Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему *Сегментація ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі*

Галузь знань: *12 – Інформаційні технології* (Шифр і назва галузі знань)

Спеціальність: *122 – Комп'ютерні науки* (Шифр і назва спеціальності)

Освітня програма: *Комп'ютерні науки* (Назва освітньої програми)

Виконав: *студент 4 курсу, група КН-19-2* (Курс, група виконавця) _____ (Підпис) *М.О. Діхтяр* (Ініціали, прізвище)

Керівник: *док. філ., ст. викладач кафедри КН* (Науковий ступінь, посада) _____ (Підпис) *П.М. Радюк* (Ініціали, прізвище)

Нормоконтроль: *к.т.н., доцент кафедри КН* (Науковий ступінь, посада) _____ (Підпис) *Р.О. Багрій* (Ініціали, прізвище)

До захисту допускаю:
Зав. кафедри КН, д.т.н., професор _____ (Підпис) *О.В. Бармак* (Ініціали, прізвище)

05 06 2023 р.

Хмельницький 2023

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет *інформаційних технологій*
Кафедра *комп'ютерних наук*
Освітній ступінь *бакалавр*
Галузь знань *12 – Інформаційні технології*
Спеціальність *122 – Комп'ютерні науки*
Освітня програма *освітньо-професійна програма підготовки бакалавра*

ЗАТВЕРДЖУЮ
Завідувач кафедри *комп'ютерних наук*

_____
(підпис)
д.т.н., професор О.В. Бармак
«*06*« *03* *2023* року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

1. Тема кваліфікаційної роботи бакалавра: *«Сегментація ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі»*

2. Завдання видано студенту *Діхтяру Максиму Олександровичу*
(прізвище, ім'я, по батькові)

3. Керівник роботи *ст. викладач кафедри КН Радюк Павло Михайлович*
(посада, прізвище, ім'я, по батькові)

4. Затверджено наказом університету від «*01*« *03* *2023*р. № *5*

5. Дата видачі завдання студенту: «*03*« *03* *2023*р.

6. Зміст пояснювальної записки (перелік задач) та вихідні дані:

Провести аналіз методів, способів та технологій цифрової сегментації за зображеннями магнітно-резонансної томографії та обрати найкращий; застосувати обраний спосіб цифрової сегментації до розв'язання задачі автоматизованої сегментації ділянок серця; реалізувати обраний спосіб у вигляді модуля програмного забезпечення; провести експериментальне тестування реалізованого модуля за еталонними наборами даних; вихідними даними є маски сегментованих ділянок серця на зображеннях магнітно-резонансної томографії.

7. Календарний план виконання кваліфікаційної роботи бакалавра:

№	Назва етапів (розділів) кваліфікаційної роботи бакалавра	Терміи виконання	Примітка
1	*Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи бакалавра з керівником*	*грудень 2022*	виконано
2	*Ознайомлення з предметною областю, формулювання мети та задач дослідження, визначення об'єкта та предмета дослідження*	*січень 2023*	виконано
3	*Робота над розділом 1 – Дослідження процесу цифрової сегментації ділянок серця за зображеннями магнітно-резонансної томографії*	*січень 2023*	виконано
4	*Робота над розділом 2 – Спосіб сегментації ділянок серця на зображеннях магнітно-резонансної томографії*	*березень 2023*	виконано
5	*Робота над розділом 3 – Програмна реалізація інформаційної системи з використанням обраного способу сегментації ділянок серця*	*квітень 2023*	виконано
6	*Оформлення пояснювальної записки згідно вимог*	*травень 2023*	виконано
7	*Попередній захист кваліфікаційної роботи бакалавра*	*травень 2023*	виконано
8	*Захист кваліфікаційної роботи бакалавра на засіданні Екзаменаційної комісії*	*червень 2023*	виконано

Виконавець: *студент 4 курсу, група КН-19-2* (Курс, група виконавця) _____ (Підпис) *М.О. Діхтяр* (Ініціали, прізвище)

Керівник: *док. філ., ст. викладач кафедри КН* (Науковий ступінь, посада) _____ (Підпис) *П.М. Радюк* (Ініціали, прізвище)

Анотація

Тема кваліфікаційної роботи бакалавра: *Сегментація ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі*

Виконавець кваліфікаційної роботи бакалавра: *студент групи КН-19-2 Діхтяр Максим Олександрович*

Керівник кваліфікаційної роботи бакалавра: *доктор філософії, ст. викладач кафедри КН Радюк Павло Михайлович*

Кваліфікаційна робота бакалавра містить:

Пояснювальна записка				Кількість додатків
Сторінок	Рисунків	Таблиць	Джерел інформації	
62	25	3	27	2

Метою кваліфікаційної роботи бакалавра є підвищення точності цифрової сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

Зміст пояснювальної записки (перелік задач) та вихідні дані: провести аналіз методів, способів та технологій цифрової сегментації за зображеннями магнітно-резонансної томографії (МРТ) та обрати найкращий; застосувати обраний спосіб цифрової сегментації до розв'язання задачі автоматизованої сегментації ділянок серця; реалізувати обраний спосіб у вигляді модуля програмного забезпечення; провести експериментальне тестування реалізованого модуля за еталонними наборами даних; вихідними даними є маски сегментованих ділянок серця на зображеннях МРТ.

Досягнення мети кваліфікаційної роботи бакалавра полягає у створенні інформаційної системи, використання якої дасть змогу підвищити точність цифрової сегментації ділянок серця на зображеннях МРТ.

Ключові слова: зображення магнітно-резонансної томографії, патології серця, цифрова сегментація, згорткова нейронна мережа.

Виконавець: *студент 4 курсу, група КН-19-2* (Курс, група виконавця) _____ (Підпис) *М.О. Діхтяр* (Ініціали, прізвище)

Зміст

Перелік скорочень

Скорочення, термін, позначення	Пояснення
МРТ	Магнітно-резонансна томографія
ПЗ	Програмне забезпечення
КРБ	Кваліфікаційна робота бакалавра
КН	Комп’ютерні науки
ЗНМ	Згорткові нейронні мережі
ШІ	Штучний інтелект
КТ	Комп’ютерна томографія

Вступ

Актуальність. Сегментація ділянок серця на зображеннях магнітно-резонансної томографії МРТ є важливим завданням в області медичного зображення. Згорткові нейронні мережі виявилися потужним інструментом для автоматичної сегментації, оскільки вони можуть виконувати навчання на великих наборах даних і відтворювати складні неоднорідності структури серця.

Актуальність полягає в тому, що точна сегментація ділянок серця дозволяє лікарям отримувати важливі клінічні показники, такі як об’єм лівого шлуночка, товщина стінок серця та об’єм маси м’язів, що допомагає в діагностиці та виборі оптимального лікування серцевих захворювань. Точна сегментація також може використовуватися для планування хірургічних втручань та оцінки ефективності терапії у пацієнтів.

Згорткові нейронні мережі демонструють високу точність та швидкість обробки, що робить їх ефективними для автоматичної сегментації зображень магнітно-резонансної томографії серця. Вони можуть виявляти навіть невеликі деталі та неоднорідності, що допомагає у виявленні патологій та аномалій серця.

Таким чином, використання згорткових нейронних мереж для сегментації ділянок серця на зображеннях магнітно-резонансної томографії має великий потенціал для покращення точності діагностики та планування лікування серцевих захворювань.

Об’єкт дослідження – процес цифрової сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

Предмет дослідження – методи, засоби та технології цифрової сегментації ділянок серця за зображеннями магнітно-резонансної томографії.

Мета кваліфікаційної роботи бакалавра – підвищення точності цифрової сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

Завдання кваліфікаційної роботи бакалавра – для досягнення поставленої мети визначено наступні завдання:

1. Провести аналітичний огляд методів, способів та технологій цифрової сегментації за зображеннями магнітно-резонансної томографії та обрати найкращий.

2. Застосувати обраний спосіб цифрової сегментації до розв’язання задачі автоматизованої сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

3. Реалізувати обраний спосіб у вигляді модуля програмного забезпечення для автоматизованої сегментації ділянок серця.

4. Провести експериментальне тестування реалізованого модуля за еталонними наборами даних.

Розділ 1 Дослідження процесу цифрової сегментації ділянок серця за зображеннями магнітно-резонансної томографії

1.1 Аналіз медичної сегментації ділянок серця на зображеннях магнітно-резонансної томографії

Захворювання серця є широко поширеною проблемою для здоров'я людини, яка має величезний вплив на незліченну кількість людей у всьому світі. Найбільший відсоток смертей українців припав якраз на серцево-судинні захворювання (рисунок 1.1) [1].

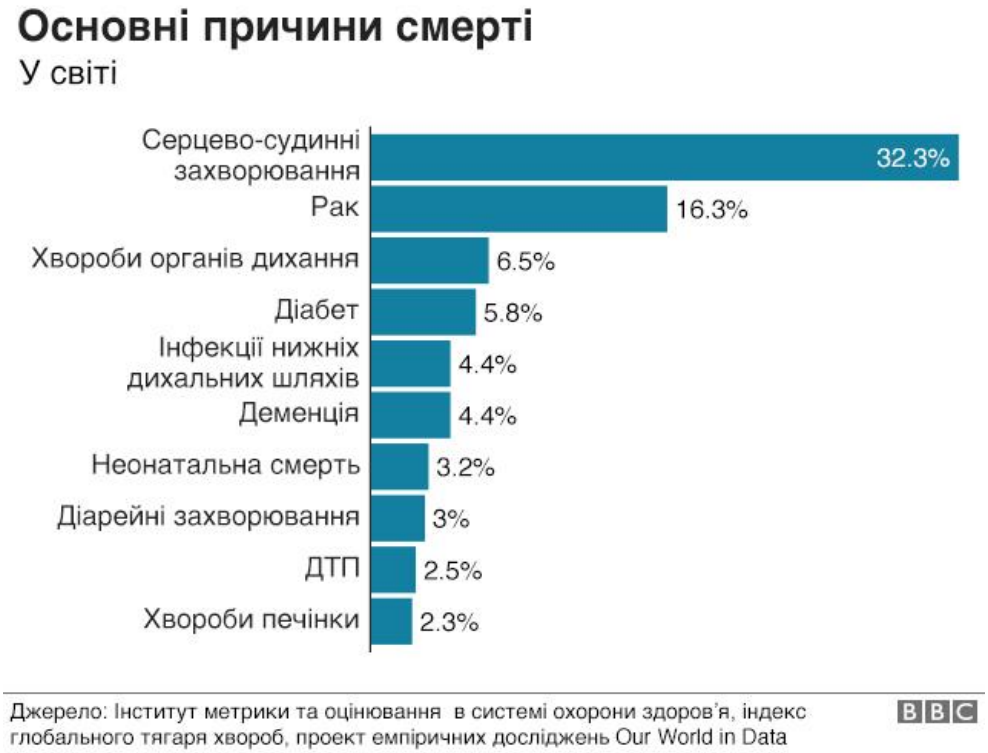


Рисунок 1.1 – Основні причини смерті [1]

Серце є життєво важливим органом нашого організму, і будь-яка проблема, пов'язана з ним, може призвести до серйозних наслідків. Для вирішення цих проблем, пов'язаних із серцем, магнітно-резонансна томографія (МРТ) стала незамінним діагностичним інструментом, який забезпечує повне зображення серця та його оточуючих структур [2]. Удосконалення штучного

інтелекту дозволило аналізувати МРТ зображення, що допомагає виявити захворювання.

Переваги використання засобів штучного інтелекту (ШІ) для аналізу МРТ зображень [3]:

– швидша та точніша діагностика;

– покращення результатів для пацієнтів;

– підвищена ефективність та зменшення навантаження на медичних працівників;

– зменшення людських помилок.

Недоліки використання ШІ для аналізу МРТ зображень:

– вартість впровадження технології ШІ;

– обмежена інтерпретація результатів;

– залежність від якості даних.

ШІ є найкращим інструментом для аналізу МРТ зображень серця, оскільки вони швидко та точно виявляють захворювання [4]. Інтеграція ШІ в аналіз МРТ зображень є важливим кроком у боротьбі з хворобами серця (рисунок 1.2).

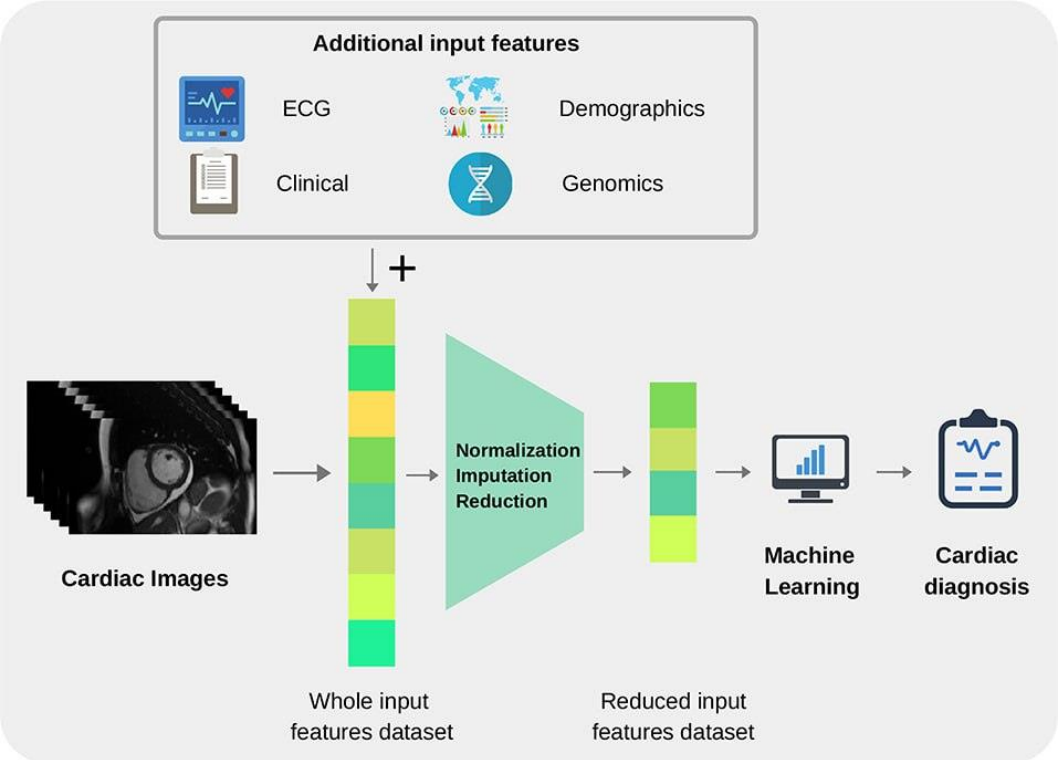


Рисунок 1.2 – отримання зображень серця за допомогою МРТ [5]

Для отримання МРТ зображень серця з використанням згорткових нейронних мереж (ЗНМ) використовуються наступні кроки [5]:

1. Збір медичних даних: здійснюється збір МРТ-зображень серця в форматі DICOM. Попередня обробка даних: перетворення зображень з формату DICOM в JPEG або PNG, використання методів фільтрації шумів. Розробка моделі ЗНМ: створення моделі для сегментації зображення серця з використанням згорткових шарів, шарів зведення і транспонованих згорткових шарів.

2. Навчання моделі: навчання моделі на наборі даних з позначеними областями серця за допомогою функції втрат, наприклад, перехресної ентропії, і оптимізатора, наприклад, Adam.

3. Валідація моделі: перевірка моделі на нових даних для оцінки її точності.

4. Використання моделі: використання моделі для сегментації зображення серця на нових зображеннях.

5. Оцінка результатів: оцінка результатів за допомогою метрик, таких як точність, чутливість, специфічність, F1-оцінка.

6. Приклади застосування: використання ЗНМ для розпізнавання патологічних станів серця і покращення діагностики та лікування пацієнтів.

У підсумку, хвороби серця є постійною проблемою, яка потребує інноваційних рішень. МРТ є життєво важливим діагностичним інструментом [6-8], і інтеграція технології штучного інтелекту в аналіз МРТ-зображень є важливим кроком у боротьбі з хворобами серця.

1.2 Аналіз методів, способів та технологій цифрової сегментації ділянок серця засобами штучного інтелекту

Сегментація серця – це процес цифрового поділу зображення серця на складові частини. Це критично важливий крок в аналізі зображень серця, оскільки дозволяє більш детально дослідити серце та його функціонування. В останні роки застосування ШІ здійснило революцію в галузі сегментації серця,

запропонувавши нові методи, засоби та технології для підвищення точності та ефективності. Мета цього звіту – надати огляд та аналіз методів, засобів і технологій, які зараз використовуються для цифрової сегментації ділянок серця за допомогою ШІ. Обсяг цього звіту охоплюватиме сучасний стан сегментації серця за допомогою ШІ, а також порівняння різних методів і технологій, а також обговорення проблем і обмежень.

У роботі розглянемо декілька програм для сегментації ділянок серця та проведемо їх аналіз. Існує багато програм для сегментації ділянок серця, відкритих і закритих. Нижче проаналізуємо найбільш популярне програмне забезпечення для сегментації ділянок серця.

Segment [9] – це відкрита програма для сегментації ділянок серця на зображеннях МРТ та КТ (рисунок 1.3). Програма має інтуїтивно зрозумілий інтерфейс користувача, що дозволяє користувачам з легкістю сегментувати ділянки серця.

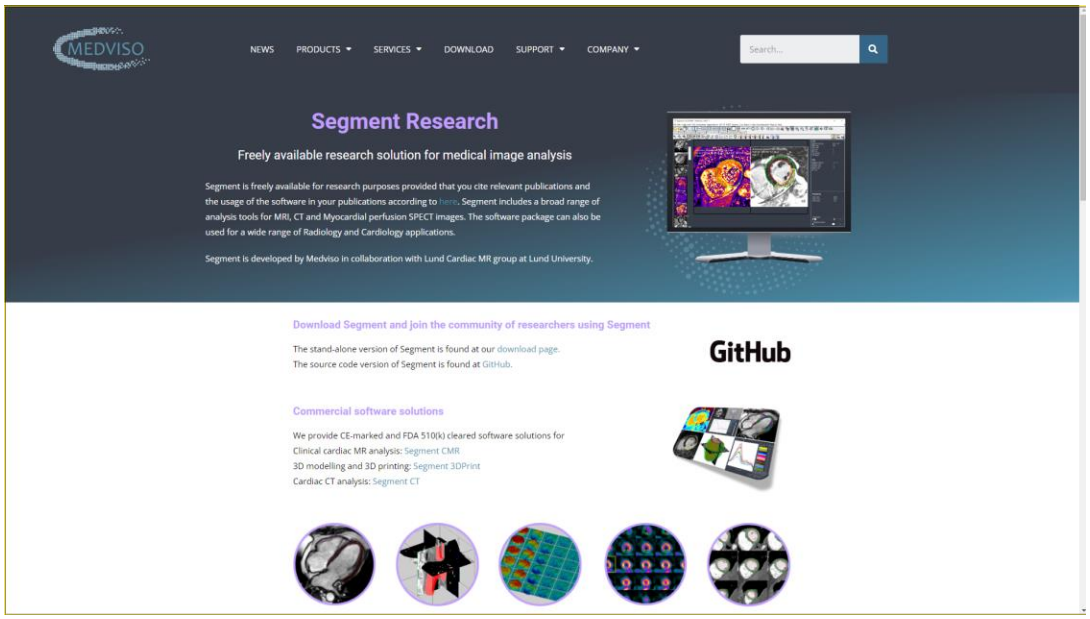


Рисунок 1.3 – Головна сторінка Segment [9]

ITK-SNAP [10] – це відкрита програма для сегментації ділянок серця на зображеннях МРТ, КТ та інших медичних зображеннях. Програма має потужну функціональність для сегментування різних ділянок серця, а також для відображення результатів у 3D (рисунок 1.4).

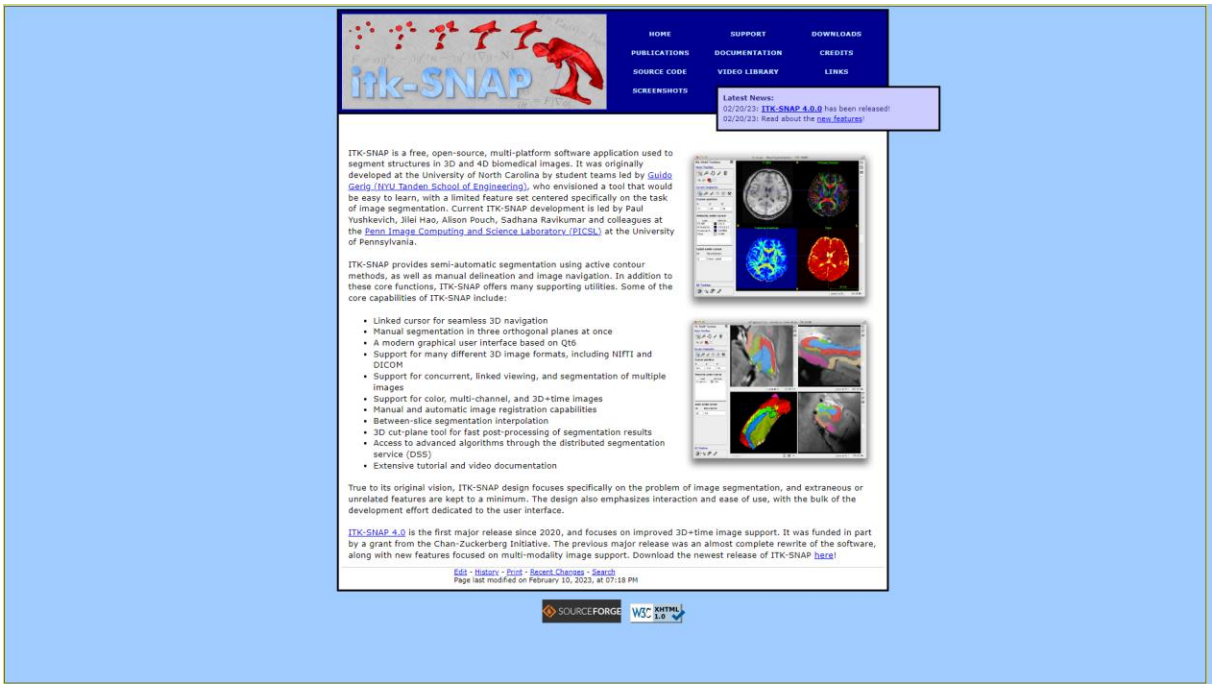


Рисунок 1.4 – Головна сторінка ITK-SNAP [10]

DeepSeg [11] – це програма для сегментації ділянок серця на зображеннях МРТ та КТ, яка використовує нейронні мережі для досягнення високої точності в сегментації (рисунок 1.5).

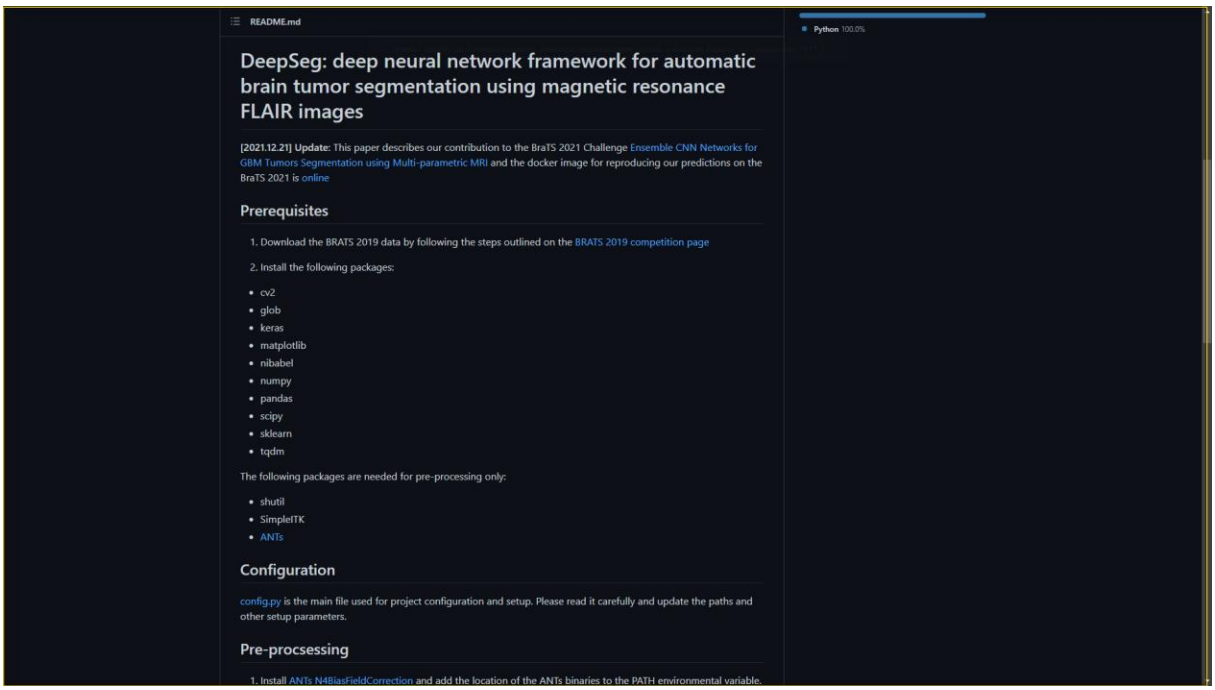


Рисунок 1.5 – Сторінка DeepSeg на GitHub [11]

Seg3D [12] – це безкоштовна програма з відкритим вихідним кодом, яка розробляється університетом Юти в США (рисунок 1.6).

