МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки
Факультет Комп'ютерних наук
Кафедра Програмної інженерії

КВАЛІФАЦІКАЦІЙНА РОБОТА

Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження методів виявлення стоматологічних захворювань на зображеннях

Виконав:
студент 2 курсу групи ІПЗм-21-1
Шевчук О. О.
(прізвище, ініціали)

Спеціальність	121 – Інженерія програмного забезпечення
Тип програми	Освітньо-наукова
Керівник	доц. Мельнікова Р. В.
	(посада, прізвище, ініціали)

Допускається до захисту

Зав. Кафедри _____ З.В. Дудар

2023

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Програмної інженерії

Рівень вищої освіти другий (магістреський)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва)

Тип програми освітньо-наукова програма

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«__»_____202_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента Шевчук Олександр Олександрович
(прізвище ім'я по батькові студента)

1. Тема роботи «Дослідження методів виявлення стоматологічних захворювань на зображеннях»

затверджена наказом університету від «29» березня 2023 р. №_302Ст

2. Термін подання студентом роботи до екзаменаційної комісії «22»_травня_2023 р.

3. Вихідні дані до роботи електронні ресурси за обраною тематикою, методи виявлення стоматологічних захворювань на зображеннях, методичні вказівки до виконання кваліфікаційної роботи магістра.

4. Перелік питань, що потрібно опрацювати в роботі аналіз предметної області, аналіз існуючих методів, постановка задачі, планування експерементальної частиники, проектування додатку, опис проведених досліджень і огляд результатів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі	03.04.2023	виконано
2	Аналіз існуючих методів	03.04.2023	виконано
3	Постановка задачі	03.04.2023	виконано
4	Проведення дослідження	21.04.2023	виконано
5	Аналіз отриманих результатів	30.04.2023	виконано
6	Написання пояснювальної записки	10.05.2023	виконано
7	Підготовка презентації та доповіді	13.05.2023	виконано
8	Перевірка на плагіат	14.05.2023	виконано
9	Нормконтроль та рецензування	15.05.2023	виконано
10	Попередній захист	20.05.2023	виконано
11	Допуск до захисту у зав. кафедри	22.05.2023	виконано

Дата видачі завдання 23 січня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Мельнікова Р. В
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 66 с., 4 таблиці, 19 джерел, 40 рисунків.

КАРІЄС, СТОМАТОЛОГІЧНІ ХВОРОБИ, RETINANET, YOLOV3, FASTER R-CNN SSD, ГЛИБОКЕ НАВЧАННЯ, СТОМАТОЛОГІЯ, LABEL STUDIO, ASP .NET CORE, ICCMS.

Об’єктом досліджень методи виявлення стоматологічних захворювань на зображення.

Предметом дослідження є аналіз наявних методів глибокого навчання для класифікації зображень.

Метою даної роботи є аналіз наявних методів досліджень зображень та розробка додатку, який буде за зображенням, видавати результат чи наявні там явні стоматологічні захворювання.

CARIES, DENTAL DISEASES, RETINANET, YOLOV3, FASTER R-CNN SSD, DEEP LEARNING, DENTISTRY, LABEL STUDIO, ASP .NET CORE, ICCMS.

The object of research is the study of methods of detecting dental diseases on images.

The subject of the study is the methods of existing deep learning algorithms for image classification.

The purpose of this work is the analysis of existing methods of image research and the development of an application that will, based on the image, give the result of whether there are obvious dental diseases.

Я, Шевчук Олександр Олександрович ,
(прізвище, ім’я, по-батькові)

студент групи ІПЗм-21-1 здобувач вищої освіти на другому (магістерському) рівні

кафедра програмної інженерії ,
(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему

Дослідження методів виявлення стоматологічних захворювань на зображеннях

що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(а) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

У наш час більшість людей зустрічається з різноманітними зубними хворобами. Більшість українців у нас час хоча б раз у житті хворіла карієсом чи пульпітом. Зубні хвороби можуть не тільки зменшити привабливість людини, через деформацію зубів, неприємний запах, деформацію черепу та неправильний прикус. Нерідко симптоми зубних хвороби заважають звичайним людям виконувати свої функції у суспільстві. До різних зубних проблем призводять найрізноманітніші причини, це недостатній гігієнічний догляд за порожниною рота, неправильний прикус зубів, дефіцит вітамінів та загальні захворювання, які напряму впливають на ротову порожнину. Нажаль у нас час тільки лікар може точно поставити діагноз і провести відповідну процедуру лікування, яка зможе позбавити людину від неприємних симптомів болю у ротовій порожнині. Усі захворювання зубів, які не підлягали відповідному лікуванню призведуть до більш складних захворювань.

Але не завжди люди зі стоматологічними проблемами можуть своєчасно дізнатись про наявні у них недуги. Наприклад виходячи з даних [1], необхідно відвідувати стоматолога не рідше ніж раз у пів року. У цьому випадку, вважають лікарі, можна максимально безпечно і швидко визначити та вилікувати хворобу.

Наявні дані демонструють, що більше ніж половина людей відвідують стоматологів раз на рік, а діти у віці від 5-12 років та молодь віком 13-18 років – майже в таких самих межах, приблизно один раз в 10-12 місяців.

Допускається, що більшість людей не хоче витрачати свій вільний час на походи до лікарів, роблячи припущення, що в них не має проблем із здоров’ям. Саме тому, було вирішено дослідити наявні способи пошуку та виявлення стоматологічних захворювань, шляхом їх аналізу їх фотографій.

Предметом дослідження є аналіз наявних методів глибокого навчання для класифікації зображень.

Метою даної роботи є аналіз наявних методів досліджень зображень та розробка додатку, який буде за зображенням, видавати результат чи наявні там явні стоматологічні захворювання.

Розроблена програмна система буде використовувати класифікацію зображень, заснований на поділі класів по системі ICCMS.

Також буде підготовлений набір даних, що містить інформацію про види карієсу та їх класифікацію відповідно до системи ICCMS. Цей набор даних був зібраний із відкритих джерел.

Варто зазначити, експериментатор не має стоматологічної освіти, що може вплинути на валідність відмічення даних, тому хоча дані і будуть класифіковані за ICCMS, але вони е були перевірені професійним стоматологом.

Буде обрано деякі моделі глибокого навчання, які найкраще підходять для пошуку проблем із ротовою порожниною, а саме: RetinaNet, YOLOv3, SSD, Faster R-CNN.

Результатом виконання дослідження, буде порівняння цих методів за наступними показниками Чутливість (TPR), Специфічність (TNR), Accuracy, Precision.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Карієс – це патологічний процес, при якому спостерігається поступове руйнування тканини кістки або зуба [2]. Причиною розвитку карієсу кісткової тканини є дистрофічні або інфекційні процеси, що виникають в кістці або надкісниці і призводять до некрозу певних ділянок кістки. В результаті цього відбувається всмоктування або відторгнення вмерлих мас та утворення дефектів в кістці .

В якості опису існуючих стадій карієсу використовується міжнародна система ICCMS (International Caries Classification and Management System), яка виділяє чотири стадії карієсу [3] (див. рис. 1).

ICCMS™ Categories (C)	Radiographic Categories (R)				
	R_0 	RA_{1-2} 	RA_3 	RB 	RC 
C_{Sound} 	$Sound_{CR}$	$Initial_{CR}$	$Initial_{CR}$	$Moderate_{CR}$	$Extensive_{CR}$
$C_{Initial}$ 	$Initial_{CR}$	$Initial_{CR}$	$Initial_{CR}$ *or* $Moderate_{CR}$	$Moderate_{CR}$	$Extensive_{CR}$
$C_{Moderate}$ 	$Moderate_{CR}$	$Moderate_{CR}$	$Moderate_{CR}$	$Moderate_{CR}$	$Extensive_{CR}$
$C_{Extensive}$ 	$Extensive_{CR}$	$Extensive_{CR}$	$Extensive_{CR}$	$Extensive_{CR}$	$Extensive_{CR}$

Рисунок 1 – Класифікації карієсу [3]

Отже перший клас – це «Чиста поверхня» (sound surfaces (ICDASTM code 0)), або зуби без зовнішніх ознак карієсу.

Рисунок 2 – Приклад здорових зубних поверхонь [3]

На здорових поверхнях зубів немає ознак карієсу (відсутність або сумнівна зміна прозорості емалі), коли виглядають чистими та після тривалого сушіння на повітрі (5 секунд). Варто зазначити, що поверхні з дефектами розвитку, такими як гіпомінералізація емалі (включаючи флюороз), зношеність зубів (стертість, стирання та ерозія), а також зовнішні або внутрішні плями будуть записані як здорові, якщо вони спостерігаються в інших ямках/тріщинах, які відповідають некаріозним звичкам (наприклад, часте вживання чаю або куріння).

Другий клас – це початкова стадія карієсу (Initial stage caries (ICDASTM codes 1 and 2)).

Рисунок 3 – Початкова стадія карієсу [3]

Перші або виразні візуальні зміни емалі, які спостерігаються як каріозне помутніння або видиме знебарвлення (біле плямисте ураження та/або коричневе каріозне знебарвлення), що не відповідає клінічному вигляду здорової емалі (ICDASTM код 1 або 2) і не має ознак руйнування поверхні або затінення нижнього дентину. Каріозне зміна кольору помітна, починаючи з основи тріщині або ямки, і може поширюватися вгору по стінці ямки/ тріщини, але явної втрати

емалі не спостерігається, тобто ямка/ тріщини зберігає свій початковий анатомічну форму.

На поверхні кореня або на цементно-емалевому з’єднанні є чітко обмежена ділянка, яка змінила колір (світло/темно-коричневий, чорний), але немає кавітації (втрата анатомічного контуру < 0,5 мм).

Третій клас – це карієс середньої тяжкості (Moderate stage caries (ICDASTM codes 3 and 4)).

Рисунок 4 – Карієс середньої тяжкості [3]

Ураження білої або коричневої плями з локалізованим руйнуванням емалі без видимого оголення дентину (код 3 за ICDASTM) або тінню дентину, що лежить під ним (код 4 за ICDASTM), яке, очевидно, виникло на поверхні, що оцінюється.

Щоб підтвердити руйнування емалі, використовується кульковий зонд WHO/CPI/PSR, його необхідно обережно провести по всій області зуба - якщо поверхня падає в мікропорожнину/розрив емалі, виявляється обмежений розрив.

Місцеве руйнування емалі в ямках і фісурах характеризується розширенням фісури/ямки через каріозну втрату структури зуба на вході або всередині нього. Незважаючи на те, що ямка або тріщина може виглядати значно та неприродно ширшою за норму, дентин не видно на стінках або основі порожнини чи розриву. Тіні дентину, що лежать під поверхнею ямок і розломів, представлені сірим, синім або коричневим знебарвленим дентином, видимим під поверхнею емалі, або непрозорим кільцем навколо ямки чи тріщини від пошкодженої емалі

На поверхні кореня є чітко обмежена ділянка, яка змінила колір (світло/темно-коричневий, чорний) і є кавітація (втрата анатомічного контуру ≥ 0,5 мм ≤ 2 мм

Четвертий клас – це екстенсивна стадія карієсу (Extensive stage caries (ICDASTM codes 5 and 6)).

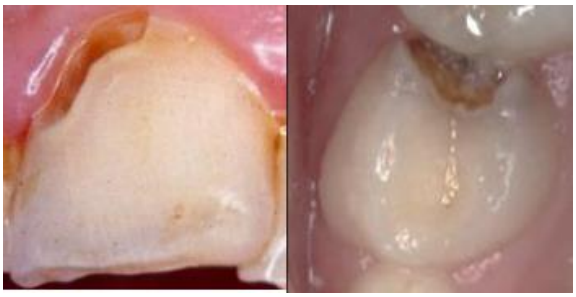

Рисунок 5 – Екстенсивна стадія карієсу [3]

Виразна порожнина в непрозорій або знебарвленій емалі з видимим дентином. Карієс викликає кавітацію, що оголює дентин під ним. Відкриття дентину може бути не відразу помітним без висушування на повітрі; спочатку зуб, який дивиться вологим, може виглядати лише так, що через емаль видно потемніння дентину. Після висихання на стінках і основі є чітко видимі ознаки кавітації з оголеним дентином. Зонди WHO/CPI/PSR можна використовувати для оцінки глибини карієсу.

На поверхні кореня є чітко обмежена ділянка, яка змінила колір (світло/темно-коричневий, чорний) і є кавітація (втрата анатомічного контуру ≥ 2 мм.

