МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
Кафедра інформаційних систем

Магістерська робота

Застосування нейромереж в медичній діагностиці

Виконала: студентка групи ПМіМ-22
спеціальності
122 комп'ютерні науки
(шифр і назва спеціальності)

Гурська Н.Ю
(підпис) (прізвище та ініціали)

Керівник Дреботій Р.Г
(підпис) (прізвище та ініціали)

Рецензент Ящук Ю.О
(підпис) (прізвище та ініціали)

Львів – 2022

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет Прикладна математика та інформатика
Кафедра інформаційних систем
Спеціальність 122 комп'ютерні науки
(шифр і назва)

«ЗАТВЕРДЖУЮ»

Завідувач кафедри Г. Шинкаренко

"06" 09 **2022 року**

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Гурська Надія Юріївна
(прізвище, ім'я, по батькові)

1 .Тема роботи Застосування нейромереж в медичній діагностиці

керівник роботи Дреботій Роман Григорович, доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені Вченою радою факультету від "_____"_____ 20_ року
№ _____

2. Строк подання студентом роботи 12 грудня 2022 року

3. Вихідні дані до роботи Нейронні мережі, способи розпізнавання зображення, мова програмування Python, модель У-нет, бібліотеки PyTorch, MONAI

4. Зміст магістерської роботи (перелік питань, які потрібно розробити)
1. Сегментація зображень.
2. Нейронні мережі.
3. Згорткові нейронні мережі.
4. Архітектура У-нет
5. Розробка нейронної мережі на основі архітектури У-нет для розпізнавання медичних знімків.

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

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 01 вересня 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1.	Постановка задачі магістерської роботи.	10.09.2022	
2.	Ознайомлення із відповідною літературою та технічною документацією.	26.09.2022	
3.	Аналіз технологій для сегментації зображень.	05.10.2022	
4.	Пошук та підготовка набору даних.	20.10.2022	
5.	Розробка нейронної мережі для розпізнавання зображень.	10.11.2022	
6.	Процес навчання та тестування мережі.	20.11.2022	
7.	Аналіз результатів.	30.11.2022	
8.	Підготовка презентації та доповіді.	03.12.2022	
9.	Підготовка матеріалів до друку.	10.12.2022	

Студент _____ Гурська Н.Ю
(підпис) (прізвище та ініціали)

Керівник роботи _____ Дреботій Р.Г
(підпис) (прізвище та ініціали)

ЗМІСТ

ВСТУП

Магістерська робота присвячена розробці нейронної мережі для розпізнавання, а тобто сегментації медичних знімків, яку можна використовувати для медичної діагностики. Медицина, а саме правильна діагностика, яка в загальному відповідає за подальший стан нашого здоров'я є невідкладною частиною нашого життя. Медична діагностика (дав.-гр. δια-γνωστικος — здатний розпізнавати) — комплекс заходів та досліджень, спрямованих на встановлення діагнозу, тобто точної причини захворювання, а також змін внутрішнього середовища організму та супутніх захворювань, та призначення ефективного лікування захворювання. Медичні зображення є основною джерелом інформації про стан організму людини за допомогою яких можна дізнатися про певні порушенні в організмі людини. Для створення таких знімків вже давно використовується велика кількість різноманітних пристрої, а от діагнози про певні порушення визначаються людиною.

Метою цієї роботи є реалізація програми, яка буде розпізнавати медичні знімки і покращить рівень діагностики. Основна відмінність між людським зором і машинним розпізнаванням полягає в тому, що людина розглядає зображення в загальному, а машина це робить більше детально за рахунок проходження і порівняння всіх пікселів, також машина може розділи зображення на потрібну кількість класів і виділити тільки їх, що допоможе сфокусуватися на проблемі.

ОСНОВНА ЧАСТИНА

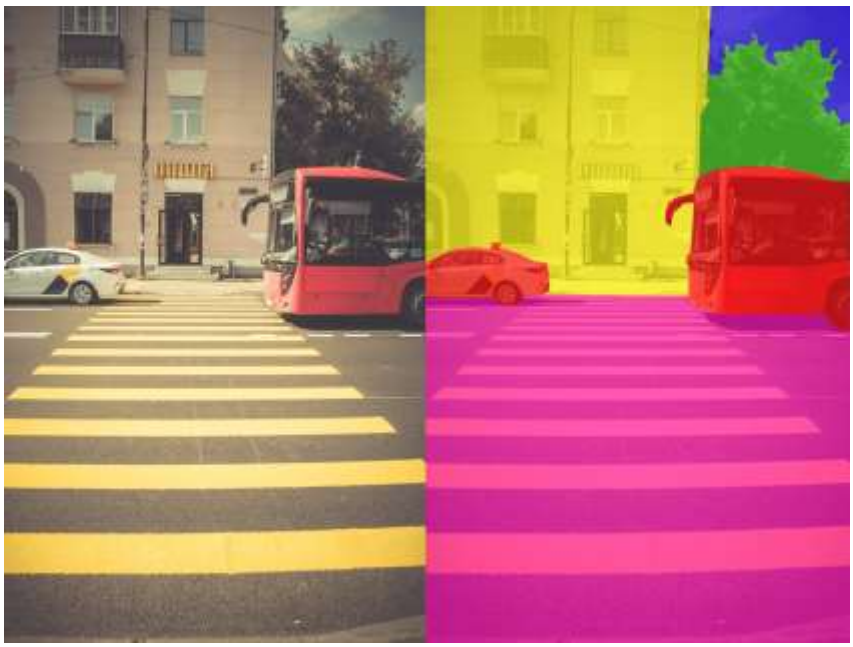
РОЗДІЛ 1. СЕГМЕНТАЦІЯ

Методи сегментації можна розділити на спектральну та просторову сегментацію. Метод спектральної сегментації призначає піксель до області відповідно до спектральної подібності. Він спирається на суто спектральні характеристики (спектральні характеристики відбиття) і розглядає лише один піксель за раз. Метод просторової сегментації класифікує пікселі зображення на основі їхнього просторового співвідношення з пікселями, які їх оточують. Він може враховувати такі аспекти, як текстура зображення, розмір елементів, спрямованість, контекст і повторення. [1.]

Порогове визначення, кластеризація, сегментація нейронних мереж, нечітка сегментація та сегментація на основі країв належать до спектральної сегментації, тоді як зростання області, сегментація на основі текстури та математична морфологія, розбиття та злиття та об'єктний підхід належать до просторової сегментації.[1.]

1.1. Семантична сегментація

Семантична сегментація — це процес присвоєння мітки класу (наприклад, людина, автомобіль або дерево) кожному пікселю зображення. Ви можете думати про це як про класифікацію, але на рівні пікселів – замість того, щоб класифікувати все зображення під однією міткою, ми будемо класифікувати кожен піксель окремо.[2]. Саме цей тип сегментації використаний в роботі. Приклад семантичної сегментації мал. 1.1.1



мал. 1.1.1

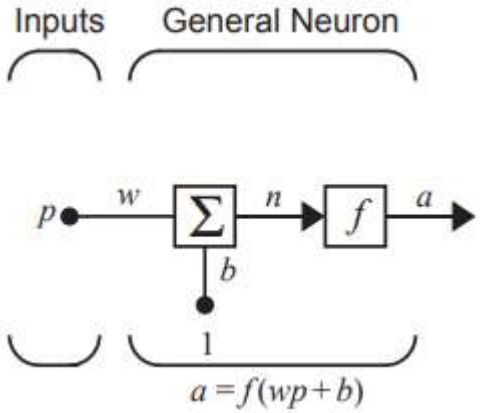
1.2. Висновок

Отже, сегментація спрощує представлення зображення і полегшує його аналіз, за допомогою розбиття зображення на класи та виділення меж.

РОЗДІЛ 2. ЗАГАЛЬНИЙ ОГЛЯД НЕЙРОННИХ МЕРЕЖ

2.1. Нейрон з одним входом

Нейрон з одним входом показаний на малюнку 2.1.1 Скалярний вхід p множиться на скалярну вагу w і тоді wp відправляється в блок суматор. Інший вхід, 1 , множиться на зміщення b, а потім передається в блок суматор. Вихід з блоку суматора n, який часто називають чистим входом, йде у активаційну функцію f, яка створює вихід скалярного нейрона a.[3] Якщо зв'язати цю просту модель з біологічним нейроном, то вага відповідає силі синапсу, тіло клітини представлено функцією підсумовування та передачі, а вихід нейрона представляє сигнал на аксоні.



Одношаровий нейрон мал. 2.1.1

Вихід з нейрона обчислюється наступним чином:

$$a = f(wp + b)$$

Якщо наприклад w= 3, p=2, а b= -1.5, тоді

$$a = f(3 * 2 - 1.5) = f(4.5)$$

Фактичний вихід залежить від обраної конкретної функції передачі. Ми обговоримо функції передачі в наступному розділі.

Зміщення дуже схоже на ваги, за винятком того, що воно має постійне значення 1. Однак, зміщення в певному нейроні не потрібне, його можна опустити.

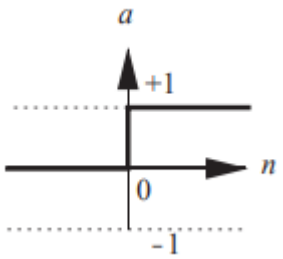
Слід зауважити, що w і b є регульованими скалярними параметрами нейрона. Зазвичай активаційна функція вибирається розробником, а потім параметри w і b

будуть налаштовані деяким правилом навчання так, щоб взаємозв'язок входу/виходу нейрона відповідав певній конкретній меті.

2.2 Активаційні функції

Активаційна функція на малюнку 2.2.1 може бути лінійною або нелінійною функцією від *n*. Конкретна функція передачі вибирається, щоб задовольнити певну специфікацію проблеми, яку нейрон намагається вирішити.[3]

Приклад активаційної функції, а саме функції одиничного стрибка, показана зліва на малюнку 2.1.2, встановлює вихід нейрона на 0, якщо аргумент функції менший за 0, або на 1, якщо її аргумент більший або дорівнює 0. Ця функція буде використовуватися, щоб створити нейрони, які класифікують вхідні дані на дві різні категорії.

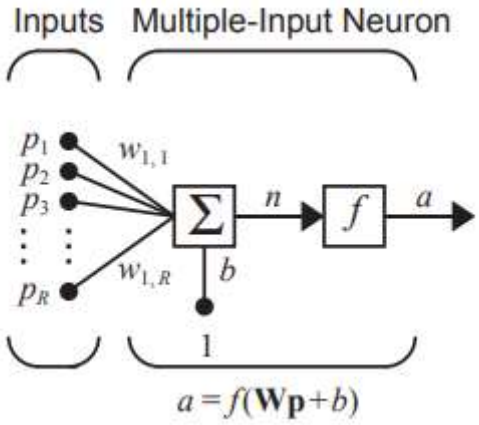


Мал.2.1.1

$$a = \begin{cases} 1, & \text{якщо } x \geq 0 \\ 0, & \text{якщо } x < 0 \end{cases}$$

2.3. Нейрон з багатьма входами

Як правило, нейрон має більше одного входу. Нейрон із R-входами показаний на малюнку 2.3.1. Кожен з окремих вхідних даних зважений відповідними елементами вагової матриці



мал 2.3.1

Нейрон має зсув b, який підсумовується зі зваженими вхідними сигналами, щоб сформувати чистий вхідний сигнал n:

$$n = w_{1,1}\, p_1 + w_{1,2}\, p_2 + \cdots + w_{1,R} p_R + b$$

Цей вираз можна записати у матричній формі:

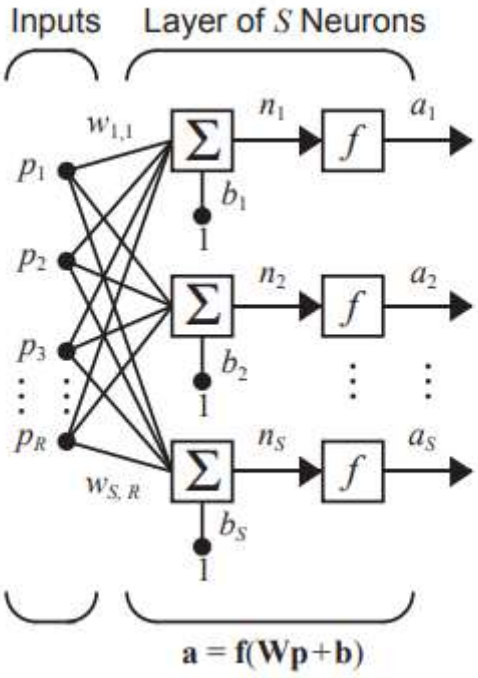
$$n = Wp + b$$

де W матриця для кожного нейрона має лише один рядок. Тепер вихід нейрона можна записати як

$$a = f(Wp + b)$$

2.4. Одношарова нейронна мережа

Одношарова мережа з *S*-нейронів показана на малюнку 2.4.1. Потрібно зауважити, що кожен із входів *R* підключено до кожного з нейронів і що вагова матриця тепер має *S* рядків.