Рисунок 1.6 – головна сторінка Seg3D [12]

В таблиці 1.1 наведемо порівняння вище перерахованих програм для сегментації серця. Проаналізуємо їх плюси та мінуси.

Таблиця 1.1 – Порівняння програм для сегментації

Програма	Плюси	Мінуси
Segment [9]	1:Висока точність сегментації. 2. Можливість використовувати різноманітні методи сегментації. 3. Інтуїтивний інтерфейс користувача. 4. Широкі можливості візуалізації та аналізу даних. 5. Підтримка різних форматів зображень.	1. Платна програма з обмеженою безкоштовною версією. 2. Вимоги до обладнання: програма працює повільно на слабкому обладнанні.
ITK– SNAP [10]	1. Безкоштовна програма з відкритим вихідним кодом.	1. Не завжди вдається отримати точну сегментацію.

	2. Підтримка різноманітних форматів зображень. 3. Можливість використовувати різні методи сегментації; 4. Інтуїтивний інтерфейс користувача.	2. Програма може працювати повільно на великих даних.
DeepSeg [11]	1. Використовує нейромережі для сегментації, що забезпечує високу точність. 2. Безкоштовна програма з відкритим вихідним кодом. 3. Можливість використовувати різноманітні нейромережі та методи сегментації. 4. Підтримка різноманітних форматів зображень.	1. Вимоги до обладнання: для роботи з нейромережами потрібен потужний комп'ютер з відеокартою. 2. Для роботи з програмою необхідні знання з машинного навчання.
Seg3D [12]	1. Безкоштовна програма з відкритим вихідним кодом. 2. Можливість візуалізувати та редагувати тривимірні медичні зображення. 3. Інтуїтивний інтерфейс користувача.	1. Обмежені можливості сегментації порівняно з іншими програмами. 2. Програма може працювати повільно на великих даних.

Кожна програма має свої переваги та недоліки залежно від вимог та задач користувача. Segment та ITK-SNAP надають широкі можливості для сегментації серця та візуалізації даних, DeepSeg забезпечує високу точність з використанням нейромереж, а Seg3D – редагування та візуалізацію медичних зображень, але має обмежені можливості з сегментації. Залежно від вимог користувача, можна обрати найкращу програму для сегментації серця. Сегментація серця використовує штучний інтелект, що покращує точність і ефективність сегментації. Існують різні методи сегментації серця, включаючи ручну, напівавтоматичну та повністю автоматизовану сегментацію [13]. Алгоритми штучного інтелекту, такі як глибоке навчання та алгоритми комп'ютерного зору, ефективно використовуються для

покращення точності сегментації серця [14, 15]. Існують різні інструменти та платформи для сегментації серця з використанням штучного інтелекту, включаючи комерційне програмне забезпечення та інструменти з відкритим кодом. Глибоке навчання є передовою технологією для сегментації серця. Реальні застосування штучного інтелекту в сегментації серця демонструють його ефективність, але все ще існують виклики та обмеження, такі як точність алгоритмів, потреба у великих навчальних даних, інтерпретація результатів та інтеграція в клінічну практику.

Застосування ШІ зробило революцію в області сегментації серця, запропонувавши нові методи, засоби та технології для підвищення точності та ефективності. Незважаючи на труднощі та обмеження, використання штучного інтелекту в сегментації серця має великі перспективи на майбутнє, і необхідні подальші дослідження, щоб продовжувати розвивати цю сферу.

Загалом, метод медичної сегментації ділянок серця на зображеннях МРТ є корисним інструментом для діагностики та планування лікування, проте він має свої обмеження. Цей метод може забирати багато І також можуть бути обмеження у здійсненні методу через необхідність обмеження руху пацієнта під час дослідження та вартістю.

1.3 Використання згорткових нейронних мереж для сегментації медичних зображень

Сегментація медичних зображень є вирішальним кроком у різних задачах аналізу медичних зображень. ЗНМ є багатообіцяючим підходом до сегментації медичних зображень, оскільки вони можуть автоматично вивчати ієрархічні представлення структур зображення. ЗНМ можна навчити на великому наборі даних анотованих зображень, щоб дізнатися про зв’язки між характеристиками зображення та цільовою сегментацією [16]. Після навчання мережу можна використовувати для сегментації нових зображень шляхом обробки зображення через мережу та створення маски сегментації. Успіх ЗНМ у сегментації медичних

зображень залежить від якості і розміру навчальних даних, дизайну архітектури мережі та методу оптимізації. Останні досягнення в глибокому навчанні призвели до розробки складніших мережевих архітектур і методів навчання, які покращили точність і ефективність ЗНМ для сегментації медичних зображень. Використання ЗНМ для сегментації ділянок серця на МРТ-зображеннях показало багатообіцяючі результати та має потенціал для революції в області медичної візуалізації.

Існує кілька популярних архітектур, які використовуються для сегментації медичних зображень [17]:

– U-Net [18] – це нейронна мережа для семантичної сегментації зображень, що має особливу архітектуру з енкодером та декодером, що дозволяє точно визначати межі об’єктів на зображенні.

– SegNet [19] – це архітектура нейронної мережі для сегментації зображень, яка використовує багатоенкодерний та багатодекодерний підхід зі спільними пікселями для підвищення швидкості та точності сегментації.

– DeepLab [20] – це архітектура нейронної мережі для семантичної сегментації зображень, яка використовує глибоку конволюційну мережу з декодером та модулем атенції для покращення точності та якості сегментації.

– U-Net++ [21] – це модифікація U-Net, що використовує структуру з підвищенням роздільної здатності зображення з кожним шаром та зворотним зв’язком між вкладеними структурами для підвищення точності та робастності сегментації.

В таблиці 1.2 наведемо всі переваги та недоліки перерахованих архітектур.

Це одні з найпоширеніших архітектур, але є багато інших з різними варіаціями та модифікаціями для вирішення конкретних проблем у сегментації медичних зображень.

U-Net та SegNet виявилися ефективними рішеннями для невеликих проблем сегментації медичних зображень і широко використовуються в цій галузі. Хоча інші архітектури також можуть підходити для невеликих наборів даних, U-Net і SegNet часто віддають перевагу через їх простоту та ефективність

обчислень. U-Net і SegNet – це дві популярні архітектури ЗНМ для сегментації медичних зображень.

Таблиця 1.2. – Порівняння архітектур

Архітектура	Плюси	Мінуси
U-Net [18]	1. Висока точність сегментації. 2. Ефективне використання глибокого навчання. 3. Можливість працювати з невеликою кількістю даних.	1. Мале розмірність зображень, що обробляються. 2. Неефективність в роботі з дуже великими даними.
SegNet [19]	1. Висока швидкість роботи. 2. Розширюваність до великих розмірів зображень. 3. Можливість роботи з відео.	1. Низька точність. 2. Висока чутливість до шуму. 3. Обмежена кількість слів.
DeepLab [20]	1. Висока точність. 2. Широкий вибір архітектур. 3. Робота з великими зображеннями.	1. Великі вимоги до обчислювальних ресурсів. 2. Високі витрати на тренування. 3. Вразливість до перенавчання.
U-Net++ [21]	1. Висока точність. 2. Менша кількість параметрів. 3. Адаптивна складність.	1. Складність реалізації. 2. Вимоги до обчислювальних ресурсів. 3. Вразливість до перенавчання.

Перевага U-Net над SegNet у задачах сегментації медичних зображень зумовлена насамперед його архітектурою, яка симетричним чином поєднує етапи кодування та декодування. На етапі кодування функції витягуються з вхідного зображення за допомогою згорткових шарів і шарів об’єднання. На етапі декодування ці функції підвищуються та об’єднуються з функціями на етапі кодування для створення остаточної карти сегментації. Ця архітектура дозволяє U-Net охоплювати як дрібні деталі [22], так і контекст високого рівня, що забезпечує точнішу сегментацію (рисунок 1.7).

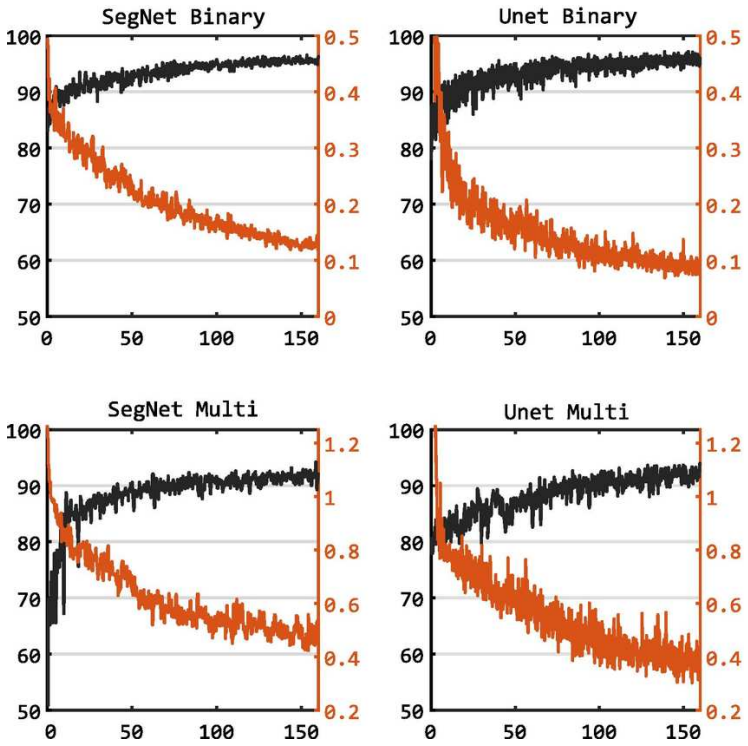


Рисунок 1.7 – Точність і втрата навчання бінарних і багатокласних сегментів SegNet і U-NET [22]

U-Net є гнучкою архітектурою з численними перевагами. Навчання наскрізне дозволяє оптимізувати мережу для конкретного завдання сегментації, підвищуючи продуктивність. Вона також може адаптуватися до різних даних і модифікуватися для включення додаткових функцій, таких як механізми уваги, що робить її вибором для трансферного навчання. U-Net також працює з різними методами медичної візуалізації, що його робить хорошим для мультимодальної сегментації. Його архітектура містить згорткові шари, зведення, транспоновані згорткові шари, конкатенацію, шари активації та відновлення [23], що дозволяє досягнути високої точності сегментації. Завдяки своїй простоті і повністю згортковому дизайну U-Net може швидко навчатися і ефективно обробляти зображення будь-якого розміру, особливо в медичних застосунках. У порівнянні з SegNet, U-Net має переваги у використанні для сегментації медичних зображень, завдяки здатності обробляти складні структури та простоті своєї архітектури.

1.4 Мета, завдання та вимоги до реалізації інформаційної системи

Метою кваліфікаційної роботи бакалавра є підвищення точності цифрової сегментації ділянок серця на зображеннях МРТ.

Для досягнення поставленої мети необхідно виконати наступні завдання:

1. Провести аналітичний огляд методів, способів та технологій цифрової сегментації за зображеннями МРТ та обрати найкращий.

2. Застосувати обраний спосіб цифрової сегментації до розв’язання задачі автоматизованої сегментації ділянок серця на зображеннях МРТ.

3. Реалізувати обраний спосіб у вигляді модуля програмного забезпечення для автоматизованої сегментації ділянок серця.

4. Провести експериментальне тестування реалізованого модуля за еталонними наборами даних.

Розділ 2 Спосіб сегментації ділянок серця на зображеннях МРТ

2.1 Спосіб виявлення області інтересу на зображеннях МРТ

МРТ зображення отримуються за допомогою магнітного поля та радіочастотних імпульсів. У процесі отримання МРТ зображення пацієнт знаходиться в тунелі МРТ апарату [13], де під дією магнітного поля відбувається вирівнювання ядер атомів водню у тканинах. Після цього застосовуються радіочастотні імпульси, що збурюють вирівняні ядра, після чого вони повертаються до своєї початкової позиції і при цьому випромінюють енергію у вигляді сигналу. Отриманий сигнал аналізується та перетворюється в цифрову форму за допомогою спеціальних приладів.

Отримане МРТ зображення можна подати у вигляді матриці чисел, де кожен елемент матриці відображає щільність сигналу в конкретній точці зображення (рисунок 2.1).

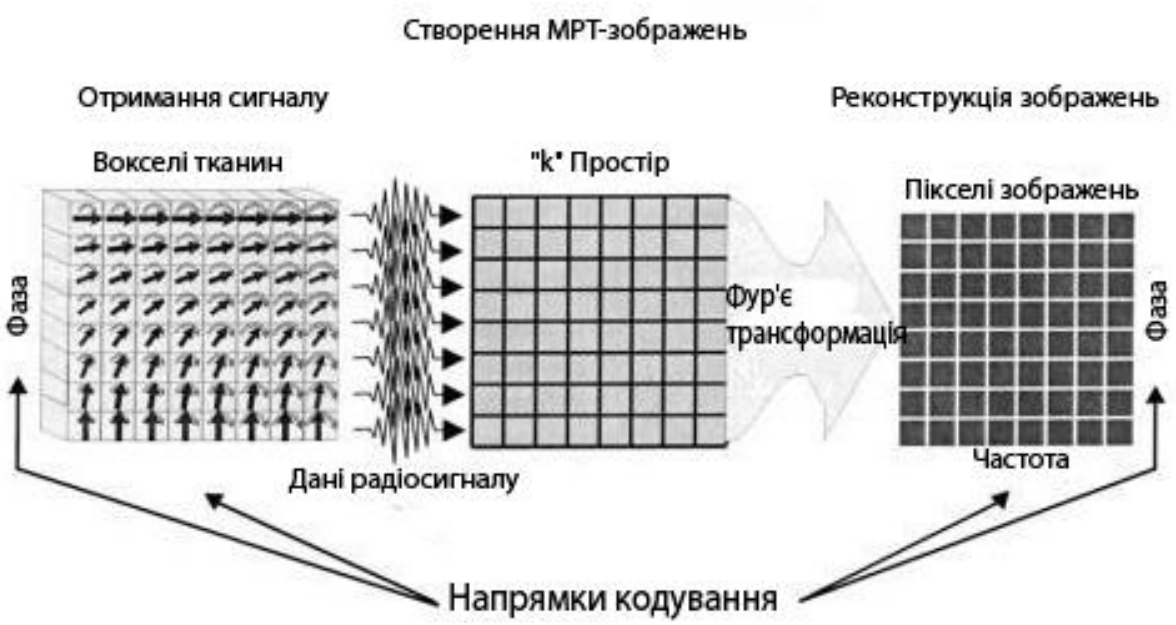


Рисунок 2.1 – Процес створення МРТ знімків [24]

Процес перетворення МРТ зображення в матрицю чисел відбувається за допомогою дискретизації. Сигнал, отриманий з МРТ, розбивається на дискретні значення, які відображають щільність сигналу в конкретних точках зображення.

Отримані значення зберігаються у вигляді матриці, де кожен елемент матриці відповідає певній точці на зображенні.

При роботі з МРТ зображеннями у вигляді матриці чисел, слід враховувати особливості їх обробки та аналізу. Оскільки МРТ зображення є тривимірними, то для їх подальшої обробки та аналізу необхідно здійснювати їх перетворення у двовимірні зображення. Також слід враховувати, що МРТ зображення містять велику кількість деталей та невеликі різниці у щільності сигналу між різними тканинами, що може ускладнювати їх аналіз та інтерпретацію.

Для перетворення МРТ зображення в матрицю чисел застосовуються різні математичні методи та способи [25]. Один з найбільш поширених способів відображення МРТ – це метод Фур'є-перетворення (рисунок 2.2).

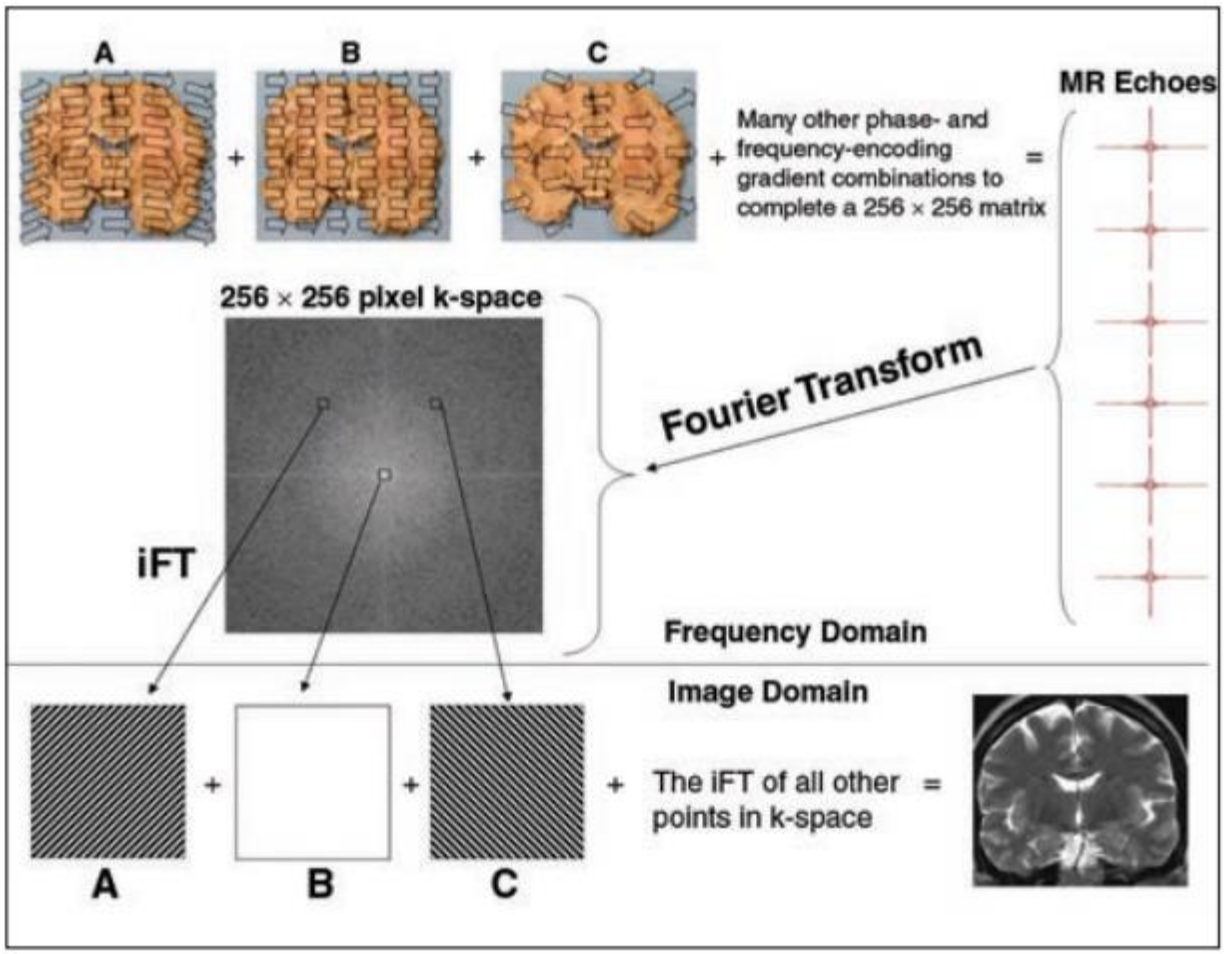


Рисунок 2.2 – Метод Фур'є [25]

У цьому методі спочатку МРТ сигнал розбивається на різні частоти за допомогою Фур’є-аналізу. Після цього виконується фільтрація частот і отримані значення записуються в матрицю. Цей метод забезпечує високу якість зображення та дозволяє отримати детальну інформацію про різні ділянки тіла.

Інший метод, який використовується для перетворення МРТ зображення в матрицю чисел – це метод оберненого радонового перетворення (рисунок 2.3).

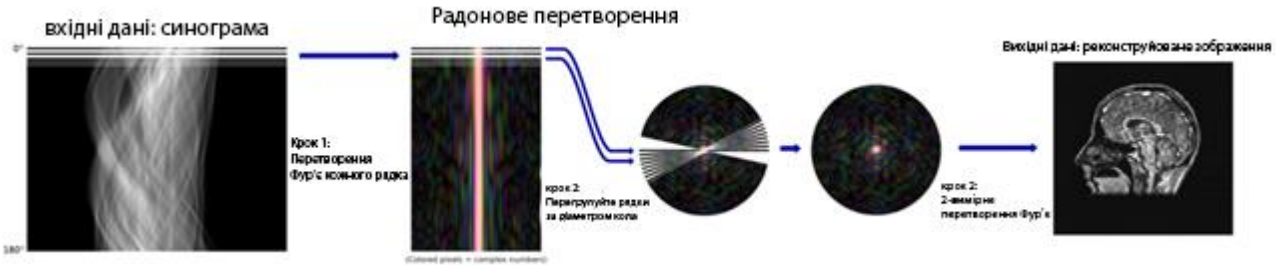


Рисунок 2.3 – Метод радонового перетворення [25]

У цьому методі МРТ зображення розбивається на багато променів, які проходять через об’єкт. Після цього виконується обернене радонове перетворення, яке дозволяє отримати значення щільності сигналу в різних точках зображення. Отримані значення записуються у вигляді матриці.

Також існують інші способи, які можуть використовуватися для перетворення МРТ зображення в матрицю чисел, такі як методи згортки та інтерполяції. Однак, способи подання та оброблення МРТ-зображень на основі методів Фур’є-перетворення та оберненого радонового перетворення є найбільш поширеними та ефективними для відображення МРТ-зображень в матриці чисел. Зображення МРТ можуть бути зісковані з різною якістю та з різними артефактами, що можуть впливати на точність та якість подальшого аналізу зображення.

Отже, з огляду на вищесказане, використовуваний спосіб сегментації ділянок серця на зображеннях МРТ подамо у вигляді схеми, що зображено на рисунку 2.4.