2 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ АБО АЛГОРИТМІВ

Що стосується карієсу зубів, наразі існують різні підходи до побудови інструментів автоматичної діагностики, наприклад застосування загальних алгоритмів інтелектуального аналізу даних щодо факторів із щорічних оглядів ротової порожнини [4] або алгоритмів класифікації, що використовують два окремі етапи: сегментацію зображення та класифікацію. [4,5].

Однак один із сучасний та відомих підходів полягає в створенні детектора об’єктів за допомогою моделей глибокого навчання [6], таких як CNN, глибока нейронна мережа (DNN), CNN на основі регіону (R-CNN), Fast R-CNN, Faster R-CNN, Mask R-CNN, You Only Look Once version 3 (YOLOv3), RetinaNet і Single-Shot Multi-Box Detector (SSD) [7–8].

Також дослідження Дінга показали, що алгоритм YOLOv3 має потенційні можливості для виявлення карієсу [9].

Д. Кім створив систему догляду за зубами вдома, використовуючи модель RetinaNet, і повідомив, що система дозволила користувачам ефективно впоратися зі своїми стоматологічними проблемами, надаючи необхідну інформацію про лікування зубів [10].

М. Естай провів дослідження з використанням Faster R-CNN для автоматичного виявлення карієсу на рентгенограмах прикусу. Це дослідження продемонструвало а багатообіцяючу ефективність для виявлення карієсу проксимальної поверхні зубного прикусу [11].

Дослідження К. Муцелоса, використання DNN Mask R-CNN для виявлення карієсу на оклюзійних поверхнях показало точність 88,9% [7]. CNN для виявлення білих плям на фотографіях зубів, повідомляючи про середню точність від 81% до 84%.

Також доступно кілька готових комерційних програм для виявлення карієсу зубів, наприклад Logicon Caries Detector для стоматологічного моніторингу [12].

Глибоке навчання – це клас штучних нейронних мереж із багатьма нейронами, які успішно застосовуються в комп’ютерному зорі, включаючи виявлення об’єктів.

Для задач виявлення об’єктів, які є загальною моделлю виявлення карієсних захворювань, використовуються різні варіанти архітектури глибокого навчання. Тому в цьому розділі ми описуємо деякі вибрані моделі глибокого навчання для автоматичного виявлення карієсу.

З точки зору техніки, можна використовувати різні архітектури глибокого навчання, наприклад Fast R-CNN, Faster R-CNN, RetinaNet, YOLOv3, SSD тощо для автоматичного виявлення ураження карієсом на внутрішньоротових знімках. Алгоритми призначені для виявлення об’єктів, зазвичай базуються на двох підходах: одноетапному виявленню об’єктів і двоетапне виявленню об’єктів.

Почнімо з двоетапних детекторів, вони мають високу точність визначення та розпізнавання. У попередніх дослідженнях було зроблено висновок, що Faster R-CNN, здається, найкращий у виявленні невеликих об’єктів завдяки своїй потужності та стабільності.

Тобто якщо зуб постраждав від карієсу несильно, то цей метод нам повністю підходить. Тому був обраний двоступеневий детектор Faster R-CNN. Нижче наведено короткий опис до архітектури Faster R-CNN, її реалізації та процесу навчання. Основні блоки моделі Faster R-CNN загалом зображені на рисунку 6 [13].

Як було описано Faster R-CNN складається з двох модулів. Перший модуль – це глибока повністю згорткова мережа, яка пропонує регіони, а другий модуль – детектор Fast R-CNN, який використовує запропоновані регіони.

Вся система являє собою єдину, уніфіковану мережу для виявлення об'єктів (див. рис. 7). На вхід у мережу поступають зображення в вигляді тензорів (висоти, ширини та глибини), які проходять через попередньо навчену CNN до проміжного шару, в результаті чого отримуємо згорнуту карту ознак, яку використовують як екстрактор ознак для наступної частини. Така техніка часто використовується в контексті навчання з перенесенням класифікатора на невеликому наборі даних з використанням ваг мереж, навчених на більшому наборі даних. Далі ми отримуємо мережу з назвою RPN, використовуючи

особливості які обчислює CNN, вона знаходить до заданої кількості областей, які можуть містити об’єкти.

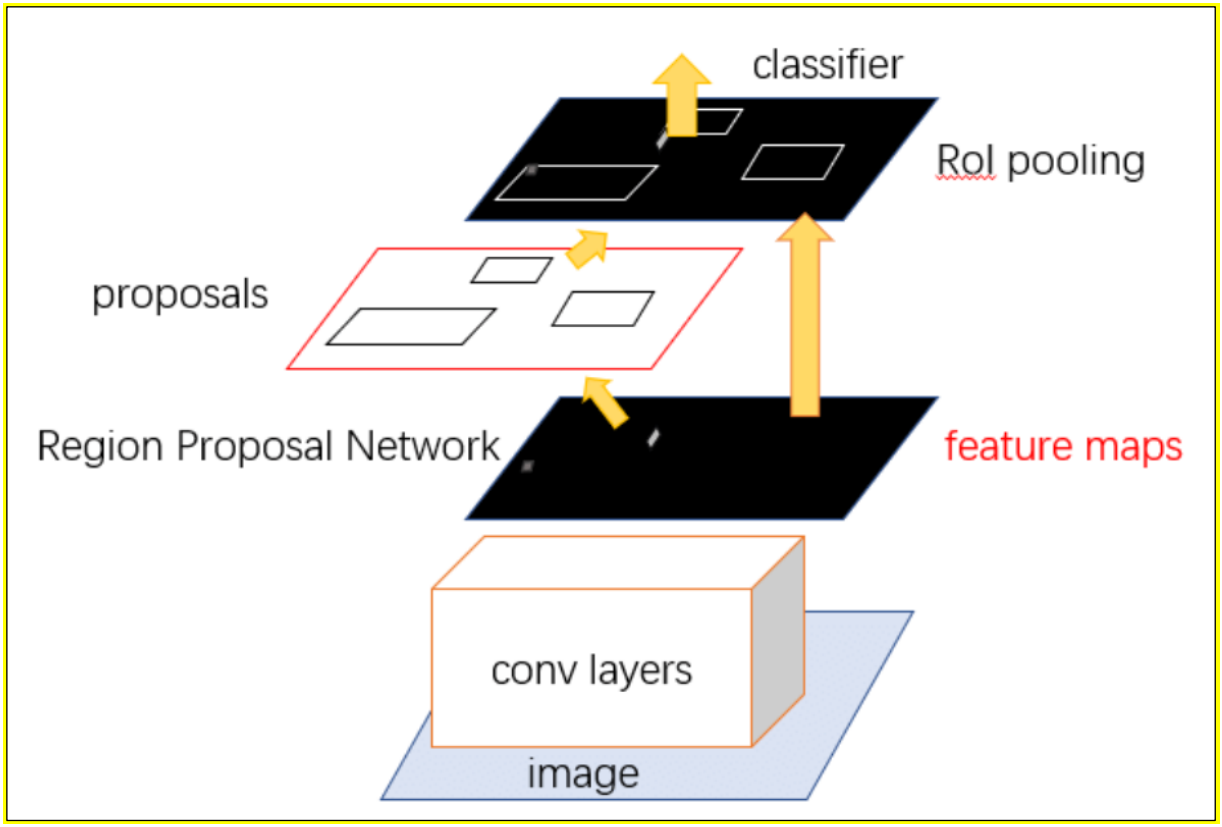

Рисунок 6 – Архітектура Faster R-CNN [13]

Найскладнішою проблемою при використанні нейронних мереж є створення списку змінної довжини. Проблема змінної довжини вирішується в RPN за допомогою якорів опорних обмежувальних рамок фіксованого розміру, які рівномірно розміщені по всьому вихідному зображенню. Замість того, щоб визначати, де знаходяться об'єкти, ми моделюємо задачу на дві частини. Перша, чи містить цей якір відповідний об’єкт. Та другий, як ми могли би налаштувати цей якір так, щоб він краще відповідав відповідному об’єкту. Далі маючи список можливих об’єктів та їх розміщення на початковому зображення, стає набагато легше виявити об’єкт. Після того, як ми використали ознаки, витягнуті CNN, та обмежувальні рамки з відповідними об'єктами, ми застосовуємо об'єднання областей інтересу (RoI) і витягаємо ті ознаки, які відповідають відповідним об'єктам, у новий тензор. І нарешті йде модель R-CNN, яка використовує

отримані дані в обмежувальній рамці для остаточного розуміння чи це необхідний об’єкт, чи це просто «background».

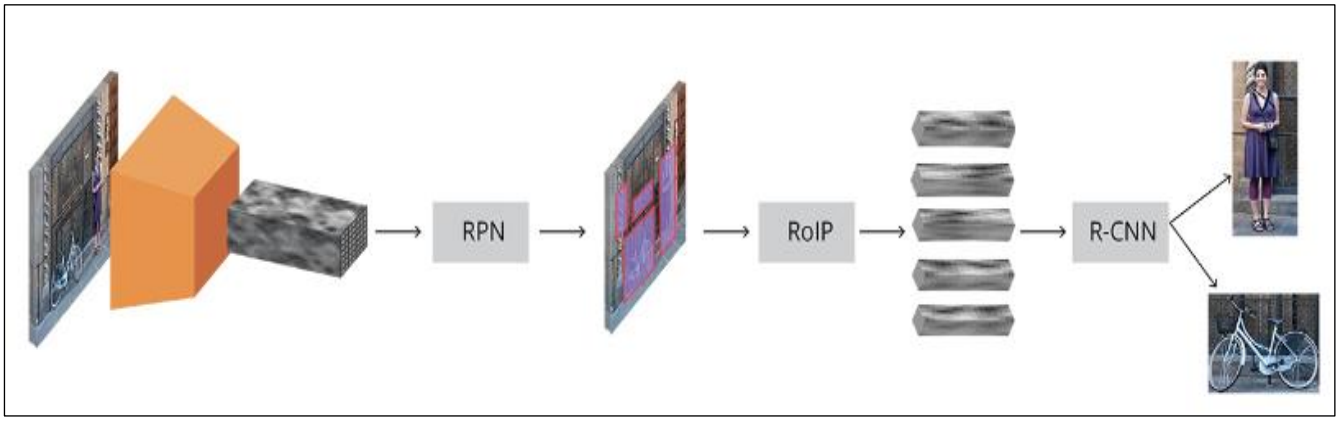

Рисунок 7 – Архітектура Faster R-CNN [13]

З іншого боку, одноступеневі детектори мають високу швидкість, що є їх суттєвою перевагою, і деякі з них можна використовувати на слабкому обладнанні. Вони включають алгоритми сімейства YOLO, сімейства SSD і RetinaNet.

YOLOv3 був представлений у 2018 році, він використовує лише одну нейронну мережу, навчену наскрізною моделлю [14]. Як і попередні версії YOLO, YOLOv3 використовує метод «розмірної кластеризації» для ідентифікації обмежувальних рамок. Архітектура мережі YOLOv3 показана на малюнку 8, який був представлений у дослідженні MAO та ін. У даній архітектурі зображення розбивається на окремі комірки і кожна з них відповідає за передбачення декількох речей. Кожна комірка відповідає за передбачення декількох рамок, що містять об'єкти та показники впевненості, а отже цей показних вказує чи містить дана рамка об'єкт чи ні. Ще одна річ за яку відповідають комірки, це показ вірогідності передбачення класів, однак це не означає, що ця комірка містить якийсь клас. Це означає, що комірка може містити частину об'єкта якогось класу. До мінусів Yolov3, можна віднести її незначну можливість шукати невеликі об’єкти та великий та необхідність мати значний об’єм пам’яті, що може бути проблемою для пристроїв з обмеженими ресурсами.

