МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему:

Нейромережева система розпізнавання вторинної сировини

Галузь знань **12 «Інформаційні технології»**

Спеціальність **122 «Комп'ютерні науки»**

Освітня програма «**Комп'ютерні науки»**

Освітній рівень: бакалавр

Виконала: студентка 4 курсу, групи КН-41

Сіриченко Л.Г.

(прізвище та ініціали)

Керівник Снитюк В.Є..

(прізвище та ініціали)

Доктор технічних наук, професор

(науковий ступінь, звання)

Випускна кваліфікаційна робота бакалавра допущена до захисту

рішенням кафедри *інтелектуальних технологій*

Протокол № 11 від 06.06.2022 р.

зав. кафедри_____ доц. Іларіонов О.Є.

Київ – 2022

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних технологій

Іларіонов О.Є.

_____

"___"_____2022 р.

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

_____

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)

_Нейромережна система розпізнавання вторинної сировини_____

_____

_____

затверджена протоколом засідання кафедри від « 23 » грудня 2021 р. №

0. Термін здачі студентом закінченого проекту (роботи) 31 травня 2022 року
0. Вихідні дані до проекту (роботи)

Список книг з використання інформаційних технологій та , закон України про «Регулювання відходів»_____

_____

_____

_____

0. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

1. Аналіз процесу забезпечення сортування вторинних відходів

2. Розробка архітектурни нейромережевої системи сортування вторинних відходів

3. Програмне забезпечення нейромережі для сортування вторинних речовин

0. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)

Об’єкт, предмет, мета та завдання дослідження (1 слайд), порівняльний аналіз існуючих рішень (3 слайди), проектування інформаційної системи (2 слайди), інформаційне забезпечення (1 слайд), структура програмного забезпечення (1 слайд), огляд процесу тестування (1 слайд), висновок (1 слайд)6. Консультанти з випускної кваліфікаційної роботи із зазначенням розділів випускної кваліфікаційної роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 15 лютого 2022 року

Керівник _____ / Снитюк В.Є. /

(підпис) (ПІБ)

Завдання прийняв до виконання _____ _____/ Сіриченко Л.Г. /

(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів випускної кваліфікаційної роботи	Примітка
1.	Аналіз літературних джерел, аналіз існуючих методів, аналіз основних процесів предметного середовища, постановка задачі	15.02.2022 – 01.03.2022	Виконано
2.	Складання датасета, проектування архітектури нейромережевої системи	02.03.2022 – 05.04.2022	Виконано
3.	Розробка та тестування результатів нейромережевої системи	06.04.2022 – 28.04.2022	Виконано
4.	Оформлення пояснювальної записки, підготовка презентації	29.04.2022 – 10.05.2022	Виконано

Студент-дипломник __ _____ /
/

(підпис) (ПІБ)

Керівник випускної кваліфікаційної роботи _____ / /

(підпис) (ПІБ)

Анотація

Студентка 4-го курсу спеціальності 122 “Комп’ютерні науки” факультету інформаційних технологій **Сіриченко Людмила Григорівна** виконала випускну кваліфікаційну роботу на тему “Нейромережева система сортування вторинних відходів”.

В цій випускній кваліфікаційній роботі проведено аналіз інформаційних технологій, що використовуються для забезпечення проблеми сталого розвитку проблематики сталого розвитку та сортування вторинної сировини. У даній роботі представлена розроблена нейромережева система та програмне забезпечення, що дозволяє класифікувати вторинну сировину.

Ключові слова: сортування вторинних відходів, аналіз, машинне навчання, нейромережева система.

Summary

Student of the last semester of major 122 - “Computer Science” of Faculty of information technology **Liudmyla Sirychenko** has done the degree project with a theme of “Neural network system of classification of recyclable materials”.

At this degree project was made the analysis of information technologies, which are used for the solution of sustainable development and classification of recyclable materials. In this bachelors project the developed Neural network system is presented which is resolving the main goal of this work.

Key words: neural networks system, analysis, machine learning, classification of recyclable materials.

ЗМІСТ

Вступ

В сучасному світі важко уявити собі якийсь процес який повністю залежить від людини, в будь якій галузі – перевірка швидкості машин на дорозі, перевірка рукопису та правильність слів або величезна кількість інших моментів які важко все уявити без участі компьютера та штучного інтелекту, а не без людини. Оптимізація всіх процесів та перенесення людини за компьютер, а не за ручну роботу. Наприклад, сортування відходів, не всі люди дотримуються нових правил сучасного світу та збереження природи для майбутніх поколінь, не всі сортують сміття, не в кожній країні взагалі є можливість відсортувати сміття по різним контейнерам, тому необхідно оптимізувати цей процес.

