Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ПОБУДОВИ МОДЕЛІ МАШИННОГО НАВЧАННЯ
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-19-1
Захар'єв О.І.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Кобилін О.А.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Захар'єву Олексію Івановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка вебзастосунку для побудови моделі машинного навчання

затверджена наказом університету від 15 травня 2023 року № 474 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 29 травня 2023 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотека для використання моделей машинного навчання TensorFlow.js.

4. Перелік питань, що потрібно опрацювати в роботі.

1. Огляд методів оптичного розпізнавання тексту на зображенні.
2. Алгоритми пошуку та обробки зображення у відеопотоці.
3. Огляд основних моделей машинного навчання.
4. Розпізнавання зображення у відеопотоці.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми обробки зображень, постановка задачі, тестові зображення.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Творошенко І.С.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	10.04.2023	
2	Аналіз завдання, підбір літератури	11.04.23-18.04.23	
3	Аналіз літератури з досліджуваної проблеми	18.04.23-20.04.23	
4	Аналіз технічних засобів	21.04.23-12.05.23	
5	Розробка застосунку	12.05.23-14.05.23	
6	Програмна реалізація	15.05.23-20.05.23	
7	Оформлення пояснювальної записки	20.05.23-26.05.23	
8	Перевірка на плагіат	31.05.23	
9	Рецензування	01.06.23	
10	Підготовка презентації та доповіді	02.06.23-03.06.23	
11	Занесення роботи в електронний архів	04.06.23	
12	Попередній захист кваліфікаційної роботи	06.06.23	

Дата видачі завдання 10 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ _____доц. Кобилін О.А._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 63 с., 2 табл., 19 рис., 30 джерел.

РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ, МАШИННЕ НАВЧАННЯ, ЛІНІЙНА РЕГРЕССІЯ, ІНТЕГРАЦІЯ РОЗПІЗНАВАННЯ ЗОБРАЖЕННЯ, ПРОЄКТУВАННЯ СЕРВІСУ.

Об’єктом роботи є процес детектування зображення у відеопотоці та розпізнавання зображення на фотографії за для його подальшої класифікації.

Мета роботи полягає у створенні вебзастосунку, який дозволятиме користувачам побудувати різні моделі машинного навчання.

У ході роботи наводиться аналітичний огляд існуючих технічних рішень та алгоритмів детектування зображення у відеопотоці та оптичного розпізнавання тексту на зображенні. Було розроблено прототип застосунку, для реалізації різноманітних моделей машинного навчання.

Використано модель машинного навчання для класифікації стану різноманітних запчастин та механізмів.

Для демонстрації програми був створений вебзастосунок який був протестований у браузерах Google Chrome та Mozilla Firefox під операційною системою Windows 10.

IMAGE RECOGNITION, MACHINE LEARNING, LINEAR REGRESSION, IMAGE RECOGNITION INTEGRATION, SERVICE DESIGN.

The object of the work is the process of image detection in a video stream and image recognition in a photograph for its further classification.

The goal is to create a web application that will allow users to build various machine learning models.

The study provides an analytical review of existing technical solutions and algorithms for image detection in a video stream and optical text recognition in an image. A prototype application was developed to implement various machine learning models.

A machine learning model was used to classify the condition of various spare parts and mechanisms.

To demonstrate the program, a web application was created and tested in Google Chrome and Mozilla Firefox browsers under the Windows 10 operating system.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

МН – машинне навчання

РЗ – розпізнавання зображень

ОС – операційна система

МРТ – магнітно-резонансна томографія

API – Application Programming Interface (прикладниий програмний інтерфейс)

GUI – Graphical user interface (графічний інтерфейс користувача)

ВСТУП

Розвиток інформаційних технологій та зростання обсягів даних стали причиною появи нових напрямків досліджень, серед яких особливе місце займає машинне навчання та обробка зображень. Ця галузь науки інтенсивно розвивається та знаходить все більше застосування в різних сферах життя – від аналізу супутникових знімків до медичної візуалізації. Здатність видобувати значущу інформацію з візуальних даних стала вирішальною в різних галузях, що зумовило попит на більш ефективні та точні методи обробки зображень.

Машинне навчання є підгалуззю штучного інтелекту, яка дозволяє комп’ютерам вчитися на основі даних та вирішувати складні завдання без явного програмування. Воно використовує алгоритми і статистичні моделі для виявлення закономірностей і прихованих структур у зображеннях, тим самим підвищуючи точність, ефективність і надійність завдань обробки зображень. Крім того, методи машинного навчання дозволяють системам адаптуватися і розвиватися з часом, що дає їм змогу вирішувати нові завдання і працювати з різноманітними наборами даних.

У даній кваліфікаційній роботі проводиться аналіз теоретичних аспектів машинного навчання, методів та алгоритмів вирішення задач, а також їх застосування.

Основним завданням роботи є огляд сучасного стану та перспектив машинного навчання, розробка та апробація алгоритмів та методів машинного навчання на практичних прикладах, а також оцінка їх ефективності та перспектив в застосуванні.

Наукова і практична новизна полягає в розробці і реалізації вебзастосунку, що використовує побудовану модель МН з класифікації зображень для подальшої взаємодії із застосунком. Беручи за основу отриману модель та програмну реалізацію, розроблену навколо неї, можна зробити безліч корисних проєктів від простих побутових до складних технічних систем.

1 ОГЛЯД МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ

Застосування машинного навчання у наш час є досить розповсюдженим і відбувається в різних галузях, таких як медицина, фінанси, автопромисловість, реклама та багато інших. Розпізнавання зображень з використанням моделей машинного навчання, зараз також дуже активно набирає обертів.

1.1 Аналіз предметної області

Розпізнавання зображень є галуззю комп'ютерної науки та штучного інтелекту, яка займається автоматичним аналізом та інтерпретацією зображень. Це означає, що системи розпізнавання зображень можуть зчитувати та розуміти зображення, які подаються у вигляді цифрових даних, і здатні виконувати завдання, які зазвичай вимагають людського зору та розуміння. Розпізнавання зображень знайшло застосування у багатьох галузях, включаючи медицину, безпеку, автомобільну промисловість, розваги, маркетинг, транспорт та інші. Наприклад, у медицині системи розпізнавання зображень можуть використовуватися для автоматичної діагностики захворювань на основі медичних зображень, таких як рентгенівські знімки та знімках магнітно-резонансної томографії. МРТ дає найдетальніші зображення м'яких тканин, таких як мозок, і широко використовується для діагностики пухлин головного мозку (рис. 1.1) [1].

Основна складність полягає в тому, що такі пухлини можуть бути різних форм і розмірів. Чим раніше буде виявлено аномалію, тим більше шансів на позитивний результат лікування. А методи машинного навчання мають потенціал значно пришвидшити локалізацію пухлини.

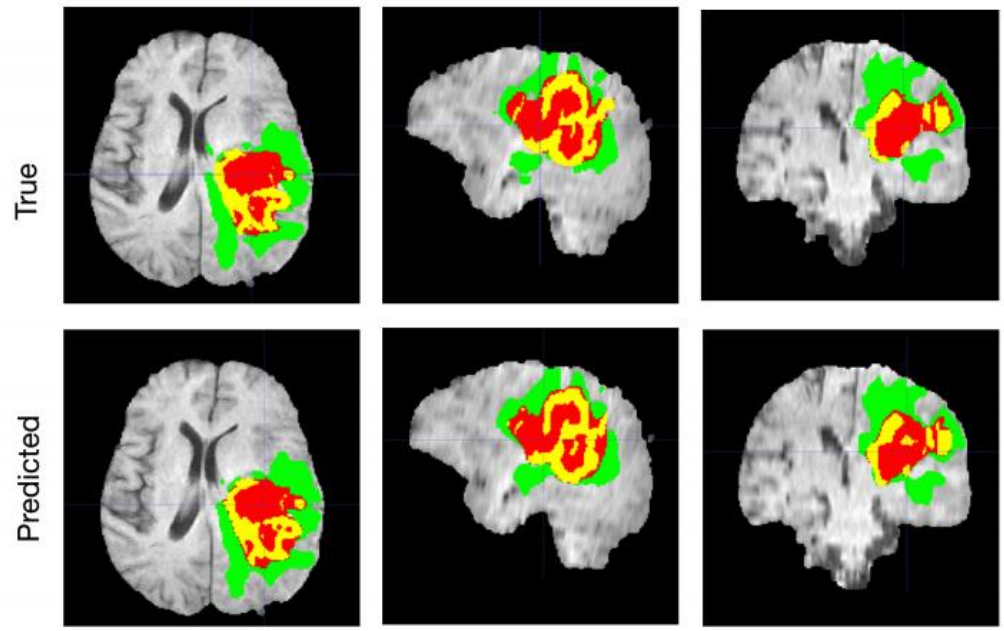

Рисунок 1.1 – Результат локалізації пухлин головного мозку за допомогою методів машинного навчання

У галузі безпеки, розпізнавання зображень можуть допомогти відслідковувати та розпізнавати обличчя людей, що перетинають кордони або проникають у обмежені зони. У розвагах розпізнавання зображень може бути використано для автоматичної ідентифікації обличчя та рухів гравців у відеоіграх. За допомогою навчання з підсиленням або глибоким навчанням, системи розпізнавання зображень можуть стати все більш точними та ефективними у виконанні завдань, що раніше вимагали багато часу та зусиль людей. Це може покращити ефективність та точність багатьох процесів, а також зменшити витрати часу та ресурсів у виробничих та інших сферах.

1.2 Огляд основних моделей машинного навчання

Машинне навчання – це галузь штучного інтелекту, що досліджує розробку алгоритмів та моделей, які дають змогу комп’ютерам вчитися на основі даних та здійснювати прогнози на основі цих даних. Моделі машинного

навчання мають широкі застосування в багатьох сферах, включаючи медицину, фінанси, маркетинг, науку про матеріали та інші. Їх використання дозволяє вирішувати складні задачі, для яких неможливо створити аналітичні моделі. Наприклад, моделі машинного навчання можуть бути використані для розпізнавання образів, прогнозування тенденцій на фондовому ринку, виявлення шахрайства, класифікації текстів та багато іншого. Крім того, моделі машинного навчання дозволяють автоматизувати процеси та зменшувати людський фактор в прийнятті рішень. Вони можуть працювати з великою кількістю даних та адаптуватися до змін у реальному часі, що дозволяє їм бути ефективними в багатьох задачах.

1.2.1 Лінійна регресія

Лінійна регресія – один з найпростіших і найпопулярніших алгоритмів машинного навчання. Це статистичний метод, який використовується для прогнозного аналізу. Лінійна регресія робить прогнози для безперервних/реальних або числових змінних, таких як продажі, зарплата, вік, ціна продукту тощо.

Алгоритм лінійної регресії показує лінійний зв’язок між залежною (y) та однією або декількома незалежними (x) змінними, тому і називається лінійною регресією. Оскільки лінійна регресія показує лінійний зв’язок, це означає, що вона знаходить, як значення залежної змінної змінюється відповідно до значення незалежної змінної.

Лінійну регресію можна розділити на два типи алгоритму. Перший тип, це проста лінійна регресія, коли одна незалежна змінна використовується для прогнозування значення числової залежної змінної, то такий алгоритм лінійної регресії називається простою лінійною регресією. Другий тип, це множинна лінійна регресія, коли для прогнозування значення числової залежної змінної

використовується більше однієї незалежної змінної, то такий алгоритм лінійної регресії називається множинною лінійною регресією.

Модель лінійної регресії представляє зв'язок між змінними у вигляді нахиленої прямої лінії (рис. 1.2).

Рисунок 1.2 – Приклад простої лінійної регресії з однією незалежною змінною

Загальна лінійна регресійна модель має вигляд:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \varepsilon,$$

де y – залежна змінна (цільова змінна);

x_1 – незалежна змінна (змінна-предиктор);

β_0 – перетин лінії (надає додатковий ступінь вільності);

β_1 – коефіцієнт лінійної регресії (масштабний коефіцієнт для кожного вхідного значення);

ε – випадкова похибка (відображає різницю між фактичними значеннями вихідної змінної та прогнозованими значеннями).

1.2.2 Метод найменших квадратів

Як правило, за допомогою лінійної регресії розв’язують задачі з підгонки прямої, яка проходить через безліч точок. Ось як це робиться за допомогою методу найменших квадратів: проводиться пряма, вимірюється відстань від неї до кожної з точок (точки і лінію з’єднують вертикальними відрізками), отримана сума переноситься вгору (рис. 1.3). У результаті та крива, в якій сума відстаней буде найменшою, і є шукана (ця лінія пройде через точки з нормально розподіленим відхиленням від істинного значення).

Лінійну функцію зазвичай використовують під час підбору даних для машинного навчання, а метод найменших квадратів – для зведення до мінімуму похибок шляхом створення метрики помилок.