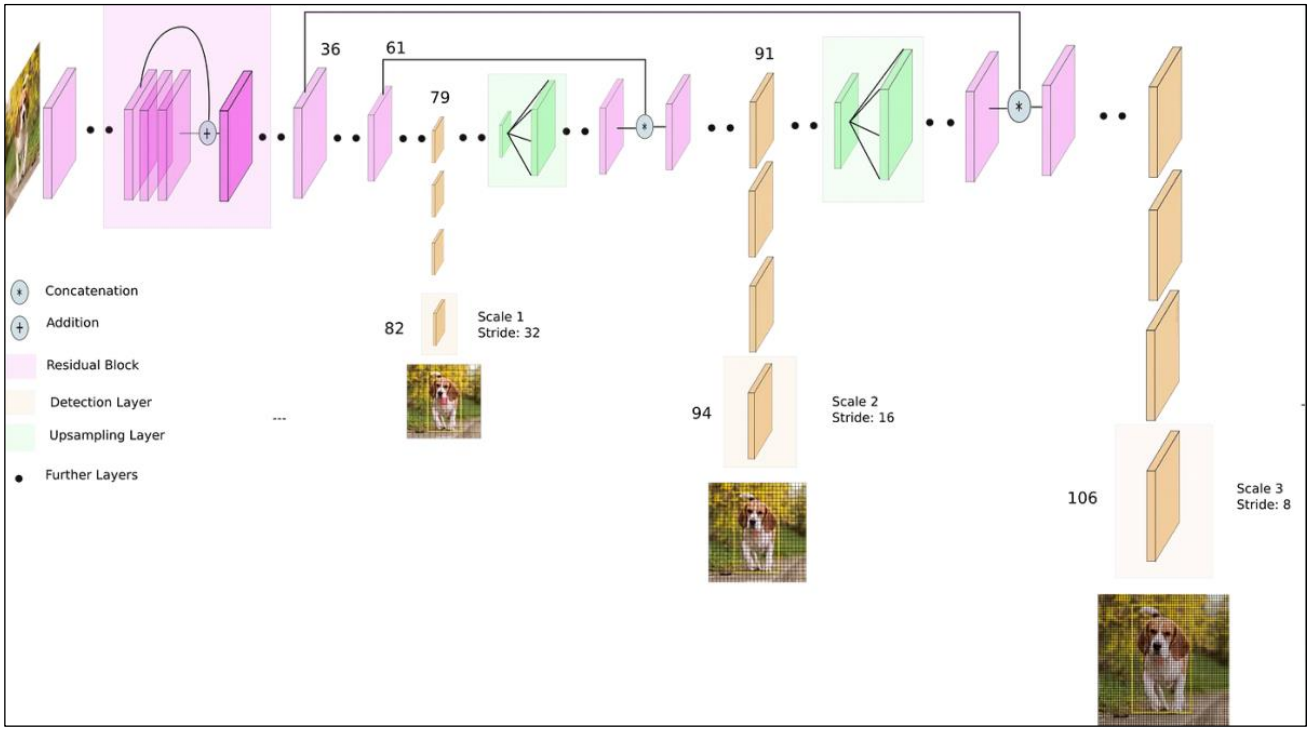

Рисунок 8 – Архітектура YOLO v3 [14]

SSD, створює набір обмежувальних прямокутників фіксованого розміру на карті об’єктів (також відомих як зміщення обмежувальної рамки) і відносні оцінки представлення мітки об’єкта, що міститься [15]. Після цього застосовується немаксимальний крок придушення з метою об’єднання згенерованих обмежувальних рамок для отримання остаточного прогнозованого результату. Подібно до YOLO, особливістю високої швидкості SSD є те, що модель використовує лише одну нейронну мережу. У SSD моделі створюється сітка з квадратів на картах функцій, і кожна клітинка називається клітинкою карти функцій. У центрі кожної комірки карти функцій визначається набір полів за замовчуванням, щоб передбачити кадр, який може охоплювати об’єкти. Процес навчання SSD має власну стратегію відповідності для уточнення ймовірності мітки і обмежувальної рамки, щоб відповідати вхідним значенням базової істинності моделі (включаючи мітки та зміщення обмежувальної рамки). Крім того, мережа поєднується з багатьма картами функцій з різною роздільною здатністю для виявлення об’єктів різного розміру та форми.

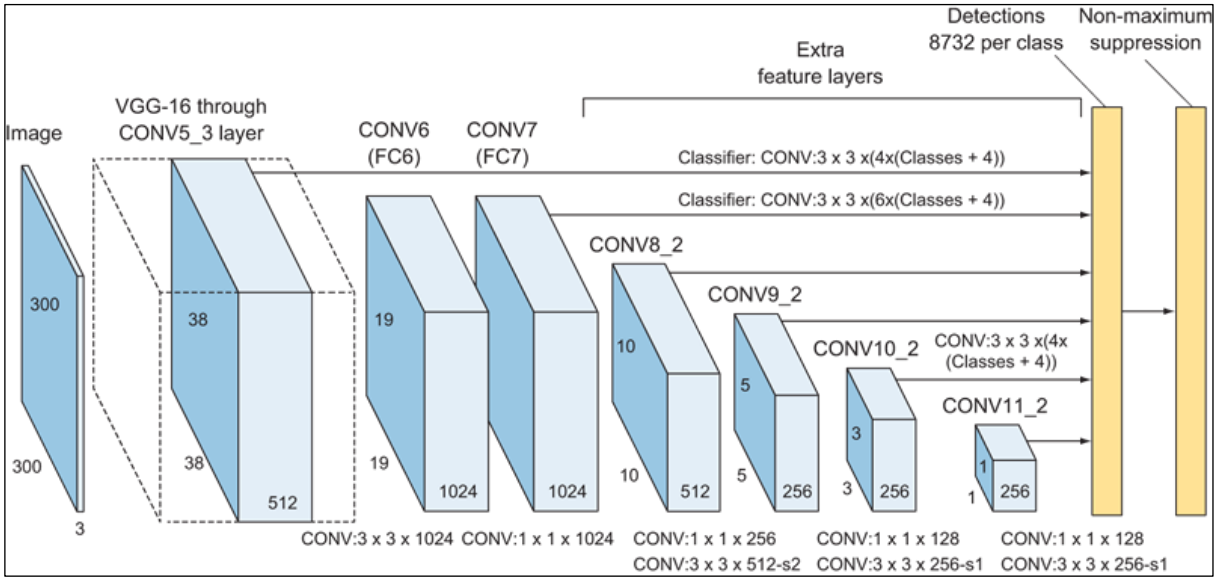

Рисунок 9 – Архітектура SSD [15]

RetinaNet – ще один одноетапний детектор об’єктів, і ця архітектура нейронної мережі фокусується на вирішенні дисбалансу між класами переднього плану та фону [16]. Вона використовує функцію фокальних втрат для усунення дисбалансу класів під час навчання. Цей детектор об’єктів виступає у ролі єдиної уніфікована мережі, що складається з магістральної мережі та двох підмереж, призначених для конкретних завдань (див. рис. 11).

Магістральна мережа відповідає за обчислення згорткової карти ознак на всьому вхідному зображенні і є автономною згортковою мережею. Перша підмережа виконує згорткову класифікацію об'єктів на виході магістральної, а друга в свою чергу виконує регресію в обмеженій області. Обидві підмережі мають простий дизайн, який дозволяє даній нейронній мережі виступати в якості одноетапного детектору об’єктів. Двоетапних моделей пошуку об’єктів, які використовують для вирішення проблеми дисбалансу класів двоступеневий каскад та евристика вибору, у них етап «пропозиції» швидко звужує кількість об’єктів, які можуть виступати в ролі шуканих, відфільтровуючи більшість фонових зразків, а другий етап включає етап класифікації, при якій застосовується евристика вибірки для забезпечення керованого балансу між переднім та заднім планом.

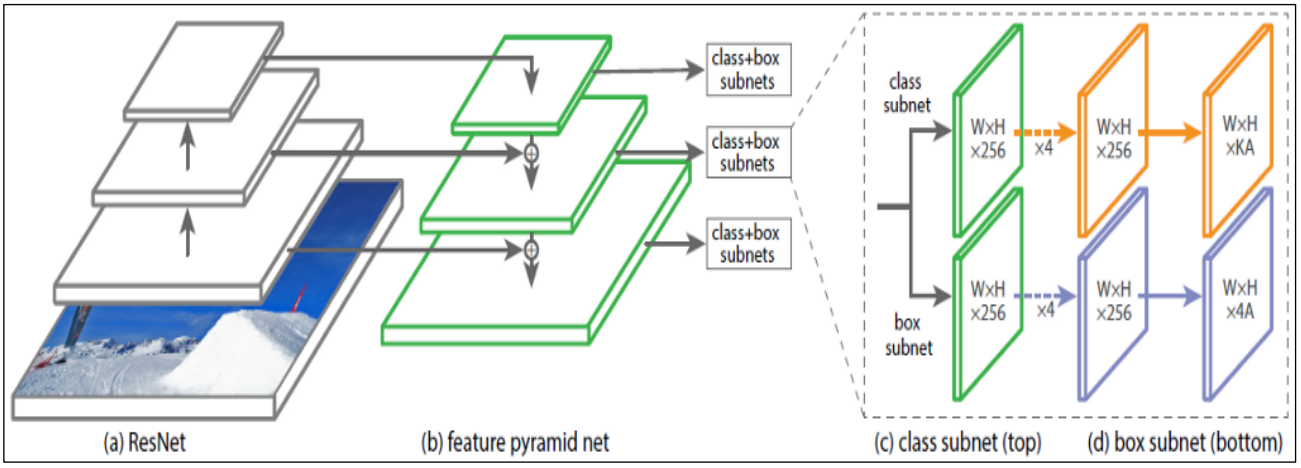

Рисунок 11 – Архітектура RetinaNet [16]

На відміну цьому одноетапний детектор повинен обробляти набагато більший набір місць розташування об'єктів-кандидатів, які регулярно відбираються по всьому зображенню. Для вирішення даної проблеми, RetinaNet використовує функцію фокальних втрат, динамічно масштабовану перехресну втрату ентропії, де коефіцієнт масштабування спадає до нуля зі збільшенням впевненості в правильному класі. Інтуїтивно зрозуміло, що цей коефіцієнт масштабування може автоматично зменшувати внесок легких прикладів під час навчання і швидко фокусувати модель на складних прикладах.

3 ПОСТАНОВКА ЗАДАЧІ

У даному дослідженні буде порівняно чотири моделі нейронних мереж глибокого навчання для пошуку карієсу, а саме YOLOv3, SSD, RetinaNet та Faster R-CNN, також виходячи з отриманих даних необхідно розробити додаток, який зможе виявляти карієсні захворювання на зображеннях, та буде дозволяти зрозуміти чи наявні у користувача проблеми з зубами.

Потрібно зазначити, що YOLOv3 та R-CNN - це дві різні моделі глибоких нейронних мереж, які використовуються для об'єктного визначення на зображеннях. Кожна з цих перерахованих моделі використовує власний формат маркування даних. Можна одразу зробити наступне допущення YOLO та SSD будуть відповідати швидше, а в свою чергу RetinaNet та Faster R-CNN будуть видавати більш точну відповідь.

Після навчання моделей, потрібно буде провести тестування на незалежному наборі даних, щоб оцінити її точність та ефективність в класифікації видів карієсу зубів згідно з системою ICCMS.

Таким чином, за допомогою розробленого додатку можна буде розв'язати задачу класифікації видів карієсу зубів згідно з системою ICCMS.

Для успішної реалізації цієї задачі необхідно мати відповідний набір даних та протестувати чотири вищеназвані моделі нейронних мереж.

4 ПЛАНУВАННЯ ЕКСПЕРИМЕНТАЛЬНОЇ ЧАСТИНИ

Усі 4 нейронні мережі YOLOv3, SSD, RetinaNet та Faster R-CNN працюють з відмінними форматами даних, які можна використовувати для тренування та використання моделей.

YOLOv3 використовує формат даних, який називається COCO (Common Objects in Context). Цей формат містить інформацію про об'єкти на зображенні, такі як їхні координати, мітки та вірогідність наявності на зображенні. Файли даних COCO мають розширення .json та містять велику кількість зображень з мітками.

SSD використовує формат даних, який називається PASCAL VOC (Visual Object Classes). Цей формат містить інформацію про об'єкти на зображенні, такі як їхні координати та мітки. Файли даних PASCAL VOC мають розширення .xml та містять велику кількість зображень з мітками.

RetinaNet також використовує формат даних PASCAL VOC, але може також використовувати формат COCO.

Faster R-CNN використовує формат даних, який називається COCO. Цей формат містить інформацію про об'єкти на зображенні, такі як їхні координати, мітки та вірогідність наявності на зображенні. Файли даних COCO мають розширення .json та містять велику кількість зображень з мітками.

Кожен з цих форматів даних можна легко конвертувати один в інший за допомогою відповідних інструментів, або можна для маркування даних використовувати готові інструменти, які можуть одразу створювати готові датасети в різних форматах.