Об’єктом дослідження даної роботи є процеси сортування вторинних відходів.

Предметом дослідження роботи є нейромережева система сортування вторинних відходів.

Мета даного дипломного проекту полягає у створенні штучного інтелекту який може відсканувати предмет та виявити до якого підрозділу його можливо відсортувати для перероблення вторинного матеріалу. Це дозволить зберегти природу нашої планети для майбутніх поколінь та оптимізувати процес сортування.

РОЗДІЛ 1. АНАЛІЗ ПРОЦЕСУ ЗАБЕЗПЕЧЕННЯ ПРОЦЕСІВ СОРТУВАННЯ ВТОРИННИХ ВІДХОДІВ ВІДХОДІВ

1.1 Проблема сортування відходів - необхідна умова забезпечення безпечного середовища проживання людини

1.1.1 Аналіз проблематики утворення відходів

Однією чи не з найбільших проблем нашого суспільства є надмірне споживання ресурсів. Надмірне споживання ресурсів неодмінно пов'язане з культурою нашого соціуму, такою як чим більше людина покупає, чим більша вона володіє речами – тим вона є успішною у суспільстві.

За даними ООН, щороку приблизно одна третина всієї виробленої їжі – що еквівалентно 1,3 мільярда тонн на суму близько 1 трильйона доларів – гниє у смітниках споживачів і роздрібних продавців або псується через погані методи транспортування та збирання. Поводження з відходами – це проблема всього світу і кожного жителя нашої планети. Щорічно утворюються мільярди тонн відходів – усі разом ми виробляємо гору сміття розміром із Ельбрус. Щороку кількість відходів душу населення зростає приблизно втричі швидше, ніж приростає саме населення світу.

За даними Європейського союзу:

- Якби люди в усьому світі перейшли на енергоефективні лампочки, світ заощадив би 120 мільярдів доларів США щорічно.
- Якщо населення планети досягне 9,6 мільярдів до 2050 року, для забезпечення природних ресурсів, необхідних для підтримки нинішнього способу життя, може знадобитися еквівалент майже трьох планет[12].

А більш ніж 85% одягу виявляється за звалищах завдяки концепції «фаст-фешн» коли такі великі конгломерати мас-маркету як Mango, Zara, Bershka та інші кожен рік випускають нову колекцію щотижня[5], [15].

Надмірне споживання створює такі світові проблеми як:

- Трата непоправних ресурсів планети.
- Утворювання сміттєве звалище, що забруднюють навколишню середу.
- Збільшення CO2 в атмосфері землі
- Нераціональне споживання вторинної сировини чи відходів.
- Збільшення різниці у фінансовому статку між різними шарами суспільства.
- Збільшення можливості природніх катаклізмів – голод, засуха, повені.

1.1.2 Класифікація відходів

Ресурсоцінні відходи – це цінний матеріал, який за сталим розвитком треба використовувати у подальших циклах переробки. Класифікують такі види вторинних матеріалів:

- паперове пакування та друкована продукція, скло, пластик і метал - це 35-50% всього сміття. Їх слід збирати окремо, повертати в економіку, створювати додаткові робочі місця та додаткову вартість у вигляді нових товарів. Органіка, харчові відходи становлять 26-50% загальної кількості сміття. Її можна і потрібно компостувати у закритих танкерах для видобутку біогазу.

- небезпечні відходи – це 1-10%. До нього можна віднести використані батарейки, енергозберігаючі лампи та ртутні градусники. Є ще залишкове сміття, яке не є цінним. Наприклад, одноразовий посуд, пакетики, дрібний пластик, зламані іграшки, гігієнічні засоби та памперси, ватні диски, серветки та все, що не увійшло до перших 3 груп».

- все сміття (мова про ТПВ — тверді побутові відходи) ділиться на чотири потоки: вторсировина, органіка, небезпечні відходи, залишкове сміття. No Waste Recycling Station працює тільки з вторинною сировиною - матеріалами, яким можна дати "друге дихання" - переробити.

1.1.3. Проблематика переробки відходів у світі

У 2018 році в ЄС 38,5% відходів було вивезено на звалища, а 37,9% перероблено. У тому ж році на одного жителя ЄС утворилося 5,2 тонни

відходів [16]. За статистикою 2017 року, 46% усіх міських відходів в ЄС переробляється або компостується. Однак практика поводження з відходами дуже різниться між країнами ЄС, і чимало країн все ще закидають великі обсяги міського сміття. 28 березня 2018 р.