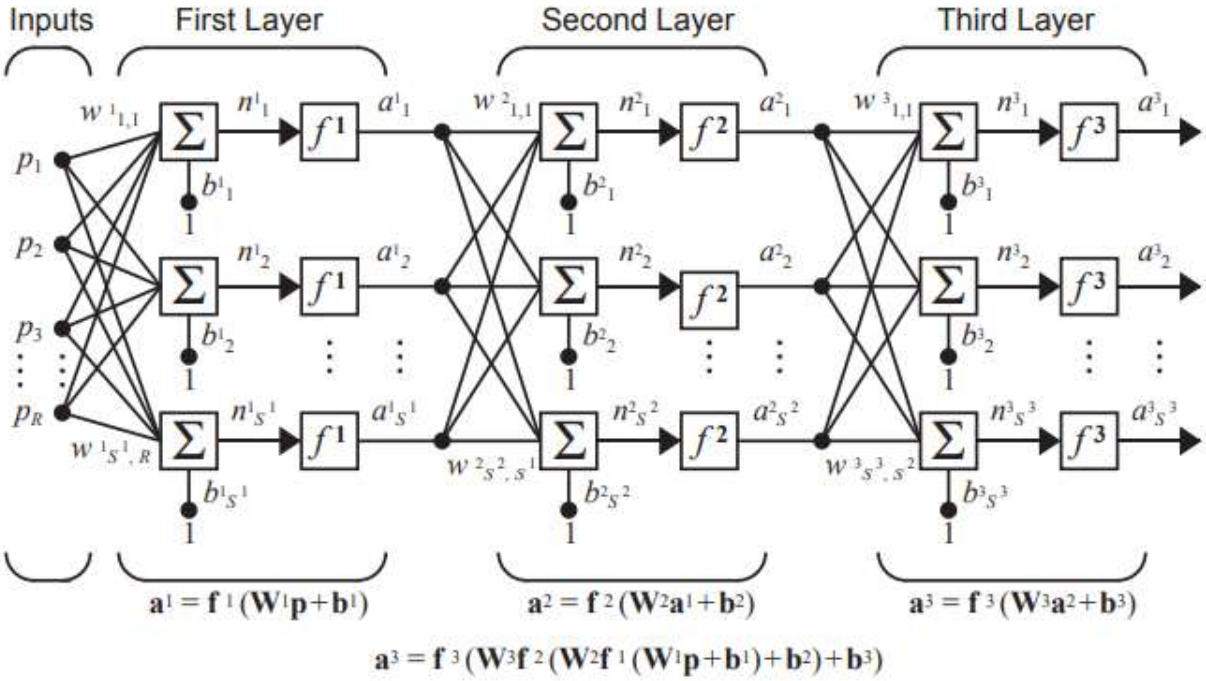
мал 2.4.1

Шар містить вагову матрицю, суми, вектор зміщення b, блоки активаційної функції та вихідний вектор a.

Кожен елемент вхідного вектора p з'єднаний з кожним нейроном через вагову матрицю W. Кожен нейрон має зміщення b_s, літо, функцію передачі f і вихід a_s. Взяті разом, виходи утворюють вектор виходу a.

2.5. Багатошарова нейронна мережа

Тепер розглянемо мережу з кількома шарами. Кожен шар має власну вагову матрицю W, власний вектор зміщення b, чистий вхідний вектор n і вихідний вектор a. Нам потрібно ввести деякі додаткові позначення, щоб розрізняти ці шари. Ми будемо використовувати верхні індекси для визначення шарів. Зокрема, ми додаємо номер шару як верхній індекс до імен для кожної з цих змінних. Таким чином, вагова матриця для першого шару записується як W^1, а вагова матриця для другого шару записується як W^2.[3.] Це позначення використовується в трирівневій мережі, показаній на малюнку 2.5.1



мал 2.5.1

2.6. Висновок

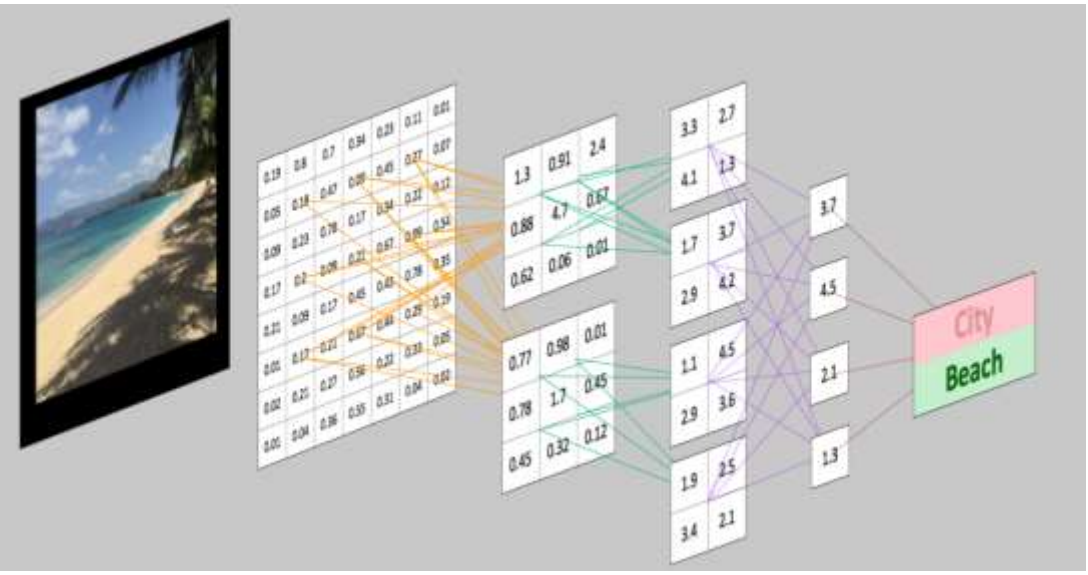
Отже, нейронні мережі є системою з'єднаних та взаємопов'язаних штучних нейронів, які отримують та відправляють певні сигнали і якщо об'єднати велику кількість таких нейронів в одну мережу, то ці нейрони разом зможуть розв'язувати складні задачі.

РОЗДІЛ 3. ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ

Метою згорткових нейронних мереж є вивчення особливостей вищого порядку в даних за допомогою згорток. Вони добре підходять для розпізнавання об’єктів із зображеннями. Вони можуть ідентифікувати обличчя, людей, вуличні знаки, качкодзьобів і багато інших аспектів візуальних даних. Згорткові нейронні мережі збігаються з аналізом тексту за допомогою оптичного розпізнавання символів, але вони також корисні при аналізі слів як окремих текстових одиниць. Вони також добре аналізують звук.

Ефективність згорткових нейронних мереж у розпізнаванні зображень є однією з головних причин, чому світ визнає силу глибокого навчання. Як показано на малюнку 3.1, згорткові нейронні мережі добре створюють незмінні функції положення та обертання з необроблених даних зображення.

Згорткові нейронні мережі сприяють значному розвитку машинного зору, який має очевидне застосування для безпілотних автомобілів, робототехніки, дронів і лікування людей із вадами зору. Згортка є потужною концепцією, яка допомагає створити більш надійний простір функцій на основі сигналу.

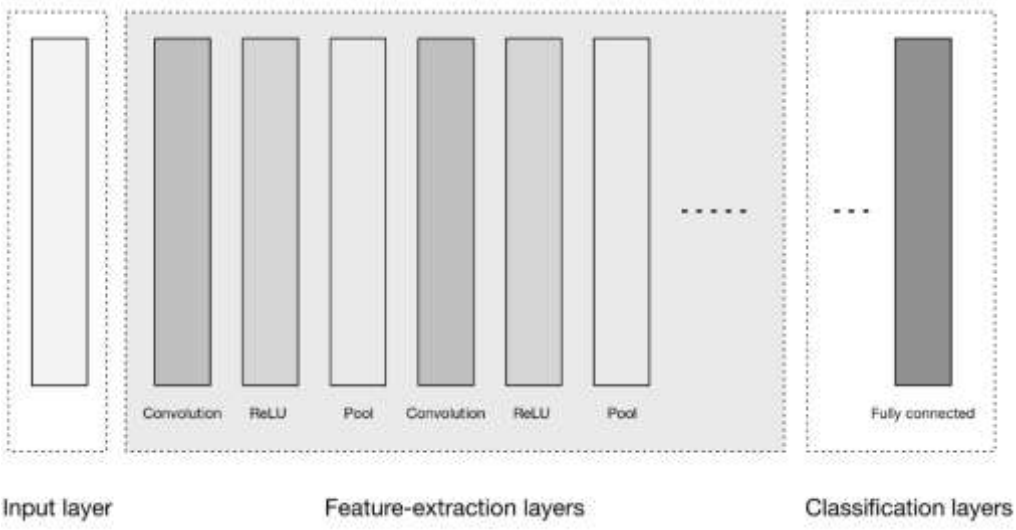


Мал 3.1

Біологічною основою згорткових нейронних мереж є зорова кора у тварин. Клітини зорової кори чутливі до невеликих субобластей вхідного сигналу. Ми називаємо це поле зору (або рецептивне поле). Ці менші підобласті об'єднані разом, щоб охопити все поле зору. Клітини добре підходять для використання сильної просторово-локальної кореляції, виявленої в типах зображень, які обробляє наш мозок, і діють як локальні фільтри над вхідним простором. У цій області мозку є два класи клітин. Прості клітини активуються, коли вони виявляють краєподібні візерунки, а більш складні клітини активуються, коли вони мають більше сприйнятливе поле та незмінні до положення візерунка.[4]

Багатошарові нейронні мережі прямого зв'язку приймають вхідні дані як єдиний одновимірний вектор і перетворюють дані за допомогою одного або кількох прихованих шарів (повністю зв'язаних). Потім мережа видає результат із вихідного рівня. Проблема, з якою ми стикаємося з традиційними багатошаровими нейронними мережами та даними зображень, полягає в тому, що ці мережі погано масштабуються за допомогою даних зображень як вхідних даних.

Згорткові нейронні мережі перетворюють вхідні дані з вхідного рівня через усі підключені рівні в набір оцінок класу, наданих вихідним рівнем. Існує багато варіантів архітектури згорткових нейронних мереж, але вони базуються на шаблоні шарів, як показано на малюнку 3.2.



Мал 3.2

На малюнку 3.2 показано три основні групи:

- Вхідний шар.
- Шари виділення функцій (навчання).
- Класифікаційні шари

Вхідний шар приймає тривимірний вхід, як правило, у формі просторового розміру (ширина × висота) зображення та має глибину, що представляє колірні канали (зазвичай три для колірних каналів RGB).