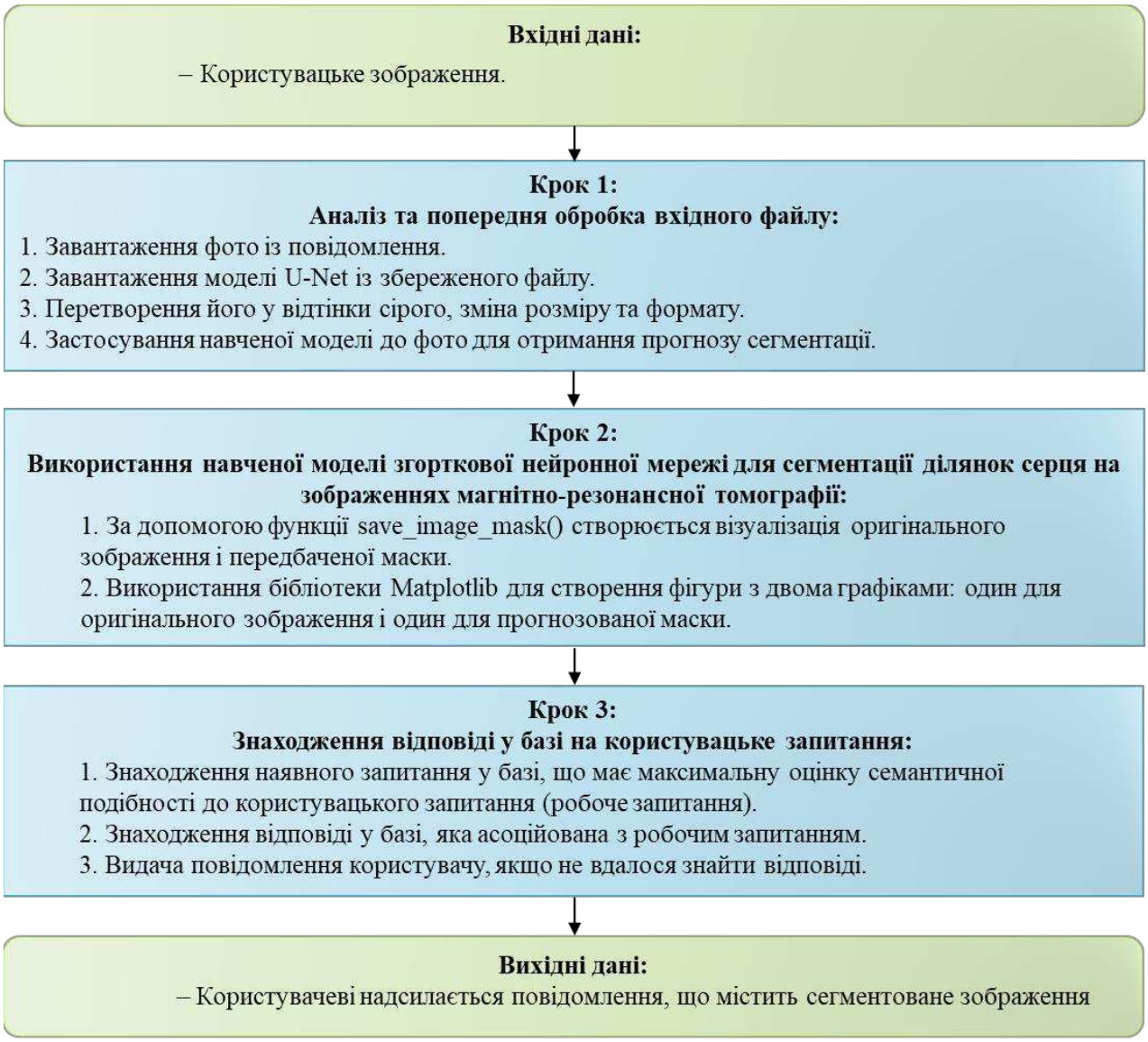


Рисунок 2.4 – Схема використаного способу сегментації ділянок серця на зображеннях МРТ

З рисунку 2.4, попереднє оброблення цифрових зображень МРТ є важливим етапом для подальшого аналізу та використання отриманих даних. В процесі отримання зображення МРТ виникає ряд проблем, які потребують корекції або виправлення.

– Затемнення: Неоднаковість інтенсивності може бути викликана різницею в поглинанні енергії залежно від місця знаходження тканини. Це призводить до затемнення на зображеннях МРТ, що ускладнює процес подальшої обробки та аналізу.

– Артефакти: Під час отримання зображення МРТ можуть виникати артефакти, такі як сигнальні інтенсивність, які не відповідають дійсному значенню, геометричні спотворення, які відбивають різні плани зрізів, або зміщення, які виникають внаслідок руху пацієнта або зміни положення зони відбору.

– Шум: Шум на зображеннях МРТ може бути викликаний різними факторами, такими як електронні коливання, рухи пацієнта або шум у системі відбору. Це може призвести до зниження якості зображення та вплинути на точність подальшої обробки даних.

Таким чином, попереднє оброблення цифрових зображень МРТ є важливим етапом для забезпечення якості даних та підвищення точності та ефективності подальшої обробки та аналізу даних.

Отже, попереднє оброблення МРТ зображення є необхідним етапом для підвищення точності та якості подальшого аналізу.

Основні методи попереднього оброблення МРТ зображень включають:

1. Накладання фільтрів: фільтрація є одним з найбільш поширених методів попередньої обробки зображень, які дозволяють зменшити шум та видалити артефакти з зображення. Різні фільтри можуть бути застосовані в залежності від характеру шуму на зображенні.

2. Виділення контурів: цей метод полягає у виділенні границь об’єктів на зображенні. Це може бути важливо для задачі сегментації об’єктів на зображенні та їх подальшого аналізу.

3. Нормалізація: цей метод дозволяє зменшити вплив різних умов на отримання зображень, таких як різниця в яскравості та контрастності. Нормалізація може зробити зображення більш однорідним та допомогти у подальшому аналізі.

Для задачі сегментації серця на зображеннях МРТ можуть бути застосовані різні методи попереднього оброблення, залежно від конкретної задачі та типу зображення. Наприклад, можуть бути застосовані методи фільтрації, щоб покращити контраст зображення та зменшити шум. Також можуть бути

використані методи видалення фону, щоб виділити тільки ті області зображення, які є серцем.

Окрім того, можуть бути застосовані методи активних контурів (snakes) або геометричного рівняння звукохвиль, щоб автоматично визначити контур серця та виділити його зображення. Такі методи можуть бути особливо корисними для сегментації серця на зображеннях МРТ з низькою якістю або з великою кількістю шуму.

Усі ці методи можуть бути поєднані та адаптовані до конкретної задачі сегментації серця на зображеннях МРТ. В результаті, попереднє оброблення зображення може значно покращити точність та швидкість сегментації, та забезпечити більш якісний та точний результат.

МРТ є надійним методом діагностики різних захворювань органів та тканин людини. Для ефективного аналізу зображень МРТ, необхідно виділити область інтересу, що відповідає області досліджуваного органу або тканини. Процес виділення області інтересу на зображенні МРТ може бути виконаний різними методами, такими як порогова обробка, кластеризація, активні контури та інші. У цьому розділі розглянемо основні методи виділення області інтересу на зображеннях МРТ, їх переваги та недоліки, а також можливі проблеми та виклики, пов’язані з цим процесом.

Існують такі методи виділення області інтересу на зображеннях МРТ.

Порогова обробка – це метод виділення області інтересу на зображеннях МРТ, який базується на встановленні певного порогового значення для яскравості пікселів зображення. Пікселі, які мають яскравість вище порогового значення, вважаються частинами області інтересу, тоді як пікселі з нижчою яскравістю виключаються з області інтересу. Цей метод є досить простим та швидким, але може бути неефективним в разі, коли зображення містить шум, що може призвести до помилкової включення або виключення деяких пікселів з області інтересу.

Кластеризація – це метод, який дозволяє групувати пікселі зображення в класи з високим степенем подібності. Цей метод дозволяє відокремлювати різні

області на зображенні МРТ та виділяти потрібні області інтересу. В процесі кластеризації пікселі зображення розбиваються на кластери, зазвичай з використанням алгоритмів, таких як k-середніх або ієрархічна кластеризація.

Алгоритм k-середніх полягає в тому, що спочатку обирається кількість кластерів, яка буде створена на зображенні, та початкові координати центрів кожного кластеру. Потім кожен піксель призначається до ближчого центру кластеру, і після цього обчислюються нові координати центрів кожного кластеру. Цей процес повторюється до тих пір, поки центри кластерів не стабілізуються.

Ієрархічна кластеризація – це інший метод кластеризації, який починається з того, що кожен піксель зображення вважається окремим кластером. Потім кластери об’єднуються в більші кластери на основі визначеної міри подібності між ними. Цей процес продовжується до тих пір, поки всі пікселі зображення не будуть об’єднані в один кластер.

Інші методи виокремлення області інтересу на зображеннях МРТ можуть включати в себе порогову обробку, яка використовує порогову значення для виділення пікселів, що відповідають області інтересу, або активні контури, які розміщуються вздовж межі області інтересу та змінюють свою форму, щоб точніше відображати границі цієї області.

Однією з основних проблем, пов’язаних з виділенням області інтересу на зображеннях МРТ, є висока залежність результатів від попереднього оброблення зображення. Якщо обробка виконана неадекватно або недостатньо, то може бути складно відрізнити область інтересу від навколишнього фону.

Іншою проблемою є високий рівень шуму на зображенні МРТ. Шум може виникнути під час зйомки, передачі або зберігання зображення. Це може призвести до неточностей під час виділення області інтересу, особливо якщо вона має складну форму або містить малі деталі.

Також, область інтересу на зображенні МРТ може бути розташована в декількох зонах зображення, що може ускладнити процес виділення. В такому випадку можуть використовуватися алгоритми, які враховують геометричні взаємозв’язки між різними частинами області інтересу. Крім того, вибір

оптимального методу виділення області інтересу може залежати від конкретних характеристик зображення МРТ, таких як його роздільна здатність, контрастність та розмір.

Таким чином, для успішного виділення області інтересу на зображеннях МРТ необхідно враховувати різні фактори, які можуть впливати на якість та точність результатів.

Трекінг серця є важливим етапом в обробленні зображень МРТ для багатьох діагностичних досліджень серцево-судинної системи. Цей процес дозволяє визначити рух серця на зображеннях МРТ та відслідковувати його зміни протягом часу. Процес трекінгу серця може бути реалізований за допомогою різних методів. Методи трекінгу серця – це алгоритми, які використовуються для визначення руху серця на зображеннях МРТ. Основна ідея полягає в тому, щоб відстежувати зміну положення серця на послідовних зображеннях та використовувати ці дані для покращення сегментації області інтересу.

Одним з методів трекінгу серця є оптичний потік. Цей метод базується на аналізі зміни інтенсивності пікселів на послідовних зображеннях. Він використовується для визначення вектора зміщення серця на кожному кадрі. Цей вектор може бути використаний для визначення руху серця та його місця на наступному кадрі.

Інший метод – це використання фільтра Калмана. Цей метод заснований на математичній моделі, яка використовується для прогнозування руху серця на наступному кадрі. Фільтр Калмана може враховувати шум та інші випадкові фактори, що можуть вплинути на точність прогнозування.

Інші методи трекінгу серця включають методи з використанням зміни форми та розміру серця на послідовних зображеннях, методи, що використовуються на основі взаємодії між сусідніми областями на зображенні, та методи, що використовуються на основі розпізнавання образів.

Отримані результати трекінгу серця можуть бути використані для покращення сегментації області інтересу, тобто для виокремлення точного контуру серця на зображенні. Для цього можна використовувати інформацію про

рух серця, яка була отримана в процесі трекінгу. Наприклад, відомо, що серце змінює свою форму положення з часом, тому можна використати цю інформацію для адаптивної сегментації серця на кожному кадрі зображення.

Одним з підходів є використання динамічної сегментації, яка базується на трекінгу серця і зміні його положення з часом. Для цього можуть бути використані методи, такі як «активні форми» або «активні контури», які дозволяють виокремлювати контур області інтересу на зображенні з урахуванням його динаміки та змін.

Інший підхід полягає в тому, щоб використовувати результати трекінгу для побудови моделі руху серця та його форми. З цією моделлю можна використовувати методи математичної обробки сигналів та комп'ютерного зору для отримання більш точної сегментації серця на зображенні МРТ.

Також можна використовувати інформацію про рух серця для підвищення точності сегментації за допомогою методу «регістрації зображень». Цей метод полягає в тому, щоб зіставити два зображення з різних часових точок з урахуванням їх зміщення та відносної ротації. Це дозволяє виокремити області, що містять серце, та використовувати цю інформацію для більш точної сегментації на кожному кадрі зображення.

Інші методи трекінгу серця включають:

Байєсовський підхід: Використовуємо статистичні моделі, щоб знайти найбільш вірогідну траєкторію серця, використовуючи при цьому інформацію з попередніх фреймів. Цей метод дуже ефективний, коли потрібно знайти нестабільний шлях руху серця.

Метод на основі функції розподілу відстані: Використовуємо функцію розподілу відстані, яка враховує розмір і форму серця. Під час трекінгу, ця функція використовується для пошуку найбільш вірогідної траєкторії серця. Цей метод дозволяє отримати точні результати трекінгу навіть при зміні форми серця.

Метод, заснований на кінематиці: Використовуємо знання про кінематику серця для визначення його траєкторії на зображеннях МРТ. Цей метод дозволяє точно визначати рух серця навіть при наявності збурень в руху.

Кожен з цих методів має свої переваги та недоліки, і вибір конкретного методу залежить від конкретної задачі. Наприклад, фільтр Калмана може бути ефективним, якщо швидкість руху серця незначна, а метод на основі функції розподілу відстані може бути корисним для розпізнавання областей, які мають незвичайні форми.

Результати трекінгу серця можуть бути використані для покращення сегментації області інтересу. За допомогою отриманих даних про рух серця можна відслідковувати зміни розміру та форми серця на зображеннях МРТ протягом часу. Це дозволяє покращити точність сегментації серця та отримати більш детальну інформацію про його стан та функцію. Проте, методи трекінгу серця можуть стикатися з різними проблемами та викликами. Наприклад, рух серця може бути досить складним та непередбачуваним, що може ускладнити процес трекінгу. Також, залежно від методу трекінгу, можуть виникати проблеми з точністю та швидкістю визначенні руху серця на зображеннях МРТ.

Крім того, важливо враховувати якість вихідних зображень МРТ, адже недостатня роздільна здатність, наявність артефактів чи шуму можуть вплинути на точність та надійність отриманих результатів трекінгу.

Також, важливо розглянути питання безпеки та етики використання методів трекінгу серця на зображеннях МРТ. Оскільки зображення МРТ містять конфіденційну медичну інформацію про пацієнта, необхідно дотримуватися відповідних правил щодо захисту цієї інформації.

У цілому, методи трекінгу серця є важливим інструментом для отримання більш детальної інформації про стан та функцію серця на зображеннях МРТ. Проте, необхідно дотримуватися відповідних принципів безпеки та етики використання цих методів для медичних досліджень та практики.

2.2 Навчання згорткової нейронної мережі

Навчання ЗНМ для сегментації ділянок серця на зображеннях МРТ може бути реалізоване в кілька етапів. Описуючи ці етапи, ми розглянемо теоретичний

підхід до навчання такої мережі, який може бути застосований під час створення програми [26].

1. Першим етапом є збір та підготовка даних для навчання. Для цієї задачі необхідно мати набір зображень серця, отриманих з МРТ, а також відповідні маски, які показують ділянки серця на кожному зображенні. Маски можуть бути створені за допомогою експертних оцінок медичних фахівців.

2. Другий етап – побудова моделі ЗНМ. Вона складається зі згорткових шарів, пулінг-шарів та повнозв'язних шарів. Згорткові шари використовуються для виявлення візуальних ознак на зображенні, пулінг-шари зменшують розмір зображення та забезпечують інваріантність до малих зміщень, а повнозв'язані шари використовуються для класифікації.

3. Третій етап – навчання моделі. На цьому етапі зображення та відповідні маски розбиваються на тренувальний та тестовий набори. Тренувальний набір використовується для навчання моделі, тоді як тестовий набір використовується для перевірки точності моделі на нових зображеннях. Під час навчання мережі використовується функція втрати, яка вимірює відстань між вихідними масками та масками, що створені моделлю. Наприклад, можна використовувати функцію перехресної ентропії для бінарної класифікації пікселів на ділянки серця та не-серця.

4. Четвертий етап – оцінка точності та підгонка параметрів. Після завершення процесу навчання моделі необхідно оцінити її точність. Це можна зробити за допомогою метрик, таких як точність, відновлення та середня оцінка відхилень (mean absolute error). Якщо точність недостатня, можливо потрібно переглянути та підібрати параметри моделі та повторити навчання. Наприклад, можна спробувати змінити кількість шарів, кількість нейронів у кожному шарі, швидкість навчання та інші параметри.

5. П'ятий етап – застосування моделі до нових зображень. Після навчання та підгонки параметрів модель можна використовувати для сегментації ділянок серця на нових зображеннях, отриманих з МРТ. Після застосування моделі до нових зображень можна оцінити її точність та відносну помилку.

Для реалізації процесу навчання ЗНМ для сегментації ділянок серця на зображеннях МРТ, було обрано мову програмування Python. Python є однією з найпопулярніших мов програмування в області машинного навчання та інтелектуального аналізу даних, завдяки своїй легкості використання, великій кількості наявних бібліотек та фреймворків, таких як TensorFlow, Keras, PyTorch, які значно полегшують розробку та реалізацію нейронних мереж.

Python також має багато інструментів для обробки та аналізу даних, що дозволяє легко зчитувати дані з файлів зображень та проводити їх попередню обробку перед подачею на вхід нейронній мережі. Завдяки цим можливостям Python було обрано для реалізації процесу навчання ЗНМ для сегментації ділянок серця на зображеннях МРТ.

2.3 Проєктування архітектури згорткових нейронних мереж для сегментації ділянок серця

Сегментація ділянок серця на зображеннях МРТ є важливою, але складною задачею в медицині. Для цього використовують ЗНМ, серед яких U-Net є популярним вибором для семантичної сегментації зображень. Розроблена у 2015 році, U-Net має унікальну структуру, яка ефективно вирішує проблеми сегментації зображень, зокрема медичних даних. Вона використовує ЗНМ та проміжні зв’язки, які інтегрують інформацію з різних масштабів зображень. Архітектура U-Net складається з двох основних частин: енкодера та декодера. Енкодер має структуру зворотного зв’язку і складається з кількох блоків згортки та зменшення розміру зображення (пулінгу), що допомагає отримати високорівневі ознаки зображення. Декодер має транспоновані згорткові шари, які збільшують розмір зображення і зменшують кількість каналів. Проміжні ланки в декодері передають інформацію від кодера. Декодер також має структуру зворотного зв’язку і складається з кількох блоків розширення розміру зображення (узгодження), що допомагає відновити точність сегментації. Загальна структура U-Net може подана на рисунку 2.5.

На цьому зображенні зображено мережу з кодером і декодером. Кодер складається з шарів згортки та об’єднання, тоді як декодер має транспоновані шари згортки та «проміжні ланки», які з’єднують відповідні шари з кодера та декодера. Всі блоки мають однакову кількість згорткових шарів з фільтрами 3x3 і функцією активації ReLU після кожного з них. Об’єднувальний шар накладається після кожного згорткового шару, окрім останнього блоку, де використовується звичайний згортковий шар з розміром фільтра 3x3. Це дозволяє зберегти розмір зображення. Декодер збільшує розмір зображення за допомогою транспонованих згорткових шарів і використовує «проміжні ланки» для з’єднання з кодером.

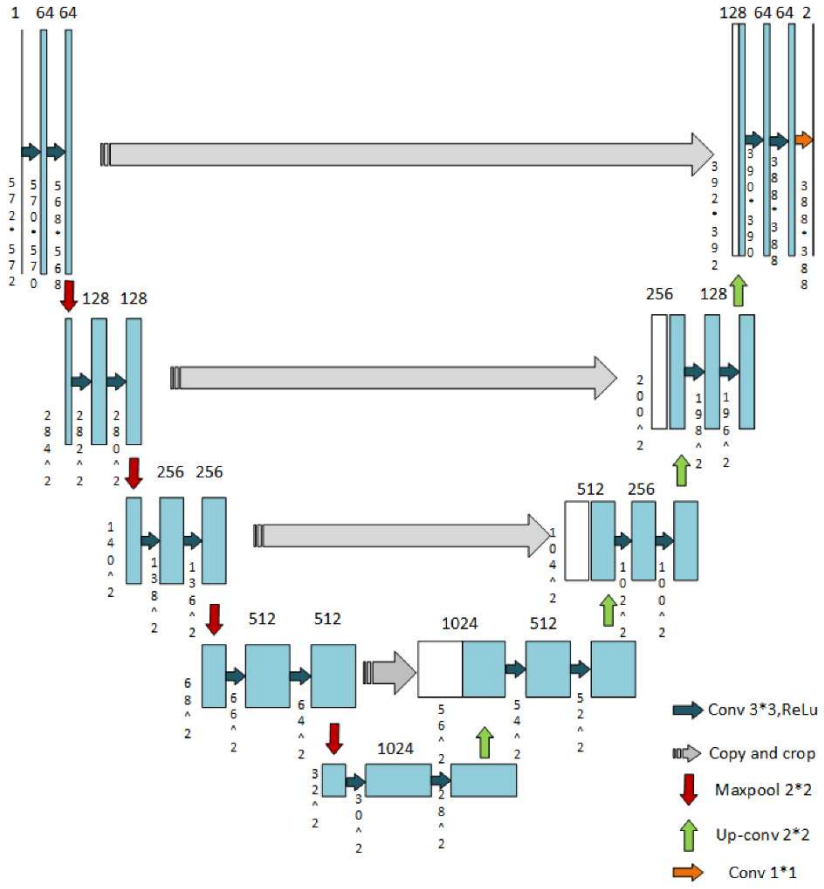


Рисунок 2.5 – Загальна структура U-Net [27]

Кожен блок у декодері має транспонований згортковий шар з фільтрами 3x3 і «проміжну ланку», яка поєднує зображення кодера і декодера. Функція активації ReLU застосовується після кожного транспонованого згорткового шару.

«Проміжні ланки» дозволяють мережі використовувати інформацію з нижчих рівнів абстракції, покращуючи точність.

U-Net – це архітектура нейронної мережі, яка зазвичай використовується для задач семантичної сегментації. Вона має декодер, який включає шар згортки 1x1, що зменшує кількість каналів до кількості класів. В результаті отримуємо зображення того ж формату і розміру, що й оригінал, але з кількістю каналів, що дорівнює кількості класів. Одним з ключових аспектів U-Net є використання пропускних з'єднань для збереження більшої кількості деталей зображення під час відтворення (рисунок 2.6).

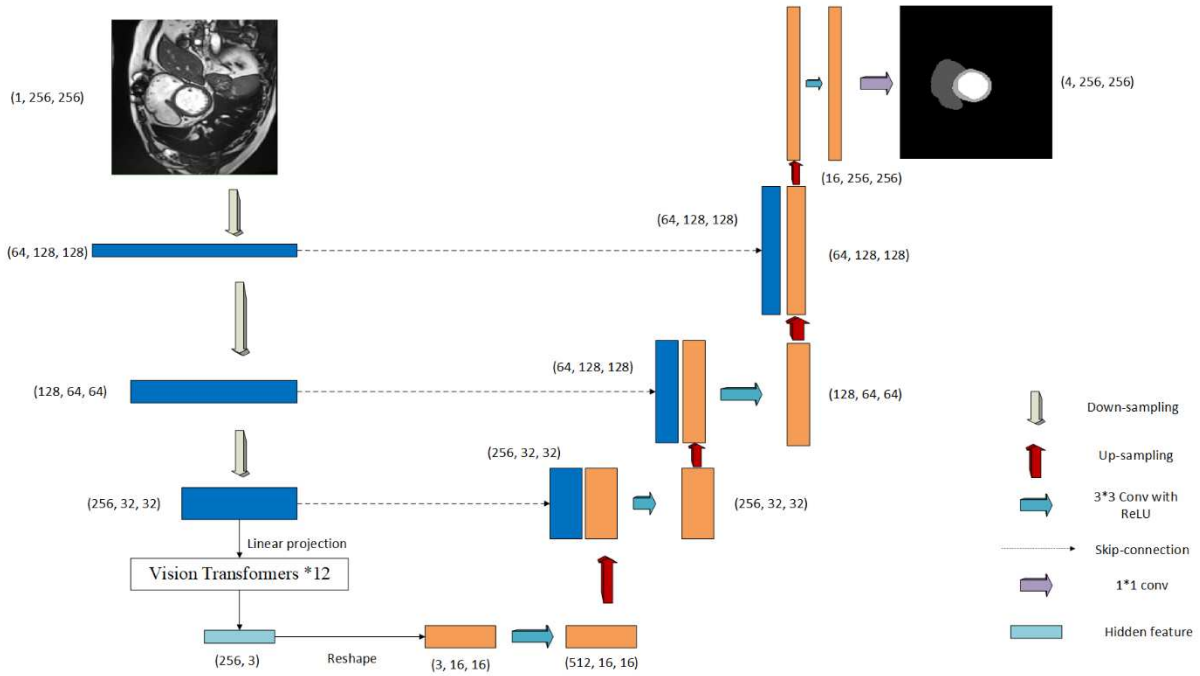


Рисунок 2.6 – Skip-connections в архітектурі U-Net [27]

Отже, основні складові архітектури U-Net включають:

– енкодер з кількома блоками згортки та пулінгу;
– декодер з кількома блоками розширення та узгодження;
– підключення ознак з енкодера до відповідного рівня декодера;
– функція активації ReLU;
– функція втрати Dice.