Розглянемо Швецію як один з найяскравіших прикладів успішної переробки вторсировини:

Згідно з інформацією, наданою шведською компанією з сортування та переробки сміття Avfall Swerige, 99% сміття переробляється: 48,6% спалюється для виробництва енергії, 50,6% стає вторинною сировиною, а решта (0,8%), з якою не можна нічого зробити, везуть на звалища. [16] Метал переробляється у 81% випадків, а скляні пляшки – у 89%. Складніше переробляють пластик у Швеції: лише 40% таких відходів можна використати. Більшість пластикових упаковок здають брудними, тому їх доводиться спалювати. З харчових відходів у Швеції отримують біогаз, а потім використовують його як паливо для транспорту.

З 2002 року у Швеції просто заборонено викидати на звалища те, з чого можна отримати енергію. "На смітті" працює навіть сама галузь: шведські сміттєвози їздять на біогазі, отриманому з відходів, або на такій же електриці.

Україна може здобути навички та практики Швеції задля свого розвитку в цій індустрії.

1.1.4.Проблематика переробки відходів в Україні.

Сьогодні технології дозволяють переробляти від 40 до 60% відходів. Зі сміття шиють новий одяг, будують дитячі майданчики, а сучасні заводи з переробки відходів обігрівають житлові квартали. В Україні її сфера поводження з відходами перебуває у ситуації, близька до катастрофічної –

близько 96% сміття не переробляється, а вирушає на поховання. Безповоротно губляться ресурси, які могли б бути використані повторно.

Дані про мусор в Україні [18]:

- 7% усієї площі України – мусорні звалища.
- 6.000 – офіційна кількість звалищ.
- 30.000 неофіційна кількість звалищ.
- 80% свалок неспроможні приймати новий мусор.
- Тільки за офіційними даними за 2016 рік в Україні утворилося 16 млн тонн.

1.2 Аналіз принципів, моделей, методів та інструментальних засобів, які використовуються для підтримки прийняття рішень при сортуванні відходів

1.2.1 Цифровізація в процесах управління сортуванням

Цифровізація перетворює 21 століття, впливаючи на всі сфери повсякденного життя, включаючи сектор екологічних технологій. Цифрові технології забезпечують більш ефективні режими поводження з відходами. Вони дозволять економіці Європи відновити більше цінних матеріалів, присутніх у потоках відходів, зменшуючи кількість видобутої або імпортованої сировини та уникаючи пов’язаного з цим впливу на навколишнє середовище та клімат.

Удосконалення використання цифрових технологій є практичним для зміни світової системи сортування в напрямку більш стійкого розвитку. Такі технології впроваджують рециркуляцію, що використовуються з використанням рециркуляцій за виробником, заощаджують заздалегідь придбання і збирання речей до споживачів, а також усувають можливі виплати витрат на рециркулятори.

Цифрові технології можна знайти на всіх етапах процесу поводження з відходами, деякі з них вже широко використовуються. Однак нинішня ситуація в Європі неоднорідна, і в різних масштабах застосовуються різні технології.

1.2.2 Інструментальні засоби які використовуються для сортування відходів

Приклади конкретних цифрових технологій, які зараз використовуються і, як очікується, в майбутньому матимуть значний вплив на ефективність індустрії відходів, включають робототехніку, Інтернет речей, хмарні обчислення, штучний інтелект та аналітику даних.

Технології, які використовують для поліпшення процесу сортування та створення індустрії сортування включають в себе:

1. Роботизовані системи забезпечують високу швидкість, якість та чистоту стрічки потоків сортування відходів. Наприклад, робот-маніпулятор, що за допомогою машинного зору, image processing, інфрачервоного зору та штучного інтелекту вміє визначати цінні матеріали вторсировини.
2. Штучний інтелект та нейромережеві системи - для запобігання використовування складних та ресурсномістких алгоритмічних систем програмування в управлінні відходами використовують Machine Learning, як у автономних підметальних машинах чи сміттєвозах.
3. Інтернет речей - сенсори, які збирають такі дані на “розумних” контейнерах як вага та рівень наповнюваності відходів, можуть збирати та передавати центральним частинам системи.
4. Облачні системи - накопичення та опрацювання даних та рішення програмного забезпечення в облачних системах поліпшує та оптимізує роботу систем сортування.
5. Аналітика даних грає велику роль в індустрії сортування відходів - вкрай важливо вміти визначати тренди та патерни в утворюванні відходів, будувати моделі та кращі рішення систем сортування.