Шари виділення функцій мають загальний повторюваний шаблон послідовності:

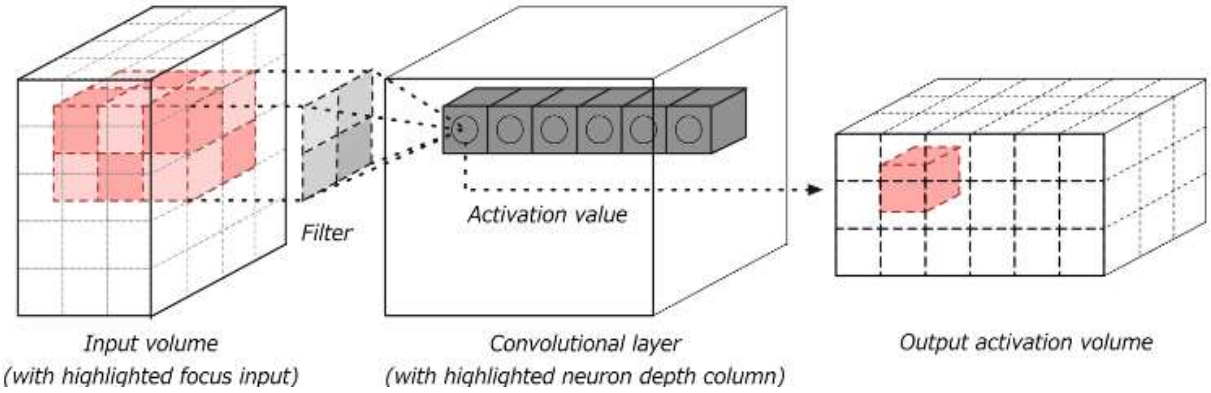
- Шар згортки. Ми представляємо як функцію активації Rectified Linear Unit (ReLU) як рівень на діаграмі тут, щоб відповідати іншій літературі.
- Шар об'єднання. Ці шари знаходять ряд функцій на зображеннях і поступово створюють елементи вищого порядку. Це безпосередньо відповідає актуальній темі глибокого навчання, за допомогою якої функції навчаються автоматично на відміну від традиційного ручного проектування.

Класифікаційні шари, в яких ми маємо один або кілька повністю пов’язаних шарів, щоб взяти ознаки вищого порядку та отримати класові ймовірності або оцінки. Ці шари повністю пов’язані з усіма нейронами попереднього шару, як випливає з їх назви. Вихідні дані цих шарів зазвичай дають двовимірний вихід розмірами [b × N], де b — це кількість прикладів у міні-пакеті, а N — це кількість класів, які ми зацікавлені оцінити.[4]

Тепер детальніше розглянемо вище згадані шари і розпочнемо з вхідних шарів. Вхідні шари – це місце, де ми завантажуємо та зберігаємо необроблені вхідні дані зображення для обробки в мережі. Ці вхідні дані визначають ширину, висоту та кількість каналів. Як правило, кількість каналів становить три для значень RGB для кожного пікселя.

Згорткові шари вважаються основними будівельними блоками архітектури згорткових нейронних мереже. Як показано на малюнку 3.3, згорткові шари перетворюють вхідні дані за допомогою фрагмента локально з’єднаних нейронів із попереднього шару. Рівень обчислить скалярний добуток між областю нейронів у

вхідному шарі та ваговими коефіцієнтами, до яких вони локально підключені у вихідному шарі.

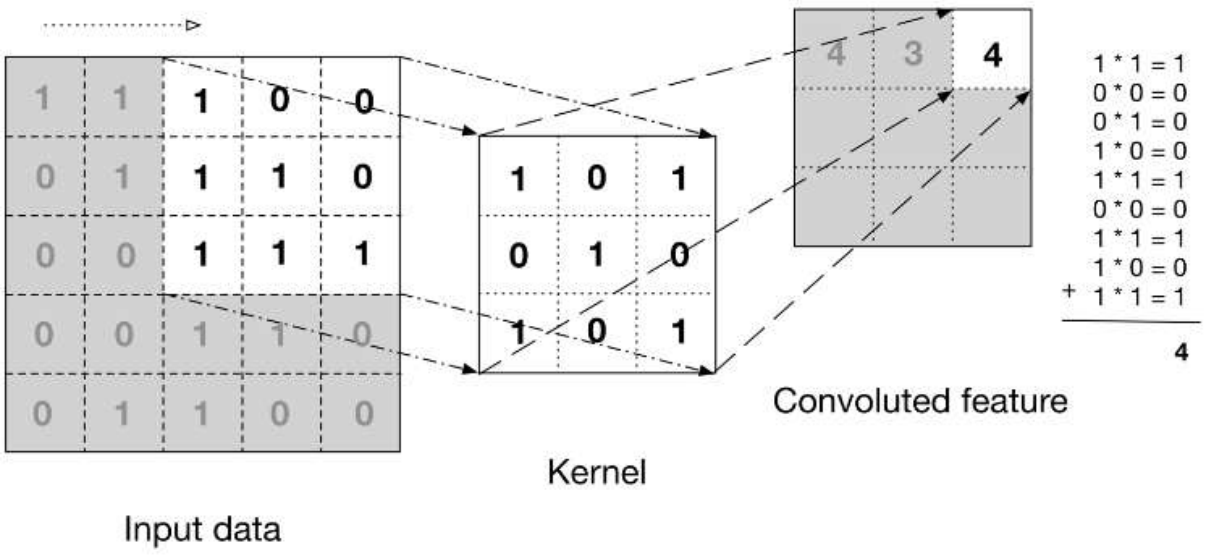


Мал 3.3

Отриманий результат зазвичай має ті самі просторові розміри (або менші просторові розміри), але іноді збільшується кількість елементів у третьому вимірі результату (вимірі глибини). Давайте ближче розглянемо ключове поняття в цих шарах, яке називається згорткою.

Згортка визначається як математична операція, що описує правило, як об'єднати два набори інформації. Це важливо як у фізиці, так і в математиці та визначає міст між просторово-часовою областю та частотною областю за допомогою перетворень Фур'є. Він приймає вхідні дані, застосовує ядро згортки та дає нам карту функцій як вихід.

Операція згортки, показана на малюнку 3.4, відома як детектор ознак згорткової нейронної мережі. Вхідними даними для згортки можуть бути необроблені дані або вихід карти ознак з іншої згортки. Його часто інтерпретують як фільтр, у якому ядро фільтрує вхідні дані для певних видів інформації; наприклад, крайове ядро дозволяє пропускати лише інформацію з краю зображення.



Мал 3.4

На малюнку показано, як ядро ковзає по вхідних даних, щоб отримати заплутані характеристики (вихідні) дані. На кожному кроці ядро множиться на значення вхідних даних у його межах, створюючи єдиний запис у вихідній карті функцій. На практиці результат буде великим, якщо функція, яку ми шукаємо, виявлена у вхідних даних.

Набори ваг у згортковому шарі називаються фільтром (або ядром). Цей фільтр згортається разом із вхідними даними, а результатом є карта функцій (або карта активації). Згорткові шари виконують перетворення обсягу вхідних даних, які є функцією активацій у вхідному обсязі та параметрів (ваги та зміщення нейронів). Карта активації для кожного фільтра складається разом уздовж вимірювання глибини для побудови об’єму 3D-виходу.

Згорткові шари мають параметри для шару та додаткові гіперпараметри. Градієнтний спуск використовується для навчання параметрів у цьому шарі, щоб оцінки класів узгоджувалися з мітками в навчальному наборі. Нижче наведено основні компоненти згорткових шарів:

- Фільтри.
- Карти активації.
- Спільне використання параметрів.

- Специфічні для шару гіперпараметри.

Параметри для згорткового шару налаштовують набір фільтрів шару. Фільтри — це функція, ширина та висота якої менші за ширину та висоту вхідного обсягу.

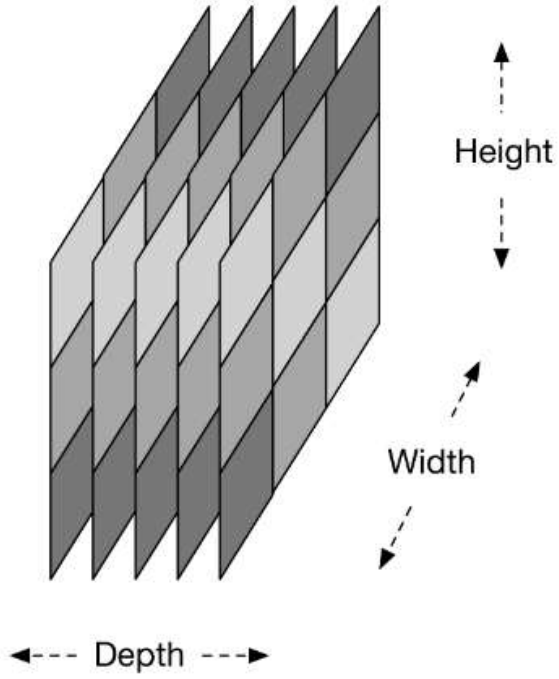
Фільтри (наприклад, згортки) застосовуються по ширині та висоті вхідного об’єму у вигляді ковзаючого вікна, як показано на малюнку 3.4. Фільтри також застосовуються для кожної глибини вхідного обсягу. Ми обчислюємо вихід фільтра, створюючи скалярний добуток фільтра на вхідну область.

Коли ми говоримо, що фільтр «активується», ми маємо на увазі, що фільтр пропускає інформацію з вхідного об’єму до вихідного об’єму.

Ми пересуваємо кожен фільтр по просторових розмірах (ширина, висота) вхідного обсягу під час прямого проходу інформації через згорткову нейронну мережу. Це створює двовимірний результат, який називається картою активації для цього конкретного фільтра.

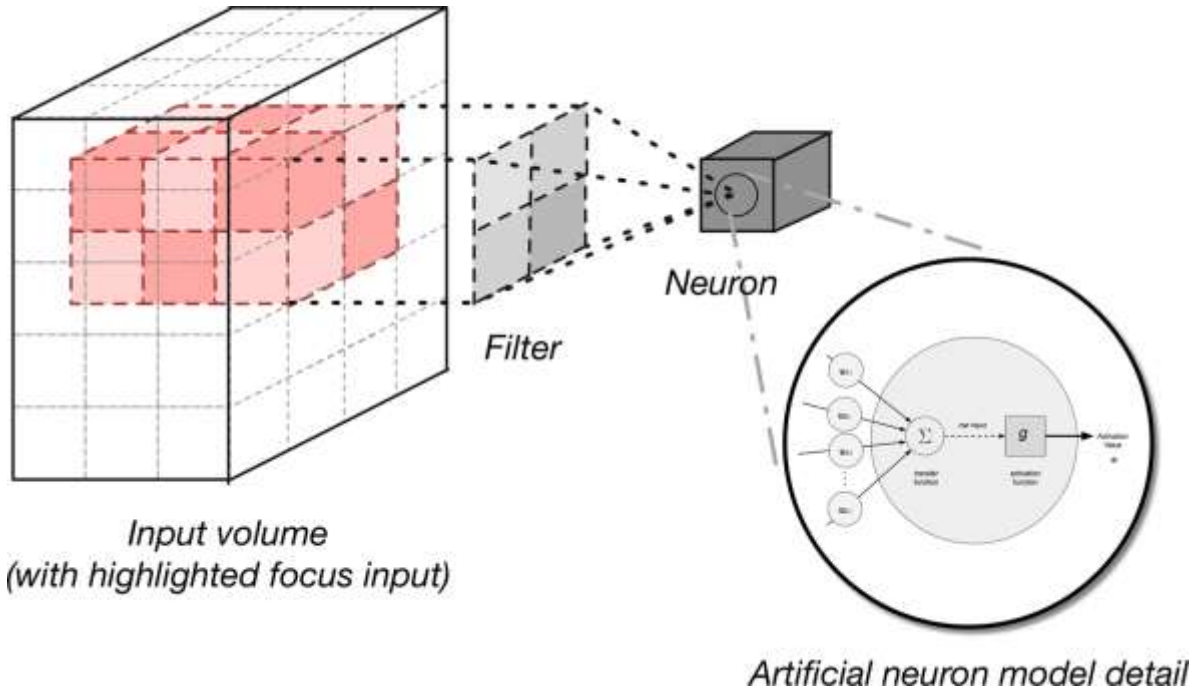
Щоб обчислити карту активації, ми проводимо фільтр по зрізу глибини вхідного обсягу. Ми обчислюємо скалярний добуток між записами у фільтрі та вхідним об’ємом. Фільтр представляє ваги, які множаться на рухоме вікно (підмножину) активацій введення. Мережі вивчають фільтри, які активуються, коли вони бачать певні типи функцій у вхідних даних у певному просторовому положенні.