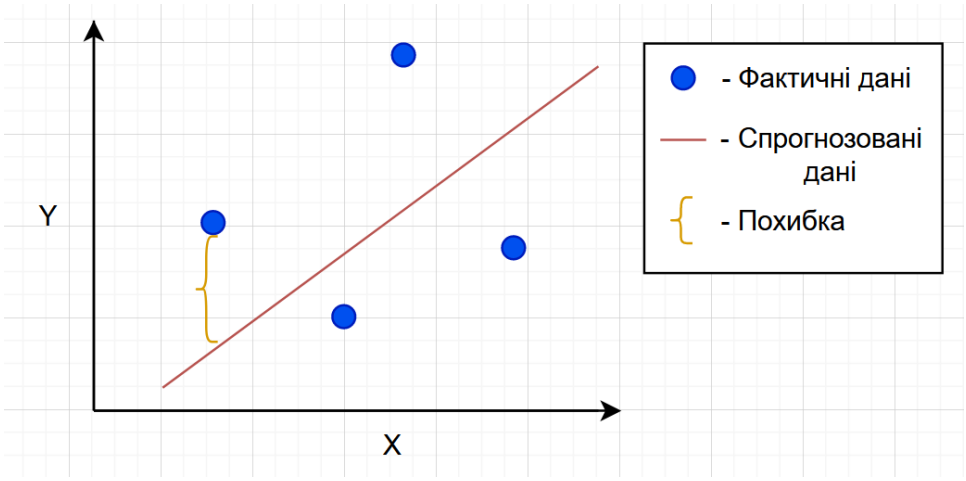

Рисунок 1.3 – Графік точок даних з лінією найменших квадратів

Основне завдання полягає в тому, щоб підібрати параметри функції моделі таким чином, щоб вона найкраще відповідала набору даних. Простий набір даних складається з n точок (пар даних) (x_i , y_i), $i = 1, ..., n$, де x_i це незалежна змінна, а y_i це залежна змінна, значення якої знаходять шляхом спостереження. Функція моделі має вигляд $f(x, \beta)$, де m регульованих параметрів містяться у векторі β. Мета полягає в тому, щоб знайти значення параметрів для моделі, яка найкраще відповідає даним. Відповідність моделі

точці даних вимірюється її залишком, який визначається як різниця між спостережуваним значенням залежної змінної та значенням, передбаченим моделлю:

$$r_i = y_i - f(x_i, \beta).$$

Метод найменших квадратів знаходить оптимальні значення параметрів шляхом мінімізації суми квадратів залишків, S:

$$S = \sum_{i=1}^{n} r_i^2.$$

У найпростішому випадку $f(x_i, \beta) = \beta$ таким чином результатом методу найменших квадратів є середнє арифметичне вхідних даних [2].

1.2.3 Наївний баєсів класифікатор

У статистиці наївні байєсівські класифікатори – це сімейство простих «імовірнісних класифікаторів», заснованих на застосуванні теореми Байєса з сильними (наївними) припущеннями про незалежність між ознаками. Вони є одними з найпростіших моделей байєсівських мереж [3]. Проте в поєднанні з ядровою оцінкою густини розподілу (kernel density estimation), можуть досягати високих рівнів точності. Наївні байєсівські класифікатори добре масштабуються, вимагаючи ряд параметрів, лінійних за кількістю змінних (ознак/предикторів) в задачі навчання. У статистичній літературі наївні моделі Байєса відомі під різними назвами, включаючи простий Байєс (simple Bayes**)** і незалежний Байєс. Всі ці назви посилаються на використання теореми Байєса в правилі прийняття рішень класифікатора, але наївний Байєс не є (обов’язково) байєсівським методом [4].

Наївний Байєс – це проста техніка побудови класифікаторів: моделей, які присвоюють мітки класів для проблемних екземплярів, представленим у вигляді векторів значень ознак, де мітки класів беруться з деякої скінченної множини. Не існує єдиного алгоритму для навчання таких класифікаторів, але є сімейство алгоритмів, заснованих на спільному принципі: всі наївні байєсівські класифікатори припускають, що значення певної ознаки не залежить від значення будь-якої іншої ознаки, заданої змінною класу.

Теорема Байєса стверджує наступну залежність між змінною класу y та залежним вектором ознак x_1 через x_n:

$$P(y \mid x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n \mid y)}{P(x_1, \dots, x_n)}.$$

Використовуючи наївне припущення про умовну незалежність:

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y).$$

Для усіх i, це співвідношення спрощено до:

$$P(y \mid x_1, \dots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)}.$$

Оскільки $P(x_1, \dots, x_n)$ постійна з урахуванням вхідних даних, можна використати таке правило класифікації:

$$P(y \mid x_1, \dots, x_n) \propto P(y)\prod_{i=1}^{n} P(x_i \mid y),$$

$$\Downarrow$$

$$\bar{y} = \arg\max_{y} P(y)\prod_{i=1}^{n} P(x_i \mid y).$$

Різні наївні класифікатори Байєса відрізняються головним чином припущеннями, які вони роблять щодо розподілу $P(x_i \mid y)$ [5].

Приведемо простий приклад, фрукт можна вважати яблуком, якщо він червоний, круглий і має діаметр близько 10 см. Наївний класифікатор Байєса вважає, що кожна з цих ознак робить незалежний внесок у ймовірність того, що цей фрукт є яблуком, незалежно від будь-яких можливих кореляцій між ознаками кольору, округлості та діаметру.

У багатьох практичних застосуваннях для оцінювання параметрів наївних байєсівських моделей використовується метод максимальної правдоподібності; іншими словами, можна працювати з наївною байєсівською моделлю, не приймаючи байєсівську ймовірність і не використовуючи жодних байєсівських методів.

Незважаючи на свою наївну конструкцію та очевидно спрощені припущення, наївні байєсівські класифікатори досить добре працювали в багатьох складних реальних ситуаціях.

Перевагою наївного Байєса є те, що він вимагає лише невеликої кількості навчальних даних для оцінки параметрів, необхідних для класифікації [6].

1.2.4 Дерево ухвалення рішень

Дерево рішень – це метод машинного навчання, що використовується для розв’язання задач класифікації та регресії. Дерево рішень представляє собою структуру, що складається з вузлів та гілок, що репрезентують рішення, які потрібно прийняти, щоб визначити кінцевий результат (рис. 1.4). Процес побудови дерева рішень розпочинається з вибору атрибута, за яким дані будуть розбиті на дві групи. Вибір атрибута залежить від того, як він відображає залежність між атрибутами та результатами. Далі дані поділяються

на дві групи на основі значень обраного атрибута, і для кожної з цих груп будується піддерево. Цей процес повторюється рекурсивно до того моменту, поки не буде досягнуто деякої умови зупинки, наприклад, коли вузол має вже задану максимальну глибину, або коли даних вже недостатньо для подальшого поділу.

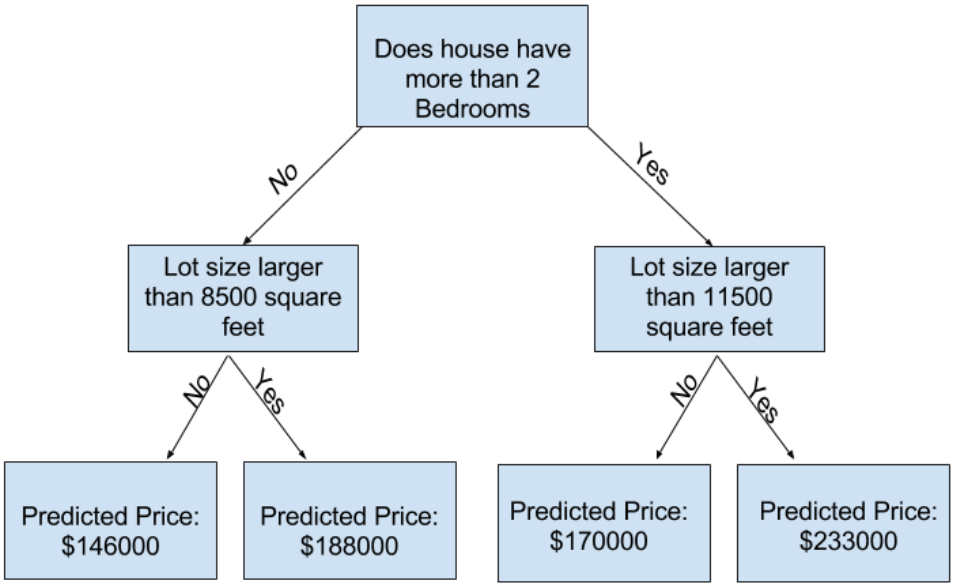

Рисунок 1.4 – Приклад дерева рішень

Під час використання дерева рішень для класифікації, для кожного листка дерева призначається клас, що найчастіше зустрічається серед тренувальних прикладів, що попали в цей листок. При класифікації нового прикладу, його атрибути проходять по дереву, і його клас визначається на основі класу, що призначений для листка, у який попадає цей приклад.

Однією з переваг методу дерева рішень є його інтерпретованість – отримане дерево можна легко зрозуміти і візуалізувати. Однак дерева рішень схильні до надмірної адаптації – коли дерево стає занадто складним, воно може вловлювати шуми в даних і погано працювати на нових прикладах. Щоб вирішити цю проблему, для спрощення дерева можна використовувати такі методи, як обрізання та встановлення критеріїв зупинки. Крім того, ансамблеві методи, такі як випадкові ліси, можуть бути використані для об’єднання декількох дерев рішень і поліпшення їх загальної продуктивності.

1.2.5 Випадковий ліс

Випадковий ліс – це популярний метод машинного навчання, який використовується для класифікації, регресії та інших завдань. Це ансамблевий метод навчання, який поєднує декілька дерев рішень для підвищення їхньої точності та надійності. Основна ідея випадкового лісу полягає у створенні набору дерев рішень, які відрізняються одне від одного, а потім об’єднанні їхніх прогнозів для отримання кінцевого результату (рис. 1.5). Щоб досягти цього, випадковий ліс вводить два джерела випадковості.

Перше джерело має назву бутстрапінг, випадковий ліс створює кілька наборів даних із початкових навчальних даних шляхом вибірки із заміною. Кожен з цих наборів даних використовується для навчання окремого дерева рішень.

Друге джерело це випадковий вибір ознак, у кожній точці розбиття дерева рішень випадковий ліс вибирає випадкову підмножину ознак для розгляду, замість того, щоб використовувати всі доступні ознаки.

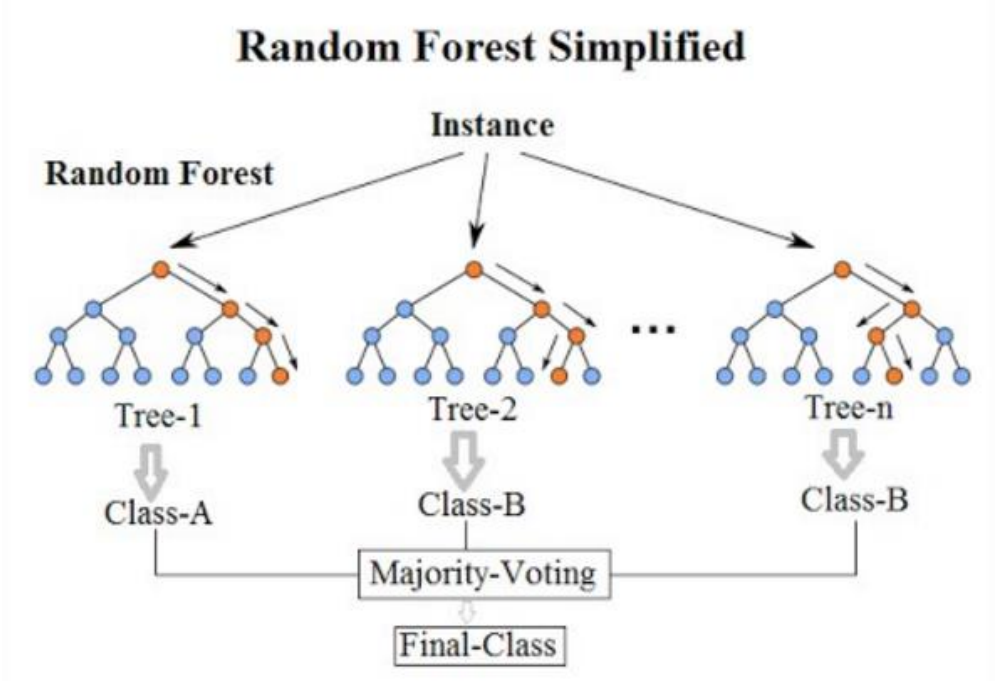

Рисунок 1.5 – Спрощений приклад випадкового лісу

Використовуючи ці джерела випадковості, випадковий ліс гарантує, що кожне дерево рішень в ансамблі відрізняється від інших і має меншу кореляцію з навчальними даними. Це зменшує ризик перенавчання і покращує ефективність узагальнення ансамблю. При використанні випадкового лісу для класифікації, прогноз ансамблю базується на більшості голосів прогнозів окремих дерев. При використанні випадкового лісу для регресії, прогноз ансамблю базується на середньому значенні прогнозів окремих дерев. Основні переваги і недоліки моделі показано у таблиці 1.1.