1.3 Формулювання задачі

Задачею роботи є отримання повністю працюючого клієнт–серверного додатку з точністю розпізнавання не менше 85%. Можна виділити наступні кроки розробки:

1. Створити скрипт для попередньої обробки зображень, що зможе змінювати розмір вхідного зображення.
2. Сформувати об’ємний датасет з фотографіями вторинної сировини, провести його обробку.
3. Спроектувати базу даних для інформації про класи розпізнавання, створити скрипт для її заповнення.
4. Спроектувати декілька нейромережевих архітектур для досліджень.
5. Навчити створені архітектури нейромереж на сформованому датасеті, зберегти отримані ваги зв'язків, вибрати кращий результат.

1.4 Аналіз основних процесів предметного середовища.

1.4.1 Аналіз предметної області

Створення системи для сортування відходів – тяжкий та відповідальний процес який потребує детального аналізу функціонування та постановки задачі яку необхідно вирішувати. Початком для створення додатку необхідно проаналізувати основні процеси роботи додатку для сортування.

Перший етап це створення контекстної діаграми “ЯК Є”(рис. 1), який відображає процес відображення класу сміття за зображенням. На вході система отримує фотографію предмету який необхідно проаналізувати та на виході відправити до користувача інформацію про сировину яку додав користувач для виявлення до якого класу необхідно відправити матеріал для перероблення. Серед елементів керування для даного процесу зазначено закон України про “регулювання відходів” – який допоможе більш вірно виявити необхідний спосіб перероблення вторинної сировини. Механізмами

у даному випадку є користувач, тобто користувач системи який завантажить зображення, та розроблюваний штучний інтелект який виявит до чого віднести вторинну сировину для сортування.

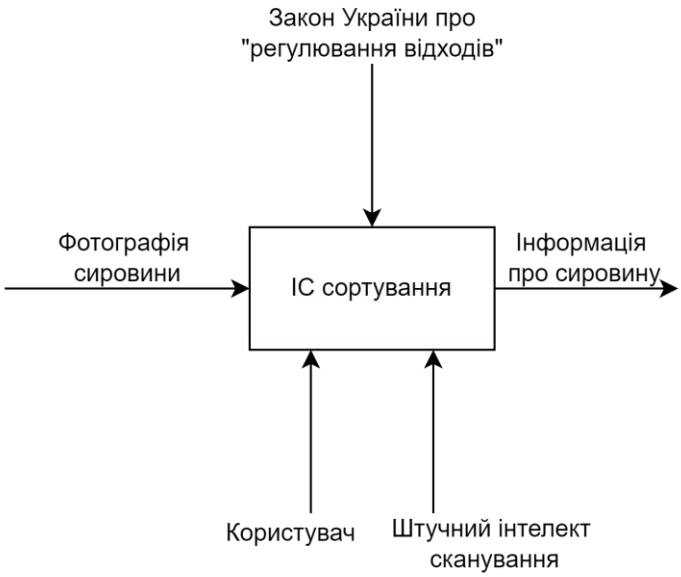

Рисунок 1 – Контекстна діаграма “ЯК Є”

Більш детально розглянути процес, який описаний раніше можливо завдяки декомпозиції основних функцій діаграми “ЯК Є”(рис. 2). Після отримання зображення система аналізує зображення для отримання інформації, що саме це за предмет, скляна тара, пластиковий пакет або взагалі олівець та інші можливі варіанти. Далі система аналізує до якого класу саме належить предмет, який знаходиться на зображенні та надає користувачу інформацію про предмет та варіанти сортування.