Детальніше про кожну з цих складових нижче:

Енкодер – це початковий шар U-Net, який виконує функцію зведення зображення до більш високорівневих ознак. У цьому блоку використовуються згорткові шари, які допомагають виявити візуальні ознаки зображення. Після кожного згорткового шару використовується шар пулінгу для зменшення розміру зображення та збільшення швидкості обчислень. Пулінг може використовувати різні методи, такі як максимальний або середній пулінг, залежно від вимог до точності та швидкості обчислень.

Декодер – це блок U-Net, який допомагає відновити розмір зображення до його початкового розміру та точно сегментувати об’єкти на зображенні. Декодер складається з кількох блоків розширення та узгодження. Розширюючий блок використовує транспоновану згортку для збільшення розміру зображення та відновлення деталей, які були втрачені в енкодері. Узгоджуючий блок використовує конкатенацію вхідного зображення та зображення з попереднього блоку для збільшення точності сегментації та збереження деталей, що допомагає уникнути втрати інформації в процесі сегментації.

Підключення ознак з енкодера до декодера допомагає зберегти деталі зображення та поліпшити точність сегментації. Для цього використовуються пропускні зв’язки, що забезпечують зв’язок між відповідними рівнями енкодера та декодера. Кожен блок декодера містить пропускний зв’язок з відповідним блоком енкодера, що допомагає зберегти ознаки, які були виявлені в енкодері, та додатково використовувати їх для підвищення точності сегментації.

Функція втрати Dice – це міра відстані між результатами сегментації та справжніми значеннями на зображенні. У випадку U-Net використовується функція втрати Dice, яка є мірою схожості між результатами сегментації та справжніми значеннями. Функція втрати Dice є ефективним способом навчання U-Net для досягнення високої точності сегментації.

U-Net – це глибока нейронна мережа, призначена для семантичної сегментації зображень. Для підвищення точності сегментації U-Net використовує методи пакетної нормалізації, відсіювання та доповнення даних. Пакетна нормалізація стандартизує вихідні дані для стабільного і швидкого навчання.

Відсіювання дозволяє уникнути надмірної адаптації та підвищити точність мережі. Доповнення даних генерує нові зображення з різними перетвореннями, такими як зміна розміру, обертання, регулювання яскравості та контрастності, які допомагають збільшити обсяг навчальних даних і підвищити точність сегментації. U-Net є однією з найпопулярніших архітектур для семантичної сегментації зображень, особливо з обмеженими навчальними даними, досягаючи високої точності та ефективності.

2.4 Функціональна структура інформаційної системи на основі згорткової нейронної мережі

2.4.1 Функціональна структура та функціональні процеси системи

При розробці автоматизованої системи Сегментації ділянок серця на зображеннях МРТ з використанням ЗНМ необхідно автоматизувати наступні функціональні процеси:

– функціональний процес «робота з введенням зображення»;
– функціональний процес «робота з підготовкою зображення»;
– функціональний процес «робота з вибором області серця»;
– функціональний процес «робота з сегментацією серця»;
– функціональний процес «робота з перевіркою результатів»;
– функціональний процес «робота з виводом результатів»;
– функціональний процес « робота зі збереженням результатів»;
– функціональний процес «робота з введенням зображення».

Цей процес відповідає за введення зображення МРТ в програму. Його можливості включають:

– отримання вхідних даних від користувача;
– перевірка формату та якості зображення;
– завантаження зображення в програму;
– відображення попереднього перегляду зображення;

– повідомлення користувача про успішне завантаження зображення або про проблеми з введенням.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.7.

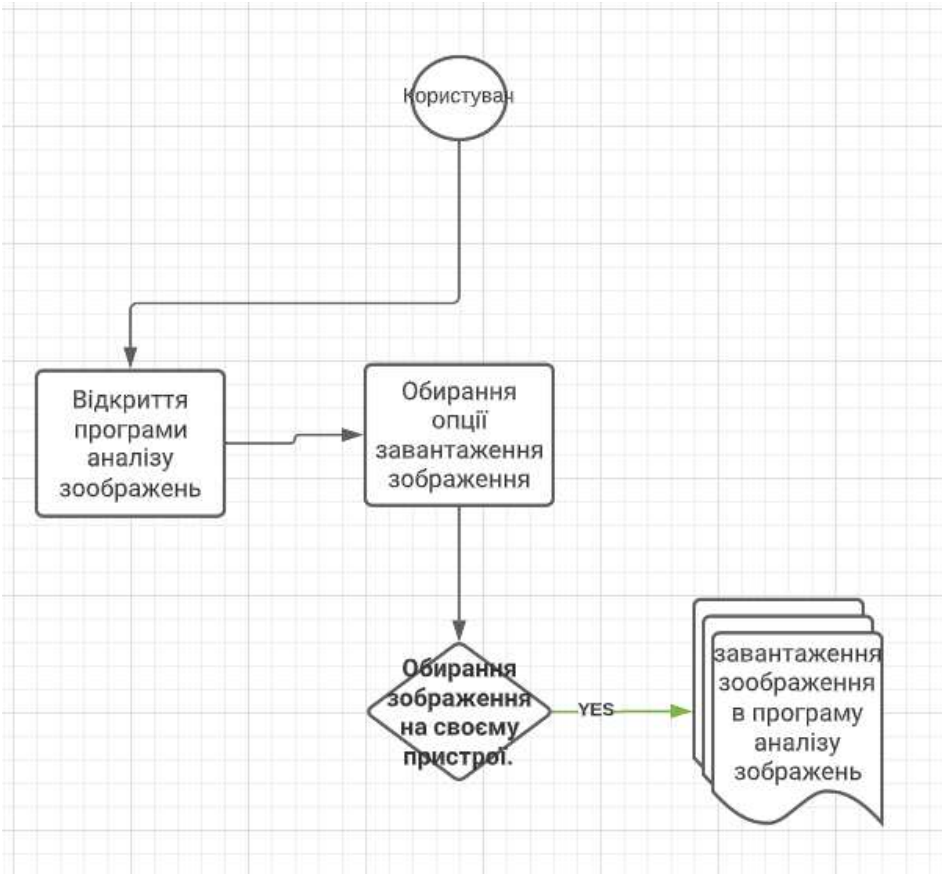


Рисунок 2.7 – Діаграма дій функціонального процесу «Робота з введенням зображення»

Функціональний процес «Робота з підготовкою зображення»:

– цей процес відповідає за підготовку зображення до подальшої обробки. його можливості включають;

– виконання фільтрації для зменшення шуму та покращення якості зображення;

– конвертація зображення до необхідного формату для подальшої обробки;

– визначення розміру зображення та його масштабування до необхідного розміру;

– відображення попереднього перегляду підготовленого зображення;

– повідомлення користувача про успішну підготовку зображення або про проблеми з обробкою.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.8.

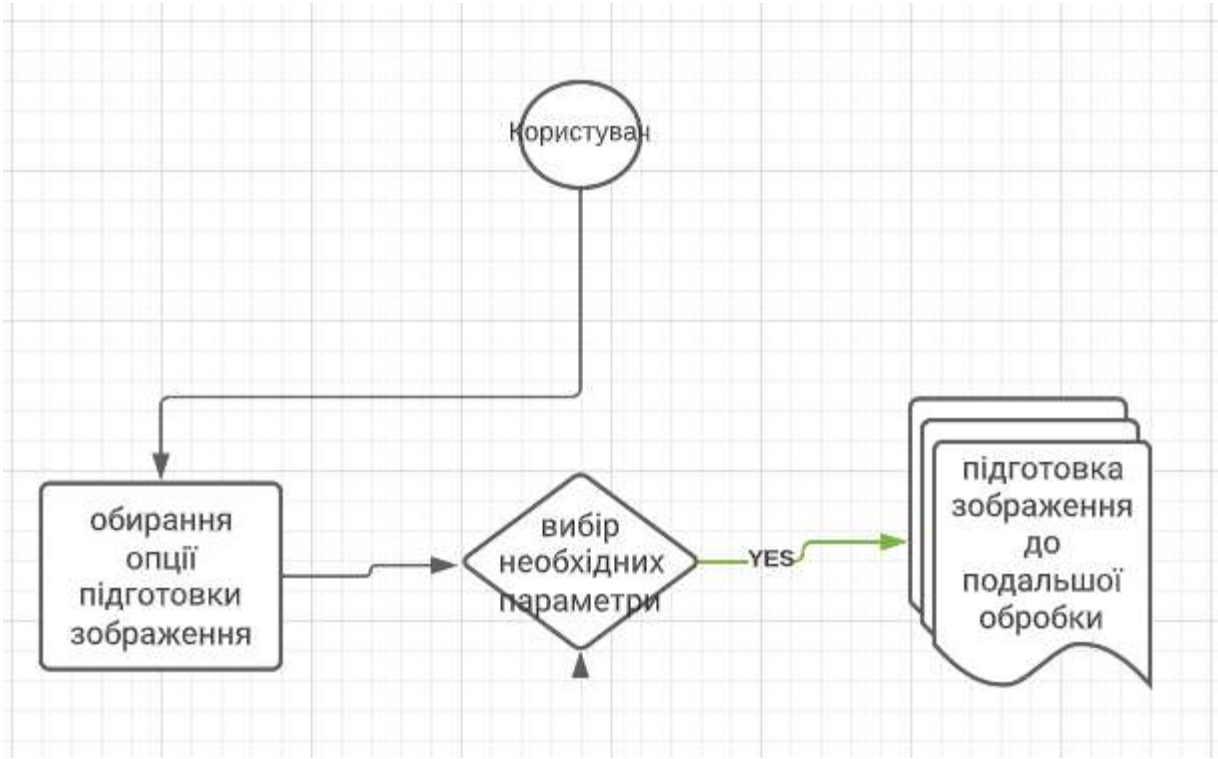


Рисунок 2.8 – Діаграма дій функціонального процесу «Робота з підготовкою зображення»

Функціональний процес «Робота з вибором області серця». Цей процес відповідає за вибір області серця на зображенні, яка буде піддаватися сегментації. його можливості включають:

– навігація по зображенню для вибору області серця;

– ручне виділення області серця або використання автоматичного алгоритму виділення області серця;

– відображення попереднього перегляду виділеної області серця;

– повідомлення користувача про успішний вибір області серця або про проблеми з вибором.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.9.

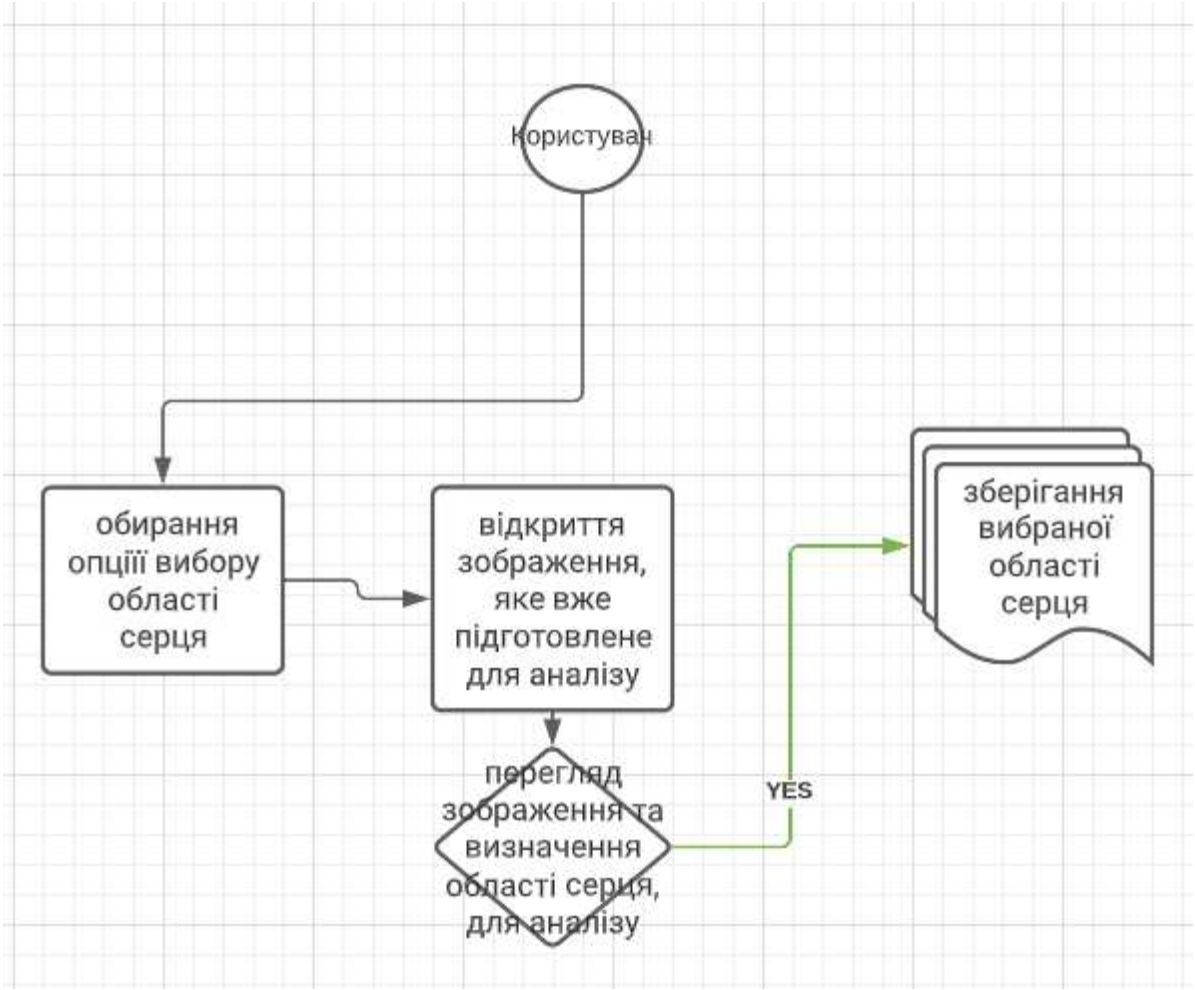


Рисунок 2.9 – Діаграма дій функціонального процесу «Робота з вибором області серця»

Функціональний процес «Робота з сегментацією серця». Цей процес відповідає за розділення зображення на окремі сегменти, які відповідають різним структурам серця, таким як міокард, перикард та ендокард. Його можливості включають:

– використання алгоритмів обробки зображень для сегментації серця;

– відображення попереднього перегляду результатів сегментації;

– можливість редагування результатів сегментації, якщо це необхідно;

– повідомлення користувача про успішну сегментацію або про проблеми з нею.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.10.

Рисунок 2.10 – Діаграма дій функціонального процесу «Робота з сегментацією серця»

Функціональний процес «Робота з перевіркою результатів». Цей процес відповідає за перевірку правильності результатів сегментації серця та визначення їх придатності для дальшого використання. Його можливості включають:

– автоматичну перевірку результатів сегментації за допомогою метрик якості, таких як чутливість та специфічність;

– можливість редагування результатів, якщо це необхідно;

– відображення попереднього перегляду результатів після редагування;

– повідомлення користувача про успішну перевірку результатів або про проблеми з ними.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.11.

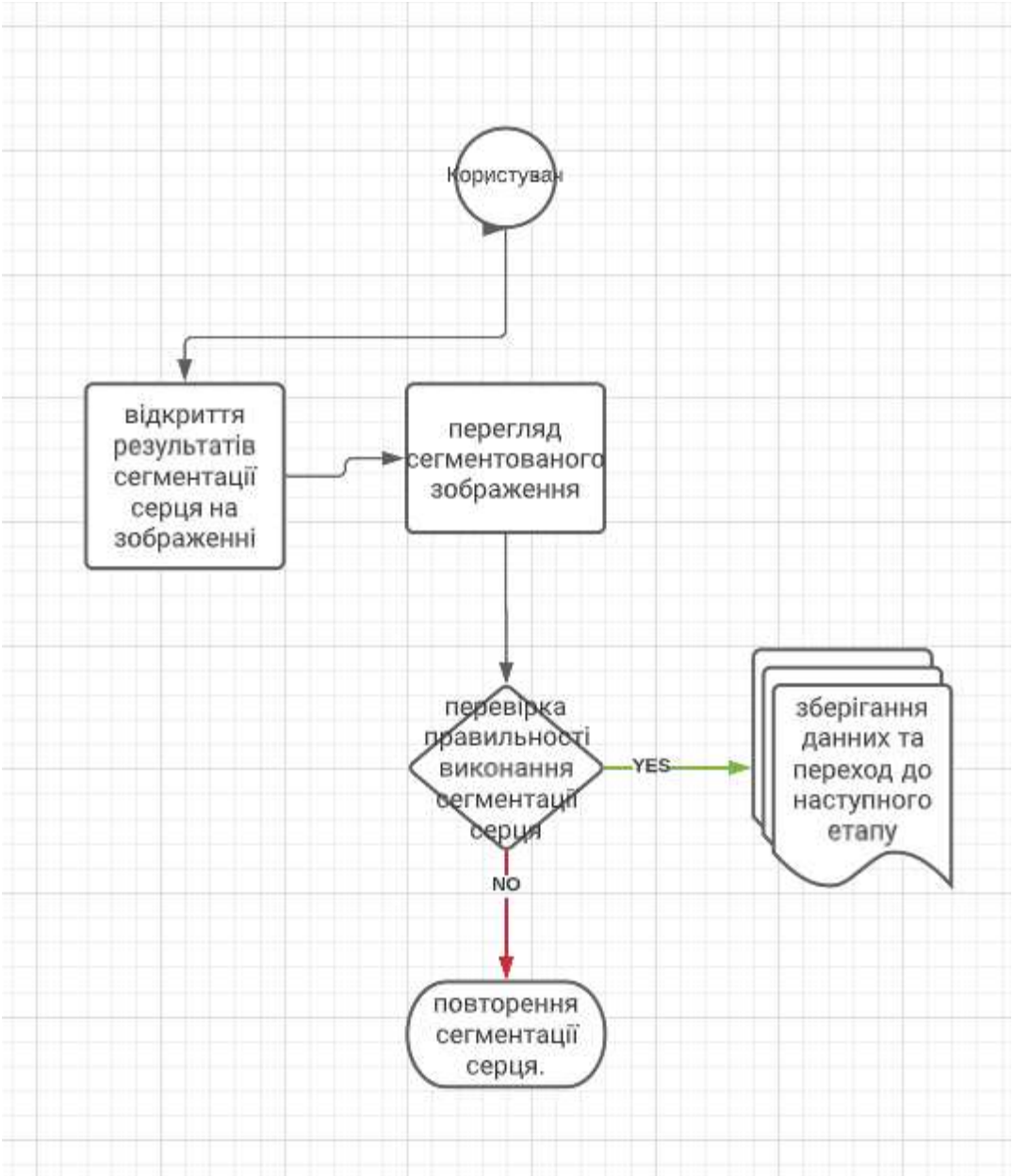


Рисунок 2.11 – Діаграма дій функціонального процесу «Робота з перевіркою результатів»

Функціональний процес «Робота з виводом результатів». Цей процес відповідає за відображення результатів сегментації серця для користувача. Його можливості включають:

– відображення зображень, які показують окремі сегменти серця;

– навігацію по різним зображенням та різним сегментам серця;

– відображення статистики про результати сегментації серця, наприклад, розмір кожного сегменту;

– можливість зберігати результати для подальшого використання або дослідження.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.12.

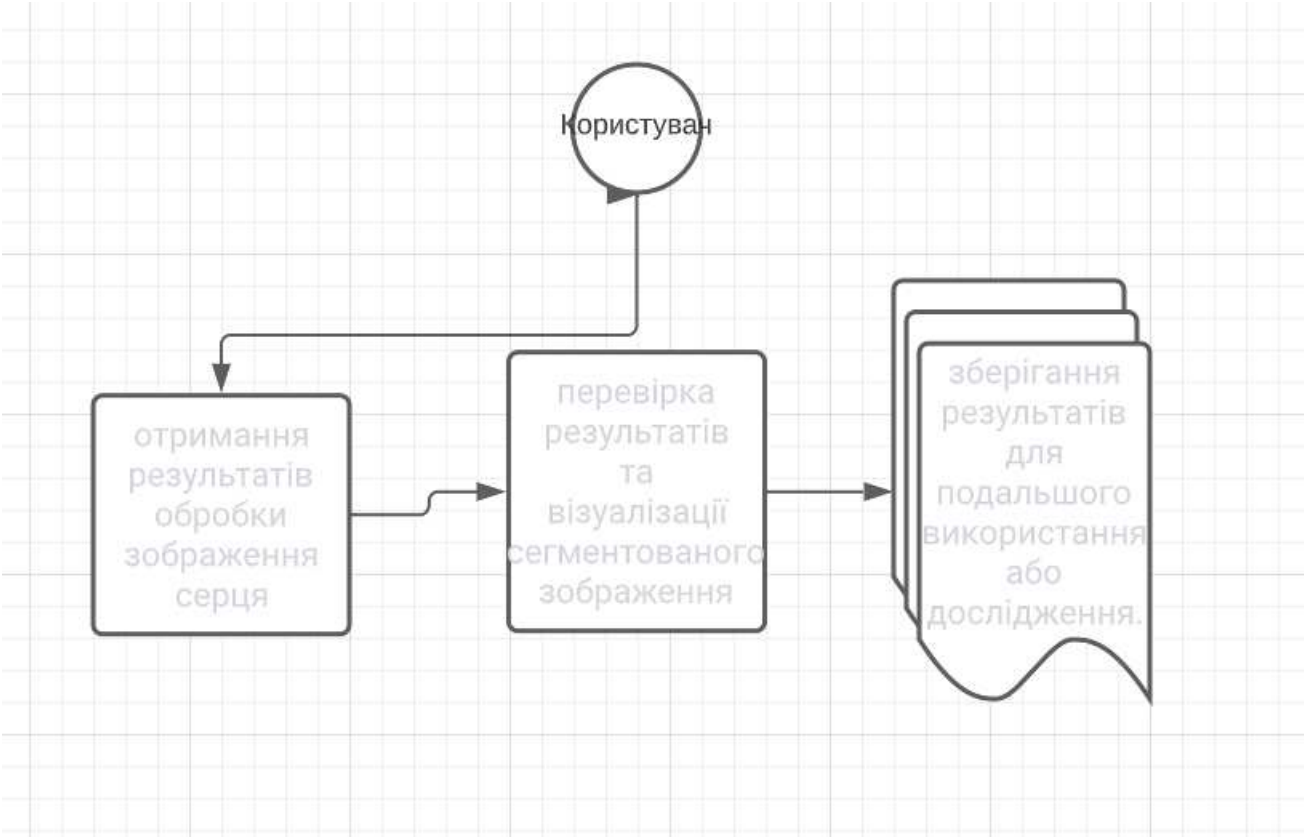


Рисунок 2.12 – Діаграма дій функціонального процесу «Робота з виводом результатів»

Функціональний процес «Робота зі збереженням результатів». Цей процес відповідає за збереження результатів сегментації серця на зображенні. Його можливості включають:

– вибір формату для збереження результатів (наприклад, DICOM, JPEG, PNG тощо);

– збереження результатів у вибраному форматі;

– визначення назви та розташування файлу з результатами;

– запис інформації про результати в базу даних, якщо така є;

– відображення повідомлення про успішне збереження результатів або про проблеми зі збереженням.

Діаграму дій для даного функціонального процесу зображено на рисунку 2.13.

Рисунок 2.12– Діаграма дій функціонального процесу «Робота зі збереженням результатів»

Цей процес дозволяє зберігати результати для подальшого використання або дослідження, що є важливою можливістю для медичної діагностики та досліджень.

2.5 Висновки до розділу 2

У даному дослідженні був розглянутий спосіб виявлення області інтересу на зображеннях магнітно-резонансної томографії (МРТ) з використанням згорткових нейронних мереж. Метою цього дослідження було розробити ефективний метод сегментації ділянок серця на зображеннях МРТ, який може бути корисним у медичній діагностиці та дослідженнях серцево-судинних захворювань.