Таблиця 1.1 – Переваги та недоліки моделі випадкового лісу

Переваги	**Недоліки**
Стійкість до шуму та перенавчання	Вимоглива до обчислювальних ресурсів при роботі з великими наборами даних та великою кількістю дерев
Може працювати з високо імпровізованими даними та пропущеними значеннями	Складність інтерпретації, особливо при використанні великої кількості дерев
Проста у використанні та наявність невеликої кількості налаштувань	
Забезпечує оцінку важливості ознак	

Модель можна використовувати у різноманітних сферах, ось приклад з галузі медицини. Припустимо, лікар хоче передбачити, чи може людина захворіти на діабет на основі її демографічних та медичних даних. У нього є набір даних з 1000 пацієнтів, кожен з яких має 10 характеристик (наприклад,

вік, вагу, кров’яний тиск тощо) і бінарну цільову змінну, яка вказує, чи є у них діабет.

Щоб побудувати модель випадкового лісу для цієї задачі, лікар спочатку створить кілька наборів даних з вихідних даних. Наприклад, це могли бути 100 вибірок, кожна з яких містила 800 пацієнтів, вибраних випадковим чином із заміною з вихідного набору даних. Далі дерево рішень навчається на кожній з вибірок, використовуючи випадкову підмножину ознак у кожній точці розбиття.

Наприклад, можна вибрати три випадкові ознаки з десяти доступних ознак для кожного розбиття. Нарешті, можна об’єднати прогнози дерев рішень, щоб отримати ансамблевий прогноз для нового пацієнта. Наприклад, лікар може передбачити, що пацієнт хворий на діабет, якщо більшість дерев рішень передбачають те саме.

Отримана модель випадкового лісу буде більш точною і надійною, ніж будь-яке окреме дерево рішень, і надасть оцінки важливості ознак, які допоможуть визначити найбільш релевантні фактори для прогнозування діабету.

1.3 Огляд систем розпізнавання зображень з використанням машинного навчання

Системи розпізнавання зображень зазвичай базуються на машинному навчанні, яке є основою багатьох інноваційних технологій сьогодні. За допомогою машинного навчання комп’ютери навчаються розпізнавати об’єкти на зображеннях, аналізувати їх та виконувати певні завдання, пов’язані з цими об’єктами. У сучасних системах РЗ використовуються різноманітні алгоритми та методи машинного навчання, такі як нейронні мережі, глибоке навчання, згорткові нейронні мережі та інші. Ці методи

дозволяють системам розпізнавання зображень знаходити складні залежності між об’єктами на зображеннях та навчатися на них.

Одним з найбільш поширених використань систем РЗ на базі машинного навчання є розпізнавання облич, відоме як «face recognition». Це дозволяє системам автоматично ідентифікувати людей на зображеннях та відео з високою точністю.

Загалом, системи розпізнавання зображень на базі машинного навчання можуть бути використані в багатьох галузях, де необхідно аналізувати та інтерпретувати великі об’єми даних, пов’язаних з зображеннями

1.3.1 Google Vision API

Google Vision API – це інструмент для розпізнавання зображень, розроблений компанією Google. Він дозволяє розпізнавати обличчя, розпізнавати об’єкти та сцени на зображеннях, визначати текст на зображеннях та аналізувати відтінки емоцій, настроїв та інших параметрів. Пропонується широкий спектр функцій, які можуть бути використані для різноманітних завдань розпізнавання зображень. Наприклад, він може розпізнавати місця на зображеннях, визначати різні види тварин та розпізнавати відповідні їм породи, а також класифікувати зображення за відповідними категоріями.

Google Vision API базується на технологіях машинного навчання, що дозволяє йому використовувати навчальні моделі для аналізу зображень та розпізнавання їх змісту. Інтерфейс Google Vision API легкий у використанні та інтегрований з іншими продуктами та сервісами Google, що робить його популярним серед розробників та користувачів.

Функціонал Google Vision API пропонує широкий спектр функцій, які можуть бути використані для різноманітних завдань розпізнавання зображень,

включаючи розпізнавання обличь, визначення тексту, класифікацію зображень та аналіз відтінків емоцій.

Google Vision API використовує технології машинного навчання, що дозволяє йому навчитися розпізнавати зображення за допомогою тренувальних даних та створити навчальні моделі, які можна використовувати для розпізнавання зображень.

Однією з переваг є висока точність системи, завдяки застосуванню машинного навчання, Google Vision API може забезпечити високу точність розпізнавання зображень, що робить його ідеальним для різних застосувань.

Не останнім критерієм є швидкість обробки данних, ще однією перевагою є те що Google Vision API може обробляти зображення в реальному часі.

1.3.2 IBM Watson Visual Recognition

IBM Watson Visual Recognition – це застосунок для розпізнавання зображень, розроблений компанією IBM. Він дозволяє користувачам аналізувати та класифікувати зображення з використанням технологій машинного навчання та штучного інтелекту.

Основні можливості IBM Watson Visual Recognition:

– можливість класифікації зображень, застосунок може автоматично класифікувати зображення на основі їх змісту та визначати, що зображено на них;

– можливість визначення об’єктів, застосунок може розпізнавати та визначати окремі об’єкти на зображенні;

– розпізнавання тексту, застосунок може визначати та розпізнавати текст на зображенні та перетворювати його на електронний формат;

– аналіз емоцій, застосунок може розпізнавати емоції на обличчях та відтінки кольору на зображеннях, що може бути корисним для маркетингових досліджень;

– аналіз контексту, застосунок може враховувати контекст зображення та враховувати його при класифікації.

IBM Watson Visual Recognition може бути використаний у галузях, таких як медицина, безпека, маркетинг та багато інших. Застосунок пропонує API для інтеграції з різноманітними платформами та розробниками.

1.3.3 Amazon Rekognition

Amazon Rekognition – це сервіс компанії Amazon, який надає можливість розпізнавання зображень з використанням технологій машинного навчання та штучного інтелекту. Цей сервіс дозволяє розпізнавати об’єкти, людей, обличчя, текст та інші елементи на зображеннях, а також проводити аналіз контенту зображення.

Основні можливості Amazon Rekognition:

– розпізнавання облич, сервіс дозволяє розпізнавати обличчя на зображеннях та пов’язувати їх з іменами людей;

– визначення елементів, Amazon Rekognition дозволяє визначати елементи на зображеннях, такі як об’єкти, логотипи, транспортні засоби, тварини та інші елементи;

– розпізнавання тексту; сервіс може визначати та розпізнавати текст на зображеннях;

– визначення відтінків емоцій, Amazon Rekognition може розпізнавати відтінки емоцій на обличчях, такі як радість, сум, гнів тощо;

– аналіз контенту, сервіс може аналізувати контент зображення, такий як склад та розмір об’єктів, що дозволяє зробити висновки про контент зображення.

Amazon Rekognition може бути використаний у галузях, таких як медицина, безпека, маркетинг, електронна комерція та багато інших. Він пропонує API для інтеграції з різноманітними платформами та розробниками.

1.4 Порівняння систем розпізнавання зображень з використанням машинного навчання

Google Vision API, IBM Watson Visual Recognition та Amazon Rekognition представляють собою три популярні сервіси комп'ютерного зору, які дозволяють користувачам аналізувати та витягувати інформацію із зображень та відео. Хоча вони мають певну схожість, кожен сервіс має свої унікальні функції та можливості. Існує кілька ключових моментів, на які слід звернути увагу при порівнянні цих трьох сервісів.

У рамках точності та продуктивності, усі три сервіси забезпечують високу точність у завданнях розпізнавання та класифікації зображень. Однак Google Vision API та Amazon Rekognition зазвичай мають вищі показники точності, ніж IBM Watson Visual Recognition. З точки зору продуктивності, Amazon Rekognition, як відомо, працює швидше, ніж інші два сервіси.

З точки зору можливостей та функціональності, Google Vision API і Amazon Rekognition пропонують широкий спектр можливостей і функцій, включаючи виявлення і розпізнавання об'єктів, розпізнавання облич, маркування зображень, OCR (оптичне розпізнавання символів) і багато іншого. IBM Watson Visual Recognition, з іншого боку, в першу чергу орієнтований на завдання розпізнавання і класифікації зображень.

Виходячи з інтеграції та простоти використання, всі три сервіси прості у використанні і пропонують просту інтеграцію з іншими платформами та застосунками. Однак Google Vision API та IBM Watson Visual Recognition відомі своєю простотою використання та зручними інтерфейсами.

Спираючись на ціноутворення усі три сервіси пропонують моделі ціноутворення за принципом «платити по мірі використання», де вартість залежить від кількості оброблених або проаналізованих зображень. Ціна може варіюватися залежно від конкретних функцій і можливостей, що вимагаються.

Загалом, хоча всі три сервіси забезпечують високу точність і широкий спектр можливостей, кожен з них має свої унікальні сильні і слабкі сторони. Зрештою, вибір сервісу залежить від конкретних потреб і вимог користувача.

1.5 Постановка задачі

Таким чином, розробка вебзастосунку з використанням моделі машинного навчання є актуальним завданням в області обробки і розпізнавання зображень.

Об’єктом роботи є процес детектування зображення у відеопотоці та розпізнавання зображення на фотографії за для його подальшої класифікації.

Мета роботи полягає у створенні вебзастосунку, який дозволятиме користувачам побудувати різні моделі машинного навчання.

Розроблений прототип вебзастосунку повинен:

– надавати можливість обробки зображення з камери пристрою або в реальному часі;

– обробляти зображення на основі завантажених користувачем файлів;

– виконувати обробку зображення та виводити певні дані, з використанням моделі МН;

– розпізнавати об’єкти що потрапляють у поле зору камери користувача;

– надавати результати обробки даних;

– мати зручний та простий у використанні графічний інтерфейс.

2 СТВОРЕННЯ АРХІТЕКТУРИ СЕРВІСУ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

2.1 Загальний алгоритм вирішення задачі

Процес вирішення задач з класифікації зображень, в загальному вигляді можливо представити у вигляді послідовності дій, зображених на рисунку 2.1.

Рисунок 2.1 – Загальна схема розпізнавання та перевірки зображень

Першим кроком у розпізнаванні та перевірці зображень є отримання зображення. Це можна зробити за допомогою цифрової камери або іншого пристрою для отримання зображень.

Після того, як зображення отримано, його потрібно попередньо обробити, щоб підготувати до розпізнавання та верифікації. Це включає зміну розміру зображення, регулювання яскравості та контрастності, а також видалення шуму.

Наступним кроком є вилучення ознак із зображення. Це передбачає виявлення візерунків, країв, текстур та інших візуальних елементів, які можна використовувати для розрізнення різних об’єктів на зображенні. Існує багато алгоритмів, які можна використовувати для виділення ознак, зокрема гістограма орієнтованих градієнтів (HOG), масштабно-інваріантне перетворення ознак (SIFT) та згорткові нейронні мережі (CNNs).

Після того, як ознаки витягнуто, наступним кроком є вибір найбільш релевантних з них для поставленого завдання. Це часто робиться за допомогою статистичних методів, таких як аналіз головних компонент (PCA) або лінійний дискримінантний аналіз (LDA).

Виділені та відібрані ознаки можуть бути використані для розпізнавання об’єкта або об’єктів на зображенні. Це можна зробити за допомогою алгоритмів машинного навчання, таких як k-найближчих сусідів (KNN), дерев рішень, машин опорних векторів (SVM) і нейронних мереж.

Після того, як об’єкт або об’єкти на зображенні розпізнано, їх можна порівняти з базою даних відомих автентичних об’єктів для перевірки їхньої автентичності. Це можна зробити за допомогою різних методів, зокрема цифрового водяного маркування, мікродруку та ультрафіолетового (УФ) друку.

Коли зображення розпізнано і перевірено, можна застосувати методи постобробки для підвищення точності результатів. Це можуть бути такі методи, як морфологічні операції, встановлення порогових значень і кластеризація.

Нарешті, продуктивність системи розпізнавання і верифікації зображень можна оцінити за допомогою таких показників, як точність, достовірність, відтворення і оцінка F1. Ця оцінка може бути використана для точного налаштування системи та покращення її продуктивності у майбутніх завданнях розпізнавання та верифікації зображень.

2.2 Модель системи розпізнавання об’єкта у відеопотоці

Використання відеопотоку дозволяє вирішувати завдання, недоступні для вирішення при аналізі одиночної фотографії. Зовнішні умови зйомки можуть привести до того, що розпізнаваємий об’єкт сильно спотворений на одиночному зображенні. Прикладом є відблиск від протяжного джерела світла, що виявляється на поверхні плоского об’єкту.

Розглянемо модель системи розпізнавання одиночного об’єкту x. Нехай задана множина, що містить M класів $C = \{c_1, c_2 ... c_m\}$. В разі розпізнання символу множиною класів може бути фіксований алфавіт. Розглядаючи завдання типізації сторінки документа на зображенні після локалізації її кордонів і проєктованого виправлення, множиною класів може виступати колекція типів сторінок документів, доступних для подальшої обробки. Окремо слід згадати, що іноді в задачах розпізнавання об’єктів і явищ допускається наявність «порожнього класу», який повинен бути відповіддю системи розпізнавання на вхідне зображення об’єкта, про який системі не відомо, або на зображення, яке не містить об’єкта.