Розроблена програмна система буде використовувати класифікацію зображень, заснований на поділі класів по системі ICCMS. Перш за все, необхідно підготувати набір даних, що містить інформацію про види карієсу та їх класифікацію відповідно до системи ICCMS. Цей набор даних був зібраний із відкритих джерел. Варто зазначити, експериментатор не має стоматологічної освіти, що може вплинути на валідність відмічення даних, тому хоча дані і будуть класифіковані за ICCMS, але не були перевірені професійним стоматологом.

В якості набору даних було використано відкриті дані, які були знайдені в репозиторіях github [17,18] (див. рис. 12).

Рисунок 12 – Приклад набору даних з хворими зубами [15]

Сумарно обидва набори даних мають приблизно 240 зображень для навчання та 60 для перевірки (див. рис. 13).

Рисунок 13 – Приклад набору даних із здоровими зубами [16]

У цьому дослідженні використовувалися загальні параметри для оцінки ефективності глибокого навчання архітектури за допомогою візуального огляду фотографій як еталонного зразка.

TP: true positive, кількість випадків, які були правильно класифіковані як позитивні.

TN: true negative, кількість випадків модель правильно класифікувала негативний зразок як негативний.

FP: false positive, кількість випадків, які були неправильно класифіковані як позитивні;

FN: false negatives, кількість випадків, які були неправильно класифіковані як негативні.

Чутливість (TPR) вимірює частку позитивних результатів, які правильно визначено згідно з формулою (1):

$$\mathrm{TPR} = \frac{\mathrm{T}P}{\mathrm{TP+FN}} \tag{1}$$

Специфічність (TNR) вимірює частку негативів, які правильно визначено відповідно до формули (2):

$$\mathrm{TNR} = \frac{\mathrm{T}N}{\mathrm{TN+FP}} \tag{2}$$

Accuracy можна представити як кількість класифікованих наборів даних, поділену на загальну кількість тестових наборів даних, і згідно з формулою (3)

$$\mathrm{Accuracy} = \frac{\mathrm{T}N + TP}{\mathrm{TN+FP + TP + FN}} \tag{3}$$

Precision вказує на правильне передбачення кількості категорій, поділених на загальну кількість даних, що підпадають під цю категорію, відповідно до формули (4):

$$\mathrm{Precision} = \frac{\mathrm{T}P}{\mathrm{TP+FP}} \tag{4}$$

Результати дослідження будуть занесені у таблицю 1, яка відповідає оцінці за вищевказаними параметрами, такі як чутливість, специфічність та точність [19].

Таблиця 1 – Результат проведеного досліду (таблиця виконана власноруч)

Модель DL	TPR	TNR	Accuracy	Precision
YOLOv3				
SSD				
RetinaNet				
Faster R-CNN				

Також наведені моделі будуть порівняні за загальним розміром, а також переглянути як моделі будуть реагувати на зображення брекетів та пожовклих зубів.

5 ПРОЄКТУВАННЯ ДОДАТКУ

ASP.NET Core є безкоштовним та відкритим програмним забезпеченням для веб-застосунків та веб серверів, розробленим компанією Microsoft та співтовариством .NET. Даний фреймворк має набагато вищу продуктивність порівняно із застарілим ASP.NET. ASP.NET Core має модульну структуру, яка може працювати як на застарілому .NET Framework, так і на більш сучасному .NET Core. Даний фреймворк забезпечує сумісність з концепціями ASP.NET MVC, і більше того об'єднує функціональність MVC, Web API та Web Pages. Завдяки сумісній реалізації вищеописаного функціоналу, було усунуте багатократне повторення коду, у цих концепціях, що призвело до загального зменшення зайнятого місця. ASP.NET Core також підтримує можливість використання різних версій при використанні різних програм, що працюють на одному комп'ютері, раніше даний функціонал був недоступний у попередніх версіях ASP.NET

Бізнес логіка передає дані на відкриті API, що дозволяє іншим розробникам використовувати їх і збільшувати популярність. Бізнес логіка також перевіряє дані які приходять, та повертає помилки. Для статичної валідації було використано DataAnotation. Атрибут [ApiController] дає змогу контролеру автоматично зупиняти невалідні дані.

Blazor – безкоштовна веб-платформа з відкритим вихідним кодом, що дозволяє розробникам створювати веб-програми з використанням C# і HTML. Розробляється корпорацією Microsoft.

Програма Blazor може взаємодіяти з JavaScript (причому вони працюють на стороні клієнта), наприклад, викликати (повторно використовувати) функції JavaScript з .NET методів.

Так, як було описано раніше, для різних моделей потрібне різне маркування даних, потрібно використовувати моделі які підтримують різні види вихідних даних такі як Yolo та COCO. Для маркування даних використовувалась Label Studio, яка підтримує усі необхідні нам формати.

Label Studio є відкритою платформою для створення, редагування та позначення даних для машинного навчання та інших завдань обробки даних. Вона дозволяє користувачам створювати і налаштовувати проекти позначення даних для різних завдань машинного навчання, таких як класифікація, визначення сутностей, семантичне розмічання тощо.

Цей проект є потужним інструментом для позначення даних, який дозволяє розширити набір даних для машинного навчання. Платформа працює на основі веб-інтерфейсу та забезпечує можливість зручно та ефективно позначати дані різними способами, що забезпечує точність та надійність моделей машинного навчання.

Студія дозволяє завантажувати та переглядати дані для позначення, створювати та настроювати завдання позначення, запрошувати інших користувачів для співпраці та контролювати якість виконаної роботи. Вона також має інтеграції з різними інструментами та платформами машинного навчання, що дозволяє ефективно використовувати позначені дані для створення моделей машинного навчання.

Основні можливості Label Studio включають:

- створення завдань позначення даних: користувачі можуть створювати проекти та завдання для позначення даних для різних завдань машинного навчання, таких як класифікація, визначення сутностей, семантичне розмічання та інші;
- конфігурування позначення: користувачі можуть встановлювати правила для позначення даних, додавати мітки та налаштовувати розмітку, щоб забезпечити якість та точність результатів;
- співпраця: програма дозволяє кільком користувачам працювати над одним проектом, що полегшує спільну роботу над проектом та забезпечує більш швидке завершення роботи;
- інтеграція: Label Studio має інтеграції з різними інструментами та платформами машинного навчання, що дозволяє використовувати позначені дані для створення моделей машинного навчання;

– експорт та імпорт даних: користувачі можуть експортувати та імпортувати дані в різних форматах, таких як JSON, CSV, та інші.

У даному випадку буде використовуватись тип проекту «Комп’ютерний Зір», а маркування даних буде відбуватись за допомогою «Виявлення об'єктів за допомогою обмежувальних рамок». Приклад розмічених даних, буде виглядати наступним чином (див. рис. 14).

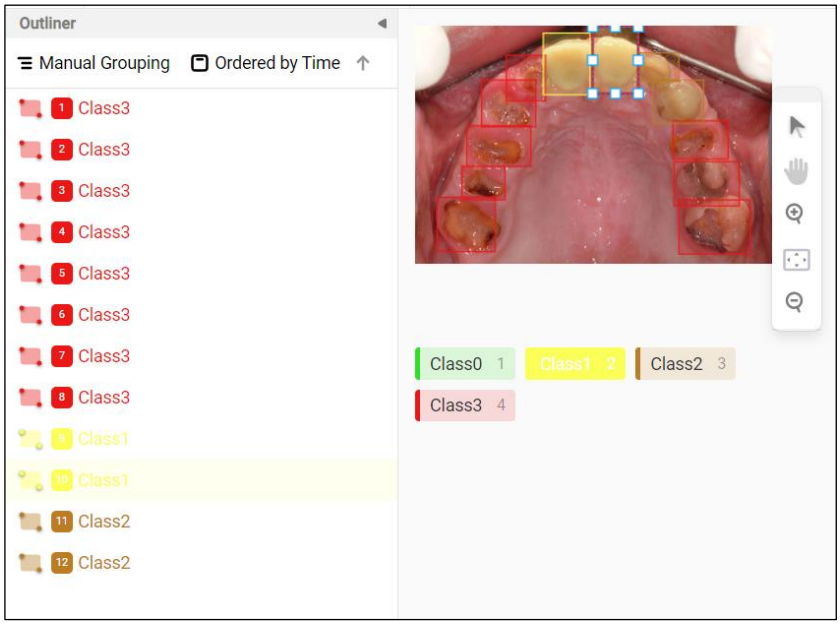

Рисунок 14 – Приклад розмічених даних (створений власноруч)

У даному проекті, Label Studio використовувався як локальний сервер (запускався на localhost).

6 ОПИС ПРОВЕДЕНИХ ДОСЛІДЖЕНЬ

Виходячи з того, що автор експерименту не мав потужних графічних процесорів, було вирішено обрати хмарний сервіс для тренування моделей.

Вибір впав на сервіс «Google Colaboratory» або «Google Colab», який надає можливість виконувати код Python через Jupyter Notebook на видаленому сервері, не встановлюючи додаткових бібліотек чи програм на своєму комп'ютері. Однією з головних причин вибору даного сервісу, була можливість безкоштовно або за невеликі кошти використовувати потужні графічні процесори GPU та TPU, що дозволяють значно пришвидшити навчання моделей.

Усі моделі для навчання використовували бібліотеку TensorFlow, так як вона має одну з найкращих інтеграцій у сервіс «Google Colab», встановлена в середу за замовчуванням тому не вимагає встановлення.

Для того, щоб кожен раз не завантажувати датасет у «Google Colaboratory», було вирішено завантажити їх у «Google Drive», а потім надавати доступ до них з «Google Colaboratory», що значно пришвидшить підготовку моделей для тренування, так як не буде існувати потреби у завантаженню даних для кожної моделі окремо, в тому випадку якщо їх формати збігаються.

Для навчання моделей використовувались наступні формати даних: YOLO (YOLOv3) та Pascal VOC XML (RetinaNet, Faster R-CNN, SSD).

Класи карієсу булі поділені наступним чином: Class1 (карієс початкової стадії), Class2 (карієс середньої тяжкості), Class3 (карієс екстенсивної стадії).

6.1 YOLOv3

Почнемо з yolov3, перед тренуванням було замінено дані у файлі конфігурації yolov3-custom.cfg: зміна параметру batch з 1 до 64; встановлено параметр max_batches на 6000, також встановлено line_step на 4800, 5400; встановлено на рядках 610, 696 та 783 значення 3; встановлено на рядках 603, 689, 776 значення 24.

Після виконання змін, було проведено навчання моделі, нижче наведені результати її виконання при поданні різних зображень.

Як можна побачити модель показує непогані результати при виявленні карієсу четвертого категорії (екстенсивна стадія), (див. рис. 15).

Рисунок 15 – Приклад знайденого карієсу екстенсивної стадії (створений власноруч)

Також модель може відносно добре знаходити карієс третього класу, якщо він має яскраво виражену помаранчеву або чорну пляму (див. рис. 16). Також модель може відрізняти чорні лінії, які також свідчать про карієс середньої тяжкості.

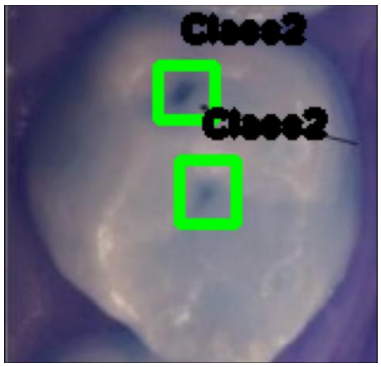

Рисунок 16 - Приклад знайденого карієсу третього класу (створений власноруч)

Найгірше модель впоралась с пошуком карієсу початкової стадії, як можна побачити з наведених малюнків, модель або бачить карієс третього класу або не бачить зовсім (див. рис. 17), на вказаному малюнку червоним кольором вказано не знайдений карієс другої стадії.