Ми створюємо тривимірний вихідний об’єм для шару згортки, складаючи ці карти активації вздовж вимірювання глибини у виводі, як показано на малюнку 3.5. Вихідний об’єм матиме записи, які ми вважаємо виходом нейрона, які переглядають лише невелике вікно вхідного об’єму.



Мал 3.5

У деяких випадках цей результат буде результатом параметрів, спільних з нейронами в одній карті активації. Кожен нейрон, що генерує вихідний об’єм, з’єднаний лише з локальною областю вхідного об’єму, як показано на малюнку 3.6.



Мал 3.6

Ми контролюємо локальну зв’язність цього процесу за допомогою гіперпараметра, який називається сприйнятливим полем, який контролює, яку частину ширини та висоти вхідного об’єму відповідає наш фільтр.

Фільтри визначають меншу обмежену область для створення карт активації з вхідних томів. Вони підключені лише до підмножини вхідного обсягу через динаміку локального зв’язку. Це дозволяє нам як і раніше мати якісне вилучення функцій, одночасно зменшуючи кількість параметрів на шар, які нам потрібно навчати. Згорткові шари ще більше зменшують кількість параметрів за допомогою техніки, яка називається спільним використанням параметрів.

Згорткові нейронні мережі використовують схему спільного використання параметрів для контролю загальної кількості параметрів. Це допомагає пришвидшити час на навчання, оскільки ми будемо використовувати менше ресурсів для вивчення набору навчальних даних. Щоб реалізувати спільний доступ до параметрів у згортковій нейронній мережі, ми спочатку позначаємо один двовимірний зріз глибини як «зріз глибини». Потім ми обмежуємо нейрони в кожному зрізі глибини, щоб використовувати однакові ваги та зміщення. Це дає нам значно менше параметрів (або ваг) для заданого згорткового шару.

Висновок

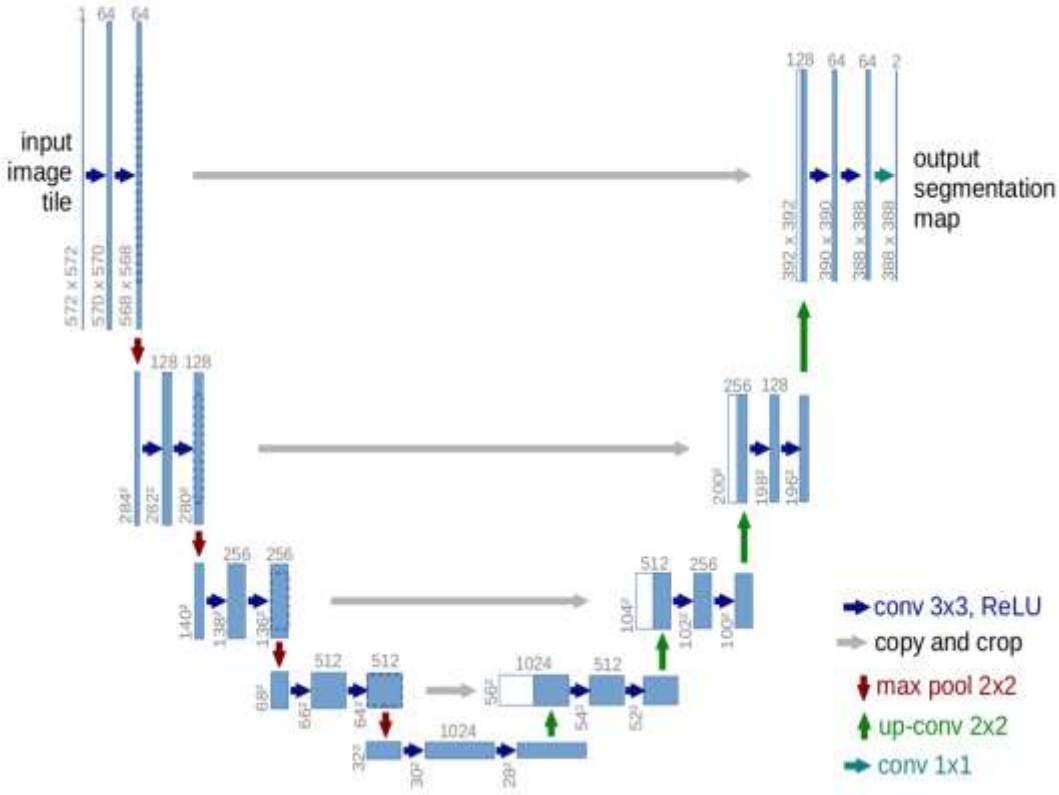
Отже, згорткова нейронна мережа, це така мережа, що використовує багатошарові перспетрони і містить один або більше згорткових шарів, які можуть бути повністю об’єднані. Ці згорткові шари створюють карти функцій, які записують область зображення, яке зрештою розбивається на прямокутники та надсилається для нелінійної обробки.

РОЗДІЛ 4. АРХІТЕКТУРА У-НЕТ

У-нет є однією з стандартних архітектур згорткових нейронних мереж для задач сегментації в яких треба сегментувати області зображення по класу, тобто потрібно створити маску, яка буде розділяти зображення на декілька класів.

Для У-нет характерно: досягнення високих результатів у різних реальних завданнях, особливо для біомедичних додатків, використання невеликої кількості даних для досягнення добрих результатів.

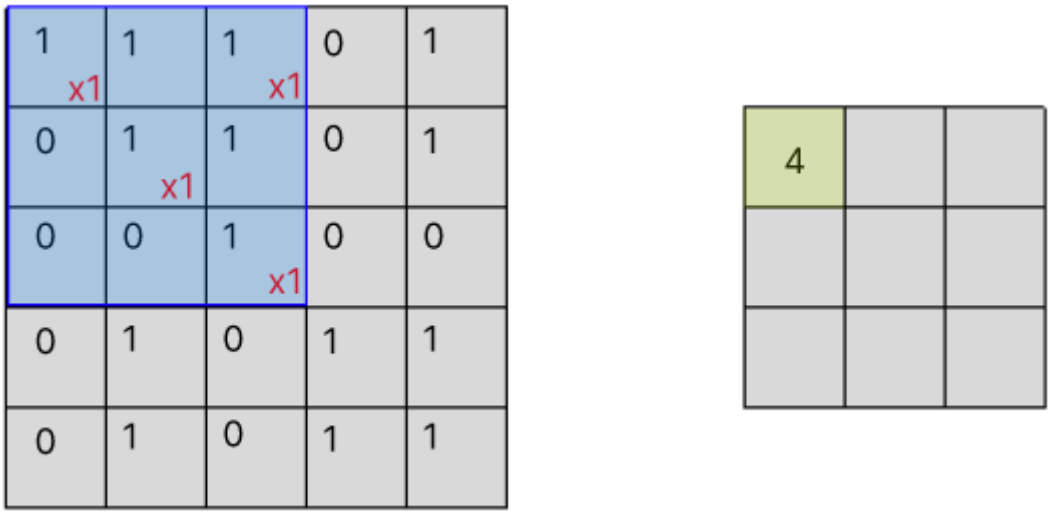
Розгляне архітектуру У-нет детальніше



Мал 4.1

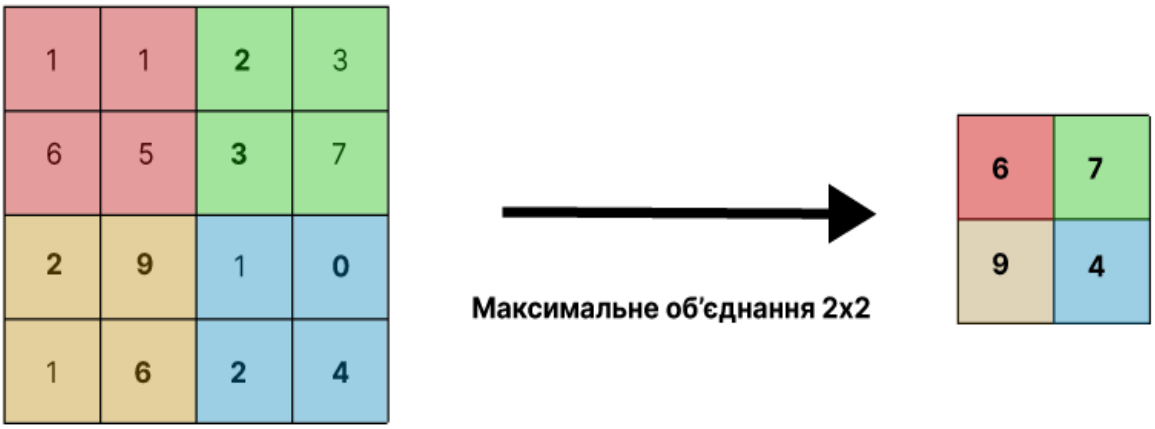
На перший погляд, вона має U-подібну форму. Архітектура є симетричною і складається з двох основних частин — ліва частина називається стискаючим шляхом, який складається із загального згорткового процесу; права частина – це розширений шлях, який складається з транспонованих 2d згорткових шарів.

Отже ліва частина складається з повторного використання двох згорток 3x3(мал 4.2.).



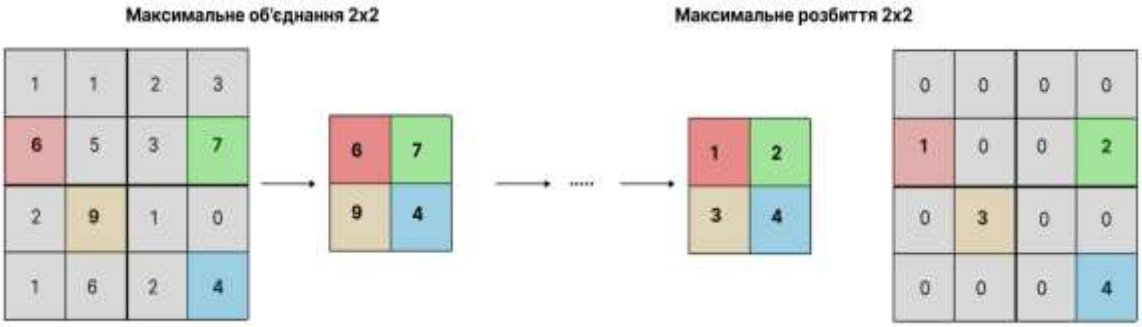
Мал 4.2.

і операції максимального об'єднання 2x2(мал 4.3.), яка використовується для зниження розширення зображення.



Мал 4.3.

Для правої частини берем останнє значення з лівої частини і застосовуємо до нього максимальне розбиття 2x2 (мал4.3) пізніше до вихідного зображення додаємо вихідне зображення з відповідно шару з лівої частини і застосовуємо до нього згортку 3x3 (4.1). Для останнього виходу потрібно застосувати згортку 1x1.



Мал 4.4.

Для навчання У-нет використовується метод стохастичний градієнтний спуск на основі вхідних зображень та відповідних їм карт сегментації. Через згортки вихідне зображення менше вхідного.

Висновок

Отже, У-нет мережа складається із звужувального шляху та розширювального шляху і є типовою згортковою мережею, яка складається з багаторазового використання застосування згорток, за кожною з яких йде випрямлена лінійна одиниця (ReLu) і операція максимального об’єднання.