Нехай задано зображення об’єкта $I(x)$ з деякої множини всіляких зображень та в рамках моделі взаємодії системи розпізнавання з користувачем існує клас $c^* \in C$, до якого належить об’єкт x. Завдання розпізнавання зображення одиночного об’єкту полягає у визначенні цього класу. Результат роботи системи розпізнавання в загальному вигляді представимо як всюди

певне відображення з множини класів C в множину оцінок приналежності: $r: C \to R$. З огляду на, що множина класів C містить рівно M елементів:

$$r(I(x)) = \{(c_1, q_1), (c_2, q_2), ..., (c_M, q_M)\}, \quad (2.1)$$

де $q_i \in R$, $i \in \{1, ..., M\}$ – речові оцінки належності об'єкту x до класу $c_i \in C$ за умови, що спостерігається зображення об'єкту $I(x)$.

В якості остаточного рішення класифікації приймається клас $c^*(I(x)) = \arg\max r(I(x))$. Тривіальна схема системи розпізнавання об'єкта у рамках описаної моделі представлена на рисунку 2.2.

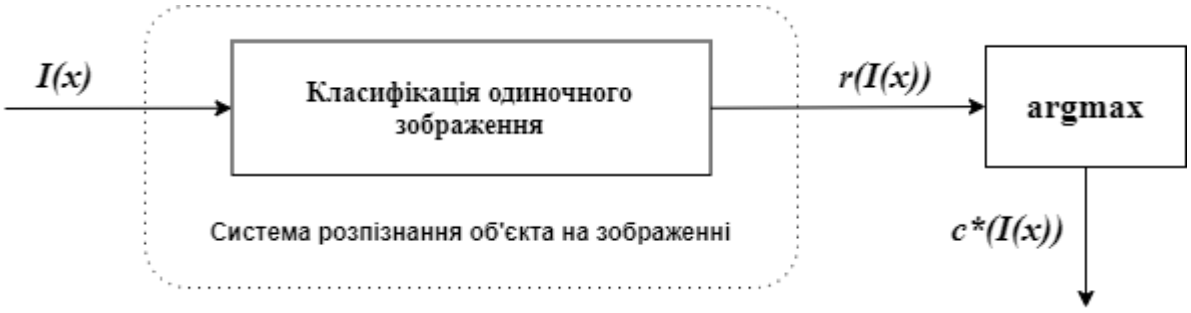

Рисунок 2.2 – Тривіальна схема системи розпізнавання одиночного об'єкта

Така система розпізнавання є статичною і не передбачає зворотних зв'язків.

Розглянемо тепер задачу розпізнавання об'єкту x у відеопотоці. Джерелом відеопотоку є деякий захоплюючий пристрій, який надає послідовність різних кадрів, кожен з яких є незалежним зображенням об'єкту x. В умовах фіксованої кількості кадрів можна розглядати задачу розпізнавання об'єкту в відеопотоці як статичну систему, аналогічну представленої на рисунку 2.2, але з більш складною моделлю входу. Тоді послідовність з N кадрів можливо розглядати як множину зображень об'єкта x: $\mathrm{I}(x) = \{I_1(x), I_2(x), ..., I_N(x)\} \subset R$. При цьому модель виходу системи залишається незмінною.

Залежно від підходу до інтеграції даних така системи може бути виражена по-різному (рис. 2.3). Можливий тривіальний розгляд процесу класифікації як «чорної скрині», яка займається обробкою відразу безлічі зображень. Інші варіанти частково або повністю використовують методи розпізнавання окремого кадру об’єкта і здійснюють інтеграцію або на рівні вхідних зображень, або на рівні результатів розпізнавання кожного окремого зображення.

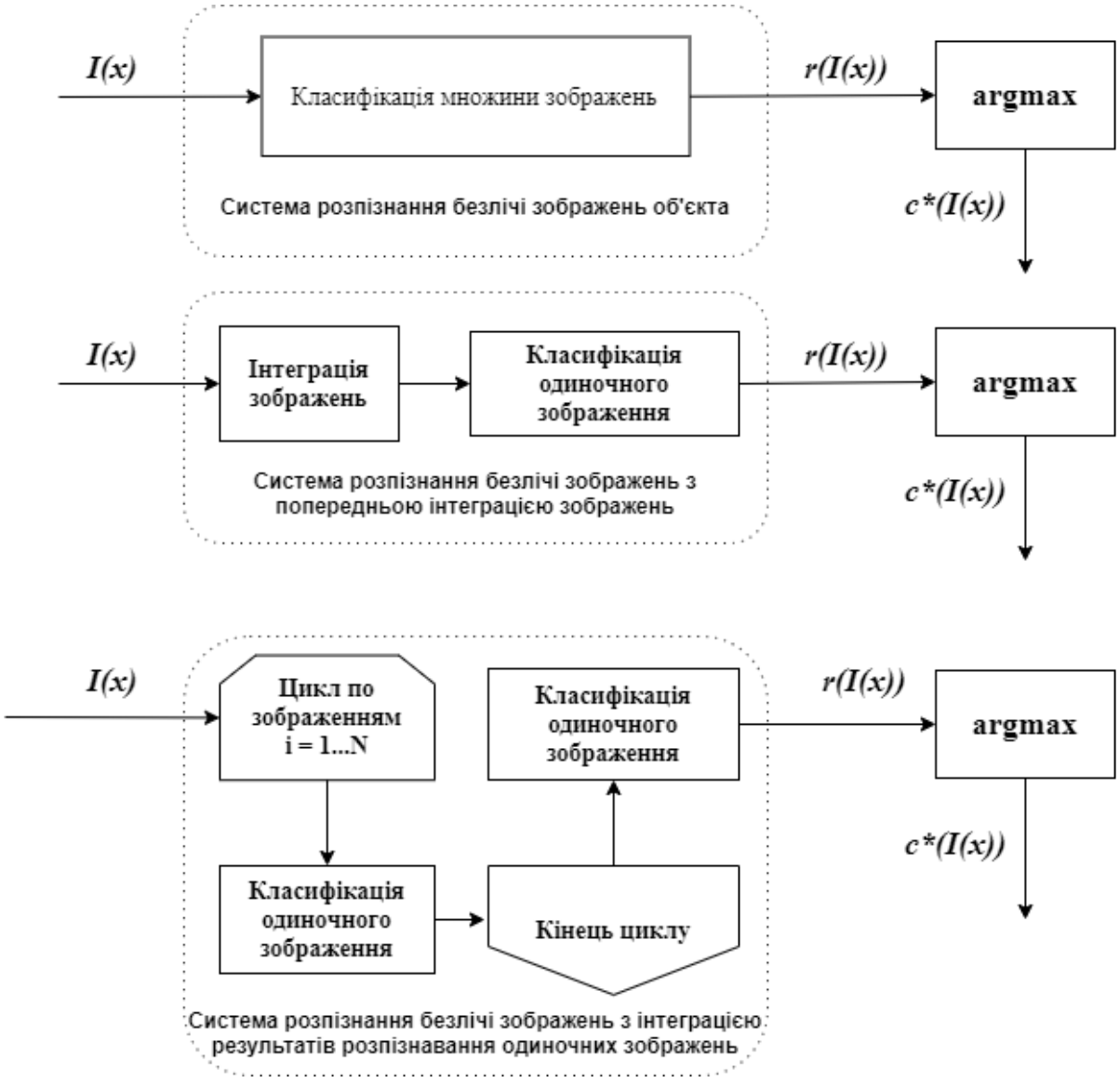

Рисунок 2.3 – Статичні системи розпізнавання безлічі зображень

Однак представлені статичні моделі системи розпізнавання об’єкта відеопотоці не в повній мірі відображають сценарій розпізнавання за

допомогою мобільного пристрою – оскільки дані моделі припускають в якості входу лише безліч кадрів і не припускають зміни стану системи в процесі зйомки. Також в умовах апаратних обмежень мобільних пристроїв зберігання і обробка безлічі зображень може бути недоцільна або неможлива. Для того, щоб більш точно відповідати процесу розпізнавання об’єкта в відеопотоці мобільного пристрою пропонується розглянути динамічну модель з дискретним часом.

2.2.1 Динамічна модель

Для цілей формалізації представимо що відеопотік це генеруюча в часі послідовність зображень об’єкта. Таким чином, задано дискретний час $t = 0,1,2...$ та відеопотік, що містить зображення спостережуваного об’єкта $I_t(x) \in R.$ Подібна дискретна модель відеопотоку відповідає принципам подання кодованого відеопотоку в програмних системах.

Для визначення системи розпізнавання об’єкта у відеопотоці, який генерується незалежно, необхідно визначити модель обслуговування, яка б була проміжним шаром між відеопотоком і безпосереднім потоком оброблюваних системою розпізнавання зображень. Найбільш очевидною є схема обслуговування, при якій зображення, що генеруються під час обробки системою розпізнавання попередніх зображення, скидаються. У випадку, якщо можливо зберігання колекції зображень альтернативною моделлю є схема обслуговування з буфером, що дозволяє накопичувати входять зображення і видавати їх за запитом системи в довільний момент часу, без обмежень, пов’язаних з дискретизацією генерації зображень джерелом [7]. З точки зору безпосередньо системи розпізнавання послідовності зображень набір методів і алгоритмів розпізнавання та інтеграції результатів не залежать від схеми обслуговування, тому в рамках даної роботи в подальшому буде

передбачатися тривіальна схема зі скиданням зображень в періоди завантаження системи.

Система розпізнавання підтримую деякий внутрішній стан $s_1 \in S$, змінний у часі. Час Δ_t, необхідний для отримання оновленого результату після вводу чергового образу $I_t(x)$, у загальному випадку є функцією від зображення та внутрішнього стану системи $\Delta_t = \Delta(I_t(x), S_t)$. Результат розпізнавання, що враховує інформацію, що міститься в зображенні, яке було захоплено в момент часу t, може бути доступний тільки в момент часу $T(t) = t + \Delta$. У початковий момент часу *t=0* ініціалізований внутрішній стан системи S_0. У кожну мить часу t відбувається захоплення зображення $I_t(x)$ з камери пристрою і відбувається перевірка, чи перебуває в даний момент часу будь-яке зображення в обробці. Аналітично дану умову можливо записати як $t < T(t_{prev})$, де t_{prev} – індекс останнього зображення, що надійшло в обробку. Однак оскільки ця умова може бути необчислюваною в момент часу t, для цілей опису моделі можна вважати, що внутрішній стан системи *t* s зберігає інформацію про те, чи знаходиться будь-яке зображення в обробці в момент часу *t*. Якщо в момент *t* система вже обробляє якесь зображення, то знову отримане зображення $I_t(x)$ скидається (в рамках тривіальної схеми обслуговування). В іншому випадку зображення $I_t(x)$ надходить на класифікацію. Результати класифікації інтегруються з накопиченими до поточного моменту результатами (які зберігаються як частина поточної системи в S_t) і стають доступні для виведення в момент часу *T(t).* В момент часу $t \in \{0,...,T(0)-1\}$ результат розпізнавання об’єкта не визначений. Результат розпізнавання, що враховує інформацію, яка міститься в N різних (послідовно захоплених) зображеннях, може бути отриманий в момент часу $T^N(0)$. При цьому індекси зображень, що надходять в обробку, рівні, відповідно, $0, T^1(0), T^2(0), ...T^{N-1}(0)$.

Методи виділення ознак і класифікації об’єктів, застосовні в статичних системах (рис. 2.3) також застосовні і в динамічній моделі, однак динамічна модель системи розпізнавання об’єкта у відеопотоці має низку специфічних властивостей. В першу чергу необхідно відзначити посилений вплив продуктивності алгоритмів розпізнавання одиночного зображення на вихід системи. Дійсно, зменшення часу Δ_t, необхідного для розпізнавання одного зображення $I_t(x)$, дозволяє обробити більшу кількість інформації за один і той же абсолютний час (тобто за той самий час з точки зору користувача). Схема описаної системи представлена на рисунку 2.4.

Рисунок 2.4 – Схема системи розпізнавання об’єкту у відеопотоці з тривіальною моделлю обслуговування

Крім цього, стосовно динамічній системі розпізнавання об’єкта у відеопотоці, незалежно від схеми обслуговування, перетворюючої вхідний відеопотік в потік оброблюваних зображень об’єкта, виникають завдання, нетипові для традиційних систем розпізнавання об’єктів на зображеннях. По перше, це задача інтеграції результатів розпізнавання одиночних об’єктів, по друге, це задача зупинки.

2.2.2 Задача інтеграції результатів та зупинки

Основним завданням традиційних систем розпізнавання об’єктів є максимізація точності розпізнавання. Завдання інтеграції результатів розпізнавання одиночних об’єктів полягає в максимізації точності результату розпізнавання безлічі різних зображень одного і того ж об’єкта при заданих результатах розпізнавання окремого кадру.