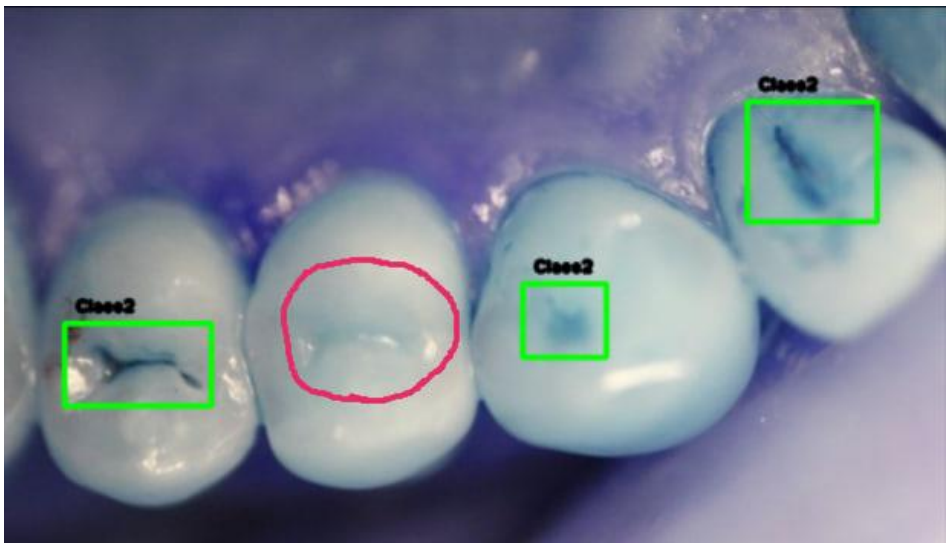

Рисунок 17 – Приклад не знайденого карієсу (створений власноруч)

Завантажимо зображення з брекетами, для перевірки чи розуміє модель, що це не карієс (див. рис. 18). Ми бачимо, що дана нейронна мережа вважає деякі брекети каріозними зубами, і помічає їх як карієс четвертої стадії. Виходячи з отриманої інформації можна зробити висновок, що модель не може відрізняти карієс від брекетів.

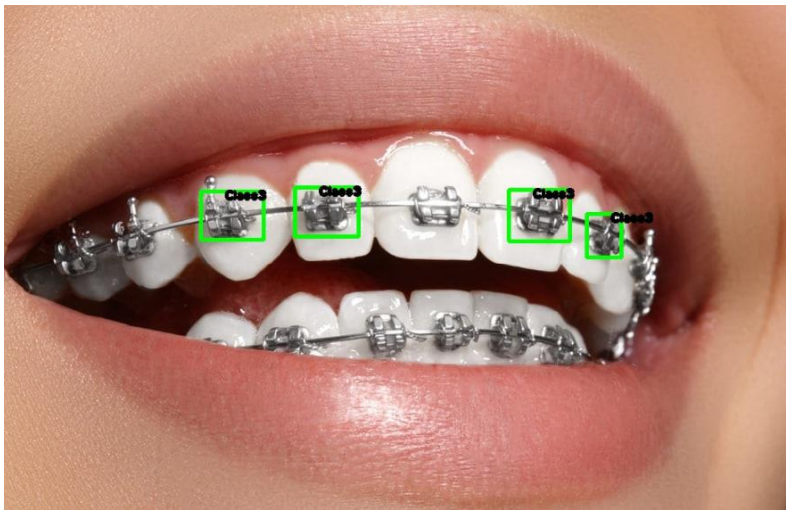

Рисунок 18 – Подання зубів з брекетами (створений власноруч)

Тепер перевіримо чи не показує дана модель карієс якщо у користувача просто жовті зуби, в своїй більшості модель показує гарний результат на звичайних жовтих зубах, і не вважає їх каріозними (див. рис. 19).

Рисунок 19 – Подання зубів з жовтим кольором (створений власноруч)

В цілому модель впоралась з пошуком карієсу важких стадій, але має проблеми з пошуком карієсу початкової стадії, та іноді плутає почервоніння ясен з карієсом, ще одним недоліком є розпізнавання брекетів, як карієсу екстенсивної стадії. Після навчання загальна вага нейронної мережі 235 мб.

6.2 SSD

При тренуванні моделі SSD, використано як попередньо навчену модель «efficientdet_d0_coco17_tpu-32.tar.gz» в якості базового пайплайну використано «ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.config».

Перед тренуванням було встановлено наступні параметри конфігурації num_steps = 40000 та batch_size = 16. Також було створено файл з переліком класів у моделі «labelmap.pbtxt».

Після внесення змін у файлах конфігурації, модель пройшла процес навчання нижче наведена її робота на різних зображеннях.

Нажаль у моделі є проблеми з пошуком карієсу екстенсивної стадії, вона або показує не усі результати карієсу останньої стадії чи не показує зовсім, червоним кольором вказано не знайдений результат (див. рис. 20).

Рисунок 20 – Приклад не знайденого карієсу екстенсивної стадії (створений власноруч)

Модель показує схожий результат з YOLOv3 при пошуку карієсу середньої тяжкості (див. рис. 21), показує найкращий результат саме при цьому пошуку. Але модель має проблеми з виявленням чорних ліній на зубі, та ідентифікує їх, в якості здорового зубу.

Рисунок 21 – Приклад знайденого карієсу середньої тяжкості (створений власноруч)

Ще одним мінусом моделі є її низька ефективність при пошуку карієсу початкової стадії, як можна побачити, модель не лише не бачить карієс початкової стадії червоне коло, а і не бачить карієс середньої тяжкості помаранчеве коло (див. рис. 22).

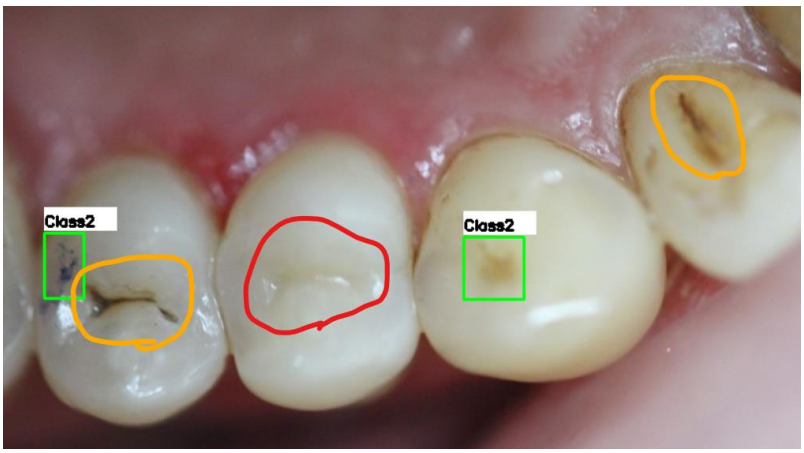

Рисунок 22 – Приклад не знайденого карієсу початкової стадії (створений власноруч)

Перевіримо, як модель реагує на зуби з брекетами (див. рис. 20), як можна побачити штучний інтелект реагує на них як на карієс останньої стадії. Нажаль ця модель також не здатна відрізнити карієс від брекетів, як і Yolov3.

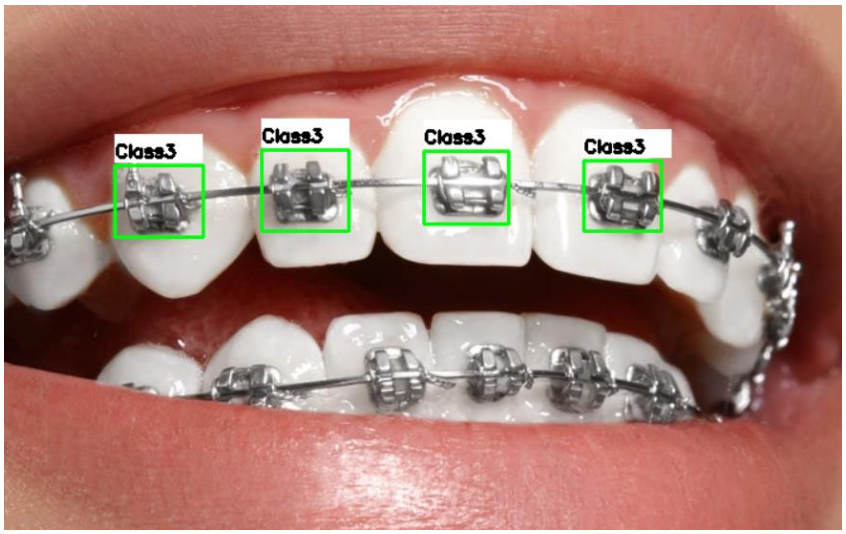

Рисунок 23 – Перевірка моделі на зубах з брекетами (створений власноруч)

Зараз ми перевіримо, як нейронна мережа реагує якщо у користувача пожовтілі зуби, як і попередня модель нейронної мережі, ця зазвичай демонструє добрі результати при звичайних жовтих зубах і не вважає їх каріозними (див. рис. 24).

Рисунок 24 – Перевірка моделі на жовтих зубах (створений власноруч)

Дана нейронна мережа показала найгірший результат при пошуку карієсу екстенсивної стадії, та має аналогічні проблеми з пошуком карієсу початкової стадії. Також вона плутає брекети з карієсом останньої стадії. Можна сказати, що дана модель найгірше впоралась з поставленим завданням.

До плюсів даної нейронної мережі можна віднести її вагу, вона займає лише 39мб, що в декілька разів менше ніж усі інші порівнювані мережі, у даній роботі. Вона показує найменшу якість при пошуку карієсу.

6.3 RetinaNet

Перед тренуванням моделі потрібно замінити у файлі «pascal_voc.py», кількість класів та замінити їх назву.

Тренування було запущено при наступних параметрах конфігурації --epochs 40 --freeze-backbone --random-transform --batch-size 1 --steps 300.

Перевіримо працездатність отриманої моделі завантаживши в неї зображення з карієсом екстенсивної стадії, в результаті опрацювання зображення ми отримуємо наступний результат (див. рис. 25). RetinaNet чудово впоралась зі своїм завданням.

Рисунок 25 – Результат пошуку карієсу екстенсивної стадії (створений власноруч)

Перегляньмо, як модель буде шукати карієс середньої тяжкості, як можна побачити дана модель не побачила другий карієс, який обведений помаранчевим кольором (див. рис. 26).

На інших переглянутих зображеннях модель має приблизно еквівалентну точність як і Yolov3.

Рисунок 26 – Результат пошуку карієсу середньої тяжкості (створений власноруч)

Нажаль модель, так само як і усі попередні показує низькі результати пошуку карієсу початкової стадії (див. рис. 27), помаранчевим колом вказано карієс який натренована нейронна модель не змогла знайти, втім модель відпрацювала краще ніж SSD, та змогла знайти карієс на четвертому зубі праворуч.

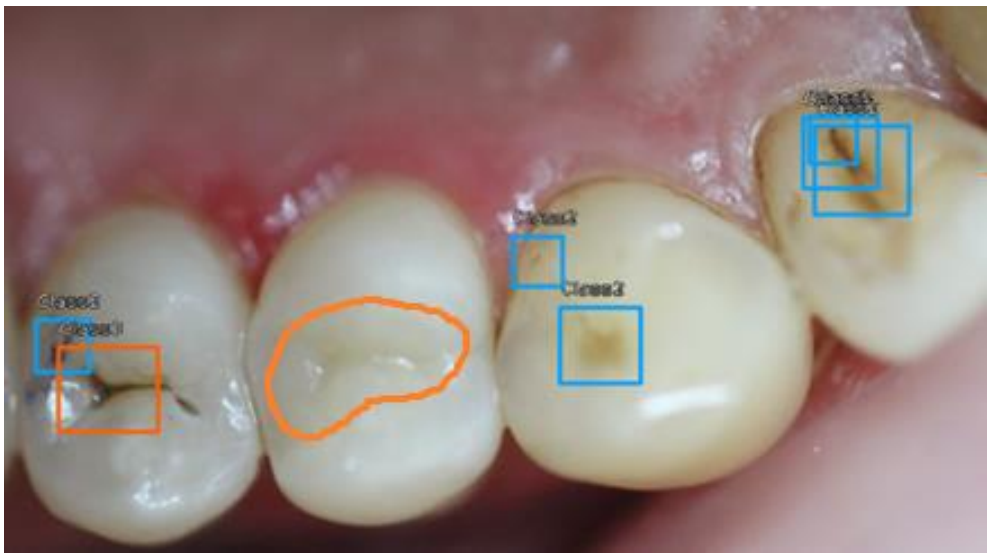

Рисунок 27 – Результат пошуку карієсу початкової стадії (створений власноруч)

Тепер перевіримо, чи дана модель реагує на зображення зубів в брекетах, як карієс (див. рис. 28). На жаль, ця нейронна мережа як і попередні також сприймає брекети як карієс екстенсивної стадії формі, та робить невірні висновки.