На початку розділу 2.2 розглянута техніка навчання згорткової нейронної мережі. Навчання включає в себе процес підготовки даних, вибір архітектури мережі. В даному дослідженні використовувалась згорткова нейронна мережа Unet, яка добре показала себе у завданнях обробки зображень.

У розділі 2.3 було проведено проєктування архітектури згорткової нейронної мережі для сегментації ділянок серця на зображеннях МРТ. При проєктуванні були враховані особливості структури серця та його зображень МРТ. Були розглянуті різні шари мережі, такі як згорткові шари, пулінгові шари та повнозв’язані шари, щоб досягти оптимальної точності сегментації.

У розділі 2.4 була описана функціональна структура інформаційної системи, яка базується на згортковій нейронній мережі. Ця система включає в себе різні функціональні процеси, які допомагають обробляти та аналізувати зображення МРТ.

Результати дослідження показали, що використання згорткових нейронних мереж разом з функціональною структурою інформаційної системи дозволяє досягти високої точності та ефективності у виявленні області інтересу на зображеннях МРТ. Цей підхід може бути використаний у клінічній практиці для допомоги лікарям у діагностиці та стеженні за хворими з серцево-судинними захворюваннями. Крім того, він може знайти застосування у дослідницьких проектах, спрямованих на вивчення різних аспектів функціонування серця та його патологій.

Розділ 3 Програмна реалізація інформаційної системи з використанням обраного способу сегментації ділянок серця

3.1 Структура та функціональне призначення програмних складових інформаційної системи

Було розроблено інформаційну систему для сегментації ділянок серця на зображеннях за допомогою бота в Telegram. Система складається з двох основних частин. Перший модуль використовується для навчання моделі ЗНМ з використанням набору даних. Другий модуль є Telegram-ботом, який надає користувачам інтерфейс для доступу до функції сегментації ділянок серця. Користувач завантажує зображення через бота, а навчена модель розпізнає та сегментує ділянки серця на цьому зображенні. Результат сегментації передається назад користувачу через Telegram-бот (рисунок 3.1).

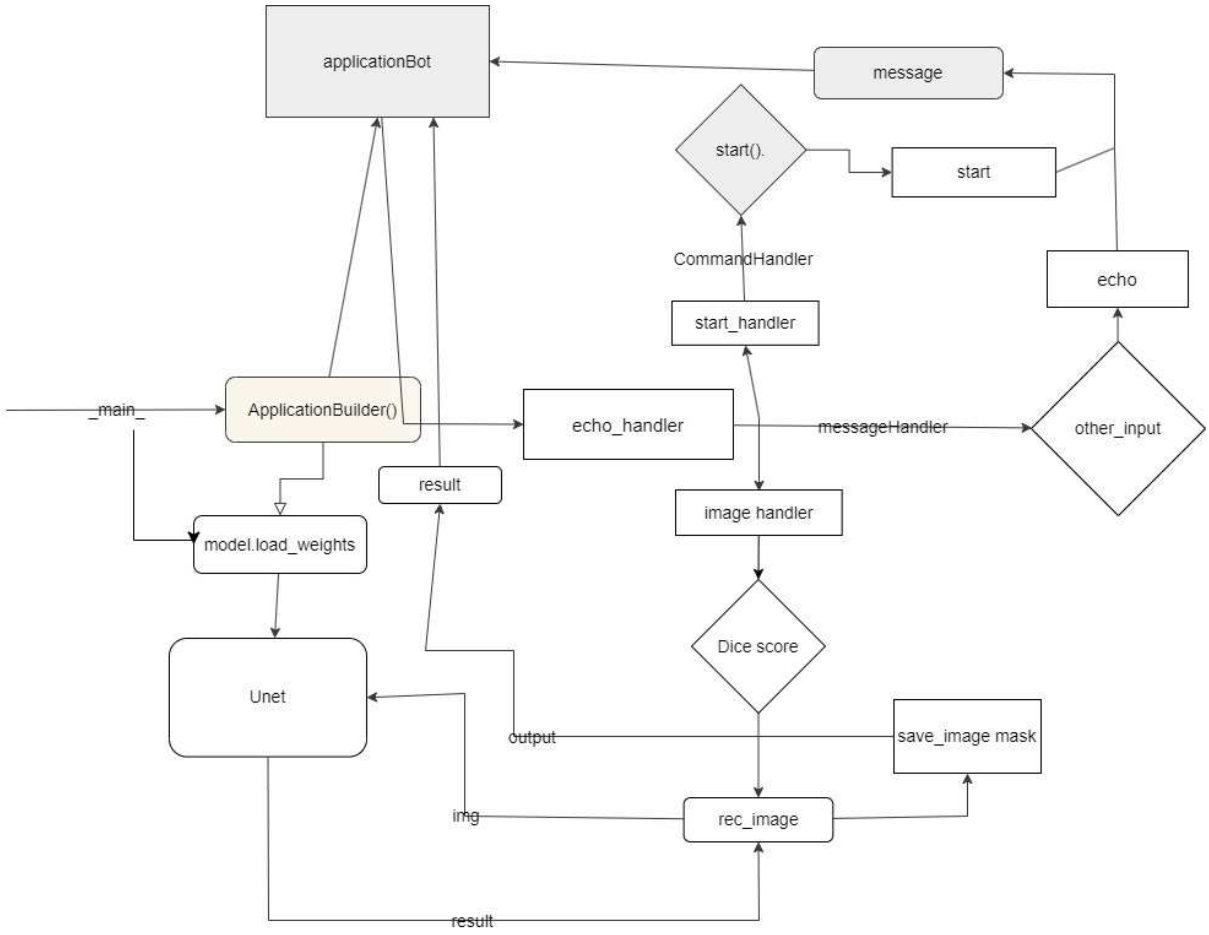


Рисунок 3.1 – Схема функціональної структури інформаційної системи

Схема на рис. 3.1. показує частину системи, що відповідає за навчання моделі U-Net для сегментації серцевих ділянок на зображеннях. Цей модуль включає кілька етапів: попередню обробку даних, побудову моделі U-Net, навчання моделі та оцінювання її роботи. Для навчання модель використовує заданий набір даних, а оцінка її роботи проводиться за допомогою балів Dice. Найкраща модель зберігається на основі отриманого балу Dice для подальшого використання.

1. Функція train_eval() тренує та оцінює модель U-Net. Функція починає роботу зі створення екземпляру моделі U-Net. Вона встановлює кількість епох та швидкість навчання.

2. Функція втрат визначається як CrossEntropyLoss, а оптимізатор встановлюється на Adam із заданою швидкістю навчання.

3. Для навчальних та валідаційних наборів даних створюються DataLoaders. Шляхи даних, кількість робітників, розміри партій і параметри перемішування встановлюються відповідно. Функція ініціалізує змінні для відстеження найкращого результату Dice score, поточного результату Dice score та журналів для навчання та валідації.

4. Для файлів журналів створюються заголовки у форматі CSV, файли журналів («trainlog.csv» і «validlog.csv») відкриваються в режимі запису і записуються заголовки. Функція входить у цикл, який повторюється для кожної епохи.

5. Модель переводиться в режим навчання за допомогою model.train().

6. У межах кожної епохи функція ітераційно переглядає завантажувач навчальних даних і виконує цикл навчання. На кожній ітерації вона отримує зображення та мітки, готує вхідне зображення для моделі, очищає градієнт оптимізатора, прогнозує маски за допомогою моделі, обчислює втрати, виконує зворотне розповсюдження та оновлює параметри моделі.

7. Після циклу навчання функція викликає функцію run_validation() з моделлю та завантажувачем навчальних даних для обчислення балів Dice score для навчального набору даних.

8. Потім вона викликає функцію avg_and_save_validation() для обчислення середніх балів Dice score і збереження їх у файлі «trainlog.csv». Загальний середній Dice score присвоюється змінній dice_score.

9. Функція знову викликає функцію run_validation(), цього разу з моделлю та завантажувачем даних для валідації, щоб обчислити бали Dice score для набору даних для валідації. Потім вона знову викликає функцію avg_and_save_validation(), щоб обчислити середні бали Dice score для набору даних валідації та зберегти їх у файлі «validlog.csv». Загальний середній Dice score присвоюється змінній dice_score. Функція перевіряє, чи поточне значення dice_score більше за best_dice (найкращий результат Dice score, досягнутий на даний момент). Якщо так, то вона оновлює best_dice до поточного значення dice_score, зберігає словник станів моделі та інші параметри навчання у файлі «./best_dice.txt». Крім того, кожні 25 епох (за винятком першої епохи) функція зберігає словник станів моделі у файл «./trained_{epoch}e.pth».

10. Функція train_eval() поєднує процеси навчання та оцінювання моделі U-Net. Вона ітераційно проходить вказану кількість епох, виконує цикл навчання, обчислює та зберігає Dice scores, а також зберігає найкращу модель на основі Dice score для перевірки.

11. Telegram bot відповідає за сегментацію ділянок серця, забезпечує користувача зручним інтерфейсом для доступу до функціональності розпізнавання та виділення ділянок серця на зображеннях. Цей модуль складається з декількох підмодулів, які працюють разом для досягнення цієї функціональності, такі як: інтеграція з телеграм-ботом, завантаження навченої моделі, розпізнавання вхідних зображень, видача результату.

12. Застосунок збирається за допомогою класу ApplicationBuilder, який надає зручний спосіб налаштування Telegram-бота. Токен Bot API встановлюється за допомогою методу token() для аутентифікації та ідентифікації бота. Екземпляр моделі U-Net створюється викликом класу U-Net(). Словник станів моделі, який містить вивчені параметри, завантажується з файлу «trained_best_dice.pth» за

допомогою torch.load(). Цей крок гарантує, що модель буде ініціалізовано навченими вагами.

13. Модель переміщується на відповідний пристрій за допомогою model.to(device). Це гарантує, що модель та її операції виконуються на обраному пристрої (CPU або GPU). Модель перемикається в режим оцінки за допомогою model.eval(). Це переводить модель у режим виводу і відключає певні операції, такі як відсікання, забезпечуючи узгодженість результатів. Для обробки вхідних повідомлень від Telegram-бота визначено різні обробники:

14. Обробник start_handler обробляє команду ‘/start’, викликаючи функцію start().

15. Обробник echo_handler обробляє будь-які текстові повідомлення (крім команд) за допомогою виклику функції echo().

16. Обробник image_handler обробляє повідомлення, що містять фотографії, за допомогою виклику функції rec_image().

17. Функція отримує ідентифікатор файлу отриманої фотографії з повідомлення про оновлення. Потім вона використовує API Telegram-бота, щоб завантажити фотографію в локальний каталог.

18. Завантажене зображення зчитується за допомогою OpenCV (cv2.imread) і конвертується у відтінки сірого, якщо воно має три канали. Зображення попередньо обробляється шляхом зміни розміру до фіксованого розміру (96x96) і перетворення в тензор PyTorch.

19. Попередньо оброблений тензор зображення пропускається через попередньо навчену U-Net-модель (модель) для сегментації. Результатом роботи моделі є тензор, що представляє передбачуваний клас (маску сегментації) для кожного пікселя зображення.

20. Тензор передбачених класів перетворюється в масив NumPy (numpy()), і викликається допоміжна функція save_image_mask для візуалізації вихідного зображення і маски сегментації поруч. Візуалізація зберігається як новий файл зображення.

21. Отриманий файл зображення надсилається назад у вигляді фотоповідомлення в чат за допомогою API Telegram-бота.

22. Виконуючи ці кроки, функція rec_image застосовує сегментацію зображення до отриманої фотографії, використовуючи попередньо навчену U-Net-модель, і надсилає оброблене зображення з накладеною сегментацією назад у чат.

Telegram-бот написаний на мові Python. Його можна використовувати для задач сегментації зображень та розпізнавання фотографій. Він використовує API бота Telegram для взаємодії з користувачами та обробки надісланих ними зображень.

Застосунок дає змогу користувачам надсилати боту фотографії, які потім сегментуються за допомогою попередньо навченої моделі U-Net. Модель U-Net завантажується зі збереженого словника станів (trained_best_dice.pth) і застосовується до отриманих зображень для сегментації. Сегментовані зображення потім надсилаються назад користувачам, відображаючи оригінальне зображення поряд з маскою сегментації.

Ця програма може бути корисною в різних сценаріях, таких як медична візуалізація, виявлення об’єктів та аналіз зображень. Він надає зручний спосіб виконувати сегментацію зображень на наданих користувачем фотографіях через інтерфейс чату в Telegram. Користувачі можуть використовувати цього бота, щоб отримувати інформацію із зображень, визначати області, що цікавлять, або витягувати значущу інформацію з візуальних даних.

3.2 Особливості реалізації програмних складових інформаційної системи

Основний модуль інформаційної системи відповідає за сегментацію ділянок серця на зображенні. Схема на рисунку 3.2 показує функції, які здійснюють попередню обробку зображення для підготовки їх до подальшого аналізу.

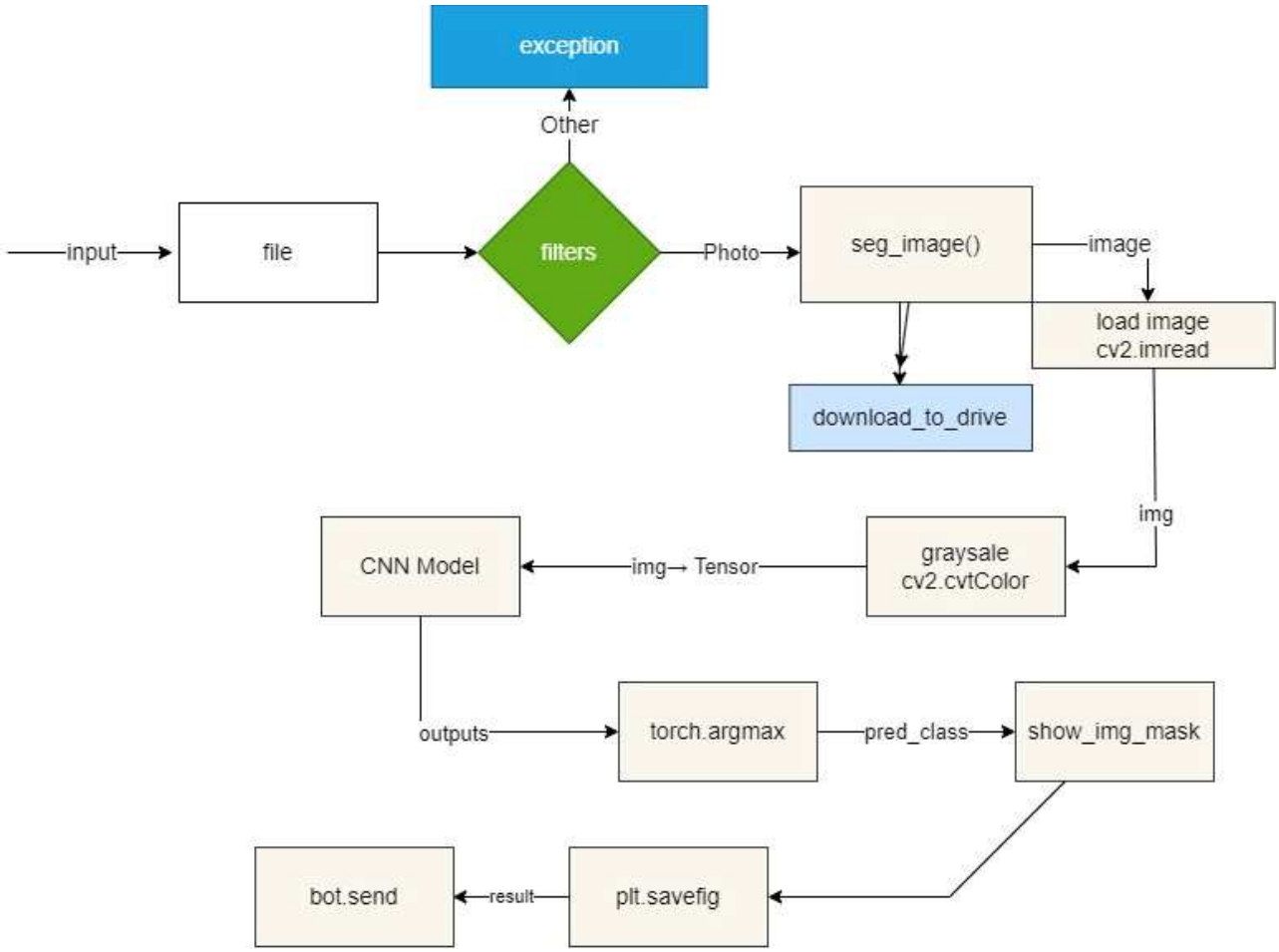


Рисунок 3.2 – Процес обробки вхідного зображення

Починаючи з отримання файлу системою та передачі його ідентифікатору до функції розпізнавання, модуль визначає тип вхідного файлу та вибирає відповідну функцію обробки, використовуючи фільтри. У разі зображення система використовує функцію сегментації ділянок серця для обробки зображення та виділення ділянок, що відповідають серцю. Ця функція виконує попередню обробку зображення, наприклад, зменшення його розміру та виконує алгоритм сегментації для виділення ділянок серця на зображенні.

1. При отриманні повідомлення, що містить фотографію, викликається функція seg_image(). Ця функція отримує в якості параметрів об’єкти update та context, які містять інформацію про повідомлення та надають доступ до функціоналу Telegram-бота.

2. Ідентифікатор файлу фотографії витягується з об’єкта update за допомогою update.message.photo[-1].file_id. Фотографія зберігається у файл в

директорії «in_image/» за допомогою методу download_to_drive() об’єкта telegram.File. Це гарантує, що фотографія зберігається локально для подальшої обробки.

3. Збережений файл зображення зчитується за допомогою функції cv2.imread() з бібліотеки OpenCV. Ця функція читає файл зображення і повертає масив NumPy, що представляє зображення. Якщо зображення має три канали (що вказує на те, що це кольорове зображення), воно конвертується у відтінки сірого за допомогою функції cv2.cvtColor() для спрощення обробки.

4. Зображення попередньо обробляється шляхом зміни розміру до фіксованого розміру (96, 96) за допомогою функції cv2.resize(). Зміна розміру зображення гарантує, що воно матиме однакові розміри для обробки.

5. Масив NumPy, що представляє попередньо оброблене зображення, перетворюється на тензор PyTorch за допомогою torch.from_numpy(). Цей крок дозволяє легко інтегруватися з моделлю PyTorch.

6. Тензор зображення передається через модель за допомогою model(image). Це виконує прямий прохід моделі, генеруючи передбачення для вхідного зображення. Отримується вихідний тензор, який представляє маску сегментації для вхідного зображення.

7. Для перетворення вихідного тензора в передбачене представлення класів використовується функція torch.argmax(). Вона знаходить індекс максимального значення вздовж заданого виміру.

8. Прогнозований тензор класів стискається і перетворюється у масив NumPy за допомогою .squeeze().cpu().numpy(). Цей крок видаляє всі синглетні виміри і приносить тензор до центрального процесора для легкої маніпуляції.

9. Функція save_image_mask() викликається для створення візуалізації оригінального зображення і передбаченої маски. Ця функція використовує бібліотеку Matplotlib для створення фігури з двома підграфіками: один для оригінального зображення і один для прогнозованої маски.

10. Візуалізація зберігається як графічний файл за допомогою plt.savefig(). Ця функція зберігає фігуру за вказаним шляхом до файлу. Користувачеві надсилається повідомлення, що містить кінцеве зображення у вигляді фотографії.

Описаний вище модуль здійснює попередню обробку зображення, включаючи зміну розміру та конвертацію у відтінки сірого, після чого застосовує алгоритм сегментації для виділення ділянок, що відповідають серцю. Результатом роботи модуля є візуалізація оригінального зображення та прогнозованої маски сегментації, які надсилаються користувачеві.

3.3 Експериментальне тестування інформаційної системи

Результати тестування дають уявлення про продуктивність інформаційної системи для сегментації ділянок серця на зображеннях МРТ. Оцінка Dice використовується як метрика оцінки для вимірювання схожості між прогнозованими масками сегментації та базовими масками (рисунок 3.3).

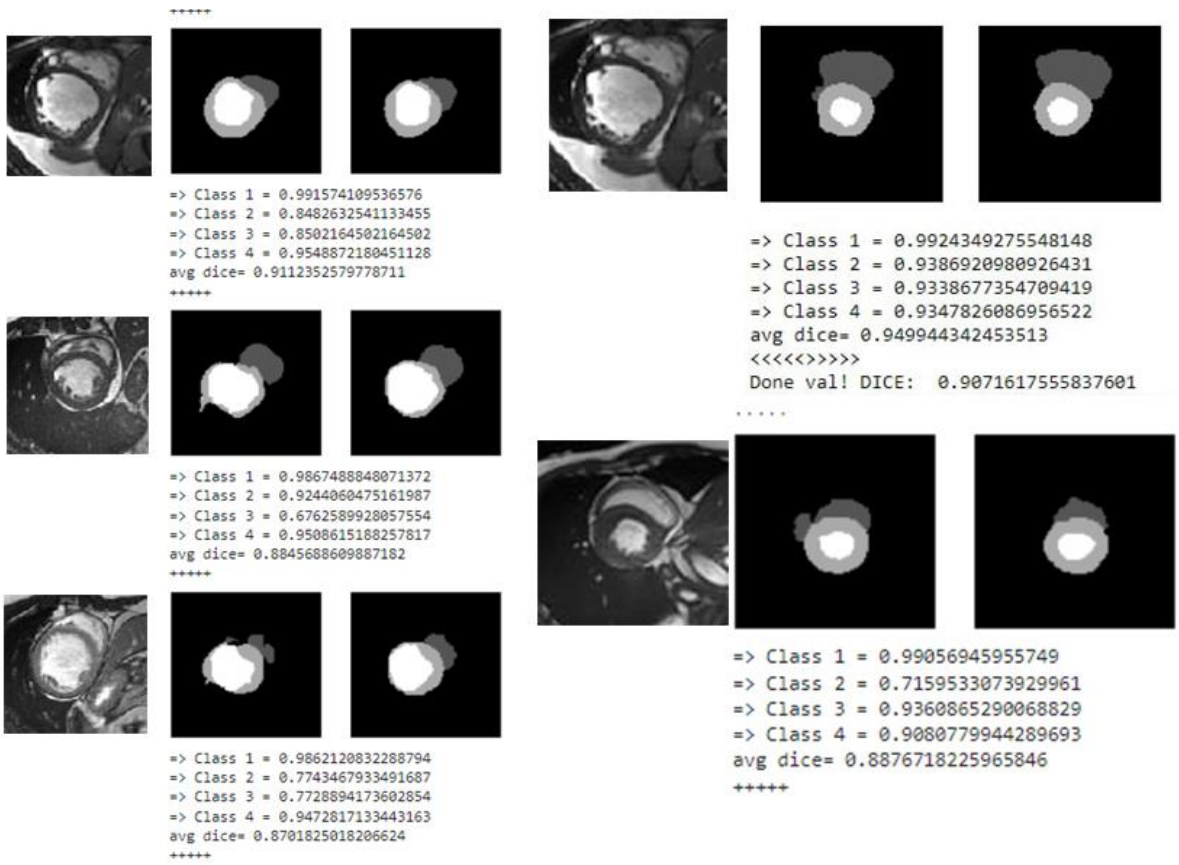


Рисунок 3.3 – Результати тестування на валідаційному наборі даних

В таблиці 3.1 можна переглянути та порівняти результати тестування.