Модель системи розпізнавання об’єкта у відеопотоці (рис. 2.4) не припускала обмеження на кількість вхідних зображень, а оскільки основною метою системи розпізнавання об’єктів є автоматизація введення, важливим параметром є абсолютне час (тобто час з точки зору оператора), необхідне для отримання остаточного результату розпізнавання. На відміну від процесу зйомки фотографії, відеопотік природним чином не обмежений у часі. Звідси випливає завдання зупинки, яка полягає в ухваленні рішення про те, що знову отриманий результат в момент часу $T(t)$ можна вважати остаточним і цикл захоплення зображень можна припинити. При розпізнаванні складних об’єктів, які складаються з безлічі незалежно розпізнаваних об’єктів, рішення про зупинку розпізнавання окремих об’єктів впливає на час Δt, необхідний для розпізнавання складеного об’єкта, а значить і на кількість інформації, оброблюваної в рамках загальної системи.

Таким чином, завдання зупинки, яке є тісно пов’язаним з завданням інтеграції, є невід’ємною складовою системи розпізнавання у відеопотоці. Це особливо важливо в рамках систем, де об’єктом розпізнавання є складений об’єкт, такий як текстове поле або документ в цілому.

Завдання зупинки включає в себе здатність визначати та локалізувати цілісні об’єкти або області інтересу в відеопотоці, що дозволяє системі взаємодіяти з ними або проводити подальші обчислення та аналіз. Наприклад, у випадку розпізнавання тексту на відеозаписі, завдання зупинки вимагає точного виявлення та виділення текстових полів для подальшого розпізнавання.

Правило зупинки в загальному вигляді формально можна уявити в вигляді предиката, що діє на відеопослідовність $P: C^* \to \{0,1\}$. Істинність предиката тягне за собою зупинку процесу захоплення і розпізнавання зображень:

$$P(\{I_1(x), I_2(x), ..., I_n(x)\}) = \begin{Bmatrix} 1: рішення\ про\ зупинку \\ 0: продовження\ роботи \end{Bmatrix}. \quad (2.2)$$

Блок-схема схема роботи системи розпізнавання об’єкта у відеопотоці мобільного пристрою в рамках описаної моделі з інтеграцією результатів розпізнавання окремого кадру з умовою зупинки, представлена на рисунку 2.5.

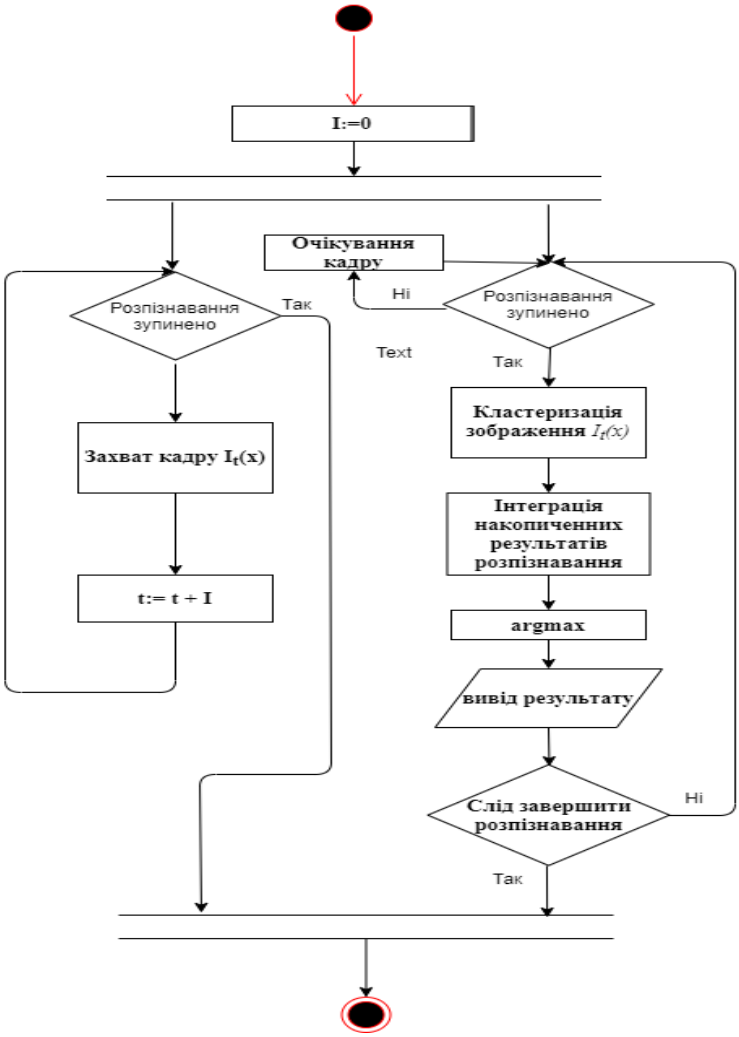

Рисунок 2.5 – Блок-схема роботи системи розпізнавання об’єкта у відеопотоці з інтеграцією результатів розпізнавання окремого кадру

2.3 Система розпізнавання символів на зображенні

Загальний алгоритм системи розпізнавання тексту на отриманому із відеопотоку зображенні документу, можливо візуалізувати в якості блок-схеми зображеної на рисунку 2.6.

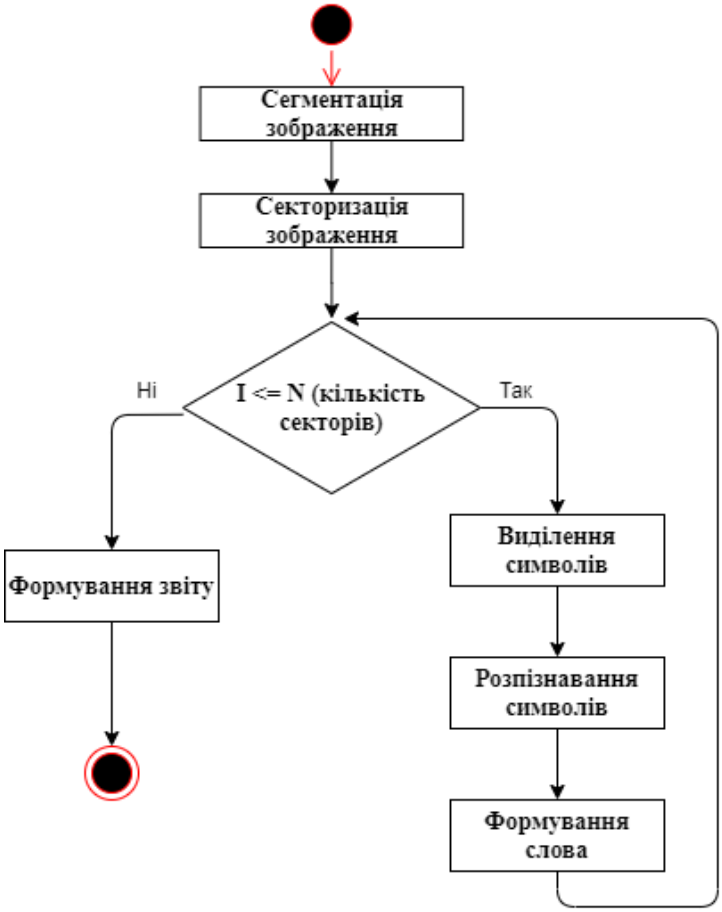

Рисунок 2.6 – Блок-схема алгоритму розпізнавання тексту у документі

2.3.1 Порогова обробка та бінаризація

Процес бінаризації – це переведення кольорового (або в градаціях сірого) зображення в двокольорове чорно-біле. Головним параметром такого перетворення є поріг t – значення, з яким порівнюється яскравість кожного пікселя. За результатами порівняння, пікселю присвоюється значення 0 або 1.

Існують різні методи бінаризації, які можна умовно розділити на дві групи – глобальні та локальні. У першому випадку величина порога залишається незмінною протягом усього процесу бінаризації. У другому зображення розбивається на області, в кожній з яких обчислюється локальний поріг [8]. Загальний алгоритм бінаризації зображення наведений на рисунку 2.7.

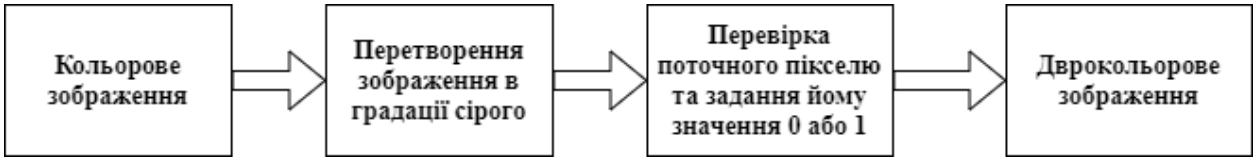

Рисунок 2.7 – Загальний алгоритм бінаризації зображення

Головна мета бінаризації, це радикальне зменшення кількості інформації, з якою доводиться працювати. Просто кажучи, вдала бінаризація сильно спрощує подальшу роботу з зображенням.

2.3.2 Сегментація рядків

Будемо розглядати зображення тексту в градаціях сірого. Початкове зображення можна представити як матрицю яскравостей точок $B = \{b_{ij}\}$.

Завдання виділення рядків зводитися до знаходження верхніх і нижніх граней рядків тексту, зображеного на вихідній картинці. Алгоритм сегментації рядків ґрунтується на тому, що середня яскравість в зображеннях міжрядкових проміжках істотно нижче середньої яскравості в зображеннях текстових рядків [9]:

– спочатку для всіх піксельних рядків вихідного зображення знаходимо їх середні значення яскравості;

$$s_j(B) = \frac{1}{n}\sum_{i=1}^{n} b_{ij}\,; \qquad (2.3)$$

– потім визначаємо середнє значення освітленості цілого зображення;

$$s(B) = \frac{1}{m}\sum_{j=1}^{m} s_j(B); \tag{2.4}$$

– так середня яскравість в міжрядкових проміжках тексту повинна бути невелика (в ідеальному випадку вона дорівнює нулю). Тому яскравість верхньої межі текстового рядка можна виразити через середню яскравість зображення:

$$s^t = k^t s(B), \tag{2.5}$$

де $0 < k^t < 1$ – коефіцієнт;

– аналогічно яскравість найнижчої межі текстового рядка, також може бути виражена через середню яскравість всього зображення:

$$s^b = k^b s(B), \tag{2.6}$$

де $0 < k^b < 1$– коефіцієнт.

Робота алгоритму сегментації рядків полягає в послідовному перегляді масиву середніх значень (s_1, s_m) та виявленні безлічі пар індексів (s^t_i, s^b_i) піксельних рядків, що відповідають верхній s^t_i та нижньої s^b_i гранях зображення рядка номер i, що задовольняють наступним умовам.

Початок текстового рядка або галузі сталого підвищення яскравості фіксується, якщо виконується наступний комплекс умов:

– яскравість поточної піксельної рядки перевищує межу s^t;

– яскравість двох попередніх піксельних рядків нижче цієї межі;

– яскравість трьох наступних рядків вище кордону s^b.

Тобто в піксельної рядку з номером і починається зображення текстового рядка якщо:

$$(s_{i-2} < s^t) \wedge (s_{i-1} < s^t) \wedge (s_i > s^b) \wedge (s_{i+1} > s^b) \wedge (s_{i+2} > s^b) \wedge (s_{i+3} > s^b). \quad (2.7)$$

Кінець області стійкого підвищення яскравості визначається, якщо виконується наступні умови:

– було зафіксовано початок області;

– яскравість поточної піксельної рядки перевищує межу s^t;

– яскравість подальшої піксельної рядки нижче межі s^b;

– було зафіксовано початок області;

– яскравість трьох наступних рядків нижче межі s^b.

Тобто в піксельної рядку з номером і закінчується зображення текстового рядка, якщо раніше було визначено, що рядок почалася, і виконується умова

$$((s_i > s^t) \wedge (s_{i+1} < s^b)) \vee ((s_{i+1} < s^b) \wedge (s_{i+2} < s^b) \wedge (s_{i+3} < s^b)). \quad (2.8)$$

У результаті формується безліч пар індексів верхніх і нижніх граней рядків. Різниця між цими індексами дає висоти текстових рядків. Однак такий алгоритм знаходить середню висоту кожної текстового рядка і «зрізає» символи, які виступають за висотою за цю середню висоту.

Щоб уникнути цього, необхідно розширити знайдені кордону. Можна запропонувати наступний алгоритм розширення меж. Серед знайдених текстових рядків визначається рядок з мінімальною висотою $H_{\min}$ і, потім все кордону з кожного боку розширюються на величину 0,3 * $H_{\min}$. Це не призводить до злиття рядків, тому що інтервали між рядками тексту, як правило, більше ніж висота рядка [10].

2.3.3 Сегментація слів

На другому етапі рішення задачі сегментації зображення тексту, з зображень рядків. Входом для алгоритму сегментації слів служить зображення, будь-якої однієї текстового рядка, яке виходить з вихідного зображення документа після застосування до нього алгоритму сегментації рядків.

