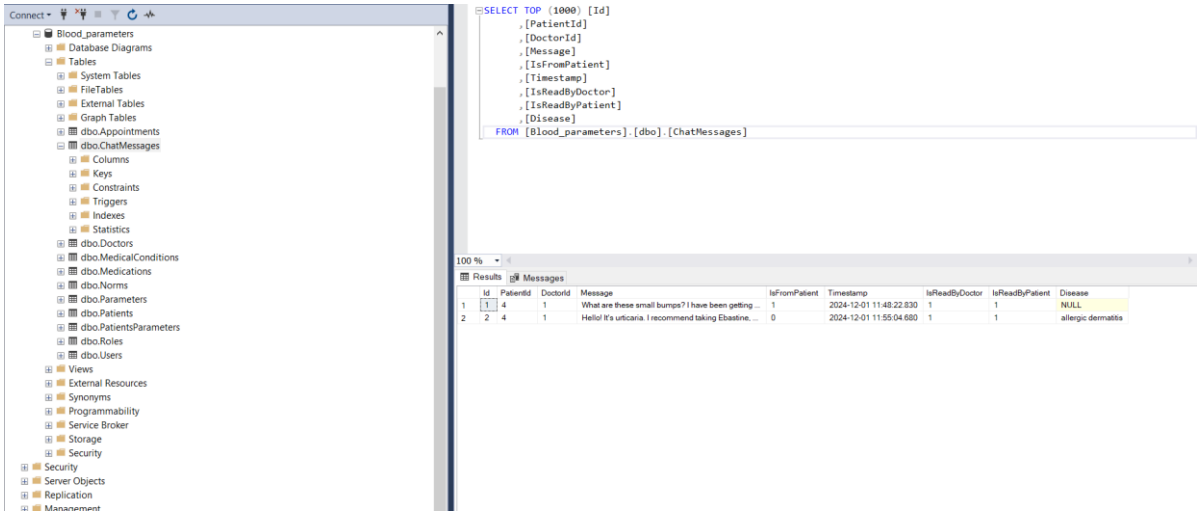


Рисунок В. 10 – Відображення запису в таблиці ChatMessages із зазначенням можливого діагнозу пацієнта

Висновки

Усі тести успішно виконані. Випробування підтвердили надійність роботи програми та її відповідність технічному завданню.