Таблиця 3.1 – Чисельні результати різних зображень

Зображення 1	=> Class 1 = 0.991574109536576 => Class 2 = 0.8482632541133455 => Class 3 = 0.8502164502164502 => Class 4 = 0.9548872180451128 avg dice= 0.9112352579778711
Зображення 2	=> Class 1 = 0.9924349275548148 => Class 2 = 0.9386920980926431 => Class 3 = 0.9338677354709419 => Class 4 = 0.9347826086956522 avg dice= 0.949944342453513
Зображення 3	=> Class 1 = 0.9867488848071372 => Class 2 = 0.9244060475161987 => Class 3 = 0.6762589928057554 => Class 4 = 0.9508615188257817 avg dice= 0.8845688609887182
Зображення 4	=> Class 1 = 0.9862120832288794 => Class 2 = 0.7743467933491687 => Class 3 = 0.7728894173602854 => Class 4 = 0.9472817133443163 avg dice= 0.8701825018206624
Зображення 5	=> Class 1 = 0.99056945955749 => Class 2 = 0.7159533073929961 => Class 3 = 0.9360865290068829 => Class 4 = 0.9080779944289693 avg dice= 0.8876718225965846

Оцінка Dice, також відома як оцінка F1, – це метрика, яка зазвичай використовується для оцінки ефективності алгоритмів сегментації зображень. Вона вимірює перекриття між прогнозованою сегментацією (отриманою від ЗНМ) і сегментацією істинної картини (наданою в наборі даних). Оцінка Dice 1.0 означає

ідеальний збіг, тоді як оцінка 0.0 вказує на повну відсутність збігу. Давайте проаналізуємо результати більш детально:

Клас 1. Оцінка Dice для класу 1 дорівнює 0.9924. Це означає, що система досягла високого рівня точності в сегментації областей класу 1, які можуть відповідати певній анатомічній структурі або особливості серця.

Клас 2. Оцінка Dice для класу 2 становить 0,9387. Цей показник свідчить про те, що система добре виконала сегментацію ділянок класу 2, продемонструвавши хорошу точність у визначенні меж і деталей цих ділянок.

Клас 3. Оцінка Dice для класу 3 становить 0,9339. Це значення вказує на те, що система ефективно сегментувала області класу 3 на МРТ-зображеннях, досягнувши високого рівня схожості з базовими масками.

Клас 4. Оцінка Dice для класу 4 становить 0,9348. Цей показник означає, що система точно сегментує області класу 4, демонструючи сильну схожість з базовими масками.

Середній показник Dice для всіх класів становить 0,9499. Цей середній показник відображає загальну ефективність системи в сегментації областей серця. Він вказує на те, що система досягла високого рівня точності та схожості з базовими масками на МРТ-зображеннях.

На додачу до оцінок для кожного класу, фрагмент також обчислює середній бал за шкалою Dice для кожного окремого зображення в наборі даних для валідації. Останній рядок результатів показує загальну оцінку Dice для валідаційного набору даних, яка становить 0,9072. Цей показник відображає середню схожість між прогнозованими масками сегментації та базовими масками для всіх зображень у валідаційному наборі даних.

Ці результати свідчать про те, що модель ЗНМ, яка використовується для сегментації, є ефективною і здатна точно ідентифікувати та сегментувати різні ділянки серця на МРТ-зображеннях. Високі оцінки за шкалою Dice вказують на сильне перекриття між прогнозованою та фактичною сегментацією, що свідчить про точні та достовірні результати. Середні та загальні оцінки Dice ще більше підвищують надійність та продуктивність системи сегментації.

Проведене тестування навченої моделі на тестовому наборі даних показало її здатність до ефективної сегментації ділянок серця на зображеннях серця. Для остаточного підтвердження повного функціонування розробленої системи необхідно провести додаткове функціональне тестування процесів, які були визначені у розділі 2.4.

Для перевірки коректної роботи системи було обрано вхідні дані, з якими проводилось тестування. Початковою точкою були фотографії, на яких зображені знімки МРТ серця. Було надіслано боту декілька зображень, кожне з яких містило різні серця. Результати розпізнавання показані на рисунку 3.4.

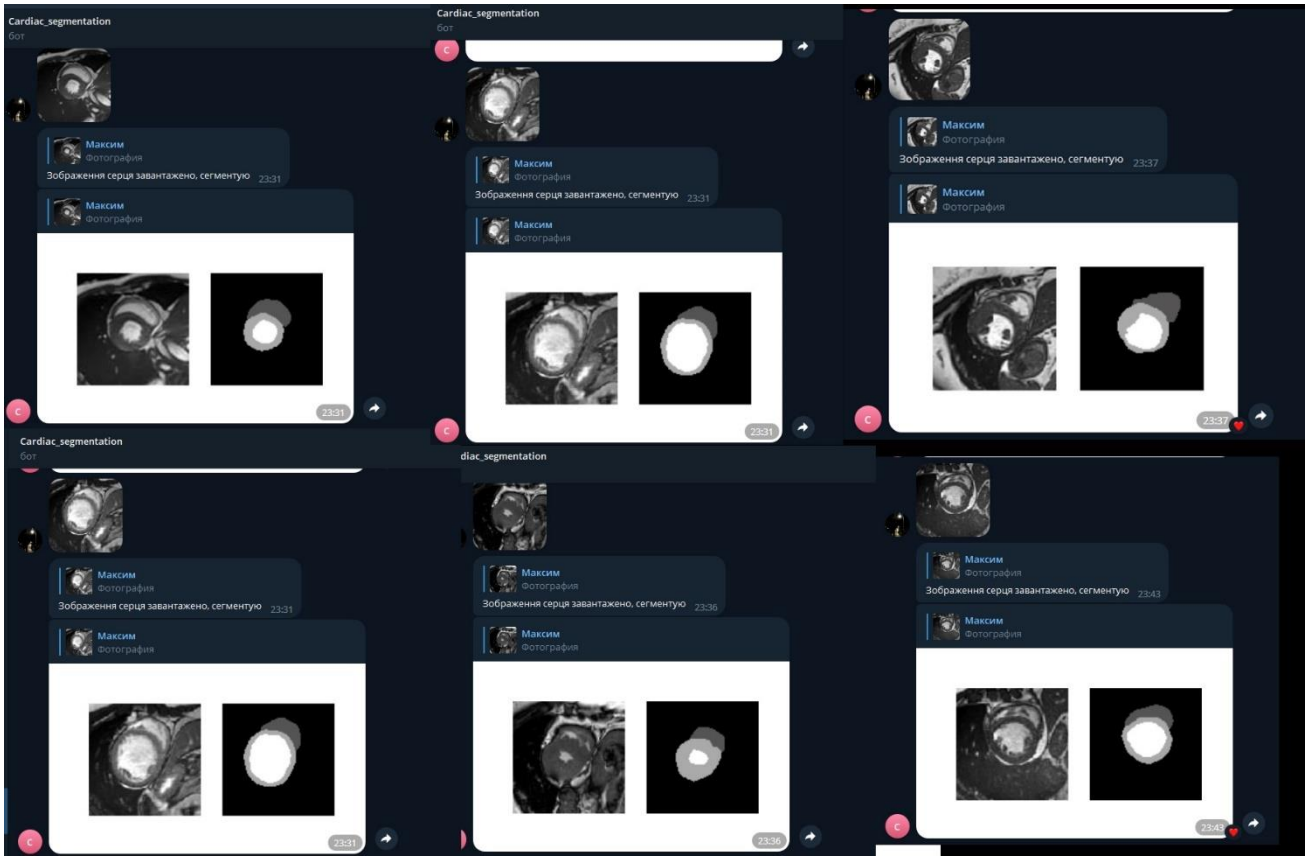


Рисунок 3.4 – Сегментація ділянок серця за допомогою Telegram-боту в настільній версії застосунку

Результати з рисунку 3.4 підтверджують правильну роботу системи з сегментації ділянок серця, що свідчить про успішність розробленої інформаційної системи у досягненні своєї основної мети.

На рисунку 3.5 продемонстровано результати сегментації, виконані у мобільній версії розробленого Telegram-боту.

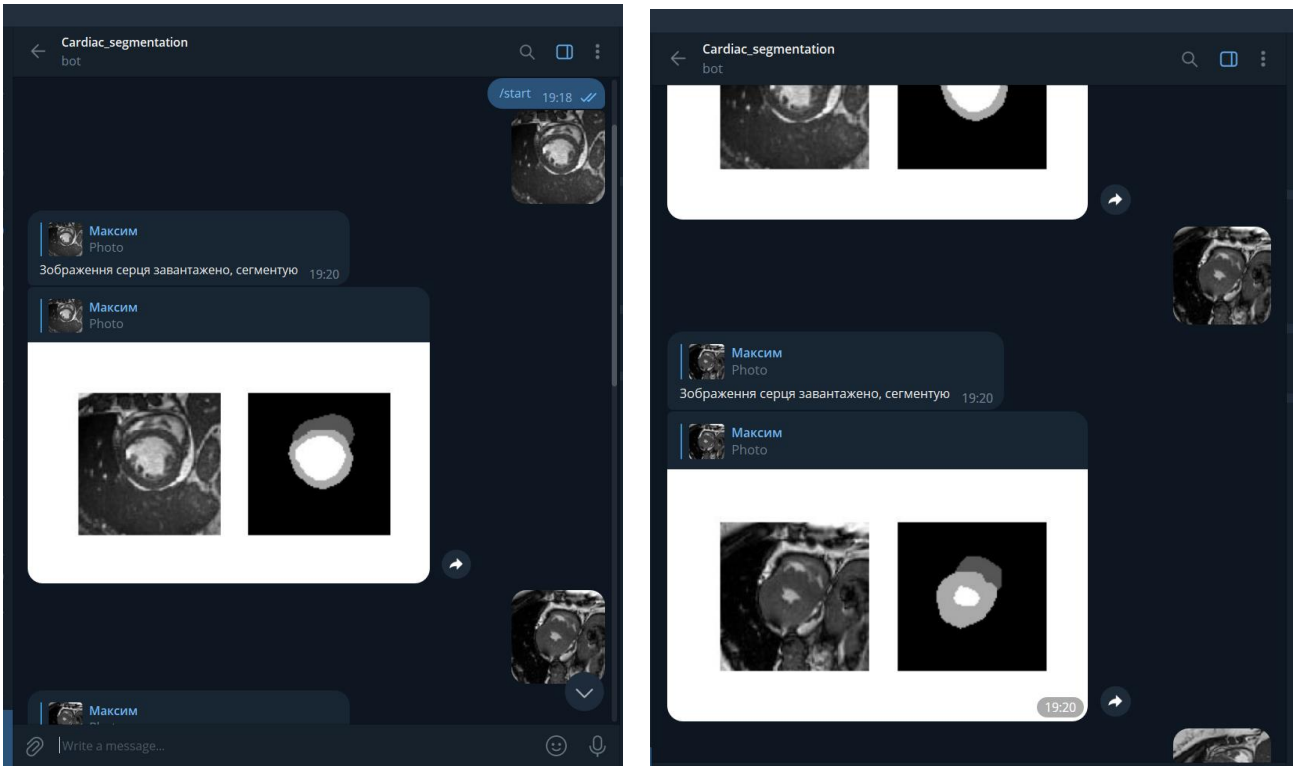


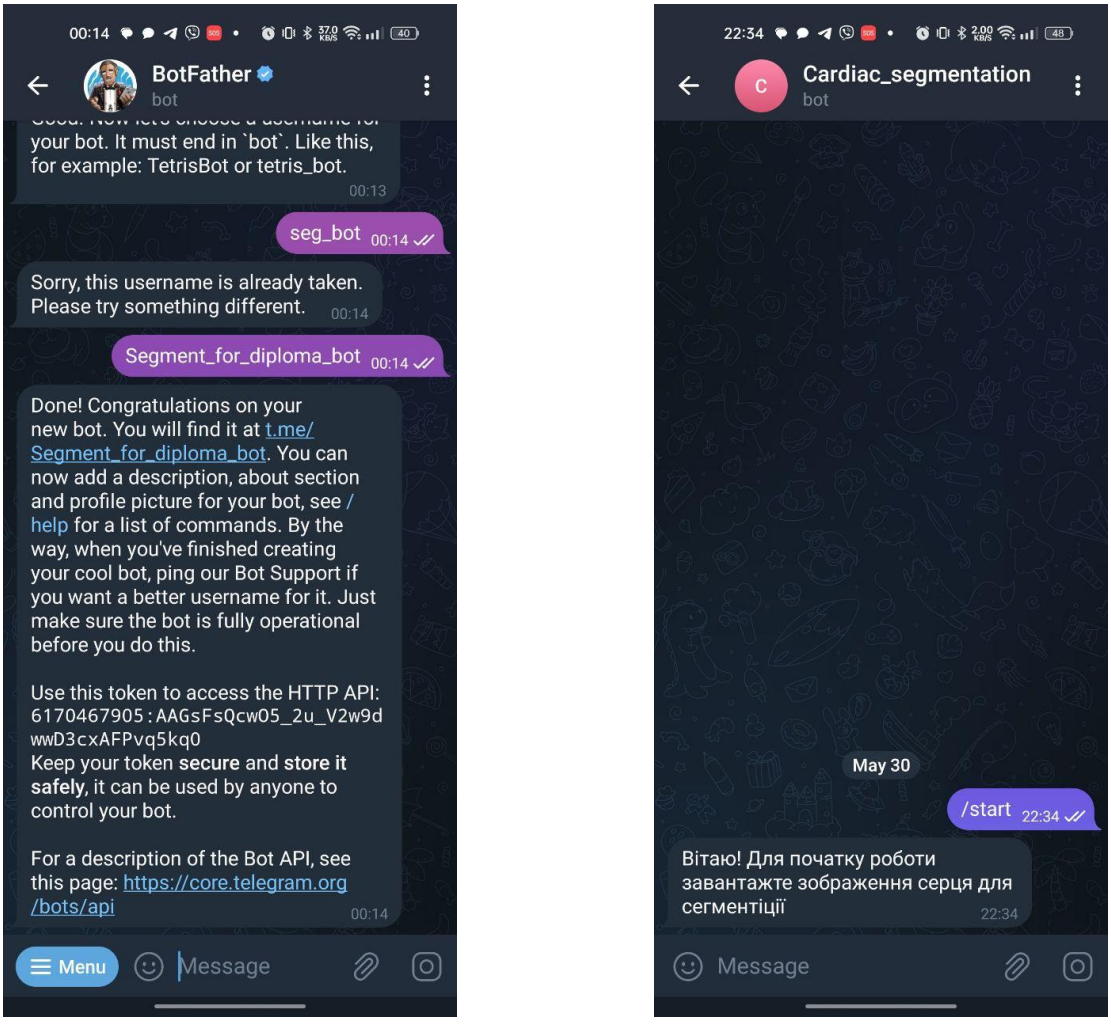
Рисунок 3.5 – Сегментація ділянок серця за допомогою Telegram-боту в мобільній версії застосунку

Загалом, результати тестування (рисунки 3.4-3.5) розробленої інформаційної системи за Telegram-ботом на вхідних даних підтверджують її здатність до точної сегментації ділянок серця на зображеннях. Це свідчить про успішність розробленого підходу до виявлення та ідентифікації різних областей серця. Під час функціонального тестування модулів системи було продемонстровано їх надійність та ефективність в роботі з різними типами зображень.

У підсумку, розроблена інформаційна система виконує свої завдання з високою точністю та забезпечує достовірну сегментацію ділянок серця на медичних зображеннях.

3.4 Вимоги до розгортання інформаційної системи та інструкція користувача

Для сегментації серця на зображеннях МРТ використовується ЗНМ. Щоб розгорнути систему на іншому комп’ютері, необхідно змінити файл bot_run.py. Зверніться до BotFather у Telegram, щоб отримати токен для створення нового бота (рисунок 3.6а).



(а) **(б)**

Рисунок 3.6 – Отримання токену з Telegram-боту (а) та початок роботи користувача з ботом (б)

Вставте отриманий токен у рядок: application = ApplicationBuilder().token(‘Ваш токен’).build(). Потім оновіть шляхи path_in та path_out у функції seg_image для збереження файлів, якщо потрібно. Після цих

змін і виконання файлу, бот буде запущено і готовий до використання. Для початку роботи з ботом знайдіть його за іменем, визначеним у BotFather. Далі потрібно відкрити чат з ботом у Telegram та надіслати команду «/start» для початку роботи. Користувач отримає привітальне повідомлення з інструкціями (рисунок 3.6б).

Далі потрібно надіслати файл боту. Натисніть значок скріпки біля поля введення повідомлення та виберіть зображення (рисунок 3.7).

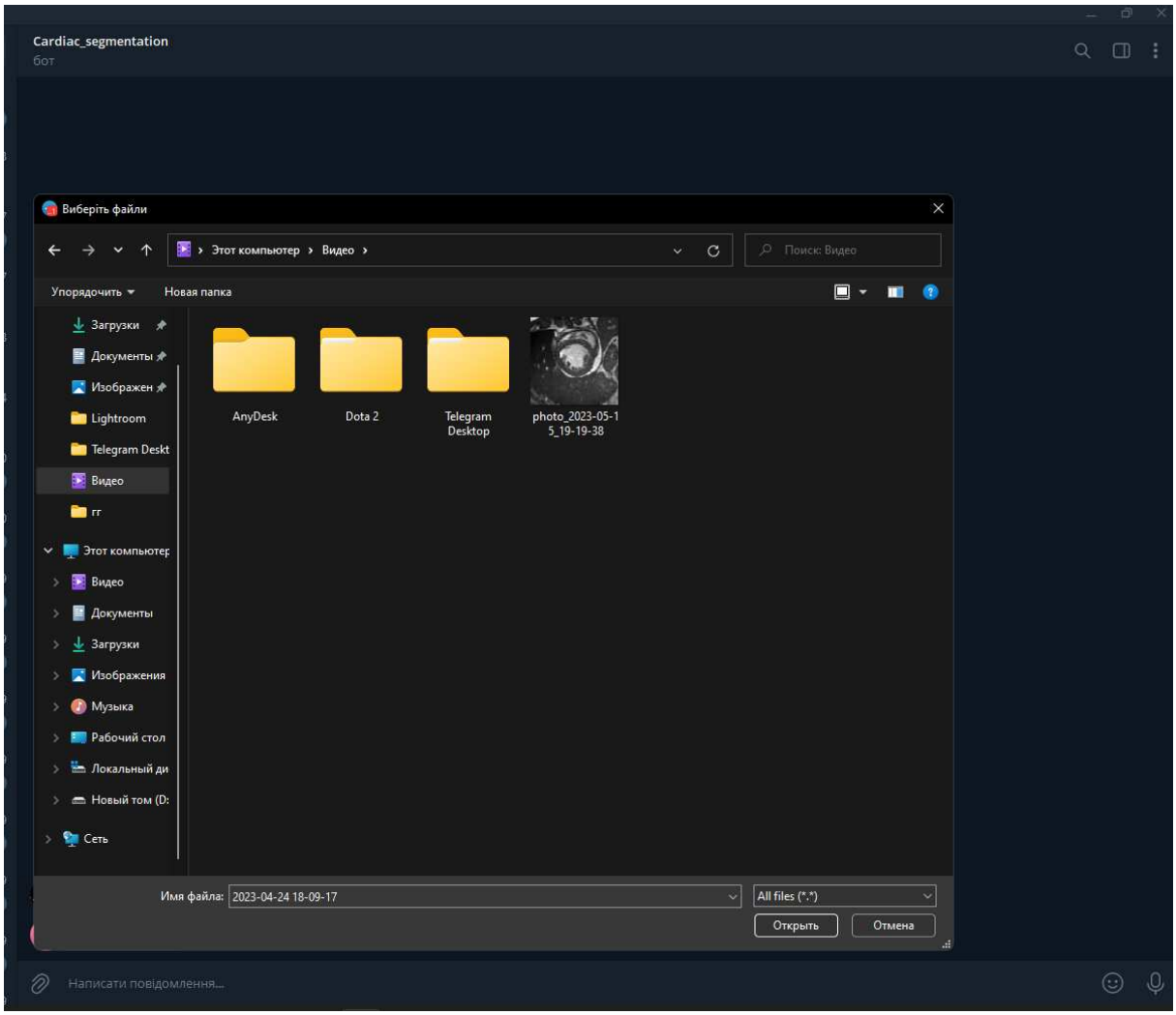


Рисунок 3.7 – Вибір зображення з носія

Після вибору зображення, необхідно надіслати його для сегментації серця. Ви отримаєте зображення з виконаною сегментацією серця (рисунок 3.8).

Після отримання результатів можете надіслати інший медіафайл або завершити роботу з системою.

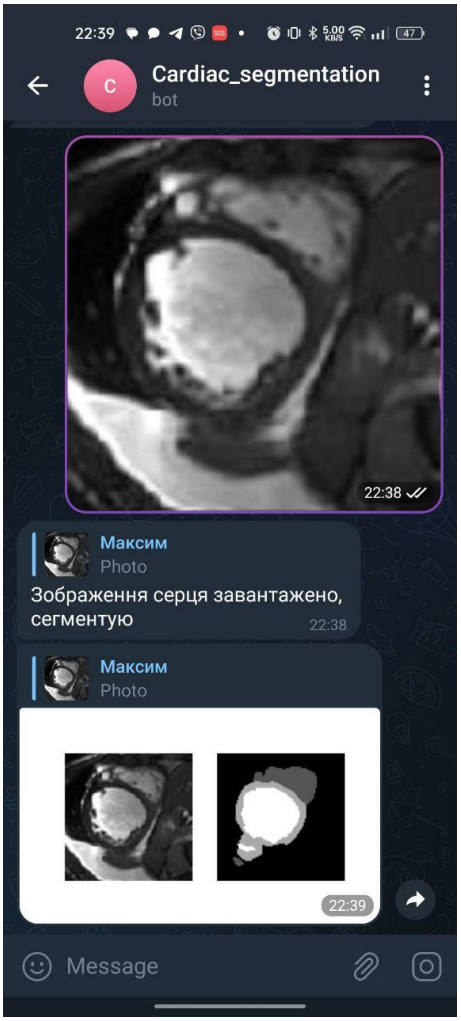


Рисунок 3.8 – Приклад користувацького інтерфейсу розробленого Telegram-бота

Інформаційна система розроблена в середовищі мови програмування Python і потребує інсталювання такого додаткового програмного забезпечення для розгортання та функціонування телеграм-боту:

1. Сумісність з платформами: Telegram-бот є сумісним з усіма основними операційними системами (Windows, macOS, Linux), веб-браузерами (Chrome, Firefox, Edge, Safari) та мобільними платформами (Android, iOS).

2. Мова програмування: Python 3.7 або вище.

3. Фреймворки глибоко навчання: TensorFlow 2.x та PyTorch 1.x для реалізації моделей сегментації МРТ серця.

3. Бібліотеки обробки медичних зображень: PyDicom і NiBabel для обробки форматів DICOM і NIFTI.

4. Попередньо навчена модель.

3.5 Висновки до розділу 3

В даному розділі була розроблена інформаційна система для сегментації ділянок серця на зображеннях за допомогою бота в Telegram. Система складається з двох основних частин: перший модуль для навчання моделі U-Net з використанням набору даних і другий модуль - Telegram-бот, який надає користувачам інтерфейс для доступу до функції сегментації ділянок серця.

У першому модулі розглядається процес навчання моделі U-Net для сегментації серцевих ділянок на зображеннях. Проводиться попередня обробка даних, побудова моделі U-Net, навчання моделі та оцінювання її роботи. Для навчання моделі використовується заданий набір даних, а оцінка її роботи проводиться за допомогою балів Dice.

Другий модуль включає роботу з Telegram-ботом, який дозволяє користувачам завантажувати зображення і отримувати результат сегментації ділянок серця на цих зображеннях. Після завантаження зображення, навчена модель розпізнає та сегментує ділянки серця на ньому, і результат сегментації передається назад користувачу через Telegram-бот.

Застосунок збирається за допомогою класу ApplicationBuilder, який надає зручний спосіб налаштування Telegram-бота. Модель U-Net із навченими параметрами завантажується з файлу для використання в боті. Після цього бот готовий обробляти команди користувачів і виконувати сегментацію ділянок серця на завантажених зображеннях.

Отже, розроблена інформаційна система забезпечує зручний та автоматизований доступ користувачів до функції сегментації ділянок серця на зображеннях через Telegram-бот.

Висновки

У кваліфікаційній роботі бакалавра розглянуто і розроблено спосіб виявлення області інтересу на зображеннях МРТ з використанням згорткових нейронних мереж. Метою було розробити ефективний метод сегментації ділянок серця на зображеннях МРТ, що може бути корисним у медичній діагностиці та дослідженнях серцево-судинних захворювань.

Дослідження розпочалося з огляду техніки навчання згорткової нейронної мережі, яка включає підготовку даних та вибір архітектури мережі. У цьому контексті була використана ЗНМ з архітектурою U-Net, яка успішно демонструє свою ефективність у завданнях обробки зображень.