Для поліпшення якості роботи алгоритму виділення слів з рядка спочатку його роботи виконуються два перетворення вхідного зображення.

Перший, це граничний фільтр підвищення контрастності, таке перетворення допомагає знизити рівень шуму, тобто прибрати значну кількість зайвих точок.

Другий, це «розмиваючий» фільтр – для кожної яскравою (чорної) точки вихідного зображення зафарбовує сусідні точки.

В результаті такого перетворення близькі точки об’єднуються в безперервну область і замість безлічі маленьких точок отримуємо картинку складається з декількох суцільних плям з досить чітким кордоном [11].

Далі виконуємо власне процедуру сегментації. Алгоритм сегментації слів ґрунтується на тому, що середня яскравість в інтервалах між словами істотно нижче середньої яскравості в зображеннях слів. Він схожий на алгоритм сегментації рядків, тільки перегляд йде по піксельним стовпцях зображення рядка.

Для всіх піксельних стовпців вихідного зображення рядка знаходимо їх середні значення яскравості.

$$c_i(B) = \frac{1}{m}\sum_{j=1}^{m} b_{ij}\,, \tag{2.9}$$

де m – висота поточного рядка в точках.

Потім визначаємо середнє значення яскравості для даного зображення рядка.

$$c(B) = \frac{1}{n}\sum_{i=1}^{n} c_i(B), \tag{2.10}$$

де n – ширина поточного рядка в точках.

Середня яскравість в інтервалах між словами повинна бути невелика (в ідеальному випадку вона дорівнює нулю [12]). Тому її ліву кордон (початок слова) можна виразити через середню яскравість зображення рядка.

$$c^l = k^l c(B), \tag{2.11}$$

де $0 < k^l < 1$ – коефіцієнт.

Аналогічно яскравість правої межі (кінець слова), також може бути виражена через середню яскравість всього зображення.

$$c^r = k^r c(B), \tag{2.12}$$

де $0 < k^r < 1$ – коефіцієнт.

Робота алгоритму сегментації слів полягає в послідовному перегляді безлічі середніх значень яскравості стовпців $(c_1,...,c_n)$ і виявленні безлічі пар індексів (c^l_i, c^r_i) піксельних рядків, відповідно до лівої c^l_i і правої c^r_i граней зображення слова номер i, що задовольняють наступним умовам.

Згідно з умовою лівої межі, початок слова або галузі сталого підвищення яскравості фіксується, якщо виконуються наступні умови:

– яскравість поточного і наступного піксельного стовпчика перевищує лівий кордон яскравості для слова c^l;

– яскравість попереднього піксельного стовпчика пройде нижче межі,

тобто в піксельному стовпці з номером *j* починається зображення слова, якщо:

$$(c_{j-1} < c^l) \wedge (c_j > c^l) \wedge (c_{j+1} > c^l). \quad (2.13)$$

Згідно з умовами правою межі Кінець області стійкого підвищення яскравості визначається, якщо виконуються наступні умови:

– було зафіксовано початок слова;

– яскравість поточного і чотирьох наступних піксельних стовпців нижче межі яскравості інтервалу між словами c^r;

– яскравість поточного і чотирьох наступних піксельних стовпців нижче межі яскравості інтервалу між словами c^r;

– яскравість двох попередніх піксельних стовпців вище межі, тобто це слово закінчується в піксельному стовпці з номером j, якщо раніше було визначено, що слово почалося, та виконується умова

$$(c_{j-2} > c^r) \wedge (c_{j-1} > c^r) \wedge (c_j < c^r) \wedge (c_{j+1} < c^r) \wedge (c_{j+2} < c^r) \wedge (c_{j+3} < c^r) \wedge (c_{j+4} < c^r). \quad (2.14)$$

2.3.4 Сегментація символів

У більшості зображень слів символи розташовані близько один до одного і інтервали між символами не так яскраво виражені, як у випадку між інтервалів між рядками та словами. Тому алгоритм сегментації символів складніший за розглянуті раніше алгоритми сегментації рядків та слів [13].

Вхідним параметром для алгоритму сегментації символів є зображення. Це може бути будь-яке слово, яке отримується з зображення текстового рядка після застосування алгоритму сегментації слів. Цей алгоритм дозволяє розділити слова на окремі символи.

Алгоритм сегментації символів ґрунтується на тому, що середня яскравість в інтервалах між символами, принаймні, нижче середньої яскравості в зображеннях символів.

Його загальна схема складається з двох основних частин:

– знаходимо індекси стовпців, що відповідають локальним мінімумам середньої яскравості стовпців c^i;

– виявляємо і видаляємо з цього списку індексів помилкові кордону символів.

Кінцева мета алгоритму – знайти індекси стовпців-кордонів між символами. Пошук локальних мінімумів середньої яскравості стовпців c^i відбувається на суміжних інтервалах зміни індексу стовпця [14]. Розмір інтервалу вибирається виходячи з висоти рядка. Для більшості шрифтів відношення ширини символу до його висоті не перевищує величину 0,3. Тому розмір інтервалу обраний $d_j = 0{,}3\ m$, де m – висота слова в точках.

Пошук мінімумів працює наступним чином:

Крок 1. Спочатку для всіх піксельних стовпців вихідного зображення знаходимо їх середні значення яскравості.

$$c_i(B) = \frac{1}{m}\sum_{j=1}^{m} b_{ij}\,, \tag{2.15}$$

де m – висота слова в точках.

Крок 2. Серед значень c^i перший мінімум є на відрізку $i = 1,...,d_j$.

Крок 3. Припустимо, що він знайшовся для індексу $i^1_{\min}$.

Крок 4. Наступний мінімум необхідно віднайти на відрізку що позначається як $i = (i^1_{\min} + 1),...,(i^1_{\min} + 1 + d_j)$.

Крок 5. Процедура пошуку повторюється, до досягнення кордону (i=n) зображення слова. Всі значення індексу $i^j_{\min}$, відповідних локальних мінімумів, зберігаються в списку W_0.

Видалення помилкових кордонів між символами будемо проводити в кілька етапів:

Крок 1. Локальний мінімум яскравості в стовпці номер i є «кандидатом» на приналежність до міжсимвольного інтервалу, якщо значення середньої яскравості c^i в цьому стовпці менше певної межі яскравості c^b і при цьому значення середньої яскравості в шпальтах віддалених від даного локального мінімуму на 2 пікселя зліва чи справа більше за кордон яскравості [15]. Кордон яскравості можна визначити через середню яскравість картинки.

$$c^b = k^b \frac{1}{n} \sum_{i=1}^{n} c_i(B), \tag{2.16}$$

де $0 < k^b < 1$ – коефіцієнт, n – ширина зображення слова в точках.

Першу умову міжсимвольних кордонів можна записати у наступному вигляді:

$$(c_i < c^b) \wedge ((c_{i-2} > c^b) \vee (c_{i+2} > c^b)). \tag{2.17}$$

В результаті зі списку індексів локальних мінімумів W_0 видаляються індекси стовпців, середня яскравість яких не задовольняє цій умові, формується другий список W_1 індексів – «кандидатів».

Крок 2. Виявлення зв’язків між стовпцями пікселів. На цьому кроці алгоритму сегментації будемо аналізувати зв’язність зображень символів і прибирати зі списку W_1 помилкові кордони, які розрізають символ на частини. Це може відбуватися з широкими слабо зв’язаної символами, наприклад символ англійського алфавіту «Н». Причому, символ може бути пов’язаний, або у верхній, або в середній, або в нижній частині піксельних стовпців. Щоб уникнути неправильної класифікації зв’язності, розділимо зображення на три рівні по вертикалі і будемо аналізувати ці рівні окремо один від одного. Поділ зображення символу на частини відбувається в наступній пропорції: верхній

рівень – 30% від висоти символу, середній рівень – 40% від висоти символу, нижній рівень – 30% від висоти символу [16].

Сформулюємо умови зв'язності серед двох сусідніх піксельних стовпців, таких як *k* і *k+1*:

– для максимумів яскравості рівнів $b_{kh_1}, b_{kl_1}, b_{km_1}$ стовпця *k*;

– максимумів яскравості трьох рівнів $b_{(k+1)h_2}, b_{(k+1)m_2}, b_{(k+1)l_2}$ стовпця *k+1* повинна виконуватися умова:

$$(h_1 = h_2) \vee (m_1 = m_2) \vee (l_1 = l_2); \tag{2.18}$$

– середня яскравість стовпця *k* повинна бути менше максимуму яскравості сусіднього стовпця *k+1*:

$$c(k) < c_{\max}(k+1). \tag{2.19}$$

– максимум яскравості в стовпці *k* повинен бути більше подвоєного абсолютного значення різниці між значеннями максимумів яскравості стовпця *k* і сусіднього стовпця *k+1*

$$c_{\max}(k) > 2\,|\,c_{\max}(k) - c_{\max}(k+1)\,|. \tag{2.20}$$

Якщо для даного стовпця виконуються всі умови зв'язності з сусідами зліва і справа то межа віддаляється як помилкова, в іншому випадку виконується ще одна перевірка. Відстань до попередньої (лівої) межі d_k має бути більше за допустимий мінімум $d_{\min}$.

$$d_k > d_{min},\ d_{min} = 0{,}4n, \tag{2.21}$$

де *n* – висота зображення слова.

В результаті зі списку індексів «кандидатів» W_1 видаляються індекси стовпців, які мають зв’язок з сусідами зліва і справа, та формується кінцевий список індексів кордонів W_2.

2.3.5 Класифікація сегментів тексту

На даному етапі набір секторів текстового блоку піддається обробці за алгоритмом *k* найближчих сусідів. На виході отримуємо розпізнаний текст. У середині алгоритму окремо розпізнається кожен символ, тому важливо якісно визначити грані кожного символу на етапі сегментації. Коли кожен символ розпізнаний, збираються слова та текстові блоки. Для використання цього алгоритму також повинна бути навчальна вибірка символів [17].

3 РОЗРОБКА ПРОГРАМНОГО ЗАСТОСУНКУ

3.1 Обґрунтування вибору середовища програмної реалізації

У рамках кваліфікаційної роботи був створений програмний застосунок з використанням моделі класифікації зображень, використовуючи можливості сервісу Teachable Machine. Оскільки програма являє собою вебзастосунок, було вирішено, що для написання основного коду буде використовуватися мова програмування JavaScript .

JavaScript це – це високорівнева інтерпретована мова програмування, яка використовується переважно для клієнтської веброзробки. Вона дозволяє розробникам додавати інтерактивності та динамічної поведінки вебсайтам, маніпулюючи об’єктною моделлю документа (DOM), обробляючи події та роблячи асинхронні запити до серверів. JavaScript широко підтримується сучасними веббраузерами і став невід’ємним компонентом веброзробки [18-25].

Переваги використання JavaScript:

– це універсальна мова, яку можна використовувати для широкого спектру цілей, включаючи веброзробку, розробку мобільних застосунків (з такими фреймворками, як React Native), серверну розробку (з Node.js) і навіть розробку десктопних застосунків (з такими фреймворками, як Electron);

– JavaScript легко інтегрується з HTML та CSS, основними компонентами веброзробки. Це дозволяє розробникам безпосередньо маніпулювати DOM, забезпечуючи динамічні оновлення та інтерактивний користувацький досвід;

– за допомогою JavaScript ви можете створювати інтерактивні та адаптивні вебсторінки, які забезпечують багатий користувацький досвід. Він забезпечує перевірку форм у реальному часі, анімацію, повзунки, каруселі та інші інтерактивні елементи, які залучають користувачів і роблять вебсайти більш привабливими;

– JavaScript підтримує асинхронне програмування, що дозволяє розробникам обробляти трудомісткі операції, такі як отримання даних з серверів, не блокуючи користувацький інтерфейс. Це підвищує продуктивність і швидкість реакції вебзастосунків;

– велика кількість бібліотек і фреймворків: JavaScript має велику екосистему бібліотек і фреймворків, таких як React, Angular і Vue.js, які спрощують і прискорюють веброзробку. Ці інструменти надають готові компоненти, управління станом та інші утиліти, що підвищують продуктивність;

– крос-платформна сумісність: JavaScript працює на різних платформах, включаючи веббраузери, мобільні пристрої та сервери, що робить його універсальною мовою для розробки крос-платформних застосунків. Це дозволяє повторно використовувати код і зменшує зусилля з розробки для різних середовищ.

Загалом, універсальність JavaScript, легкість інтеграції, покращений користувацький досвід, підтримка асинхронного програмування, велика кількість бібліотек або фреймворків та кросплатформенна сумісність сприяють його популярності та ефективності в сучасній веброзробці.

Розробка сервісу здійснювалася за допомогою редактор вихідного коду Visual Studio Code, яке надає можливості легко та зручно розробляти як графічний інтерфейс користувача так і працювати з базою даних, створювати алгоритми обробки зображень за допомогою сторонніх бібліотек та тестувати розроблюваний сервіс. Також обране середовище розробки допомагає уникнути помилок у процесі написання програмного коду, адже одразу вказує на проблемне місце, та навіть автоматично вставляє загублені символи і форматує програмний код. Даний редактор вихідного коду доступний для ОС Windows, macOS і Linux .