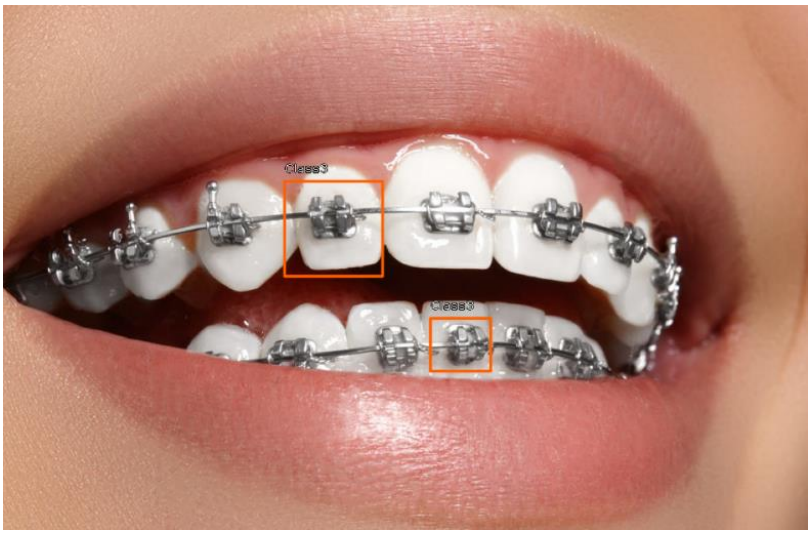

Рисунок 28 – Перевірка на фотографію з брекетами (створений власноруч)

Перегляньмо як модель впорається з розпізнаванням жовтого нальоту на наступному зображенні (див. рис. 29). Так само як SSD та YOLOv3, RetinaNet демонструє можливість розпізнавання пожовклих зубів, в якості здорових, а не каріозних.

Рисунок 29 – Перевірка моделі на жовтих зубах (створений власноруч)

Ця нейронна мережа займає 140мб, і показує непогані можливості у пошуку карієсу екстенсивної форми та карієсу середньої тяжкості. Втім як і вище описані моделі у неї є проблеми з розпізнавання карієсу початкової стадії. Нажаль вона також має проблеми з розпізнаванням брекетів, але з іншої сторони може відрізняти пожовкла зуби від каріозних. Дана модель має високу якість пошуку карієсу не дивлячись на поганий результат пошуку карієсу початкових стадій.

6.4 Faster R-CNN

Перед тренуванням замінені данні конфігурації у папці «data_configs», було замінено шлях до даних тренування та тестування, замінено назви класів (клас «__background__», не можна замінювати) та змінено параметр NC (кількість класів) на 4, бо 3 класи та клас «__background__». Тренування було запущено з наступними параметрами: --data data_configs/ppe.yaml --epochs 200 --model fasterrcnn_resnet50_fpn --name ppe_training --batch 16.

Після навчання, перевіримо як працює створена нами модель, для початку завантажимо в неї зображення з карієсом екстенсивної стадії (див. рис. 30). Як можна побачити з результату, модель також бачить карієс середньої тяжкості на місці між зубами, де є затемнення від зубів.

Рисунок 30 – Приклад пошуку карієсу екстенсивної форми (створений власноруч)

Давайте поглянемо як модель впорається з карієсом середньої тяжкості. Нейронна мережа може з високою точність знаходити карієс середньої тяжкості, але існують рідкісні моменти (див. рис. 31), коли вона не може знайти карієс третього класу, але це виключення із правила, помаранчевим кольором зображено карієс який модель не змогла знайти.

Рисунок 31 – Приклад пошуку середньої тяжкості (створений власноруч)

Перейдемо до перегляду карієсу початкової стадії (див. рис. 32), як можна побачити, ця нейронна мережа також не змогла знайти карієс початкової стадії, але досить добре відображує карієс середньої тяжкості на наведеному малюнку, але як було описано вище інколи модель сприймає тінь від зубів чи перехід між зубом та ясною, видно на третьому зубі вгорі.

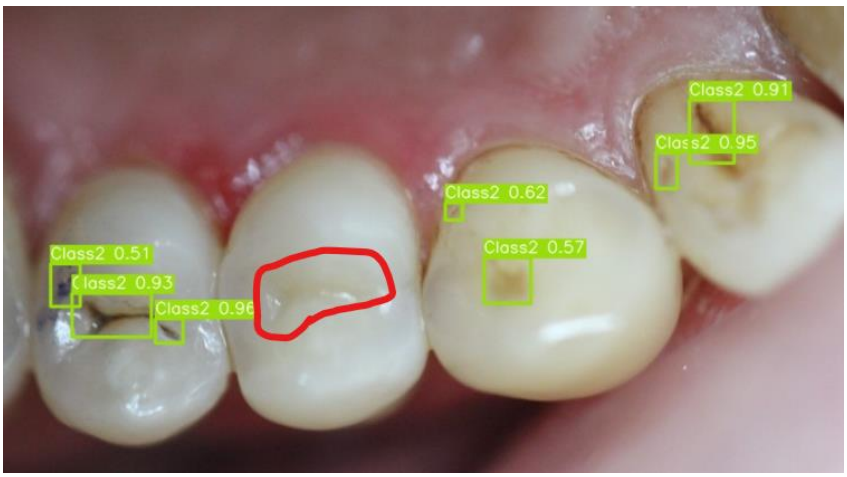

Рисунок 32 – Приклад невдалого пошуку карієсу початковою стадії (створений власноруч)

Faster R-CNN, як і інші моделі не може впізнати де карієс, а де брекети. І як всі вище описані моделі сприймає брекети за карієс екстенсивної форми. З чого можна зробити висновок, що проблема в датасеті, який не бере до уваги зуби з брекетами (див. рис. 33).

Рисунок 33 – Приклад роботи моделі з брекетами (створений власноруч)

Тепер давайте перевіримо, як дана нейронна мережа працює з зубами які мають жовтий наліт (див. рис. 34). Можна побачити, що Faster R-CNN знову ж таки, сприйняла ясну між двома зубами за карієс, бо вони затіняються. В цілому можна сказати, що дана модель може зрозуміти коли зуби просто жовті, а коли вони каріозні.

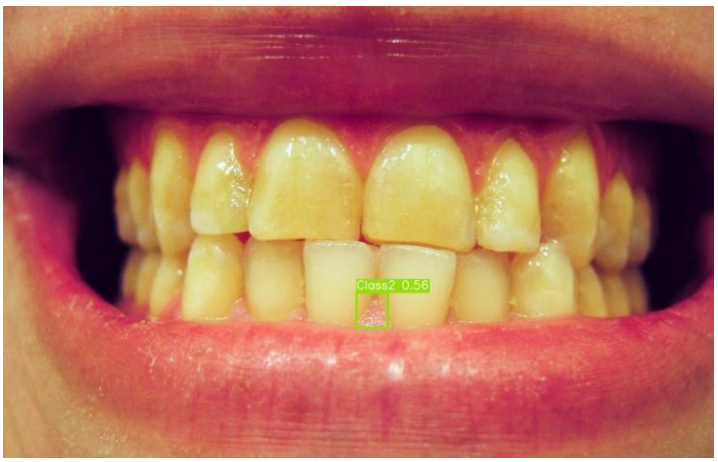

Рисунок 34 – Приклад роботи моделі з жовтими зубами (створений власноруч)

Дана модель займає найбільше місця приблизно 660мб, вона дуже добре знаходить карієс екстенсивної форми та середньої тяжкості, але дуже часто робить неправильні здогадки коли зуби створюють тінь один на одний, або коли під зубом є запалення ясен. Також модель може розрізняти, коли зуби просто жовті, а коли вони мають каріозні плями.

7 АНАЛІЗ РЕЗУЛЬТАТІВ ПРОВЕДЕНОГО ДОСЛІДЖЕННЯ

Як можна було побачити, з вище оглянутих досліджень усі моделі мають великі проблеми з виявленням карієсу початкової стадії, насправді цей карієс є одним з найважчих для діагностики і навіть досвідчені стоматологи іноді можуть робити невірні припущення. Результати `порівняння моделей наведені нижче у таблиці 2, варто зазначити, що модель SSD зробила нуль вірних виборів (TP). Найбільшу точність має модель Faster R-CNN, далі за нею йде модель, далі за нею йде YOLOv3. Жодна модель не показала задовільної точності для її можливого використання для пошуку карієсу початкової стадії, втім краще за все себе показала Faster R-CNN, але модель часто робить невірні допущення через відсвічування світла від спалаху на зубах чи яснах.

Таблиця 2 – Результат проведеного досліду для карієсу початкової стадії (виконана власноруч)

Модель DL	TPR	TNR	Accuracy	Precision
YOLOv3	22	14	18,1	18,6
SSD	0	21	13	0
RetinaNet	10	32	23,6	8
Faster R-CNN	27	23	25	23

Огляньмо результати оцінки моделей при пошуку карієсу середньої тяжкості у таблиці 3.

Таблиця 3 – Результат проведеного досліду для карієсу середньої тяжкості (виконана власноруч)

Модель DL	TPR	TNR	Accuracy	Precision
YOLOv3	80	14	60,6	66
SSD	61	36,8	50	55
RetinaNet	77	13	45,8	47,4
Faster R-CNN	78,9	50	66,3	67,1

Цей карієс легко знаходиться будь-яким стоматологом, тому моделі також показали у цілому непоганий результат при пошуку карієсу третьої стадії, через яскраву вираженість даної стадії та помітні чорні смуги по всьому зубу.

Перейдемо до результатів роботи моделі з карієсом екстенсивної стадії. В даному випадку, моделі показали чудовий результат, через яскраво виражену деформацію зуба, оголений дентин та чорний колір. Усі моделі окрім SSD показали гарний результат у цьому завданні.

Таблиця 4 – Результат проведеного досліду для карієсу екстенсивної стадії (виконана власноруч)

Модель DL	TPR	TNR	Accuracy	Precision
YOLOv3	83	62,5	76	80
SSD	59	37	46	40
RetinaNet	73	51	64,1	67
Faster R-CNN	82,8	63	76,7	82

Підсумовуючи результати проведених експерементів, можна зробити висновок, що найкраща з усіх обраних моделей штучного інтелекту для пошуку карієсу себе показала Faster R-CNN, після неї йде YOLOv3 яка теж показала непоганий результат. Відповідно найгірше себе продемонстрували RetinaNet та SSD. Можна зробити висновок, що обрані моделі можуть непогано знайти карієс середньої та важкої стадії, втім мають значні проблеми з пошуком карієсу початкової стадії. Тож результати дослідження вказують на великий потенціал YOLOv3 і Faster R-CNN у практичному застосуванні.

Варто зазначити, що кожна модель плутала брекети з карієсом екстенсивної стадії. Хоча суто з точки зору логіки перед встановленням брекетів власник проходить повне лікування зубів, також протягом кожного одного-двох місяців необхідно приходити до ортодонта, який постійно проводить огляд.

До плюсів навчених моделей також можна віднести їх здатність розрізняти карієс від просто пожовклих зубів, кожна модель впоралась з поставленим завданням.

До мінусів можна віднести, те що спалах світла від камери часто призводить до розпізнавання моделями, як карієс початкової чи середньої тяжкості.

Серед усіх нейронних мереж найбільше місця займає Faster R-CNN приблизно 660мб, далі йде Yolov3 і займає 235мб, за нею йде RetinaNet з розміром 140мб і завершує найменша модель SSD з розміром лише 39мб.

Важливо відмітити, що навчання проводилось на відносно обмеженому наборі даних, і при їх збільшенні результати можуть поліпшитись. Також можна використовувати модифікації вищеописаних моделей.

8 РЕАЛІЗАЦІЯ ДОДАТКУ

Для того, щоб кожен міг дізнатися чи є в нього карієс було вирішено реалізувати додаток який опрацьовує відправлений POST запит, і повертає відповідь з координатами та класом карієсу, в якості WEB API. Проект має наступну структуру (див. рис. 35).

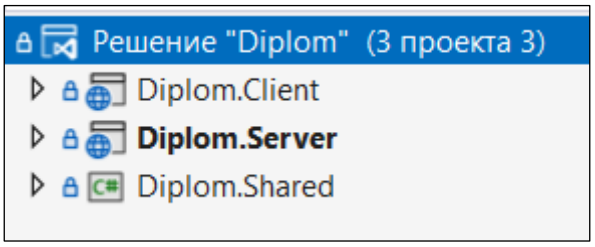

Рисунок 35 – Структура проекту (створений власноруч)

У проекті Diplom.Client знаходиться BlazorWebAssembly код, який відповідає за UI користувача. У проекті Diplom.Server знаходиться код який відповідає за обробку даних з контролерів та повертає дані користувачу. У проекті Diplom.Shared знаходяться DTO контракти.