РОЗДІЛ 5. ТЕХНОЛОГІЇ

Для реалізації даної програми було використано середовище розробки Visual studio code. Мовою реалізації даної програми є Python. В процесі реалізації даної роботи було використані наступні бібліотеки:

- PyTorch — відкрита бібліотека машинного навчання на основі бібліотеки Torch, що використовують для таких застосувань, як комп'ютерне бачення та обробка природної мови. Розробляє її переважно група дослідження штучного інтелекту компанії Facebook. Вона є вільним та відкритим програмним забезпеченням, що випускають під ліцензією Modified BSD.
- MONAI — це фреймворк з відкритим кодом, створений проектом MONAI. MONAI є доступним безкоштовно фреймворком, який підтримується спільнотою, заснований на PyTorch фреймворку для глибокого навчання зображень у сфері охорони здоров’я. Він надає оптимізовані для домену базові можливості для розробки робочих процесів навчання зображень у сфері охорони здоров’я.

РОЗДІЛ 6. РЕАЛІЗАЦІЯ

Для успішної реалізації було виконано наступні кроки:

6.1. Пошук набору даних

Дані для навчання було взято набір даних з відкритої бази даних Medical Segmentation Decathlon[5]

6.2. Підготовка даних

Цей етап можна розбити на дві частини. Першим етапом було розбиття даних на групи. Перетворення розширення файлів з dicom на nii. Перевірка даних на пусті файли. Для цього були додані функції в яких були виконані дані команди.

Під час другого етапу підготовки даних відбулися наступні перетворення даних.

Розбиття зображення на канали. Зображення було розбито на 2 класи, а саме бекграунд та об’єкт в якому зацікавленні.

Пізніше узагальнили всі розширення зображення, а саме висоту ширину та глибину зображення, адже якщо зображення будуть з різними розширеннями, тоді вихідні дану будуть не точними.

Встановили однакову орієнтацію для всіх зображень.

На наступному етапі налаштували контраст зображення, а також змінили вокселі зображень.

Пізніше обрізали пусті місця на зображеннях.

Останнім етапом перетворень було перетворення зображень в тензор, щоб можна було з легкістю проводити обчислення.

Під час підготовки даних було додано кешування, яке допомогло пришвидшити процес навчання нейронної мережі.

6.3. Навчання мережі

Для навчання нейронної мережі було використано тренувальну вибірку, яка складалася з 100 пацієнтів. Під час навчання відбувався прохід по всіх пацієнтах і після кожного пацієнта відбувалося обчислення помилки. Ці обчислення потрібні, щоб зрозуміти чи добре навчається мережа, чим блище значення до 0, тим кращий результат можна отримати. Для цього обчислення було використано наступну формулу

$$dice\ loss = 1 - \frac{y \cap y_{pred}}{y + y_{pred}}, \text{ де } y - \text{реальне значення}$$

$$y_{pred} - \text{передбачене значення}$$

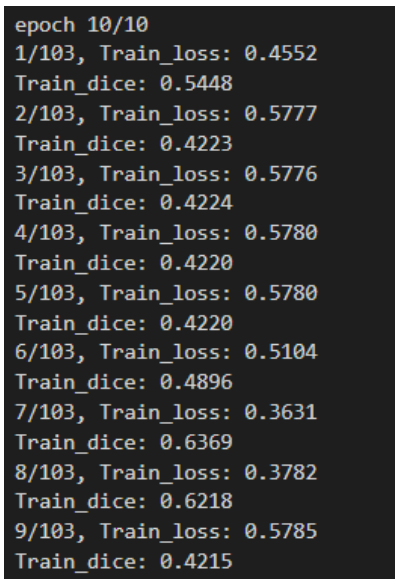
Пізніше результати з епохи з найменшою помилкою використовувалися для тестування мережі.

6.4. Тестування мережі

Для тестування була використані зрізи медичних знімків, які не використовувалися в тренуванні мережі. Для тестування були виккористана мережа, яка повертала найменшу помилку.

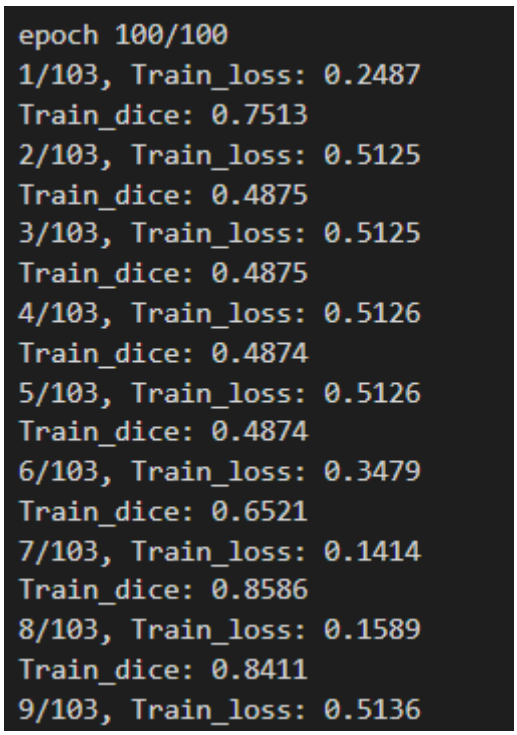
6.5. Результати

Спочатку відбулася ініціалізація навчання нейронної мережі. Навчання запускалося декілька разів з різною кількістю епох для порівняння обчислювання і як змінюються значення. На малюнку 6.5.1. можна побачити приклад з виведенням помилки для кожного пацієнта при 10 епохах.



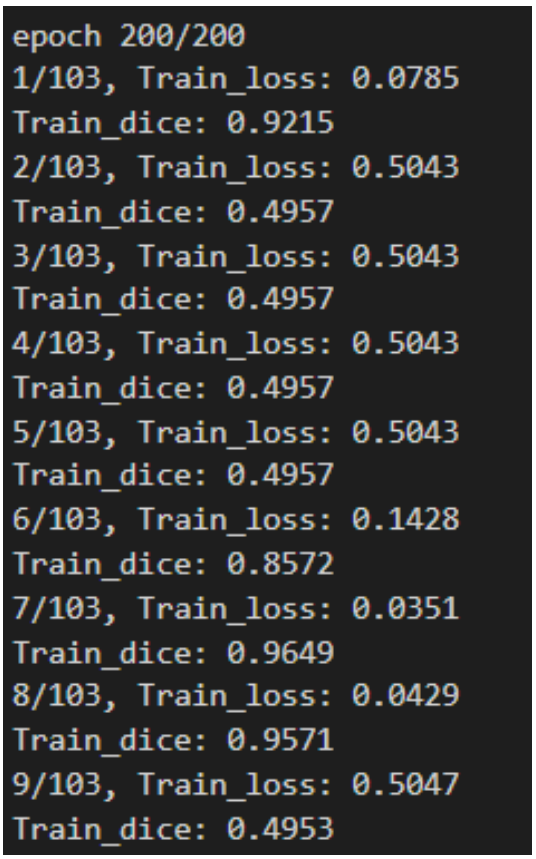
Мал. 6.5.1.

На малюнку 6.5.2. можна побачити приклад з виведенням помилки для кожного пацієнта при 100 епохах.



Мал. 6.5.2.

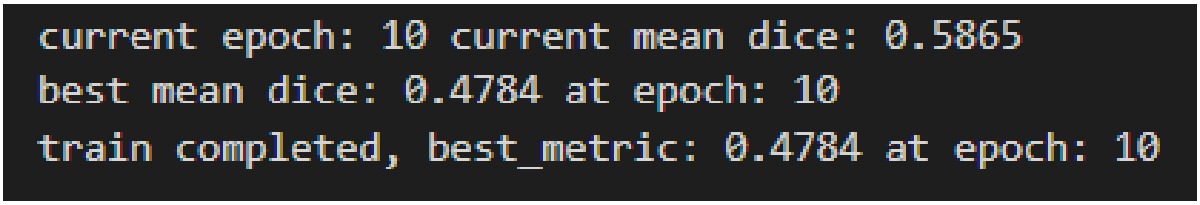
На малюнку 6.5.3. можна побачити приклад з виведенням помилки для кожного пацієнта при 200 епохах.



Мал 6.5.3.

Після проходження кожної епохи відбувалося порівняння помилки поточної епохи з попередніми і якщо помилка поточної була більшою найменше значення з попередніх обчислень, тоді записувалося значення найменшої помилки.

На малюнку 6.5.4. можна побачити найкраще значення помилки для 10 епох.



Мал. 6.5.4.

На малюнку 6.5.5. можна побачити найкраще значення помилки для 100 епох.

```
current epoch: 100 current mean dice: 0.7449
best mean dice: 0.5711 at epoch: 97
train completed, best_metric: 0.5711 at epoch: 97
```

Мал. 6.5.5.

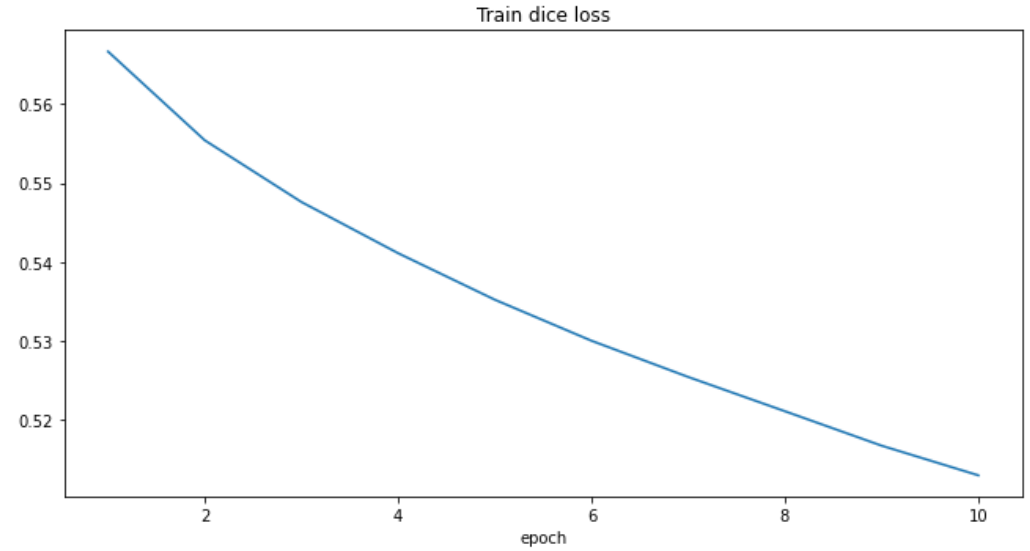
На малюнку 6.5.6. можна побачити найкраще значення помилки для 200 епох.