Для тестування базового функціоналу сервісу використовувався браузер Google Chrome, та Mozilla Firefox.

У таблиці 3.1 наведені версії усіх використовуваних технологій та компонентів.

Таблиця 3.1 – Версії технологій розробки

Технологія	Версія
JavaScript	ECMAScript 2021
Visual Studio Code	1.77.0
Google Chrome	113.0.5672.93
Mozilla Firefox	112.0.2

3.2 Програмна реалізація прототипу сервісу

Прототип сервісу класифікації зображень складається з модулю для роботи з камерою, модулю для роботи з готовими файлами та модулю обробки та класифікації зображення. Кожен модуль сервісу відповідає за конкретну частину роботи прототипу. Модель машинного навчання для обробки зображень було згенеровано за допомогою сервісу Teachable Machine, та успішно використано у програмному застосунку. Великою перевагою сервісу Teachable Machine є можливість генерації власних моделей під будь-які найунікальніші потреби користувача. Отримана модель виходить досить гнучкою й дозволяє без проблем підключити її до програмного застосунку, таким чином користувач зможе використовувати власні моделі і перетворити прототип на конкретну систему у залежності від своїх вподобань.

Корисною функцією стане обробка зображення у режимі реального часу, що дозволить розпізнавати об’єкти що потрапляють у поле зору об’єктиву, без завантаження фотографії одна за одною Залежності між модулями прототипу сервісу представлені за допомогою діаграми (рис. 3.1).

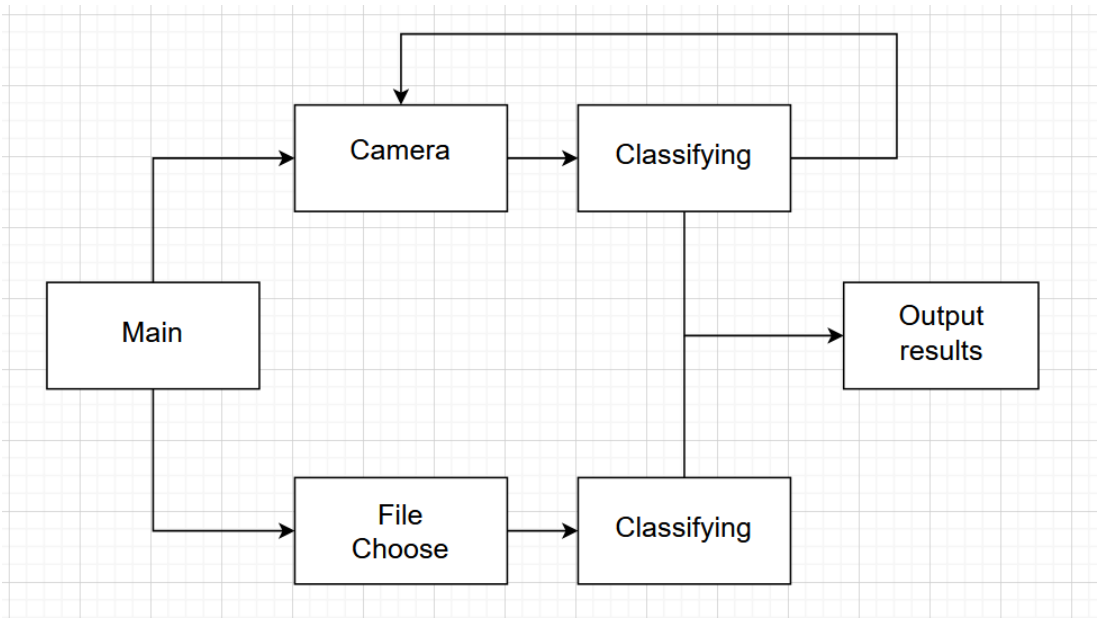

Рисунок 3.1 – Схема сервісу прототипу

Основний модуль сервісу відповідає за взаємодію з користувачем, тобто реагує на команди, які подає користувач через графічній інтерфейс. Також Main модуль виконує функції менеджменту сервісу, тобто виклика інші модулі. У цій реалізації використовується один з основних шаблонів проєктування систем – Посередник. Посередник – поведінковий шаблон проєктування, що забезпечує взаємодію безлічі об’єктів, формуючи при цьому слабку зв’язаність і позбавляючи об’єкти від необхідності явно посилатися один на одного. Тобто у сервісі перевірки документів, саме Main модуль є посередником, адже контролює основний потік виконання програми та інші модулі сервісу.

Взаємодія з камерою пристроя відбувається завдяки стандартним можливостям веббраузерів Google Chrome та Mozilla Firefox та реалізації інтерфейсів взаємодії, які надає мова розмітки HTML.

Для детекції зображення у відеопотоці або файлі та подальшої обробки та класифікації зображення використовується TensorFlow.js.

TensorFlow.js (TF.js) – це бібліотека JavaScript, яка дозволяє виконувати завдання машинного навчання безпосередньо в браузері. Вона дозволяє

працювати з попередньо навченими моделями для класифікації зображень, полегшуючи розпізнавання об’єктів на зображеннях.

Для початку потрібно імпортувати бібліотеку TensorFlow.js у ваш JavaScript-проєкт. Це можна зробити, включивши скрипт TensorFlow.js у ваш HTML-файл або встановивши його через менеджер пакетів, наприклад, npm.

Після налаштування TensorFlow.js ви можете завантажити попередньо навчену модель для класифікації зображень. TF.js надає попередньо навчені моделі, такі як MobileNet і ResNet, які були навчені на великих наборах даних і можуть точно класифікувати різні об’єкти на зображеннях. Ви можете завантажити ці моделі у свій застосунок за допомогою функції tf.loadLayersModel(). Ця функція завантажує архітектуру моделі та її вивчені ваги.

Перед тим, як завантажити зображення в модель, його потрібно попередньо обробити, щоб воно відповідало вимогам попередньо навченої моделі. Зазвичай це передбачає зміну розміру зображення до вхідних розмірів, очікуваних моделлю, і нормалізацію значень пікселів. Для попередньої обробки зображення можна використовувати функції маніпулювання зображеннями TensorFlow.js або полотно HTML.

Після того, як зображення підготовлено, ви можете пропустити його через завантажену попередньо навчену модель за допомогою функції model.predict(). Ця функція приймає на вхід попередньо оброблене зображення і генерує прогноз або розподіл ймовірностей за класами, на яких була навчена модель. На виході ви отримаєте інформацію про ймовірність належності зображення до різних класів [26-30].

Щоб інтерпретувати результати, ви можете вивчити передбачені ймовірності для кожного класу і визначити клас з найбільшою ймовірністю як передбачувану мітку для вхідного зображення. Це дасть вам результат класифікації, який вказує на те, що зображення, найімовірніше, представляє.

Використовуючи TensorFlow.js і попередньо навчені моделі, можна виконувати завдання класифікації зображень безпосередньо в браузері, що дозволяє створювати інтерактивні застосунки і сервіси, які можуть розпізнавати і класифікувати об’єкти на зображеннях, не покладаючись на обробку на стороні сервера.

3.3 Вхідні та вихідні дані сервісу

У якості вхідних даних для сервісу може виступати відеопотік одержуваний з камери пристрою в режимі реального часу, або окремі файли фотографій у форматах PNG, WebP, або SVG.

Для коректного розпізнавання тексту документа важливо, щоб він знаходився у фронтальній площині та мав нахил від горизонтальної осі не більш ніж 20 градусів. Також слід звернути увагу на якість зображення, слід використовувати зображення високої якості та чіткості. Розмиті, спотворені або низькоякісні зображення можуть ускладнити процес розпізнавання та погіршити точність розпізнавання.

Вихідними даними сервісу за нормальних умов є:

- назви класів;
- вірогідності відношення зображення до певного класу;
- перегляд вибраного зображення.

3.4 Інструкція користувача

По-перше користувачу необхідно завантажити будь-який сучасний веббраузер на свій комп’ютер, або смартфон працюючий на операційній системі IOS або Android. Взаємодія користувача з сервісом перевірки документів відбувається через GUI, який має дуже простий та інтуїтивно

зрозумілий дизайн. На рисунку 3.2 представлена головна сторінка розробленого застосунку.

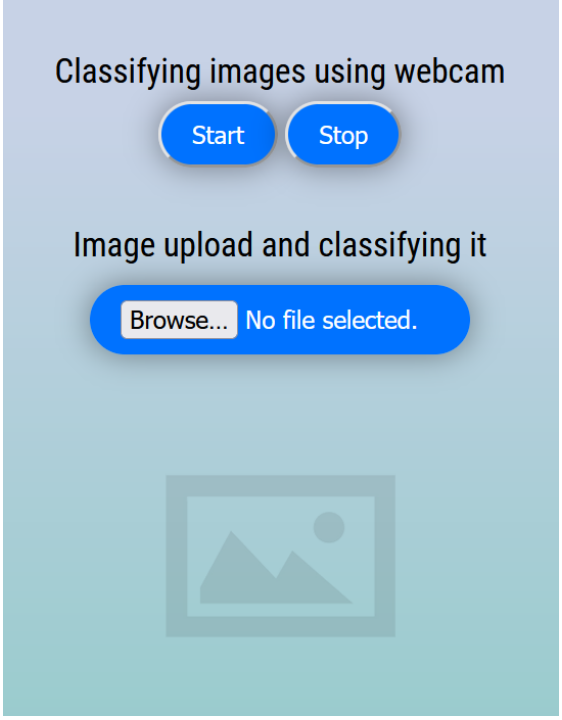

Рисунок 3.2 – Головна сторінка сервісу

Після відкриття застосунку необхідно надати йому доступ до камери, браузер автоматично запросить дозвіл до камери при натисненні на кнопку «Start». Якщо користувач забажає класифікувати зображення у режимі реального часу з використанням вебкамери, йому слід натиснути «Allow» (рис. 3.3).

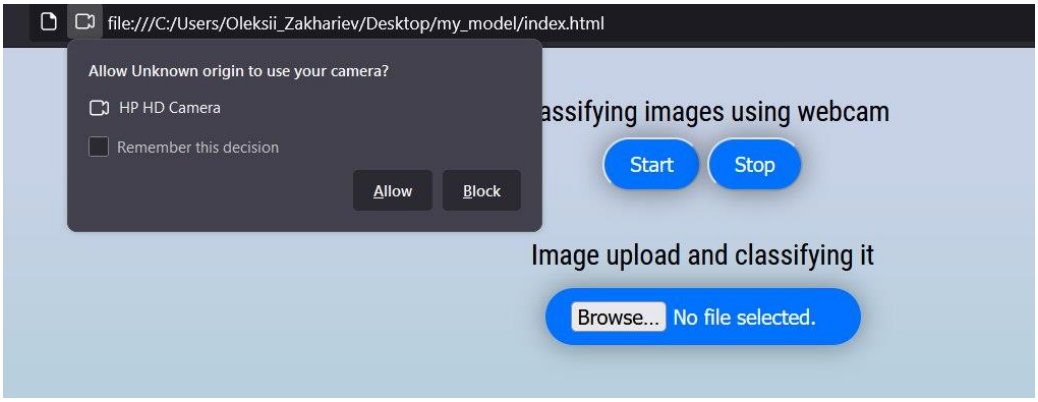

Рисунок 3.3 – Вікно доступу до камери

На початковій сторінці є вибір з джерела отримання зображень, це може бути або постійний відеопотік або окремий файл. Відеопотік запускається кнопкою «Start», після надання доступу яку відкривається камера для розпізнавання зображення. Далі користувачу потрібно лише навести камеру на об’єкт аби сервіс автоматично класифікував зображення згідно моделі машинного навчання що запрограмована у нього. Як результат з’явиться поле з даними (рис. 3.4).

Рисунок 3.4 – Результат розпізнавання зображення у відеопотоці

Проте слід пам’ятати що обробка зображення, джерелом якого виступає вебкамера, відбувається до тих пір поки працює програма, тобто вона буде

фіксувати всі зміни що трапляються з зображенням та проводити класафікацію на основі цих даних. Беручи до уваги припущення щодо зображення на рисунку 3.4, програма на 99% впевнена що перед нею деталі що являються заржавілими, та на 1% впевнена що вони чисті. Але якщо трохи змінити кут зображення (рис. 3.5). Тоді вихідні дані трохи зміняться, а саме програма на 93% буде впевнена що перед нею деталі що являються заржавілими, та на 7% впевнена що вони чисті.

Рисунок 3.5 – Результат розпізнавання зображення зі зміненим кутом

Користувач також може використовувати для класифікації й заздалегідь підготовлені файли, для цього у застосунку слід натиснути кнопку «Browse» під написом «Image upload and classifying it».

Після натискання відкриється вікно вибору файлів, у ОС Windows 10 воно виглядає як показано на рисунку 3.6.