Почнемо опис з проекту Diplom.Shared, він містить файли DTO контрактів, якими відповідають WEB API контролеру, а також усі енами які використовуються розробленою бізнес логікою, клас приклад DTO відповіді виглядає наступним чином:

```
public class CariesViewItem
{
   public string CariesType { get; set; } = default!;

   public float X { get; set; }

   public float Y { get; set; }

   public float Height { get; set; }

   public float Width { get; set; }

   public float Confidence { get; set; }
}
```

Як можна побачити, цей клас містить координати карієсу та його точність знаходження, а також назву типу карієсу.

В свою чергу клас CariesResponce, містить колекцію CariesViewItem, для опису карієсу на зображені.

Перейдемо до проекту Diplom.Server, в цьому проекті знаходяться уся бізнес логіка розробленого додатку, на вхід до контролерів надходять фотографії зубів, далі контролер викликає бізнес логіку яка повертає користувачам дані про координати карієсу. Для прикладу візьмемо контролер, який відповідає за розпізнавання карієсу за допомогою моделі yolov3 має наступний вигляд:

```
[ApiController]
    [Route("api/[controller]")]
    public class Yolov3Controller : ControllerBase
    {
        private readonly IYolov3CariesPredictor _service;

        public Yolov3Controller(IYolov3CariesPredictor service)
        {
            _service = service;
        }

        [HttpPost("{minScore:float?}")]
        public async Task<IActionResult> GetCariesCoords(IFormFile file,
float? minScore)
        {
            var result = minScore is null
                ? await
_service.GetCariesResponceAsync(file.OpenReadStream(), file.FileName)
                : await
_service.GetCariesResponceAsync(file.OpenReadStream(), file.FileName,
minScore.Value);

            return Ok(result);
        }
    }
```

На вхід до нього надходить файл та мінімальний відсоток точності, при якому на відповідь будуть повертатись координати. У конструкторі він отримує сервіс який відповідає за розпізнавання карієсу за допомогою моделі Yolov3. Сервіс реалізує інтерфейс ICariesPredictor, який має одну функцію GetCariesResponceAsync, яка повертає дані про координати карієсу і іншу допоміжну інформацію. На вхід цієї функції надходять наступну аргументи

Stream, filename та необов'язковий аргумент minScore. Реалізація сервісу Yolov3CariesPredictor має наступний вигляд:

```
public class Yolov3CariesPredictor : IYolov3CariesPredictor
    {
        private readonly MLContext _mlContext;
        private readonly string imagesFolder;
        private readonly string modelFilePath;

        public Yolov3CariesPredictor(MLContext mlContext,
IWebHostEnvironment environment)
        {
            _mlContext = mlContext;
            modelFilePath = Helper.GetModelPath(AIModels.Yolov3,
environment);
            imagesFold+er =
Path.Combine(Helper.GetImagesFolder(environment), new Guid().ToString());
            Directory.CreateDirectory(imagesFolder);
        }

        public Task<CariesResponce> GetCariesResponceAsync(Stream stream,
string filename, float minScore)
        {
            var responce = new CariesResponce();

            using (var fileStream = File.Create(Path.Combine(imagesFolder,
filename)))
            {
                stream.Seek(0, SeekOrigin.Begin);
                stream.CopyTo(fileStream);
            }

            List<ImageNetData> images =
ImageNetData.ReadFromFile(imagesFolder).ToList();
            IDataView imageDataView =
_mlContext.Data.LoadFromEnumerable(images);

            var score = new Yolov3ModelScorer(imagesFolder, modelFilePath,
_mlContext);

            // Use model to score data
            IEnumerable<float[]> probabilityList =
modelScorer.Score(imageDataView);

            // Post-process model output
            YoloOutputParser parser = new YoloOutputParser();

            var boundingBoxes =
                probabilityList
                .Select(item => parser.ParseOutputs(item))
                .Select(item => parser.FilterBoundingBoxes(item, 5,
minScore));

            for (var i = 0; i < images.Count(); i++)
            {
```

```
                responce.FileName = images.ElementAt(i).Label;
                responce.Caries = boundingBoxes
                    .ElementAt(i)
                    .Select(item => new CariesViewItem()
                    {
                        Confidence = item.Confidence,
                        Height = item.Dimensions.Height,
                        Width = item.Dimensions.Width,
                        X = item.Dimensions.X,
                        Y = item.Dimensions.Y,
                        CariesType = item.Label
                    }).ToList();

                string imageName = images.ElementAt(i).Label;
                IList<YoloBoundingBox> detectedObjects =
boundingBoxes.ElementAt(i);

                Helper.LogDetectedObjects(imageName, detectedObjects);
            }

            return Task.FromResult(responce);
        }
    }
```

Цей клас реалізує IYolov3CariesPredictor, який є простим наслідником вище описаного класу ICariesPredictor. У конструктор класу до нас приходить створена модель контексту та змінна оточення, далі конструктор створює необхідні директорії та створює змінні шляху для моделі та файлів. Тепер розглянемо як працює метод GetCariesResponceAsync, першу за все він зберігає файл у створеній директорії для запиту, далі зчитується усі зображення які були збережені у директорії зображень. Після цього ці зображення завантажуються у модель, далі створюється трансформер який є допоміжним класом для бізнес логіки. Після цього виконуємо аналіз завантажених зображень обраною моделлю. Далі створюємо допоміжний клас YoloOutputParser, який допомагає обробляти вже отриманні дані і відкидає знайдені невалідні результати. Після цього дані перетворюються у DTO клас CariesResponce, логує виконані дії та повертає результат користувачу.

Для вибору моделі яку необхідно завантажити використовується допоміжний клас, який зберігає шляхи до кожної нейронної моделі, код методу отримання шляху до цієї моделі наведено нижче:

```
public static string GetModelPath(AIModels model, IWebHostEnvironment
environment = null!)
    {
        switch (model)
        {
            case AIModels.Yolov3:
                return Path.Combine(GetAssetsPath(environment), "Models",
"yolov3_model.onnx");
            case AIModels.SSD:
                return Path.Combine(GetAssetsPath(environment), "Models",
"ssd_model.onnx");
            case AIModels.RetinaNet:
                return Path.Combine(GetAssetsPath(environment), "Models",
"retinanet_model.onnx");
            case AIModels.Faster_Rcnn:
                return Path.Combine(GetAssetsPath(environment), "Models",
"faster_rcnn_model.onnx");
        }

        throw new NotImplementedException($"Such model don`t implemented");
         }
```

Як можна побачити, до методу надходить енам з назвою моделі та змінною оточення ASP.NET Core, а функція повертає шлях до цієї моделі. Усі моделі знаходяться у папці Asset (див. рис. 36).

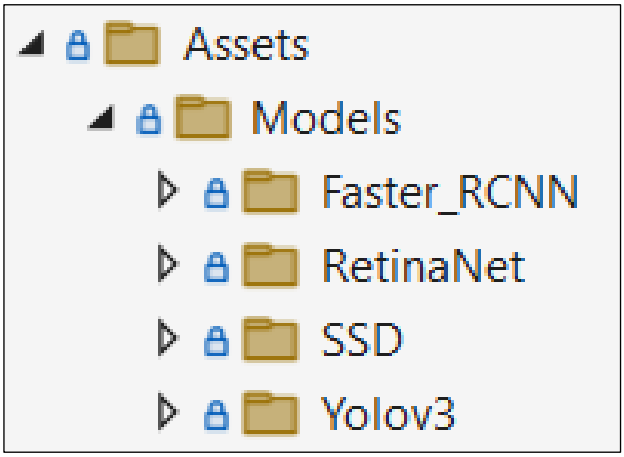

Рисунок 36 – Папка з нейронними мережами (створений власноруч)

За допомогою Swagger, ми можемо побачити наявні WEB API (див. рис. 37), для візуалізації файлу свагера використовується бібліотека «Swashbuckle.AspNetCore.SwaggerUI».

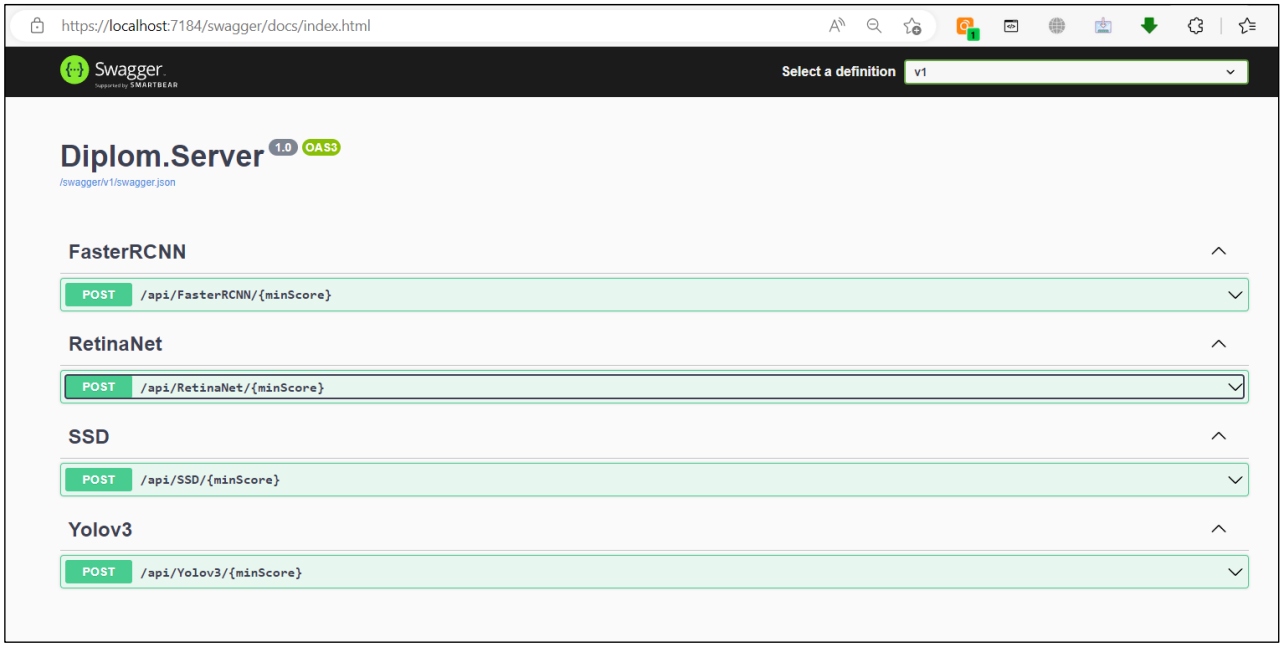

Рисунок 37 – SwaggerUI (створений власноруч)

Тепер огляньмо інтерфейс користувача, він перейдемо на сторінку і виберемо пункт Yolov3, завантажена сторінка буде мати наступний вигляд (див. рис. 38).

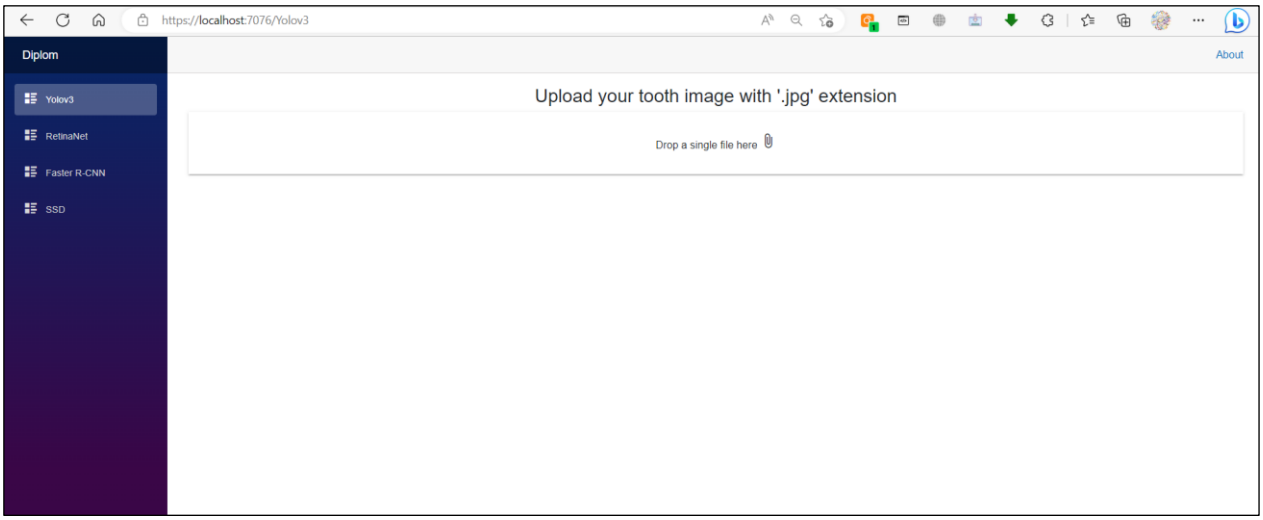

Рисунок 38 – Інтерфейс користувача на сторінці yolov3 (створений власноруч)

Тепер натиснемо кнопку завантаження зображення, і після завантаження зліва появиться завантажене зображення, а після опрацювання серверу справа з’явиться оброблена фотографія (див. рис. 39), з вказаним класом карієсу.