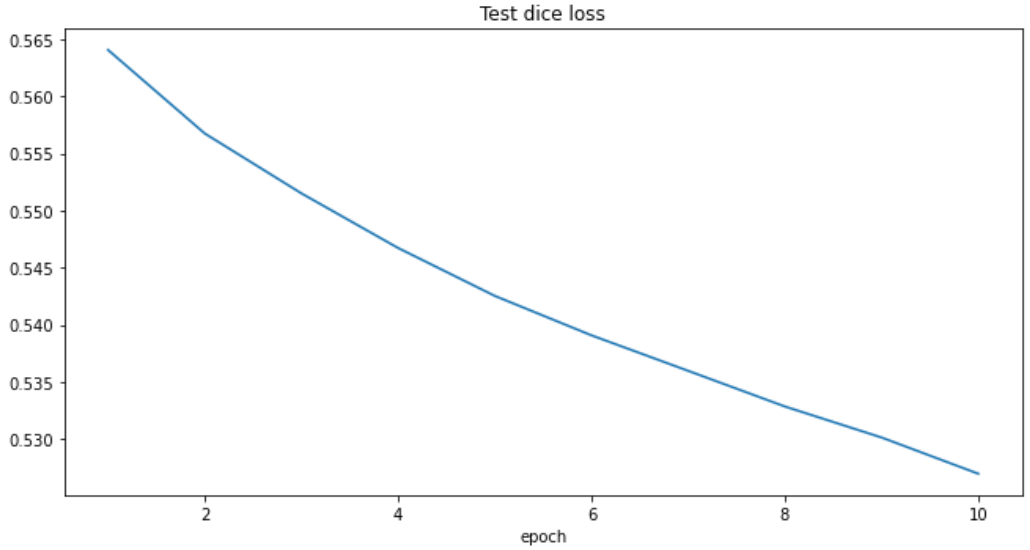
```
current epoch: 200 current mean dice: 0.8591
best mean dice: 0.6101 at epoch: 193
train completed, best_metric: 0.6101 at epoch: 193
```

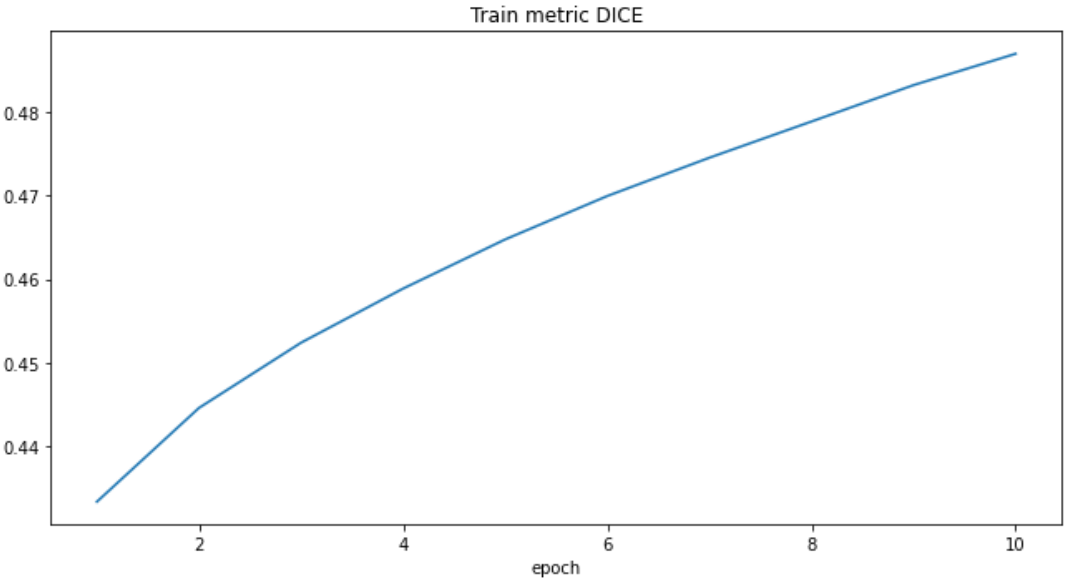
Мал. 6.5.6.

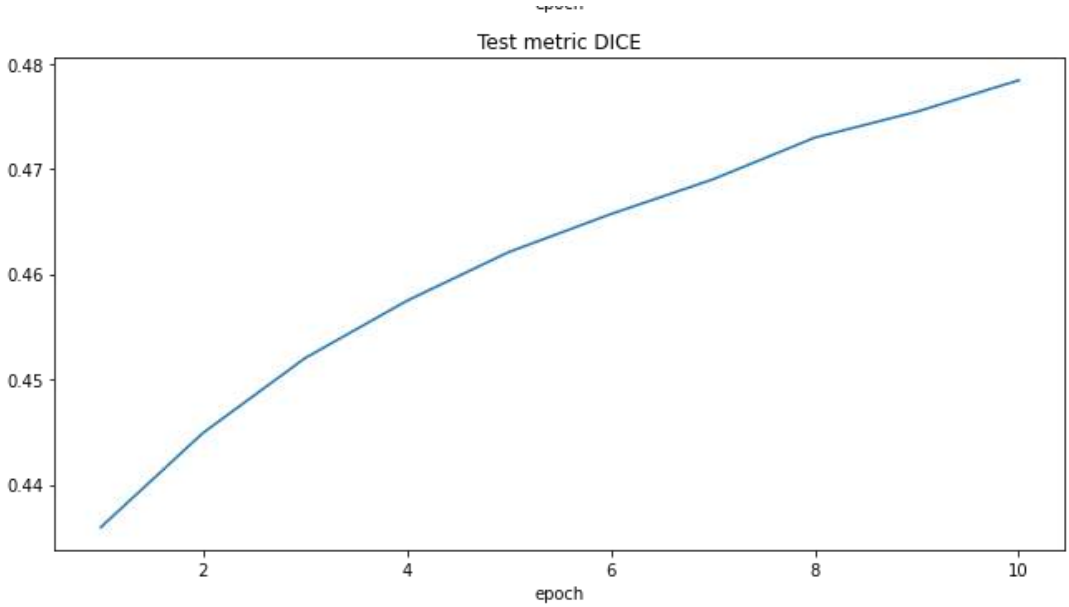
З результатів обчислень можна зауважити, що в останній ітерації не завжди буде найменше значення помилки.

З малюнку 6.5.7.можна побачити, як змінювалися метрики для 10 епох.



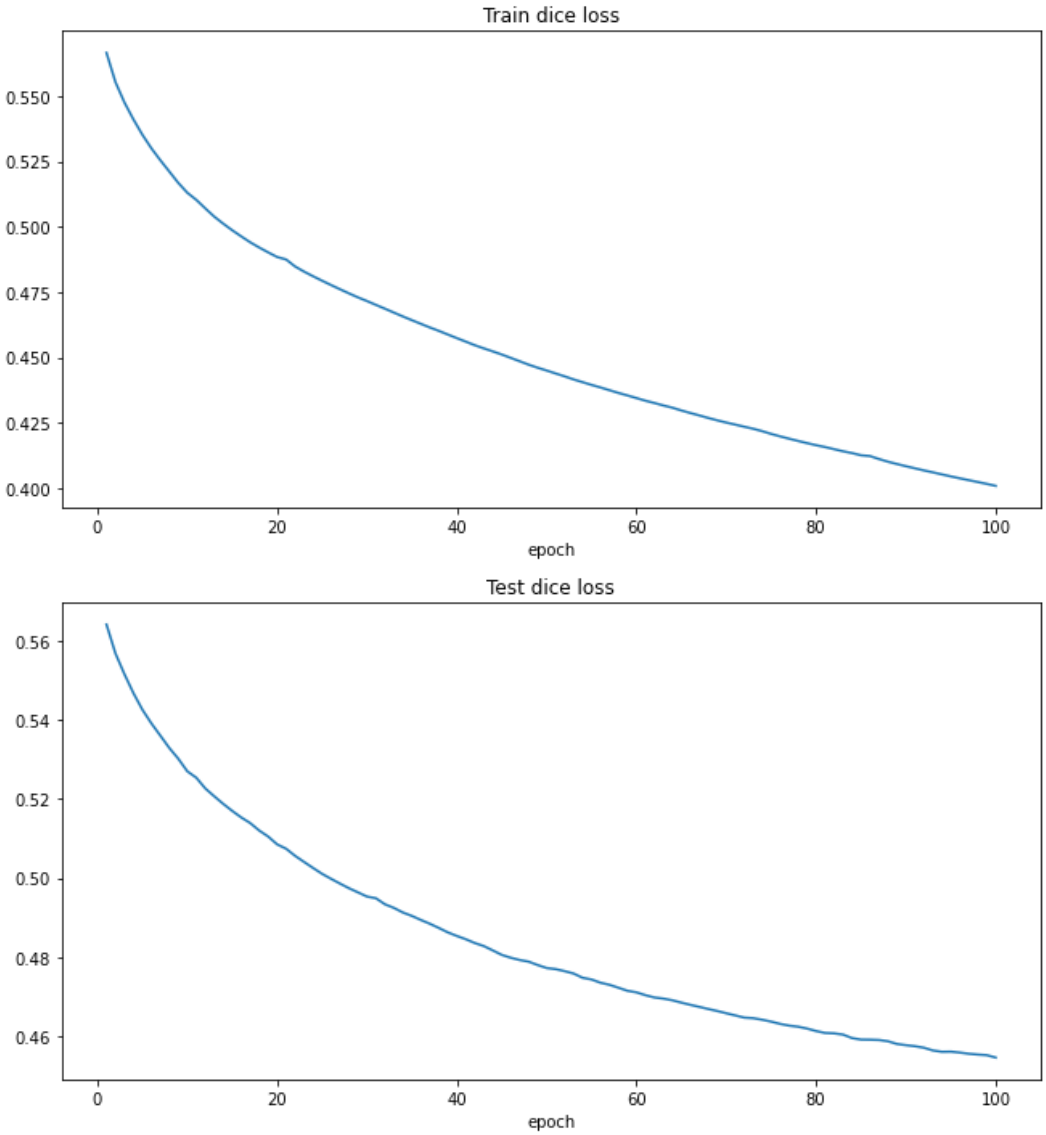




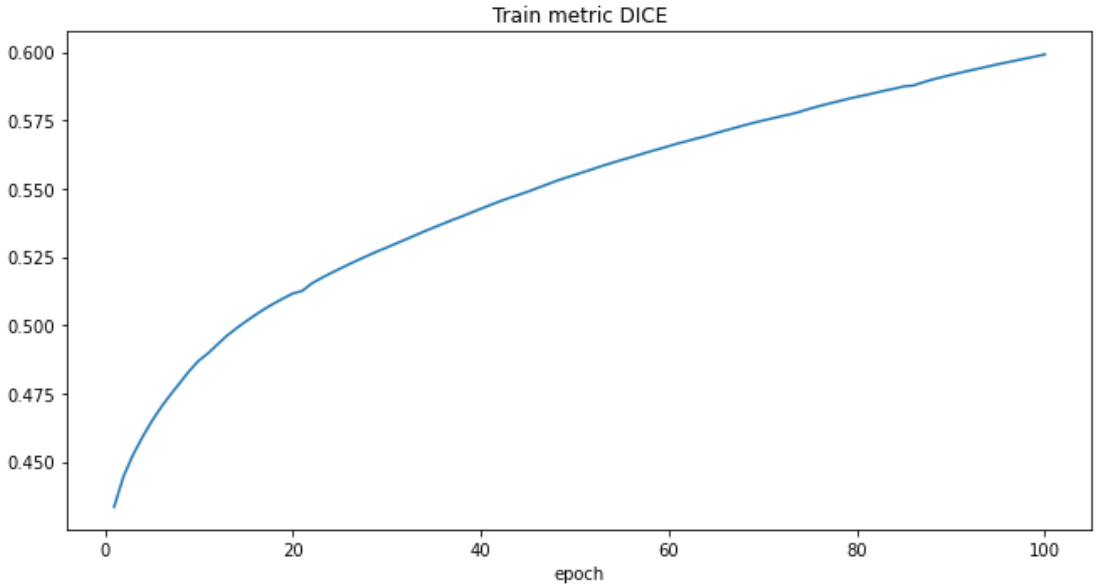


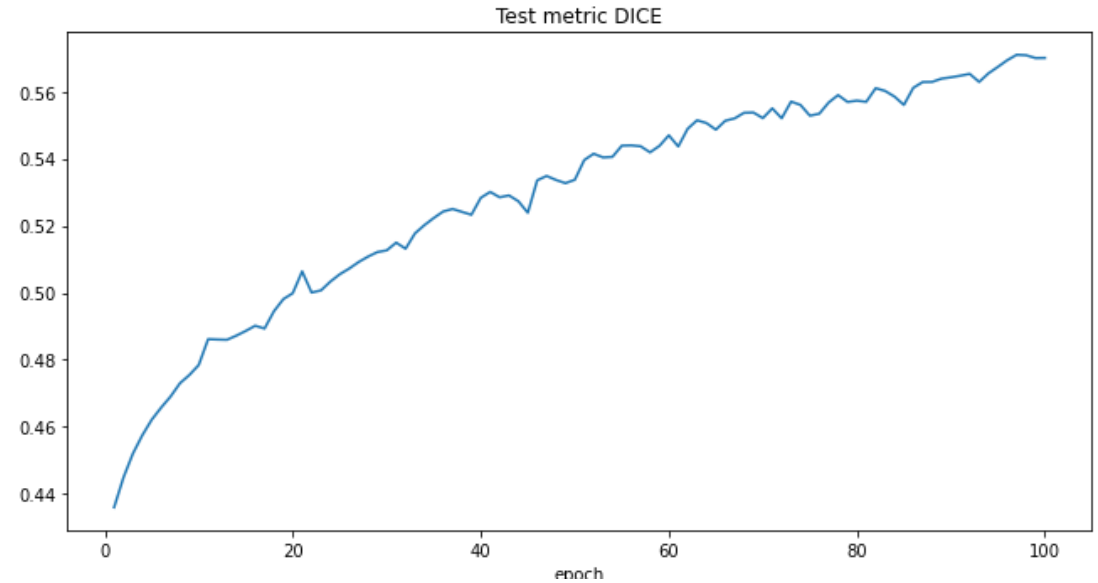
Мал. 6.5.7

З малюнку 6.5.9. можна побачити, як змінювалися метрики для 100 епох.