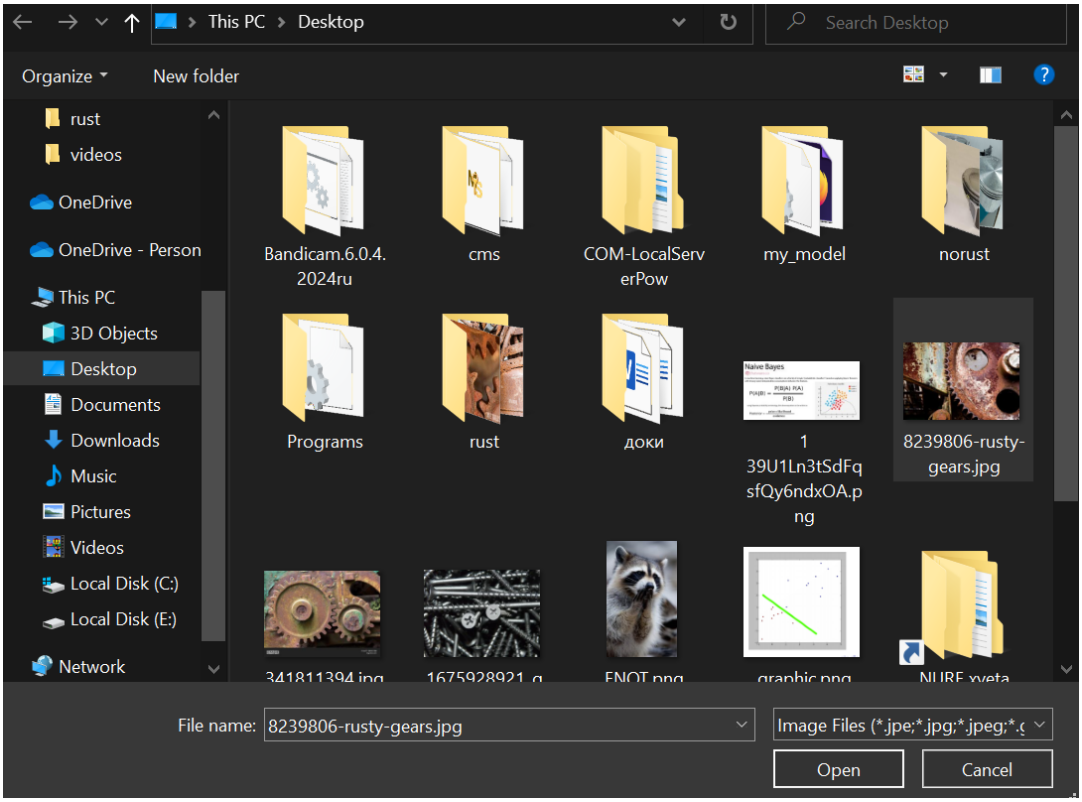

Рисунок 3.6 – Вікно вибору файлів

Після відкриття файлу зображення, (підтримуються формати файлів .apng, .gif, .ico, .cur, .jpg, .jpeg, .jfif, .pjpeg, .pjp, .png, .svg та інші) зображення буде оброблено автоматично, користувачу не потрібно буде робити ніяких додаткових дій.

Аналогічно як і з обробкою через відеопотік, буде виведено окреме поле для перегляду завантаженого зображення, та назви класів з їх вірогідностями (рис. 3.7).

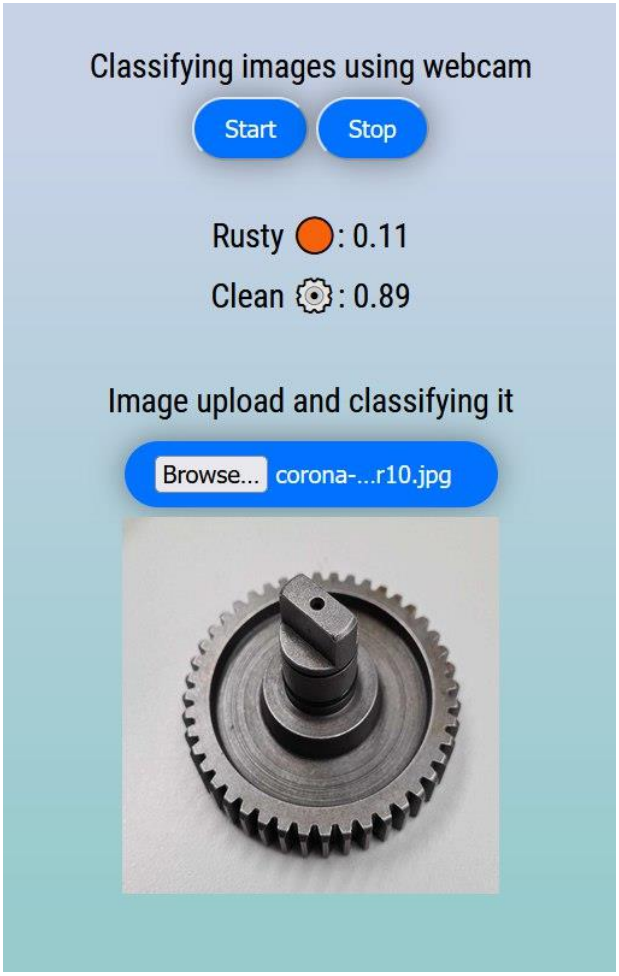

Рисунок 3.7 – Обробка зображення з використанням файлу

ВИСНОВКИ

У рамках роботи був розроблений і реалізований прототип сервісу для реалізації моделей МН на прикладі розпізнавання та обробки зображень використовуючи файли чи відеопоток. В ході роботи було вивчено кілька методів і методологій для вирішення завдання класифікації. Це включало використання передових алгоритмів комп'ютерного зору, моделей глибокого навчання та методів обробки зображень. У проєкті було використано поєднання попередньо навчених нейронних мереж і спеціального навчання на невеликих наборах зображень, що дозволило системі розпізнавати і класифікувати об'єкти, сцени або специфічні візуальні особливості на зображеннях.

Впровадження такої системи дає проєкту кілька значних переваг. По-перше, вона забезпечує автоматизовану та ефективну за часом класифікацію зображень, усуваючи необхідність ручного сортування та аналізу. Це може бути особливо корисно в ситуаціях, коли потрібно швидко обробити і класифікувати велику кількість зображень або відеокадрів.

Крім того, проєкт сприяє підвищенню точності та надійності завдань класифікації зображень. Використання моделей глибокого навчання дозволяє системі вчитися на великих обсягах навчальних даних, покращуючи її здатність точно розпізнавати і класифікувати різні об'єкти і сцени. Це може мати практичне значення в таких галузях, як медична візуалізація, системи безпеки та модерація контенту.

Також, проєкт аналізую аспект класифікації зображень у реальному часі, що дозволяє системі обробляти зображення або відеокадри «на льоту». Ця можливість має вирішальне значення у застосунках, які вимагають негайного прийняття рішень або реагування, таких як відеоаналітика в реальному часі або автономні системи.

Однак важливо відзначити, що в області класифікації зображень у відеопотоці все ще існують певні обмеження і проблеми. До них відносяться

обробка змін в умовах освітлення, перешкод і наявності складного фону. Для вирішення цих проблем і підвищення загальної продуктивності та надійності системи необхідні подальші дослідження і розробки.

Підводячи підсумки, проєкт з розробки системи класифікації зображень у відеопотоці або з використанням файлів зображень, створює основу для автоматизованого і точного аналізу візуального контенту. Проєкт демонструє потенціал розвитку комп’ютерного зору, машинного навчання та методів обробки зображень у різних галузях, приносячи користь промисловості, науковим дослідженням та іншим практичним застосуванням.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. *Automatically Segmenting Brain Tumors with AI | NVIDIA Technical Blog*. (2022, August 21). NVIDIA Technical Blog. https://developer.nvidia.com/blog/automatically-segmenting-brain-tumors-with-ai/ (дата звернення 24.04.2023)

2. Dekking, F. M., Kraaikamp, C., Lopuhaä, H. P., & Meester, L. E. (2005). *A Modern Introduction to Probability and Statistics: Understanding why and how* (Vol. 488). London: Springer.

3. McCakkum, A. (2019). Graphical Models, Lecture2: Bayesian Network Representation.

4. Russell, S. J. (2010). *Artificial intelligence a modern approach*. Pearson Education, Inc.

5. 1.9. Naive Bayes. (n.d.). URL: https://scikit-learn.org/stable/modules/naive_bayes.html (дата звернення 24.04.2023).

6. Why does Naive Bayes work better when the number of features >> sample size compared to more sophisticated ML algorithms? URL: https://stats.stackexchange.com/questions/379383/why-does-naive-bayes-work-better-when-the-number-of-features-sample-size-comp (дата звернення 24.04.2023).

7. Lyashenko, V., Kobylin, O., & Baranchykov, Y. (2018, October). Ideology of Image Processing in Infocommunication Systems. In *2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)* (pp. 47-50). IEEE.

8. Lyashenko, V., Kobylin, O., & Ahmad, M. A. (2014). General methodology for implementation of image normalization procedure using its wavelet transform. *International Journal of Science and Research (IJSR)*, *3*(11), 2870-2877.

9. Lyashenko, V., Lyubchenko, V., Mohammad, A., Alveera, K., & Kobylin, O. (2016). The methodology of image processing in the study of the properties of fiber as a reinforcing agent in polymer compositions.

10. Лащенко Костянтин Сергійович. Розробка программного забезпечення для распізнавання друкованого тексту. URL: https://masters.donntu.ru/2017/fknt/lashchenko/diss/indexu.htm (дата звернення 24.04.2023).

11. Lyashenko V., Kobylin O., Selevko O. (2020) Wavelet Analysis and Contrast Modification in the Study of Cell Structures Images. International Journal of Advanced Trends in Computer Science and Engineering. 9(4). – 4701-4706.

12. Lyashenko, V., Matarneh, R., Kobylin, O., & Putyatin, Y. (2016). Contour detection and allocation for cytological images using Wavelet analysis methodology.

13. Wemhoener, D., Yalniz, I. Z., & Manmatha, R. (2013, August). Creating an improved version using noisy OCR from multiple editions. In *2013 12th International Conference on Document Analysis and Recognition* (pp. 160-164). IEEE.

14. Nguyen, T. T., Nguyen, T. T. T., Pham, X. C., & Liew, A. W. C. (2016). A novel combining classifier method based on variational inference. *Pattern Recognition*, *49*, 198-212.

15. Park, S. C., Park, M. K., & Kang, M. G. (2003). Super-resolution image reconstruction: a technical overview. *IEEE signal processing magazine*, *20*(3), 21-36.

16. Chen, Q. (2004). *Evaluation of OCR algorithms for images with different spatial resolutions and noises* (Doctoral dissertation, University of Ottawa (Canada)).

17. Mateus, B. G., & Martinez, M. (2019). An empirical study on quality of Android applications written in Kotlin language. *Empirical Software Engineering*, *24*(6), 3356-3393.

18. Han, S., Noh, Y., Lee, U., & Gerla, M. (2019). Optical-acoustic hybrid network toward real-time video streaming for mobile underwater sensors. *Ad Hoc Networks*, *83*, 1-7.

19. Bukhtoyarov, V. V., Tynchenko, V. S., & Petrovsky, E. A. (2019, June). Multi-stage intelligent system for diagnostics of pumping equipment for oil and gas industries. In *IOP Conference Series: Earth and Environmental Science* (Vol. 272, No. 3, p. 032030). IOP Publishing.

20. Su, J., Vargas, D. V., & Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, *23*(5), 828-841.

21. Kobylin, O. A., Gorokhovatskyi, V. O., Tvoroshenko, I. S., & Peredrii, O. O. (2020). The application of non-parametric statistics methods in image classifiers based on structural description components. Telecommunications and Radio Engineering, 79(10).

22. Skjæveland, M. G., Forssell, H., Klüwer, J. W., Lupp, D., Thorstensen, E., & Waaler, A. (2019). Pattern-based ontology design and instantiation with reasonable ontology templates. *A Higher-Level View of Ontological Modeling*, 69.

23. Griffiths, D., & Griffiths, D. (2019). *Head First Kotlin: A Brain-friendly Guide*. O'Reilly Media.

24. Kuzminska, O., Mazorchuk, M., Morze, N., & Kobylin, O. (2019, June). Digital learning environment of ukrainian universities: The main components to influence the competence of students and teachers. In *International Conference on Information and Communication Technologies in Education, Research, and Industrial Applications* (pp. 210-230). Springer, Cham.

25. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.

26. Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. Information, 10(4), 150.

27. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Zeghid M. (2022) Tools for fast metric data search in structural methods for image classification, *IEEE Access*, 10, pp. 124738-124746.

28. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, *Advances in Electrical and Electronic Engineering*, 21(1), pp. 19-27.

29. Гороховатський В.О., Творошенко І.С., Чмутов Ю.В. (2022) Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень, *Сучасні інформаційні системи*, 6(3), С. 5-12.

30. Гороховатський В., Передрій О., Творошенко І., Марков Т. (2023) Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень, *Сучасні інформаційні системи*, 7(1), С. 5-13.