Виконавець: студентка групи КІ-61, Волинець К. А. _____

Додаток Г

Лістинг коду «Модель прогнозування діагнозу»

```
import torch
import torch.nn as nn
import re
import nltk
from tqdm import tqdm

# Ensure NLTK tokenizer is available
nltk.download('punkt')

# Define constants
EMBED_SIZE = 256
HIDDEN_SIZE = 512
NUM_LAYERS = 1
MAX_LEN = 50

# Preprocessing function
def preprocess_text(text):
    """
    Preprocess text by lowercasing, removing special characters, and tokenizing.
    """
    text = text.lower()
    text = re.sub(r'[^a-z0-9\s]', '', text)
    tokens = nltk.word_tokenize(text)
    return tokens

# Encode text to indices
def encode_text(tokens, word2idx):
    """
    Convert tokens into indices using the provided vocabulary (word2idx).
    """
    return [word2idx.get(token, word2idx['<UNK>']) for token in tokens]

# Pad sequences to a fixed length
def pad_sequence(seq, max_len, pad_value):
    """
    Pad or truncate a sequence to the maximum length with the pad_value.
    """
    seq = seq[:max_len]
    seq += [pad_value] * (max_len - len(seq))
    return seq

# Define Encoder model
class Encoder(nn.Module):
    def __init__(self, input_size, embed_size, hidden_size, num_layers):
        super(Encoder, self).__init__()
        self.embedding = nn.Embedding(input_size, embed_size)
        self.lstm = nn.LSTM(embed_size, hidden_size, num_layers, bidirectional=True)

    def forward(self, x):
        embedding = self.embedding(x)
        outputs, (hidden, cell) = self.lstm(embedding)
        hidden = hidden.view(NUM_LAYERS, 2, -1, HIDDEN_SIZE).sum(dim=1)
        cell = cell.view(NUM_LAYERS, 2, -1, HIDDEN_SIZE).sum(dim=1)
        return outputs, (hidden, cell)

# Define Attention mechanism
```

```
class Attention(nn.Module):
    def __init__(self, encoder_hidden_dim, decoder_hidden_dim):
        super(Attention, self).__init__()
        self.attn = nn.Linear((encoder_hidden_dim * 2) + decoder_hidden_dim,
decoder_hidden_dim)
        self.v = nn.Linear(decoder_hidden_dim, 1, bias=False)

    def forward(self, hidden, encoder_outputs):
        src_len = encoder_outputs.shape[0]
        hidden = hidden.unsqueeze(1).repeat(1, src_len, 1)
        encoder_outputs = encoder_outputs.permute(1, 0, 2)
        energy = torch.tanh(self.attn(torch.cat((hidden, encoder_outputs), dim=2)))
        attention = self.v(energy).squeeze(2)
        return nn.functional.softmax(attention, dim=1)

# Define Decoder model
class Decoder(nn.Module):
    def __init__(self, output_dim, embed_size, hidden_size, num_layers, attention):
        super(Decoder, self).__init__()
        self.output_dim = output_dim
        self.attention = attention
        self.embedding = nn.Embedding(output_dim, embed_size)
        self.lstm = nn.LSTM((hidden_size * 2) + embed_size, hidden_size, num_layers)
        self.fc_out = nn.Linear((hidden_size * 2) + hidden_size + embed_size,
output_dim)

    def forward(self, input, hidden, cell, encoder_outputs):
        input = input.unsqueeze(0)
        embedded = self.embedding(input)
        a = self.attention(hidden[-1], encoder_outputs)
        a = a.unsqueeze(1)
        encoder_outputs = encoder_outputs.permute(1, 0, 2)
        weighted = torch.bmm(a, encoder_outputs).permute(1, 0, 2)
        lstm_input = torch.cat((embedded, weighted), dim=2)
        output, (hidden, cell) = self.lstm(lstm_input, (hidden, cell))
        embedded = embedded.squeeze(0)
        output = output.squeeze(0)
        weighted = weighted.squeeze(0)
        prediction = self.fc_out(torch.cat((output, weighted, embedded), dim=1))
        return prediction, hidden, cell

# Define Seq2Seq model
class Seq2Seq(nn.Module):
    def __init__(self, encoder, decoder, device):
        super(Seq2Seq, self).__init__()
        self.encoder = encoder
        self.decoder = decoder
        self.device = device

    def forward(self, src, trg, teacher_forcing_ratio=0.5):
        batch_size = src.shape[1]
        max_len = trg.shape[0]
        trg_vocab_size = self.decoder.output_dim
        outputs = torch.zeros(max_len, batch_size, trg_vocab_size).to(self.device)
        encoder_outputs, (hidden, cell) = self.encoder(src)
        input = trg[0, :]
        for t in range(1, max_len):
            output, hidden, cell = self.decoder(input, hidden, cell, encoder_outputs)
            outputs[t] = output
            top1 = output.argmax(1)
            input = trg[t] if torch.rand(1).item() < teacher_forcing_ratio else top1
        return outputs
```

```
# Load the vocabulary and model
word2idx = torch.load("word2idx.pth", weights_only=False)
idx2word = {idx: word for word, idx in word2idx.items()}
vocab_size = len(word2idx)

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

attn = Attention(HIDDEN_SIZE, HIDDEN_SIZE)
enc = Encoder(vocab_size, EMBED_SIZE, HIDDEN_SIZE, NUM_LAYERS)
dec = Decoder(vocab_size, EMBED_SIZE, HIDDEN_SIZE, NUM_LAYERS, attn)
model = Seq2Seq(enc, dec, device).to(device)

# Load trained model weights
model.load_state_dict(torch.load("seq2seq_attention_model.pth", weights_only=False))
model.eval()

# Function for translating multi-turn dialogs
def translate_dialog(dialog, max_len=MAX_LEN):
    """
    Translate a multi-turn dialog into a summary using the Seq2Seq model.
    """
    model.eval()
    dialog_text = " ".join(dialog)  # Concatenate all dialog turns
    tokens = preprocess_text(dialog_text)
    indices = encode_text(tokens, word2idx)
    indices = [word2idx['<SOS>']] + indices + [word2idx['<EOS>']]
    indices = pad_sequence(indices, max_len, word2idx['<PAD>'])
    src_tensor = torch.tensor(indices).unsqueeze(1).to(device)  # (seq_len,
batch_size=1)

    encoder_outputs, (hidden, cell) = model.encoder(src_tensor)

    trg_indices = [word2idx['<SOS>']]
    for _ in range(max_len):
        trg_tensor = torch.tensor([trg_indices[-1]]).to(device)
        with torch.no_grad():
            output, hidden, cell = model.decoder(trg_tensor, hidden, cell,
encoder_outputs)
        pred_token = output.argmax(1).item()
        trg_indices.append(pred_token)
        if pred_token == word2idx['<EOS>']:
            break

    trg_tokens = [idx2word[idx] for idx in trg_indices]
    return " ".join(trg_tokens[1:-1])  # Remove <SOS> and <EOS>
import sys
# Test the model with a dialog
test_dialog = sys.argv[1].split("|")

# Generate and print the summary
output = translate_dialog(test_dialog)
#print(f"Input Dialog:\n{' '.join(test_dialog)}")
print(f"{output}")
```

код виклику скрипта для отримання діагнозу

```
          int idFirstMessage = 0;
                int idLastMessage = 0;
                string aiDisease;
                List<string> conversation = new List<string> { };
                foreach (var message in messages)
                {
```

```
                    idLastMessage++;
                    if (message.Disease != null)
                    {
                        idFirstMessage = idLastMessage;
                    }
                }
                if (idLastMessage != 0 && idLastMessage-idFirstMessage != 0)
                {
                        for (int i = idFirstMessage; i < idLastMessage; i++)
                        {
                            if (messages[i].IsFromPatient == true) {

                                conversation.Add("Patient: " + messages[i].Message);
                            }
                            else {
                                conversation.Add("Doctor: " + messages[i].Message);
                            }
                        }

                    string pythonExe = "python";
                    string scriptPath = Path.Combine(Environment.CurrentDirectory,
"test3.py");

                    var processStartInfo = new System.Diagnostics.ProcessStartInfo
                    {
                        FileName = pythonExe,
                        Arguments = $"{scriptPath} \"{string.Join("|",
conversation)}\"",
                        RedirectStandardOutput = true,
                        RedirectStandardError = true,
                        UseShellExecute = false,
                        CreateNoWindow = true
                    };

                    using (var process =
System.Diagnostics.Process.Start(processStartInfo))
                    {
                        string output = process.StandardOutput.ReadToEnd();
                        string error = process.StandardError.ReadToEnd();
                        process.WaitForExit();

                        if (process.ExitCode != 0)
                        {
                            Console.WriteLine($"Error executing Python script:
{error}");
                            return View("Error");
                        }

                        aiDisease = output.Trim();
                    }
                    messages[idLastMessage-1].Disease = aiDisease;
                    db.SaveChanges();
                }
```