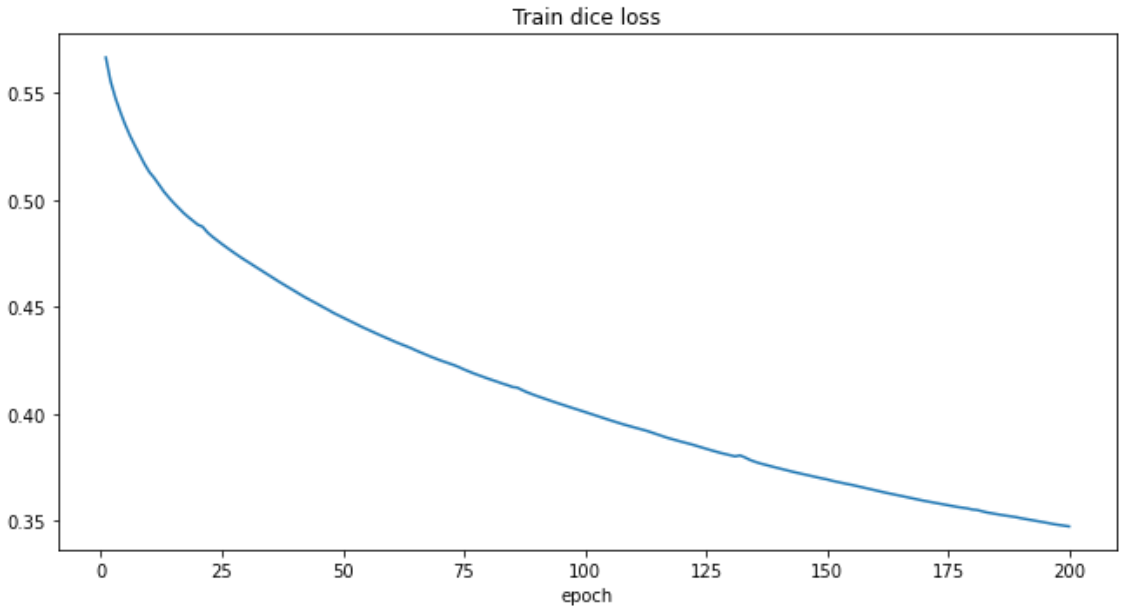
Мал. 6.5.9.


Train metric DICE
0.600
0.575
0.550
0.525
0.500
0.475
0.450
0
20
40
60
80
100
epoch

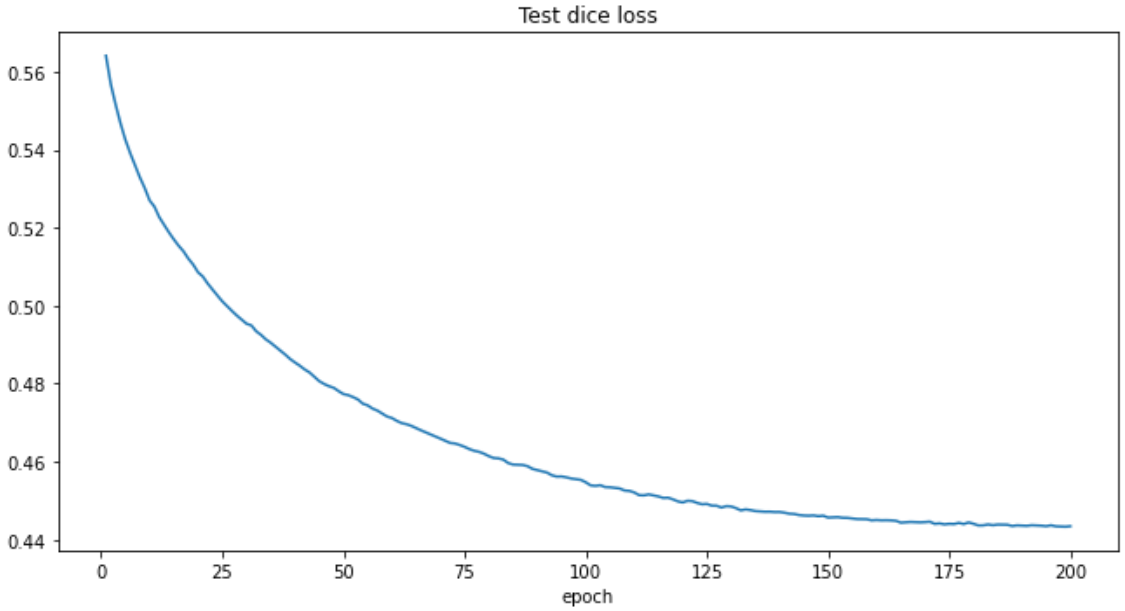

Test metric DICE
0.56
0.54
0.52
0.50
0.48
0.46
0.44
0
20
40
60
80
100
epoch

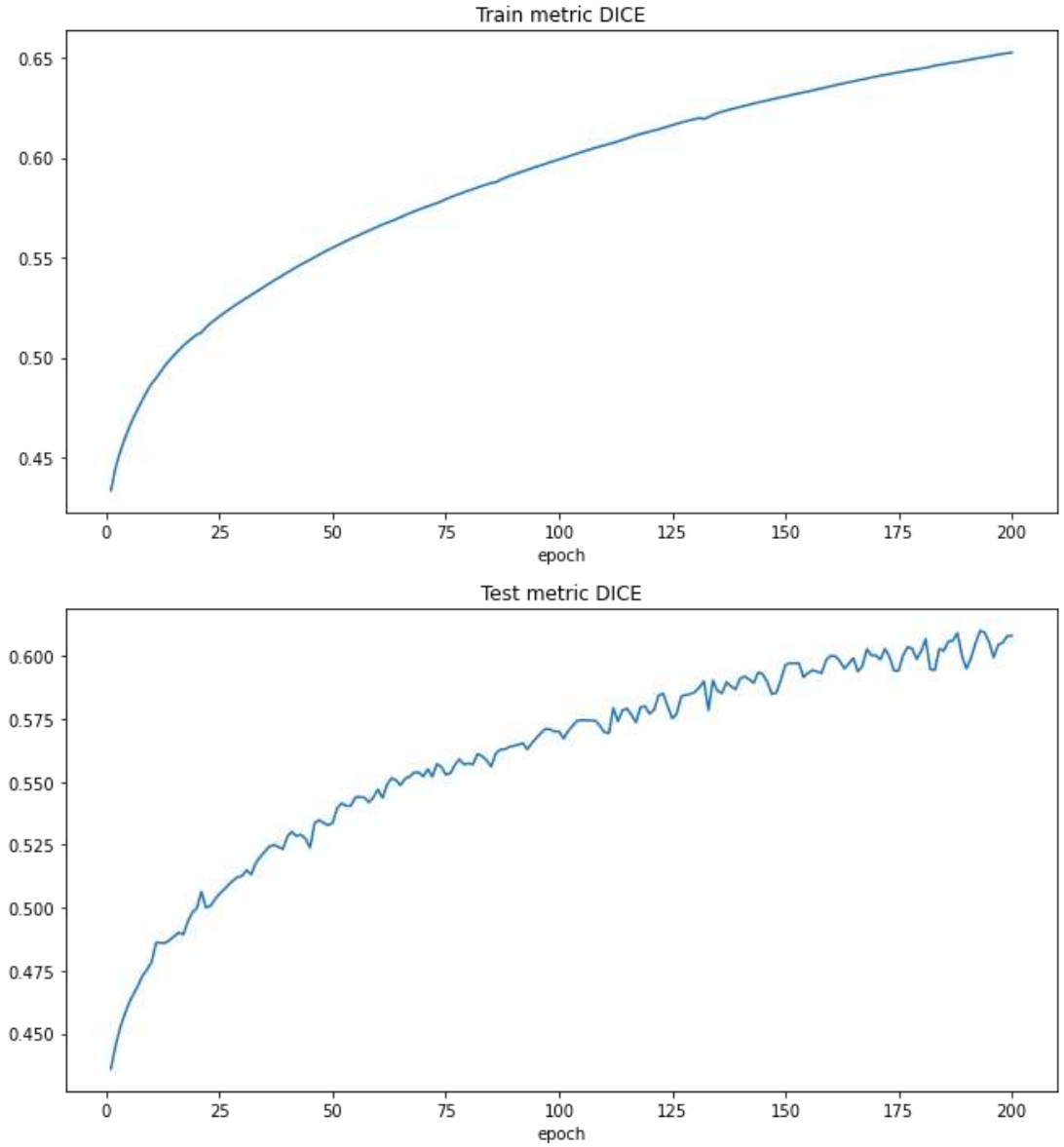
Мал. 6.5.9.

З малюнку 6.5.10. можна побачити, як змінювалися метрики для 200 епох.



Мал. 6.5.10.