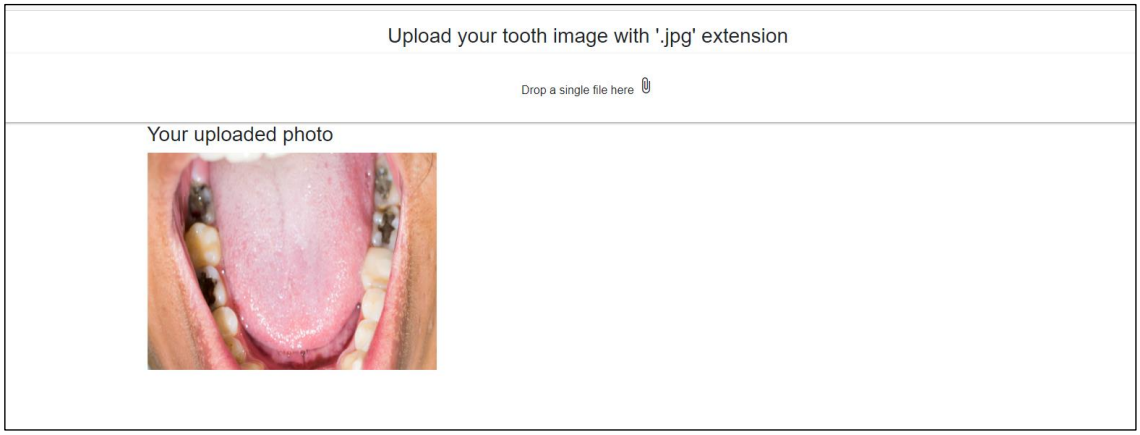

Рисунок 39 – Завантаження зображення (створений власноруч)

Під час очікування відповіді з’являється колесо яке показує очікування відповіді. І через деякий час, після отримання відповіді від серверу буде завантажено малюнок з позначеним карієсом (див. рис. 40).

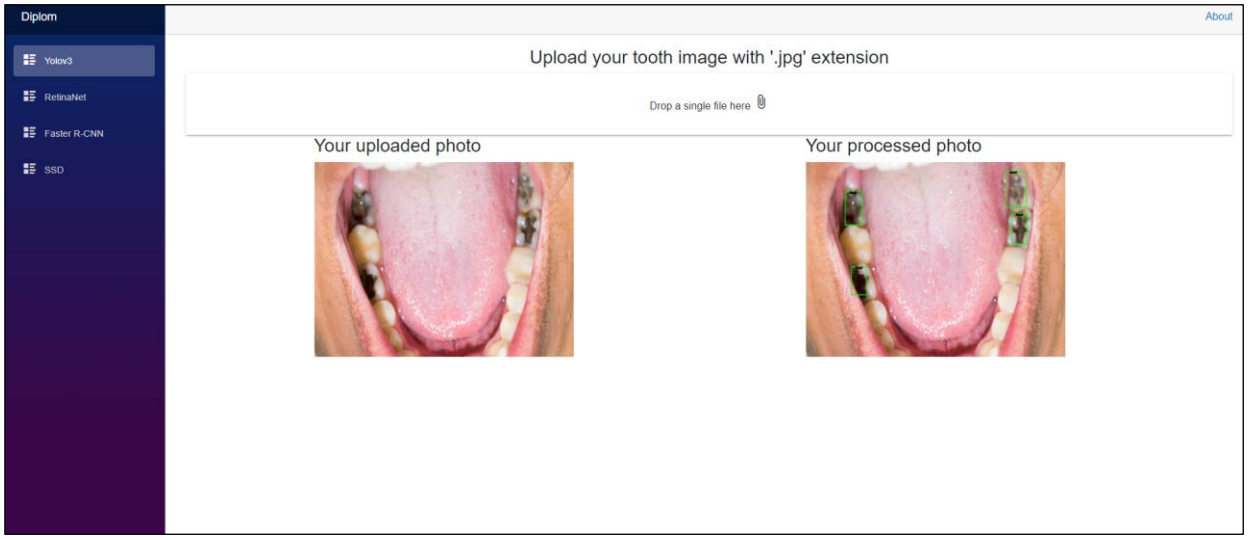

Рисунок 40 – Приклад отриманої відповіді від серверу (створений власноруч)

Завдяки тому, що був використаний Blazor ми можемо використовувати той самий DTO контракт, що значно зменшує час налаштування користувацького інтерфейсу, втім через те зо ми використали WEB API в будь який час можна буде написати інший інтерфейс користувача, використовуючи ті самі API.

ВИСНОВКИ

В результаті дослідження методів виявлення стоматологічних захворювань на зображеннях, було проведено аналіз предметної області, а саме основні дентальні хвороби людей. Було проаналізовано наявну статистику про кількість стоматологічних захворювань в Україні. Проаналізована проблема великої кількості пізніх стадій захворювань у українців. Була розглянута класифікація карієсу за допомогою міжнародної системи ICCMS.

Було визначено основні методи глибокого навчання, які найкраще підходять для пошуку проблем із ротовою порожниною, а саме: RetinaNet, YOLOv3, SSD, Faster R-CNN.

Було спроектовано та реалізовано програмний додаток, який допомагає зрозуміти чи наявні у користувача проблеми з зубами.

Результатом виконання дослідження, стало порівняння цих методів за наступними показниками Чутливість (TPR), Специфічність (TNR), Accuracy, Precision.

Результат наведених досліджень може бути покращений за рахунок збільшення кількості фото у наявному датасеті. Також у майбутньому можна використовувати модифікації вищеописаних методів, які вузькоспеціалізовані на стоматологічній тематиці.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Як часто потрібно відвідувати стоматолога: особливості візиту та що все-таки очікувати пацієнту? [Електронний ресурс] // Вісник Розділля. – Режим доступу: Як часто потрібно відвідувати стоматолога: особливості візиту та що все-таки очікувати пацієнту? - «Вісник Розділля» (visrozdil.lviv.ua) (Дата звернення: 25.02.2023).

2. Карієс – Вікіпедія [Електронний ресурс] // wikipedia.org. – Режим доступу: Карієс зубів – Вікіпедія (wikipedia.org) (Дата звернення: 25.01.2023).

3. Дентальний карієс [Електронний ресурс] // iccms-web.com. – Режим доступу: 59284654c0a6f822230100.pdf (iccms-web.com) (Дата звернення: 28.02.2023).

4. Construction of a dental caries prediction model by data mining [Електронний ресурс] // jst.go.jp. – Режим доступу: Construction of a dental caries prediction model by data mining (jst.go.jp) (Дата звернення: 15.03.2023).

5. Algorithmic analysis for dental caries detection using an adaptive neural network architecture: [Електронний ресурс] // Heliyon (cell.com). – Режим доступу: Algorithmic analysis for dental caries detection using an adaptive neural network architecture: Heliyon (cell.com) (Дата звернення: 15.03.2023).

6. Study of Methods for Determining Types and Measuring of Agricultural Crops due to Satellite Images [Електронний ресурс] // ieeexplore.ieee.org. – Режим доступу: Study of Methods for Determining Types and Measuring of Agricultural Crops due to Satellite Images | IEEE Conference Publication | IEEE Xplore (Дата звернення: 21.03.2023).

7. JCM | Free Full-Text | Dental Caries Diagnosis and Detection Using Neural Networks: A Systematic Review [Електронний ресурс] // Journal of Clinical Medicine (mdpi.com). – Режим доступу: JCM | Free Full-Text | Dental Caries Diagnosis and Detection Using Neural Networks: A Systematic Review (mdpi.com) (Дата звернення: 15.03.2023).

8. Detecting white spot lesions on dental photography using deep learning: A pilot study [Електронний ресурс] // ScienceDirect. – Режим доступу: Detecting white

spot lesions on dental photography using deep learning: A pilot study - ScienceDirect (Дата звернення: 15.03.2023).

9. Detection of dental caries in oral photographs taken by mobile phones based on the YOLOv3 algorithm - Ding [Електронний ресурс] // Annals of Translational Medicine (amegroups.com). – Режим доступу: Detection of dental caries in oral photographs taken by mobile phones based on the YOLOv3 algorithm - Ding - Annals of Translational Medicine (amegroups.com) (Дата звернення: 15.03.2023).

10. A smart home dental care system: integration of deep learning, image sensors, and mobile controller [Електронний ресурс] // SpringerLink. – Режим доступу: A smart home dental care system: integration of deep learning, image sensors, and mobile controller | SpringerLink (Дата звернення: 15.03.2023).

11. Evaluation of a deep learning system for automatic detection of proximal surface dental caries on bitewing radiographs [Електронний ресурс] // PubMed (nih.gov). – Режим доступу: Evaluation of a deep learning system for automatic detection of proximal surface dental caries on bitewing radiographs - PubMed (nih.gov) (Дата звернення: 15.03.2023).

12. Home - DentalMonitoring [Електронний ресурс] // DentalMonitoring. – Режим доступу: Home - DentalMonitoring (dental-monitoring.com) (Дата звернення: 10.02.2023).

13. Faster R-CNN: Down the rabbit hole of modern object detection [Електронний ресурс] // tryolabs.com. – Режим доступу: Faster R-CNN: Down the rabbit hole of modern object detection | Tryolabs (Дата звернення: 19.03.2023).

14. YOLO for Object Detection, Architecture Explained [Електронний ресурс] // medium.com. – Режим доступу: YOLO for Object Detection, Architecture Explained! | by Sairaj Neelam | Analytics Vidhya | Medium (Дата звернення: 19.03.2023).

15. SSD architecture [Електронний ресурс] // manning.com. – Режим доступу: base-network in deep-learning (manning.com) (Дата звернення: 19.03.2023).

16. How RetinaNet works? [Електронний ресурс] // developers.arcgis.com. – Режим доступу: How RetinaNet works? | ArcGIS API for Python (Дата звернення: 19.03.2023).

17. Detecting tooth decay and cavities using Tensorflow Object Detection API [Електронний ресурс] // github.com – Режим доступу: atul-g/object_detection_on_cavities: Detecting cavities or tooth decay on images of teeth using Tensorflow's Object Detection API. (github.com) (Дата звернення: 28.02.2023).

18. Classifying Images of Teeth Having Cavity [Електронний ресурс] // github.com. – Режим доступу: https://github.com/atul-g/cavity_detection (Дата звернення: 28.02.2023).

19. Шевчук О. О. Пошук каріозних захворювань на зображеннях за допомогою штучного інтелекту// Радіоеектроніка та молодь у XXI столітті. Тези доповідей 27-го Міжнародний молодіжний форуму 10 – 12 травня 2023 року.– Харків, 2023, т. 6, ч. 1. с. 357.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

6. Study of Methods for Determining Types and Measuring of Agricultural Crops due to Satellite Images [Електронний ресурс] // ieeexplore.ieee.org. – Режим доступу: Study of Methods for Determining Types and Measuring of Agricultural Crops due to Satellite Images | IEEE Conference Publication | IEEE Xplore (Дата звернення: 21.03.2023).

19. Шевчук О. О. Пошук каріозних захворювань на зображеннях за допомогою штучного інтелекту // Радіоеектроніка та молодь у XXI столітті. Тези доповідей 27-го Міжнародний молодіжний форуму 10 – 12 травня 2023 року.– Харків, 2023, т. 6, ч. 1. с. 357.