У роботі підготовлено та навчено модель ЗНМ U-Net для сегментації ділянок серця на зображеннях МРТ. При проєктуванні були враховані особливості структури серця та зображень МРТ. Використовувалися різні типи шарів, такі як згорткові, пулінгові та повнозв’язані шари, з метою досягнення оптимальної точності сегментації.

Для реалізації цього підходу була розроблена функціональна структура інформаційної системи, яка ґрунтується на ЗНМ. Ця система включає функціональні процеси, які допомагають обробляти та аналізувати зображення МРТ для виявлення області інтересу, зокрема сегментацію ділянок серця. Цей підхід може бути використаний у медичній практиці для допомоги лікарям у діагностиці та стеженні за хворими з серцево-судинними захворюваннями, а також у дослідницьких проектах, спрямованих на вивчення функціонування серця та його патологій.

Крім того, була розроблена інформаційна система, яка забезпечує зручний та автоматизований доступ користувачів до функції сегментації ділянок серця на зображеннях МРТ за допомогою Telegram-бота. Таке подання інформаційної системи дає змогу швидко та ефективно отримувати результати сегментації безпосередньо на мобільному пристрої. Програмне забезпечення складається з двох основних модулів.

У першому модулі розглядається процес навчання моделі U-Net для сегментації серцевих ділянок на зображеннях. Проводиться попередня обробка даних, побудова моделі U-Net, навчання моделі та оцінювання її роботи. Для навчання моделі використовується заданий набір даних, а оцінка її роботи проводиться за допомогою балів Dice.

Другий модуль включає роботу з Telegram-ботом, який дозволяє користувачам завантажувати зображення і отримувати результат сегментації ділянок серця на цих зображеннях. Після завантаження зображення, навчена модель розпізнає та сегментує ділянки серця на ньому, і результат сегментації передається назад користувачу через Telegram-бот.

Застосунок збирається за допомогою класу ApplicationBuilder, який надає зручний спосіб налаштування Telegram-бота. Модель U-Net із навченими параметрами завантажується з файлу для використання в боті. Після цього бот готовий обробляти команди користувачів і виконувати сегментацію ділянок серця на завантажених зображеннях. В результаті проведених експериментальних тестувань встановлено, що розроблена інформаційна система забезпечує зручний та автоматизований доступ користувачів до функції сегментації ділянок серця на зображеннях через Telegram-бот.

У підсумку, використання ЗНМ разом з функціональною структурою інформаційної системи дозволяє досягти високої точності та ефективності у виявленні області інтересу на зображеннях МРТ. Цей підхід має потенціал для впровадження у клінічну практику та дослідницькі проекти, спрямовані на вивчення та діагностику серцево-судинних захворювань.

Перелік посилань

1. Від чого помирають люди у світі? - BBC News Україна. *BBC News Україна*. URL: https://www.bbc.com/ukrainian/features-47474480

2. Катеренчук І.П. Огляд основних положень рекомендацій європейського товариства кардіологів із діагностики та лікування серцево-судинних захворювань під час пандемії COVID-19 (2021). Частина 2: Лікування та подальші спостереження. *Практикуючий лікар*. 2022. №1. С. 17-25.

3. Risks of Implementing Artificial Intelligence in Computer Systems / K. Marchenko et al. *Central Ukrainian Scientific Bulletin. Technical Sciences*. 2022. Vol. 1, No. 5(36). Pp. 119-124.

4. Human-in-the-loop approach based on MRI and ECG for healthcare diagnosis / P. Radiuk et al. *The 5th International Conference on Informatics & Data-Driven Medicine (IDDM-2022)* : CEUR-Workshop Proceedings. Vol. 3302. (Lyon, France, 18-20 November 2022). CEUR-WS.org, Aachen, 2022. Pp. 9-20.

5. Image-based cardiac diagnosis with machine learning: A Review / C. Martin-Isla et al. *Frontiers in Cardiovascular Medicine*. 2020. Vol. 7. Pp. 1-19.

6. Segmentation and classification in MRI and US fetal imaging: Recent trends and future prospects / J. Torrents-Barrena et al. *Medical Image Analysis*. 2019. Vol. 51. Pp. 61-88.

7. Бармак О.В., Радюк П.М. Інформаційна технологія візуального аналізу рентгенівських зображень для інтерпретації результатів діагностування пневмонії. *Вісник Хмельницького національного університету. Технічні науки*. 2021. № 295(2). С. 52-55.

8. Neuromonitoring, neuroimaging, and neurodevelopmental follow-up practices in neonatal congenital heart disease: A European survey / M. Feldmann et al. *Pediatric Research*. 2022. Vol. 93. Pp. 168-175.

9. Segment Research – Medviso. *Medviso – Medical software solutions*. URL: https://medviso.com/segment/

10. ITK-SNAP. *ITK-SNAP Home*. URL: http://www.itksnap.org/pmwiki/pmwiki.php?n=Main.HomePage

11. DeepSeg: Deep neural network framework for automatic brain tumor segmentation using magnetic resonance FLAIR images / R.A. Zeineldin et al. *International Journal of Computer Assisted Radiology and Surgery*. 2020. Vol. 15, No. 6. Pp. 909-920.

12. Seg3D: Segmentation Image Processing. *The NIH/NIGMS Center for Integrative Biomedical Computing*. URL: https://www.sci.utah.edu/cibc-software/seg3d.html

13. Синявський В.І. Аналіз методів сегментації зображень/сигналів в задачах медичного діагностування : дипломна робота магістра за спеціальністю 122 – комп'ютерні науки / В.І. Синявський. Тернопіль: ТНТУ, 2019. 129 с.

14. A novel feature vector for ECG classification using deep learning / O. Kovalchuk et al. *The 4th International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS-2023)* : CEUR-Workshop Proceedings. Vol. 3373. (Khmelnytskyi, Ukraine, 22-24 March 2023). CEUR-WS.org, Aachen, 2023. Pp. 227-238.

15. An overview of deep learning methods for left ventricle segmentation / M.A. Shoaib et al. *Computational Intelligence and Neuroscience*. 2023. Vol. 2023. Pp. 1-26.

16. Radiuk P., Barmak O., Krak Iu. An approach to early diagnosis of pneumonia on individual radiographs based on the CNN information technology. *The Open Bioinformatics Journal*. 2021. Vol. 14, No 1. Pp. 93-107.

17. NAS-HRIS: Automatic design and architecture search of neural network for semantic segmentation in remote sensing images / M. Zhang et al. *Sensors*. 2020. Vol. 20, No. 18. Pp. 5292.

18. Liu Z., Huang J. Semantic segmentation network of UAV image based on improved U-Net. *IOP Conference Series: Earth and Environmental Science*. 2019. Vol. 330, No. 5. P. 052050.

19. SegNet-based left ventricular MRI segmentation for the diagnosis of cardiac hypertrophy and myocardial infarction / Z. Yan et al. *Computer Methods and Programs in Biomedicine*. 2022. Vol. 227. Pp. 107197.

20. Automatic segmentation of cardiac magnetic resonance images based on multi-input fusion network / J. Shi et al. *Computer Methods and Programs in Biomedicine*. 2021. Vol. 209. Pp. 106323.

21. Improving myocardial pathology segmentation with U-Net++ and EfficientSeg from multi-sequence cardiac magnetic resonance images / H. Cui et al. *Computers in Biology and Medicine*. 2022. Vol. 151, No. A. P. 106218.

22. Saood A., Hatem I. COVID-19 lung CT image segmentation using deep learning methods: U-Net versus SegNet. *BMC Medical Imaging*. 2021. Vol. 21, No. 1. Pp. 1-10.

23. Radiuk P.M. Applying 3D U-Net architecture to the task of multi-organ segmentation in computed tomography. *Applied Computer Systems*. 2020. Vol. 25, No 1, Pp. 43-50.

24. Chapter 9 - Spatial Characteristics of the Magnetic Resonance Image. *Magnetic Resonance Imaging*. URL: https://bit.ly/427eIIJ

25. John J. Image processing techniques for identifying tumors in an MRI image. *arXiv:2103.15152*. 2021. Pp. 1-6.

26. Radiuk P.M. Application of a genetic algorithm to search for the optimal convolutional neural network architecture with weight distribution. *Herald of Khmelnytskyi National University. Technical sciences*. 2020. Vol. 281, No 1. Pp. 7-11.

27. nn-TransUNet: An automatic deep learning pipeline for heart MRI segmentation / L. Zhao et al. *Life*. 2022. Vol. 12, No. 10. Pp. 1570.

ДОДАТКИ

Додаток А

Лістинг програмного коду

Модуль завантаження та оброблення даних:

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

print(torch.cuda.is_available())
print('device count', torch.cuda.device_count())
print('current', torch.cuda.current_device())
print('GPU', torch.cuda.get_device_name(0))

from matplotlib import pyplot as plt

def show_image_mask(img, mask, cmap='gray'): # visualisation
    fig = plt.figure(figsize=(5,5))
    plt.subplot(1, 2, 1)
    plt.imshow(img, cmap=cmap)
    plt.axis('off')
    plt.subplot(1, 2, 2)
    plt.imshow(mask, cmap=cmap)
    plt.axis('off')
    plt.show() # draw the images immediatelly>>>>> DATA LOADERS <<<<<
(including fix for linux glob)



class TrainDataset(data.Dataset):
    def __init__(self, root=''):
        super(TrainDataset, self).__init__()
        self.img_files = glob(os.path.join(root,'image','*.png'))
        #self.img_files = self.img_files[0:10] # only using part of the dataset
        self.mask_files = []
        for img_path in self.img_files:
            basename = os.path.basename(img_path)
            self.mask_files.append(os.path.join(root,'mask',basename[:-
4]+'_mask.png'))


    def __getitem__(self, index):
            img_path = self.img_files[index]
            mask_path = self.mask_files[index]
            data = cv2.imread(img_path, cv2.IMREAD_UNCHANGED)
            label = cv2.imread(mask_path, cv2.IMREAD_UNCHANGED)
            return torch.from_numpy(data).float(), torch.from_numpy(label).float()

    def __len__(self):
        return len(self.img_files)
```

```
class TestDataset(data.Dataset):
    def __init__(self, root=‘‘):
        super(TestDataset, self).__init__()
        #self.img_files = glob(os.path.join(root,’image’,’*.png’))
        self.img_files = natsort.natsorted(glob(os.path.join(root,’image’,’*.png’)))

    def __getitem__(self, index):
            img_path = self.img_files[index]
            print("get "+img_path)
            data = cv2.imread(img_path, cv2.IMREAD_UNCHANGED)
            return torch.from_numpy(data).float()

    def __len__(self):
        return len(self.img_files)

    def __insert__(self, newimgfiles, newmaskfiles):
        #self.img_files = torch.cat[self.img_files, newimgfiles]
        #self.mask_files = torch.cat[self.mask_files, newmaskfiles]
        return 1
```

Клас, який реалізує архітектуру мережі U-Net:

```
class UNet(nn.Module):
    def __init__(self):
        super(UNet, self).__init__()

        kernel_size = 5
        padding = 2
        dropout_rate = 0.5

        # Down Layers

        self.enc_11 = nn.Sequential(*[
            nn.Conv2d(in_channels=1,
                      out_channels=64,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True)])

        self.enc_12 = nn.Sequential(*[
            nn.Conv2d(in_channels=64,
                      out_channels=64,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True)
        ])

        self.enc_21 = nn.Sequential(*[
            nn.Conv2d(in_channels=64,
                      out_channels=128,
                      kernel_size=kernel_size, padding=padding),
```

```
    nn.BatchNorm2d(128),
    nn.ReLU(inplace=True)])

self.enc_22 = nn.Sequential(*[
    nn.Conv2d(in_channels=128,
              out_channels=128,
              kernel_size=kernel_size, padding=padding),
    nn.BatchNorm2d(128),
    nn.ReLU(inplace=True)
])

self.enc_31 = nn.Sequential(*[
    nn.Conv2d(in_channels=128,
              out_channels=256,
              kernel_size=kernel_size, padding=padding),
    nn.BatchNorm2d(256),
    nn.ReLU(inplace=True)])

self.enc_32 = nn.Sequential(*[
    nn.Conv2d(in_channels=256,
              out_channels=256,
              kernel_size=kernel_size, padding=padding),
    nn.BatchNorm2d(256),
    nn.ReLU(inplace=True),
    nn.Dropout2d(0.5)
])

self.enc_41 = nn.Sequential(*[
    nn.Conv2d(in_channels=256,
              out_channels=512,
              kernel_size=kernel_size, padding=padding),
    nn.BatchNorm2d(512),
    nn.ReLU(inplace=True)])

self.enc_42 = nn.Sequential(*[
    nn.Conv2d(in_channels=512,
              out_channels=512,
              kernel_size=kernel_size, padding=padding),
    nn.BatchNorm2d(512),
    nn.ReLU(inplace=True)
])

self.enc_51 = nn.Sequential(*[
    nn.Conv2d(in_channels=512,
              out_channels=1024,
              kernel_size=kernel_size, padding=padding),
    nn.BatchNorm2d(1024),
    nn.ReLU(inplace=True)
])

self.enc_52 = nn.Sequential(*[
```

```
            nn.Conv2d(in_channels=1024,
                      out_channels=1024,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(1024),
            nn.ReLU(inplace=True),
            nn.Dropout2d(0.5)])


        self.dec_4t = nn.Sequential(*[nn.ConvTranspose2d(in_channels=1024,
out_channels=1024, kernel_size=2, stride=2)])
        self.dec_4u = nn.Sequential(*[nn.Upsample(scale_factor=2, mode='bilinear',
align_corners=True)])

        self.dec_41 = nn.Sequential(*[
            nn.Conv2d(in_channels=1024,
                      out_channels=512,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(512),
            nn.ReLU(inplace=True)
        ])

        self.dec_42 = nn.Sequential(*[
            nn.Conv2d(in_channels=1024,
                      out_channels=512,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(512),
            nn.ReLU(inplace=True),
            nn.Conv2d(in_channels=512,
                      out_channels=512,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(512),
            nn.ReLU(inplace=True),
            nn.Dropout2d(dropout_rate)
        ])

        self.dec_3t = nn.Sequential(*[nn.ConvTranspose2d(in_channels=512,
out_channels=512, kernel_size=2, stride=2)])
        self.dec_3u = nn.Sequential(*[nn.Upsample(scale_factor=2, mode='bilinear',
align_corners=True)])

        self.dec_31 = nn.Sequential(*[
            nn.Conv2d(in_channels=512,
                      out_channels=256,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True)])

        self.dec_32 = nn.Sequential(*[
            nn.Conv2d(in_channels=512,
                      out_channels=256,
                      kernel_size=kernel_size, padding=padding),
```

```
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True),
            nn.Conv2d(in_channels=256,
                      out_channels=256,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True),
            nn.Dropout2d(0.5)
        ])

        self.dec_2t = nn.Sequential(*[nn.ConvTranspose2d(in_channels=256,
out_channels=256, kernel_size=2, stride=2)])
        self.dec_2u = nn.Sequential(*[nn.Upsample(scale_factor=2, mode='bilinear',
align_corners=True)])

        self.dec_21 = nn.Sequential(*[
            nn.Conv2d(in_channels=256,
                      out_channels=128,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(128),
            nn.ReLU(inplace=True)])

        self.dec_22 = nn.Sequential(*[
            nn.Conv2d(in_channels=256,
                      out_channels=128,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(128),
            nn.ReLU(inplace=True),
            nn.Conv2d(in_channels=128,
                      out_channels=128,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(128),
            nn.ReLU(inplace=True),
            nn.Dropout2d(0.5)])

        self.dec_1t = nn.Sequential(*[nn.ConvTranspose2d(in_channels=128,
out_channels=128, kernel_size=2, stride=2)])
        self.dec_1u = nn.Sequential(*[nn.Upsample(scale_factor=2, mode='bilinear',
align_corners=True)])

        self.dec_11 = nn.Sequential(*[
            nn.Conv2d(in_channels=128,
                      out_channels=64,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True)])

        self.dec_12 = nn.Sequential(*[
            nn.Conv2d(in_channels=128,
                      out_channels=64,
                      kernel_size=kernel_size, padding=padding),
```

```
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True),
            nn.Conv2d(in_channels=64,
                      out_channels=64,
                      kernel_size=kernel_size, padding=padding),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True),
            nn.Dropout2d(0.5)])

        self.final = nn.Conv2d(64, 4, kernel_size=1)

        self.init_kaiming_weights()

    def forward(self, x):
        enc11 = self.enc_11(x)
        #print("\nEnc1: ", enc11.shape)
        enc12 = self.enc_12(enc11)
        pool1 = F.max_pool2d(enc12, kernel_size=2, stride=2)

        enc21 = self.enc_21(pool1)
        #print("Enc2: ", enc21.shape)
        enc22 = self.enc_22(enc21)
        pool2 = F.max_pool2d(enc22, kernel_size=2, stride=2)

        enc31 = self.enc_31(pool2)
        #print("Enc3: ", enc31.shape)
        enc32 = self.enc_32(enc31)
        pool3 = F.max_pool2d(enc32, kernel_size=2, stride=2)

        enc41 = self.enc_41(pool3)
        #print("Enc4: ", enc41.shape)
        enc42 = self.enc_42(enc41)
        pool4 = F.max_pool2d(enc42, kernel_size=2, stride=2)

        enc51 = self.enc_51(pool4)
        #print("Enc5: ", enc51.shape)
        enc52 = self.enc_52(enc51)

        dec4t = self.dec_4u(enc52)
        #print("Dec4t: ", dec4t.shape)
        dec41 = self.dec_41(dec4t)
        #print("Dec41: ", dec41.shape)
        dec4c = torch.cat([dec41, enc42], dim=1)
        #print("Dec4c: ", dec4c.shape)
        dec42 = self.dec_42(dec4c)
        #print("Dec42: ", dec42.shape)

        dec3t = self.dec_3u(enc42)
        #print("Dec3t: ", dec3t.shape)
        dec31 = self.dec_31(dec3t)
        #print("Dec31: ", dec31.shape)
```

```
        dec3c = torch.cat([dec31, enc32], dim=1)
        #print("Dec3c: ", dec3c.shape)
        dec32 = self.dec_32(dec3c)
        #print("Dec32: ", dec32.shape)

        dec2t = self.dec_2u(dec32)
        #print("Dec2t: ", dec2t.shape)
        dec21 = self.dec_21(dec2t)
        #print("Dec21: ", dec21.shape)
        dec2c = torch.cat([dec21, enc22], dim=1)
        #print("Dec2c: ", dec2c.shape)
        dec22 = self.dec_22(dec2c)
        #print("Dec22: ", dec22.shape)

        dec1t = self.dec_1u(dec22)
        #print("Dec1t: ", dec1t.shape)
        dec11 = self.dec_11(dec1t)
        #print("Dec11: ", dec11.shape)
        dec1c = torch.cat([dec11, enc12], dim=1)
        #print("Dec1c: ", dec1c.shape)
        dec12 = self.dec_12(dec1c)
        #print("Dec12: ", dec12.shape)

        return self.final(dec12)

    def init_kaiming_weights(self):
        # Encoder 1
        nn.init.kaiming_normal_(self.enc_11[0].weight)
        nn.init.kaiming_normal_(self.enc_12[0].weight)
        # Encoder 2
        nn.init.kaiming_normal_(self.enc_21[0].weight)
        nn.init.kaiming_normal_(self.enc_22[0].weight)
        # Encoder 3
        nn.init.kaiming_normal_(self.enc_31[0].weight)
        nn.init.kaiming_normal_(self.enc_32[0].weight)

        # Encoder 4
        nn.init.kaiming_normal_(self.enc_41[0].weight)
        nn.init.kaiming_normal_(self.enc_42[0].weight)

        # Encoder 5
        nn.init.kaiming_normal_(self.enc_51[0].weight)
        nn.init.kaiming_normal_(self.enc_52[0].weight)

        # Decoder 4
        nn.init.kaiming_normal_(self.dec_41[0].weight)
        nn.init.kaiming_normal_(self.dec_42[0].weight)

        # Decoder 3
        nn.init.kaiming_normal_(self.dec_31[0].weight)
        nn.init.kaiming_normal_(self.dec_32[0].weight)
```

```
        # Decoder 2
        nn.init.kaiming_normal_(self.dec_21[0].weight)
        nn.init.kaiming_normal_(self.dec_22[0].weight)

        # Decoder 1
        nn.init.kaiming_normal_(self.dec_11[0].weight)
        nn.init.kaiming_normal_(self.dec_12[0].weight)

        # Final
        nn.init.kaiming_normal_(self.final.weight)
        import numpy as np

def categorical_dice(mask1, mask2, label_class=1):
    mask1_pos = (mask1 == label_class).astype(np.float32)
    mask2_pos = (mask2 == label_class).astype(np.float32)
    denom = (np.sum(mask1_pos) + np.sum(mask2_pos))
    if(int(denom) == 0):
        return 0
    dice = 2 * np.sum(mask1_pos * mask2_pos) / (np.sum(mask1_pos) +
np.sum(mask2_pos))
    return dice

def dice_class_score(mask1, mask2):
    dice_scores = [
        categorical_dice(mask1, mask2, 0),
        categorical_dice(mask1, mask2, 1),
        categorical_dice(mask1, mask2, 2),
        categorical_dice(mask1, mask2, 3)
    ]
    return dice_scores

def average_dice(mask1, mask2, verbose):
    dice_scores = dice_class_score(mask1, mask2)
    if(verbose):
        for i in range(len(dice_scores)):
            print("=> Class {} = {}".format(i+1,dice_scores[i]))
    total_dice = sum(dice_scores) / len(dice_scores)
    return total_dice


def run_validation(model,dataloader):
    model.eval()
    dices = []
    with torch.no_grad():


        for iteration, sample in enumerate(dataloader):
            img, mask = sample
```

```
            img = img.unsqueeze(1)
            img = img.to(device)
            outputs = model(img)

            mask = mask.type(torch.LongTensor)

            mask_pred = torch.argmax(outputs, dim=1).detach().cpu()

            numsamples = mask.shape[0]
            for i in range(numsamples):
                dice = dice_class_score(mask[i,...].numpy(),
mask_pred[i,...].numpy())
                dices.append(dice)
    return dices

def avg_and_save_validation(filename,dices):
    numclasses = len(dices[0])
    totals = [0 for i in range(numclasses)]
    for scores in dices:
        for i in range(len(scores)):
            totals[i] += scores[i]

    numdices = len(dices)
    classavg_dices = [t / numdices for t in totals]
    dice_score = sum(classavg_dices) / len(classavg_dices)

    stringdices = [str(d) for d in classavg_dices]
    with open(filename,"a") as f:
        line = ", ".join(stringdices)
        print(line)
        f.write(line+"\n")
    return dice_score


def train_eval():
    model = UNet()

    model = UNet()
    epochs = 200
    lr = 0.01
    loss_fn = torch.nn.CrossEntropyLoss()
    optimiser = optim.Adam(model.parameters(), lr=lr)


    device = torch.device("cpu")
    if (torch.cuda.is_available()):
        print("CUDA")
        device = torch.device("cuda")

    model = model.to(device)
    loss_fn = loss_fn.to(device)
```

```
    train_data_path = ‘./data/train’
    validate_data_path = ‘./data/val’
    num_workers = 0
    batch_size = 10
    val_batch_size = 2
    train_set = TrainDataset(train_data_path)
    validate_set = TrainDataset(validate_data_path)
    validation_data_loader = DataLoader(dataset=validate_set,
num_workers=num_workers, batch_size=val_batch_size,
                                        shuffle=True)
    training_data_loader = DataLoader(dataset=train_set, num_workers=num_workers,
batch_size=batch_size,
                                      shuffle=True)


    print(‘=> Training Data Length:’ , len(training_data_loader))
    best_dice = 0
    dice_score = 0

    train_log = []
    validation_log = []