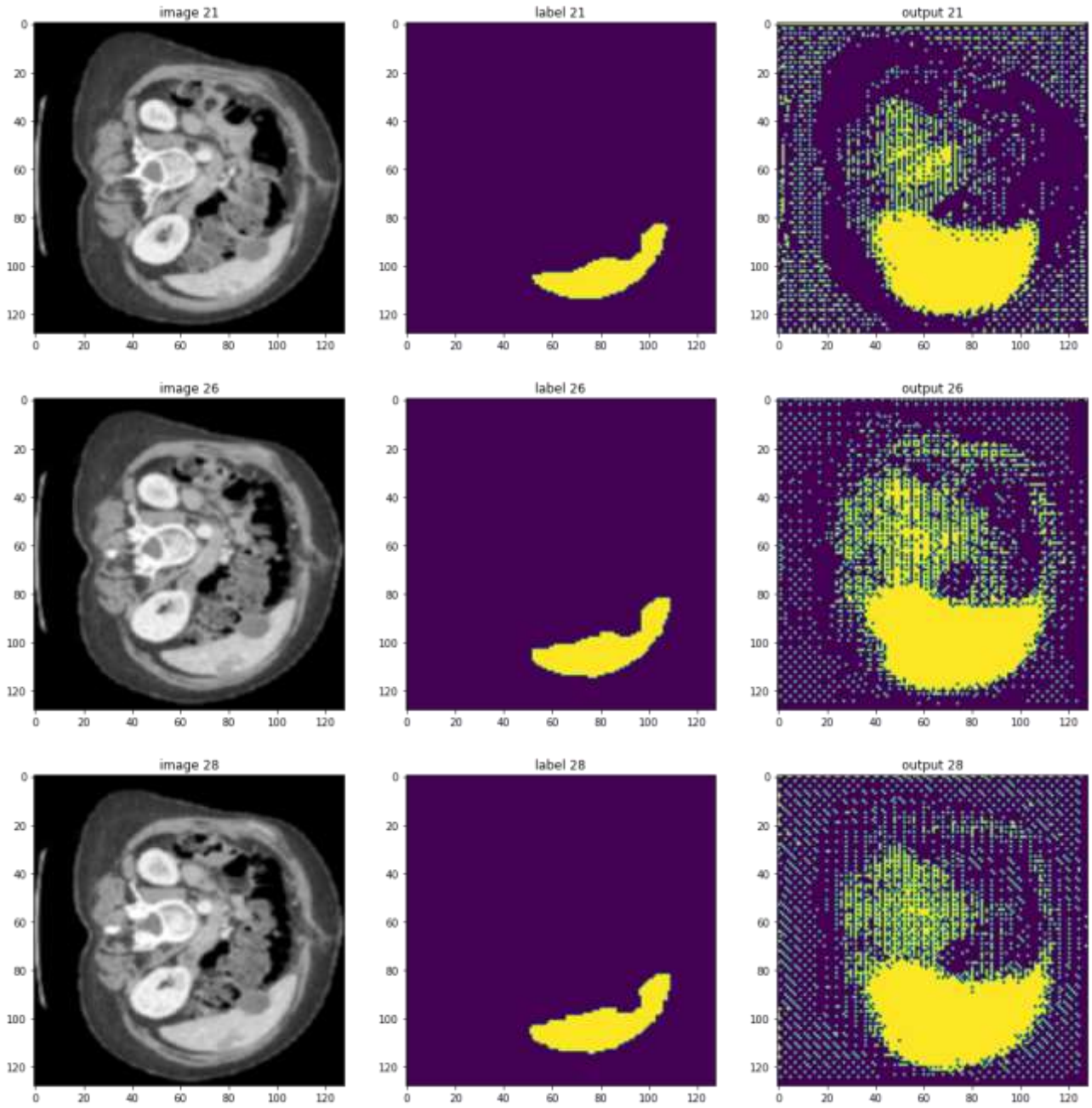




Мал. 6.5.10.

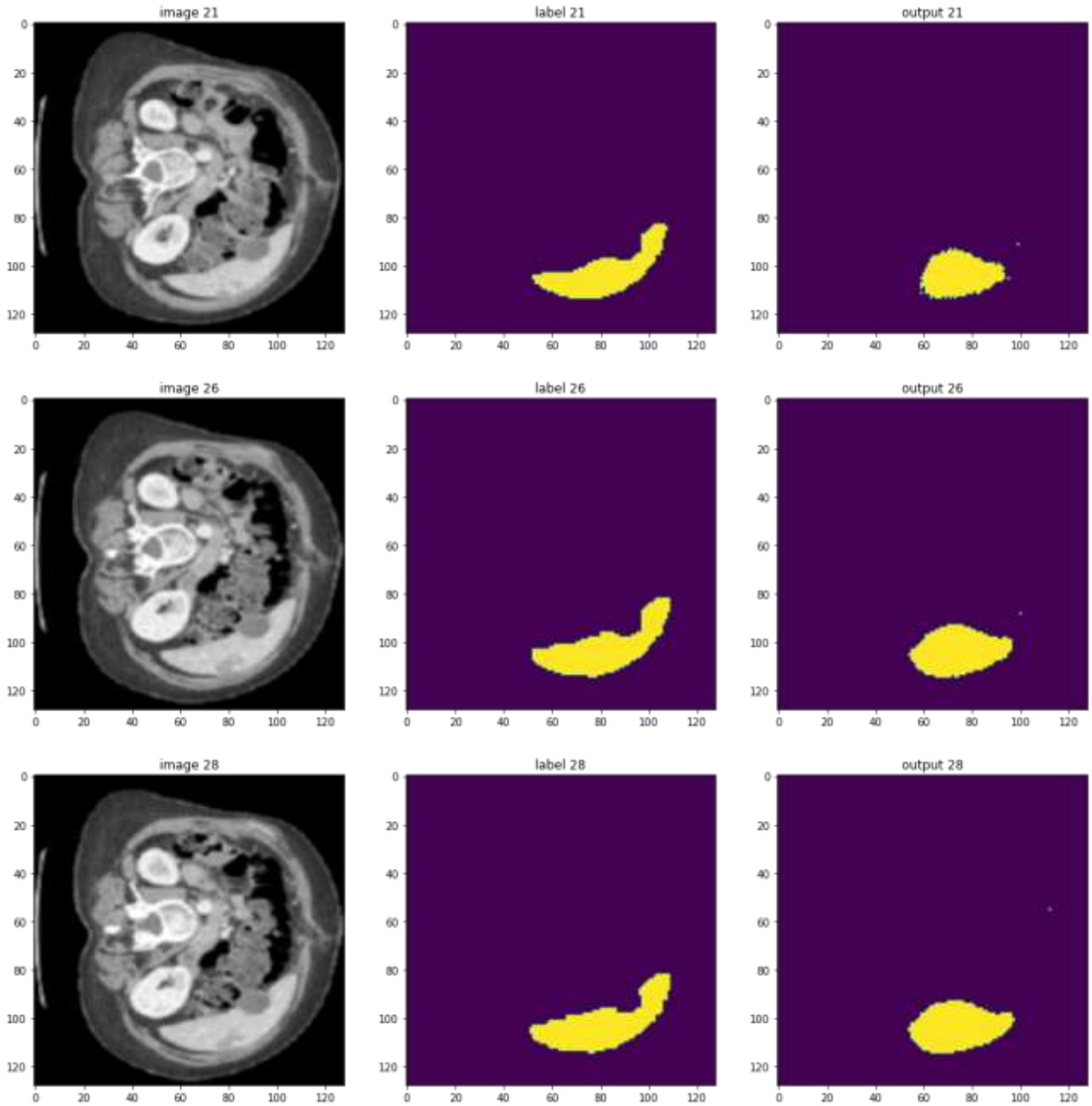
На наступних малюнках можна побачити результат перевірки мережі на тестових даних, які нейронна мережа не бачила до цього моменту. З графіків можна сказати, що наша мережа здатна розпізнати об’єкт, який потрібно.

На малюнках 6.5.11 можна побачити як розпізнає модель медичні зрізи після 10 ітерацій.



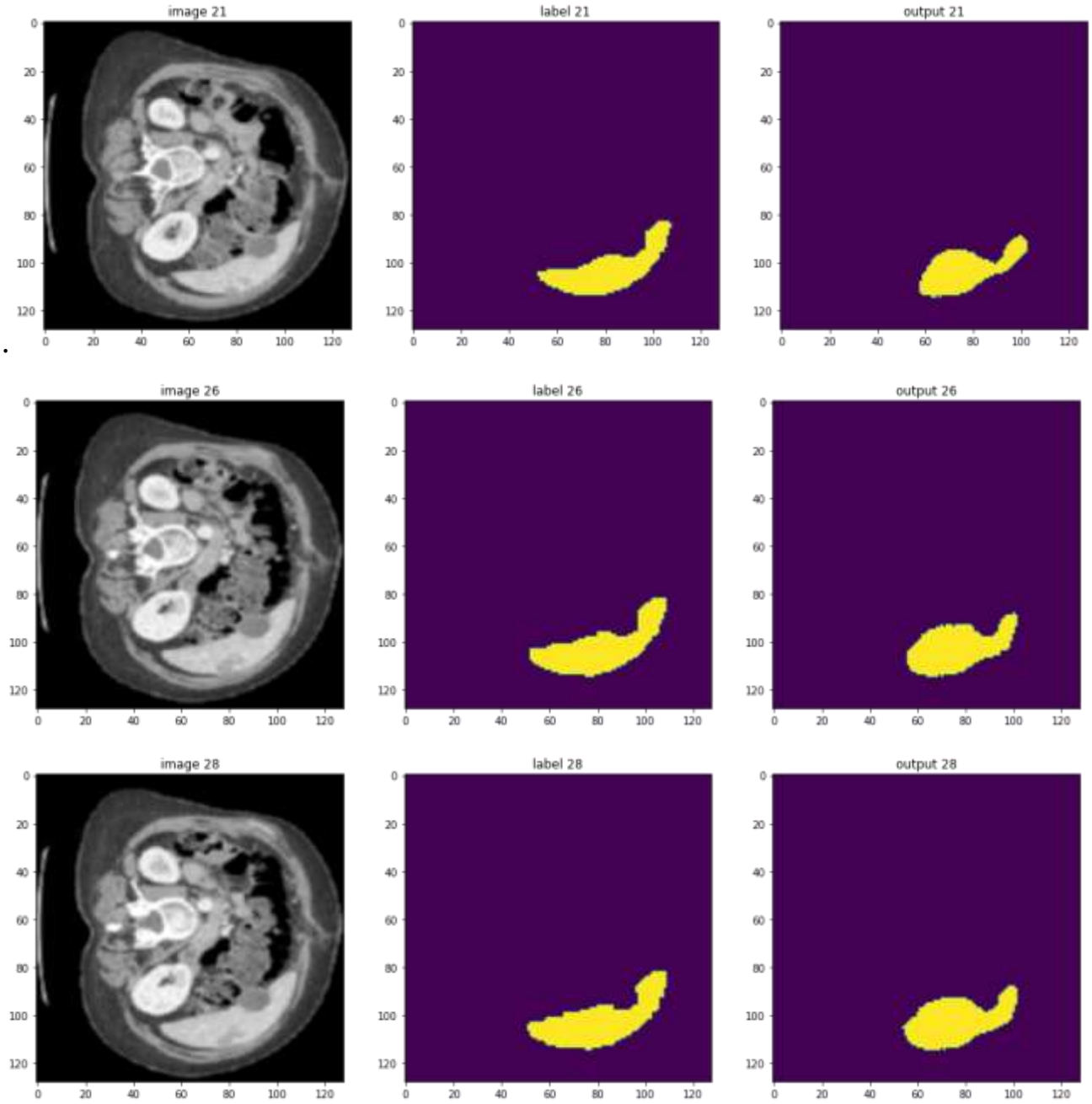
Мал. 6.5.11

На малюнках 6.5.12 можна побачити як розпізнає модель медичні зрізи після 100 ітерацій.



Мал 6.5.12.

На малюнках 6.5.13 можна побачити як розпізнає модель медичні зрізи після 200 ітерацій.



Мал 6.5.13.

ВИСНОВОК

В даній магістерській роботі було продемонстровано як працюють нейронних мереж для сегментації медичних знімків. За допомогою сегментації можна виділити клас об’єкта, який ми хочемо розпізнати, що і було продемонстровано в результатах.

Якщо розглянути поточні результати, то потрібно сказати, що при даному наборі даних та кількості епох, які пройшла нейронна мережа результати є не ідеальними. Для покращення потрібно провести більше навчання мережі, а також збільшити розмір навчальної вибірки. Але не зважаючи на ці фактори результат є досить не поганим, можна побачити, що дана модель після проходження 10 епох може приблизно розуміти в якому місці знаходиться об’єкт, а після 100 епох вже розпізнає його, але все одно допускає значної похибки. Після 200 епох можна побачити, що результат місцями збігається з оригінальним зображенням.

Одним з елементів складності тут є те, що медичні знімки не завжди високої якості і тому передбачити де і який об’єкт знаходиться досить складно. Одним з мінусів даної моделі є довгий час обчислень, адже для обчислення 10 епох потрібно було 328 хвилин, для обчислення 100 епох було затрачено 1360 хвилин, а для 200 епох 3120 хвилин. Не зважаючи на мінуси і складності модель впоралася з своїм завданням і змогла розпізнати зображення.

ДЖЕРЕЛА

1. Image segmentation Techniques by Goutam Sen, Sugata Hazra and Debasish Chakraborty.
2. Python Deep Learning: Exploring Deep Learning Techniques and Neural Network Architectures with PyTorch, Keras, and TensorFlow by Ivan Vasilev.
3. Neural network design by Martin Hagan
4. Deep Learning by Josh Patterson, Adam Gibson
5. http://medicaldecathlon.com/
6. Monai framework: https://monai.io/community.html
7. Pytorch: https://pytorch.org/docs/stable/community/contribution_guide.html