    CSV_HEADER = "class0, class1, class2, class3\n"
    with open("trainlog.csv","w") as f:
            f.write(CSV_HEADER)
    with open("validlog.csv","w") as f:
            f.write(CSV_HEADER)

    for epoch in range(epochs):

        model.train()

        for iteration, sample in enumerate(training_data_loader):


            img, mask = sample


            inp_img = img.view(batch_size, 1, 96, 96)

            inp_img = inp_img.to(device)

            optimiser.zero_grad()

            mask_pred = model(inp_img)

            mask_pred = mask_pred.view(batch_size, 4, 96, 96)


            mask = mask.long()
```

```
            mask = mask.to(device)

            loss = loss_fn(mask_pred, mask)
            loss.backward()
            optimiser.step()


        dices = run_validation(model,training_data_loader)
        print("Epoch:",epoch,"train raw: ",end=‘‘)
        dice_score = avg_and_save_validation("trainlog.csv",dices)
        print("Epoch:",epoch, "training dice:  ", dice_score)



        dices = run_validation(model,validation_data_loader)
        print("Epoch:",epoch,"valid raw: ",end=‘‘)
        dice_score = avg_and_save_validation("validlog.csv",dices)
        print("Epoch:",epoch, "validation dice:", dice_score)


        if(dice_score > best_dice):
            print("=> New Personal Best! {}".format(dice_score))
            print("=> Saving...")
            best_dice = dice_score
            with open("./best_dice.txt","w") as f:
                f.write("avg_dice={}\n".format(best_dice))
                f.write("epoch={}\n".format(epoch))
                f.write("totalepochs={}\n".format(epochs))
                f.write("lr={}\n".format(lr))
                f.write("train_data={}\n".format(train_data_path))
                f.write("validation_data={}\n".format(validate_data_path))
                f.write("train_batch_size={}\n".format(batch_size))
            PATH = ‘./trained_best_dice.pth’.format(epoch)
            torch.save(model.state_dict(), PATH)
            print("=> Saved Personal Best")



        if epoch % 25 == 0 and epoch != 0:

            PATH = ‘./trained_{0}e.pth’.format(epoch)
            torch.save(model.state_dict(), PATH)
train_eval()
```

Модуль інтерфейсу Telegram-боту:

```
logging.basicConfig(
    format=‘%(asctime)s - %(name)s - %(levelname)s - %(message)s’,
    level=logging.INFO
)
```

```
async def start(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await context.bot.send_message(chat_id=update.effective_chat.id,
text="test_start")

async def echo(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await context.bot.send_message(chat_id=update.effective_chat.id,
text='test_exception')

async def rec_image(update: Update, context: ContextTypes.DEFAULT_TYPE):
    file_id = update.message.photo[-1].file_id

    photo1 = await context.bot.get_file(file_id)

    file_name = 'img_{}.png'.format(update.message.message_id)

    path_1 = 'in_image/' + file_name
    path_2 = 'out_image/' + file_name
    await photo1.download_to_drive(path_1)

    image = cv2.imread(path_1, cv2.IMREAD_UNCHANGED)
    num_channels = image.ndim

    if num_channels == 3:
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Preprocess image
    image = cv2.resize(image, (96, 96))
    #image = image.transpose((2, 0, 1)) # convert to channel-first format
    image = image[np.newaxis, ...] # add batch dimension
    image = image[np.newaxis, ...] # add batch dimension

    image = torch.from_numpy(image).to(device).float()

    await context.bot.send_message(chat_id=update.effective_chat.id,
reply_to_message_id=update.message.message_id, text='Зображення серця завантажено,
сегментую')

    # forward
    with torch.no_grad():
        outputs = model(image)

    # convert output to predicted class so it can be visualised
    pred_class = torch.argmax(outputs, dim=1).squeeze().cpu().numpy()
    save_image_mask(image.squeeze().cpu(), pred_class, path_2)

    await context.bot.send_photo(chat_id=update.effective_chat.id,
reply_to_message_id=update.message.message_id, photo=path_2)


def save_image_mask(img, mask, path_2, cmap='gray'): # visualisation
```

```
    fig = plt.figure(figsize=(5,3))
    plt.subplot(1, 2, 1)
    plt.imshow(img, cmap=cmap)
    plt.axis('off')
    plt.subplot(1, 2, 2)
    plt.imshow(mask, cmap=cmap)
    plt.axis('off')
    plt.savefig(path_2)
    plt.close(fig)


if __name__ == '__main__':

    application =
ApplicationBuilder().token('6140598554:AAEkMedQ0rxHSAKMdb03Rv3yaIAzrEpk4F0').build()

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

    model = UNet()
    model.load_state_dict(torch.load("trained_best_dice.pth"))
    model.to(device)
    model.eval() # switch model to evaluation mode

    start_handler = CommandHandler('start', start)
    echo_handler = MessageHandler(filters.TEXT & (~filters.COMMAND), echo)
    image_handler = MessageHandler(filters.PHOTO, rec_image)

    application.add_handler(start_handler)
    application.add_handler(echo_handler)
    application.add_handler(image_handler)
    application.run_polling()
```

Додаток Б

Презентаційний матеріал

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

СЕГМЕНТАЦІЯ ДІЛЯНОК СЕРЦЯ НА ЗОБРАЖЕННЯХ МАГНІТНО-РЕЗОНАНСНОЇ ТОМОГРАФІЇ З ВИКОРИСТАННЯМ ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ

Виконав:
студент 4 курсу, групи КН-19-2
Діхтяр Максим Олександрович

Керівник:
старший викладач кафедри КН
Радюк Павло Михайлович

Актуальність

- Сегментація ділянок серця на зображеннях магнітно-резонансної томографії є важливим завданням в області медичного зображення. Згорткові нейронні мережі (ЗНМ) виявилися потужним інструментом для автоматичної сегментації, оскільки вони можуть виконувати навчання на великих наборах даних і відтворювати складні неоднорідності структури серця.
- Актуальність полягає в тому, що точна сегментація ділянок серця дозволяє лікарям отримувати важливі клінічні показники, такі як об'єм лівого шлуночка, товщина стінок серця та об'єм маси м'язів, що допомагає в діагностиці та виборі оптимального лікування серцевих захворювань. Точна сегментація також може використовуватися для планування хірургічних втручань та оцінки ефективності терапії у пацієнтів.
- Згорткові нейронні мережі демонструють високу точність та швидкість обробки, що робить їх ефективними для автоматичної сегментації зображень магнітно-резонансної томографії серця. Вони можуть виявляти навіть невеликі деталі та неоднорідності, що допомагає у виявленні патологій та аномалій серця.
- Таким чином, використання згорткових нейронних мереж для сегментації ділянок серця на зображеннях магнітно-резонансної томографії має великий потенціал для покращення точності діагностики та планування лікування серцевих захворювань.

3

Мета і задачі роботи

Об’єкт дослідження – процес цифрової сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

Предмет дослідження – методи, засоби та технології цифрової сегментації ділянок серця за зображеннями магнітно-резонансної томографії.

Мета кваліфікаційної роботи бакалавра – підвищення точності цифрової сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

Для досягнення поставленої мети необхідно виконати **наступні завдання**:

1. Провести аналітичний огляд методів, засобів та технологій цифрової сегментації за зображеннями магнітно-резонансної томографії та обрати найкращий.

2. Застосувати обраний метод цифрової сегментації до розв’язання задачі автоматизованої сегментації ділянок серця на зображеннях магнітно-резонансної томографії.

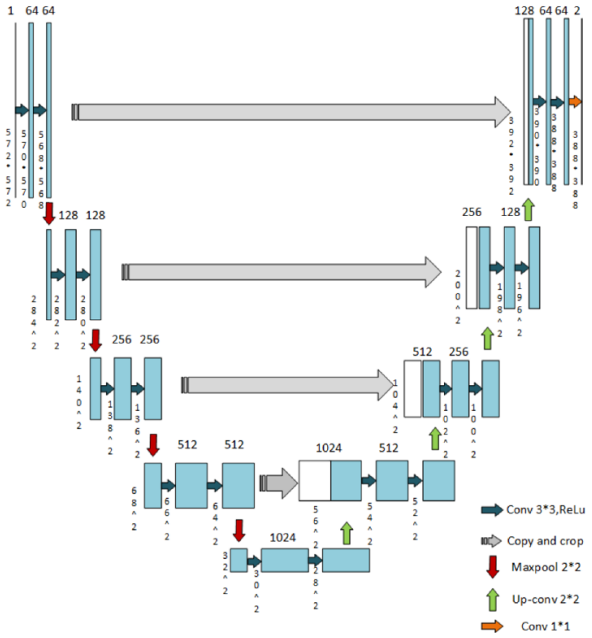
3. Реалізувати обраний метод у вигляді модуля програмного забезпечення для автоматизованої сегментації ділянок серця.

4. Провести експериментальне тестування реалізованого модуля за еталонними наборами даних.

4

Схема способу сегментації ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі

5



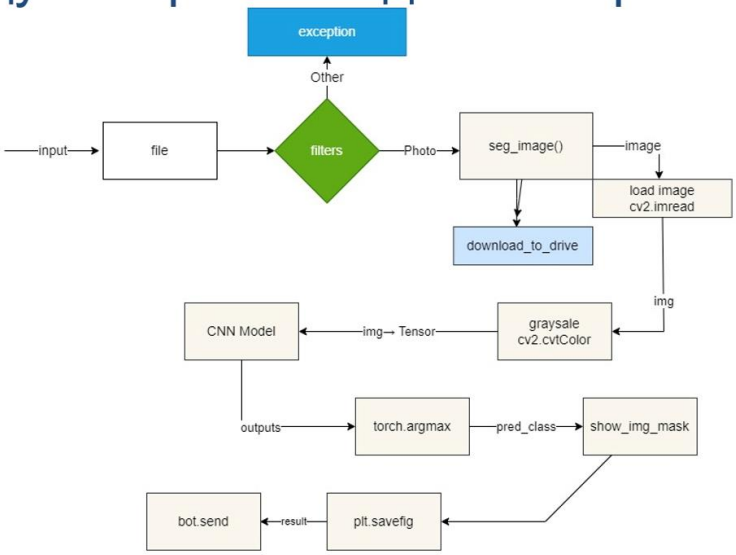
Структура використаної нейронної мережі U-Net

6

Схема функціональної структури інформаційної системи

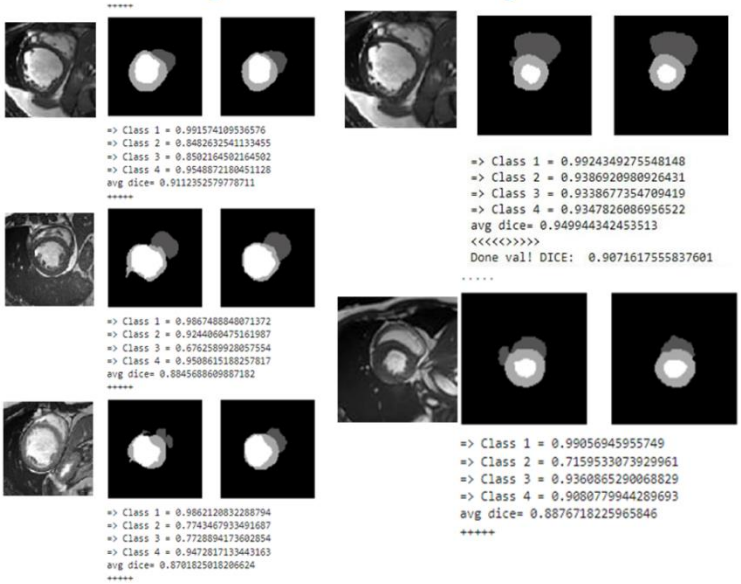
7

Модуль обробки вхідного зображення



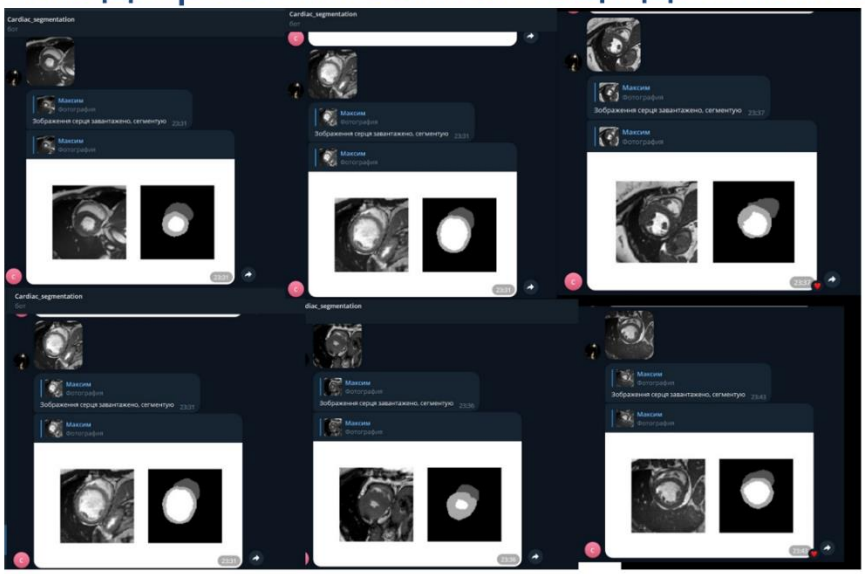
8

Результати тестування



9

Приклади роботи сегментації ділянок серця



10

Приклади роботи сегментації ділянок серця

Висновки

У рамках виконання кваліфікаційної роботи бакалавра було виконано **розробку й апробацію методу** сегментації ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі. Зокрема, було проведено аналіз предметної області й досліджено сучасні підходи до сегментації ділянок серця на зображеннях магнітно-резонансної томографії , розглянуто існуючі програмні реалізації за цим напрямком.

Розроблена **програмна реалізація** сегментації ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі з використанням мови Python виконує наступні основні функції:

- Завантаження та обробку користувацького зображення
- За допомогою моделі U-Net сегментує ділянки серця на зображеннях магнітно-резонансної томографії
- Автоматично відправляє сегментоване зображення користувачу

У якості засобів розробки було обрано платформу PyTorch, мову програмування Python, редактор програмного коду Visual Studio та Anaconda.

Thu Jun 01 19:09:32 EEST 2023, Петровський Сергій Степанович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 3.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 9%**

ID: 114502 Назва: КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА Додано в БД: 2023-06-01 Автора: М.О. Діхтяр Керівники: П.М. Радюк Консультанти: Опоненти:	Документ		Сумарний збіг по Базі Даних	
	Символи	Лексеми	Символи	Лексеми
	62062	945	3075 (5%)	49 (5%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK

Ім'я користувача:
Кафедра КН

Дата перевірки:
01.06.2023 20:01:39 EEST

Дата звіту:
01.06.2023 20:14:48 EEST

ID перевірки:
1015376514

Тип перевірки:
Doc vs Internet + Library

ID користувача:
100005671

Назва документа: **КН-19-2 Діхтяр**

Кількість сторінок: 60 Кількість слів: 10360 Кількість символів: 78238 Розмір файлу: 2.51 MB ID файлу: 1015042204

8.17% Схожість

Найбільша схожість: 3.25% з джерелом з Бібліотеки (ID файлу: 1014975805)

6.31% Джерела з Інтернету	637	Сторінка 62
4.8% Джерела з Бібліотеки	54	Сторінка 65

0.16% Цитат

Цитати	1	Сторінка 66

Не знайдено жодних посилань

0% Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 8 слів та 0%)

0% Вилучення з Інтернету	1	Сторінка 67
0% Вилученого тексту з Бібліотеки	1	Сторінка 67

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ КАФЕДРИ КОМП'ЮТЕРНИХ НАУК
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: *Сегментація ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі*

Автор: *студент групи КН-19-2 Діхтяр Максим Олександрович*

Спеціальність: *122 – Комп'ютерні науки*

Освітня програма: *освітньо-професійна*

Науковий керівник: *док. філ., ст. викл. Радюк П.М.*

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	***відповідає***
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі Діхтяра М.О., не є плагіатом, оскільки: запозичення розміщені в розділі огляду існуючих підходів, не описують безпосередньо авторську роботу і не стосуються її результатів; усі запозичення фрагментарні; до запозичень входять фрагменти програмного коду, що не мають авторства і містять поширені конструкції; поміж запозичень знаходяться загальновідомі терміни та скорочення.

Обсяг запозичень, визначений системами виявлення збігів/ідентичності/схожості, складає:

- за системою Anti-Plagiarism: 3.0%;

- за системою Unicheck: 8.17 %.

Отже, запозичення є допустимими та відносяться до описаних вище і адресуються до першоджерел, що, з урахуванням наведених обґрунтувань, свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ *Павло РАДЮК*

Гарант ОП _____ *Олександр МАЗУРЕЦЬ*

Завідувач кафедри КН _____ *Олександр БАРМАК*

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МОН УКРАЇНИ

Кафедра комп'ютерних наук

ВІДГУК НАУКОВОГО КЕРІВНИКА

на кваліфікаційну роботу бакалавра

студента *гр. КН-19-2 Діхтяра Максима Олександровича*

за темою *Сегментація ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі*

1. Актуальність теми

Автоматизована сегментація ділянок серця дає можливість медичним фахівцям отримувати важливі клінічні показники, що може сприяти більш точному діагностуванню серцево-судинних захворювань. Заразом, точна сегментація серця засобами штучного інтелекту на зображеннях магнітно-резонансної томографії є відкритою проблемою. Відповідно актуальною є задача розроблення нового програмного рішення на основі згорткової нейронної мережі, яке дало б змогу здійснювати сегментування ділянок серця з високим показником точності.

2. Відповідність роботи предметній області Стандарту спеціальності 122 Комп'ютерні науки

Відповідно до стандарту бакалавра вищої освіти України спеціальності 122 – Комп'ютерні науки, описом предметної галузі, об'єктом та предметом вивчення є математичні, інформаційні та імітаційні моделі реальних явищ, об'єктів, систем і процесів та методи й технології отримання, зберігання, обробки, передачі та використання інформації. Метою поданої роботи є підвищення точності цифрової сегментації ділянок серця на зображеннях магнітно-резонансної томографії. Мету роботи досягнуто завдяки використанню методів, засобів та технологій розв'язання теоретичних і прикладних задач, що виникають у процесі проєктування та розроблення інформаційних технологій. Отже, результати виконання кваліфікаційної роботи відповідають стандарту бакалавра спеціальності 122 – Комп'ютерні науки.

3. Професійні та особистісні якості бакалавра

Під час виконання кваліфікаційної роботи студент Діхтяр М.О. проявив себе кваліфікованим фахівцем та дисциплінованим студентом, вчасно виконуючи поставлені перед ним завдання. Студент опанував компетентності та засвоїв результати навчання, що відповідають виконанню освітньо-професійної програми рівня вищої освіти «Бакалавр» за спеціальністю 122 – Комп'ютерні науки.

4. Ступінь самостійності під час виконання кваліфікаційної роботи

Студент особисто одержав результати кваліфікаційної роботи та обґрунтував їхню практичну значущість внаслідок виконання ним усіх поставлених завдань.

5. Ступінь оволодіння методами дослідження

У процесі аналізу способу сегментації ділянок серця на зображеннях магнітно-резонансної томографії і розроблення на його основі інформаційної системи студент Діхтяр М.О. продемонстрував задовільний рівень компетентностей та володіння необхідними інструментами й обладнанням, методами, методиками та технологіями галузі інформаційних технологій.

6. Повнота та якість розкриття теми роботи

Тема роботи повністю обґрунтована й розкрита; актуальність предметної галузі та відомі дослідження щодо обраної тематики проаналізовані достатньо. Студентом успішно виконані усі поставлені перед ним завдання. Розроблене студентом програмне забезпечення для валідації та верифікації інформаційної системи сегментації ділянок серця на основі згорткової нейронної мережі відповідає технічним вимогам спеціальності 122 – Комп'ютерні науки.

7. Логічність, послідовність, аргументованість, літературна грамотність викладення матеріалу

Матеріал кваліфікаційної роботи Діхтяра М.О. подано логічно, послідовно, аргументовано та є таким, що відповідає поставленій меті. Мова і стиль викладення роботи відповідають стандартам, що забезпечує доступність сприймання матеріалу і відповідає вимогам до сучасних кваліфікаційних робіт.

8. Можливість практичного застосування кваліфікаційної роботи бакалавра, окремих її частин

Розроблене в роботі програмне забезпечення може бути використане медичними фахівцями або їхніми асистентами під час діагностування серцево-судинних захворювань за зображеннями магнітно-резонансної томографії в клінічних умовах, наприклад, для надання додаткової інформації про стан серця в режимі реального часу.

9. Висновок про можливість допуску кваліфікаційної роботи бакалавра до захисту, на яку оцінку заслуговує робота

*З огляду на рівень виконання та забезпечення всіх необхідних вимог, вважаю, що кваліфікаційна робота Діхтяра Максима Олександровича може бути допущена до захисту. Рекомендована оцінка – **«задовільно»**.*

Керівник _____ *док. філ., ст. викл. каф. КН Павло РАДЮК*

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МОН УКРАЇНИ

Кафедра комп'ютерних наук

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента *гр. КН-19-2 Діхтяра Максима Олександровича*
за темою: *Сегментація ділянок серця на зображеннях магнітно-резонансної томографії з використанням згорткової нейронної мережі*

1. Актуальність обраної теми

Сегментація ділянок серця на зображеннях магнітно-резонансної томографії (МРТ) є важливим завданням у медичній обробці зображень. Точна сегментація дає змогу отримати інформацію про стан серцево-судинної системи пацієнта, що допомагає в точній діагностиці та плануванні лікування. Використання згорткових нейронних мереж (ЗНМ) слугує відмінним інструментом для обробки зображень та дає можливість досягти високої точності та швидкості обробки зображень, що має велике значення для клінічної практики. Обрана тема є актуальною, оскільки поліпшення точності та швидкості сегментації допоможе покращити діагностику та лікування серцево-судинних захворювань.

2. Повнота розкриття мети та завдань роботи

У процесі виконання кваліфікаційної роботи були детально розкриті мета та завдання. Автором проведено глибокий аналіз предметної галузі сегментації ділянок серця на зображеннях МРТ засобами штучного інтелекту. Проведено систематизацію основних принципів та методів, що використовуються в задачах аналізу медичних зображень. Визначено ключові виклики та труднощі, з якими зустрічаються дослідники в разі сегментації серця на зображеннях МРТ. Загалом, подану роботу можна вважати повною та збалансованою в розкритті мети та завдань.

3. Зміст кожного розділу роботи

У першому розділі кваліфікаційної роботи проведено дослідження процесу цифрової сегментації ділянок серця за зображеннями магнітно-резонансної томографії. Другий розділ присвячений проєктуванню способу сегментації ділянок серця на зображеннях магнітно-резонансної томографії. У третьому розділі наведено та описано програмну реалізацію інформаційної системи з використанням обраного способу сегментації ділянок серця. Насамкінець оформлено висновки до виконаних у роботі завдань.

4. Оцінка розробленої інформаційної системи, її практична цінність

Розроблена інформаційна система для сегментації ділянок серця на зображеннях МРТ із використанням ЗНМ може бути ефективна та мати практичну цінність у медичній галузі. Її точність у сегментації серцевих структур на зображеннях МРТ гарантує отримання детальної інформації про стан серцево-судинної системи пацієнта, що є ключовим для правильної діагностики та планування лікування. Крім того, система відрізняється високою швидкістю обробки зображень, завдяки використанню згорткової нейронної мережі. Практична цінність розробленої інформаційної системи полягає в її

потенціалі покращити діагностику та лікування серцево-судинних захворювань. У підсумку, розроблена інформаційна система для сегментації ділянок серця на зображеннях МРТ із використанням ЗНМ справляє позитивні враження своєю точністю, швидкістю та практичною цінністю.

5. Якість оформлення кваліфікаційної роботи бакалавра

Записка до кваліфікаційної роботи підготовлена автором якісно; вона характеризується логічним структурованим викладом матеріалу та вмістом, що переконливо підтверджує висунуті аргументи. Автор проявив високу грамотність та здатність використовувати влучну та адекватну лексику, що додає роботі вагомості та доречності.

6. Недоліки кваліфікаційної роботи бакалавра

Робота також містить недоліки. Експериментальне тестування із інформаційною системою у роботі розкрито недостатньо. Не розкрито питання зберігання та використання бази завантажених МРТ зображень. У тексті використано кілька абревіатур, але їх не було включено до переліку скорочень. Деякі твердження в роботі потребують посилань на додаткові джерела.

7. Загальний висновок (допускається чи не допускається до захисту), та оцінка на яку заслуговує кваліфікаційна робота.

*З огляду на рівень виконання та забезпечення всіх необхідних вимог, вважаю, що подана кваліфікаційна робота бакалавра може бути допущена до захисту. Рекомендована оцінка – **«добре»**.*

Рецензент _____ Марочник В.