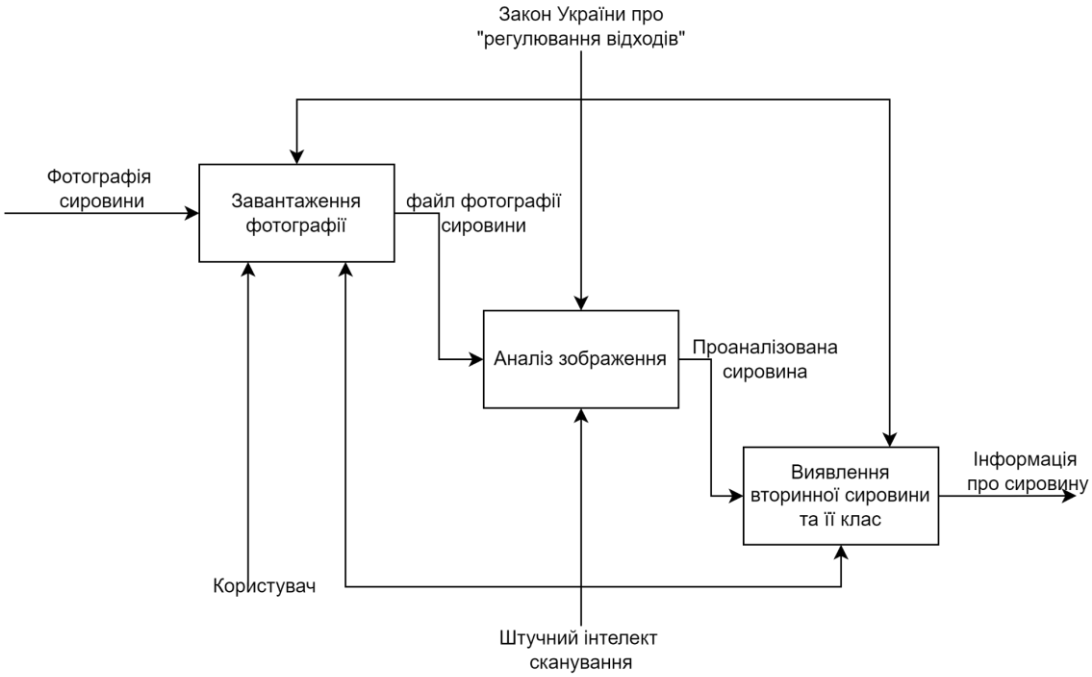

Рисунок 2 – Декомпозиція основних функцій діаграми ЯК Є.

1.5 Постановка задачі на розробку штучного інтелекту для сортування вторинної сировини

Головна задача програми – сортувати сировину. Головні критерії для користування програмою це максимальний простий функціонал та можливість додавати нові зображення та класи для сортування.

Функціональні вимоги для програми сортування:

1. Зручний інтерфейс додатку

Користувачу повинно бути зрозумілим як необхідно завантажити зображення та максимально зрозуміла інформація, що необхідно робити с предметом сортування.

2. Навчання системи

   Штучний інтелект повинен навчатись, як маленька дитина – яка вивчає нові предмети для себе та розуміє куди скласти скло, а куди пластик і розуміти, що для чого.

3. Історія сортування

Для зручності користуванням системи, необхідно розуміти, що ми вже завантажували та переглядати прогрес програми сортування.

4. Звітність завантажених зображень

Для перевірки відсотку відсортованого матеріалу, та розуміння скільки программа змогла відсортувати, необхідно отримувати інформацію за деякі

проміжки часу сортування, для отримання звіту найпопулярнішого вторинного матеріалу.

5. Адаптивно зрозумілий функціонал

Для людини головне розуміти, що можно переглядати та доступність функціоналу сортування, программа повинна бути зручною та зрозумілою для користувача, для поліпшення ситуації сортування сировини.

1.6 Функціональні та нефункціональні вимоги

1.6.1 Функціональні вимоги

Функціональні вимоги (functional requirements) визначають функціональність ПЗ, яку розробники повинні побудувати, щоб користувачі змогли виконати свої завдання в рамках бізнес–вимог. Іноді вони називаються вимогами поведінки (behavioral requirements), вони містять положення з традиційним «повинен» або «повинна»: «Система повинна по електронній пошті відправляти користувачеві підтвердження про замовлення».

До функціональних вимог можна віднести:

1. Система повинна мати модуль для попередньої обробки зображення.
2. Датасет повинен бути об’ємним, містити 12 класів для розпізнавання вторсировини.
3. Система повинна мати модуль для завантаження даних до бази з попереднім формуванням файлу з назвами класів розпізнавання.
4. Система повинна розпізнавати зображення вторсировини та правильно віднести до класу.

1.6.2 Нефункціональні вимоги.

Нефункціональні вимоги описують цілі і атрибути якості. Атрибути якості (quality attributes) представляють собою додатковий опис функцій продукту, виражені через опис його характеристик, важливих для користувачів або розробників. До таких характеристик відносяться:

1. Легкість і простота використання.
2. Цілісність.
3. Ефективність і стійкість до збоїв.

До нефункціональних вимог можна віднести:

1. Нейромережева система повинна мати зрозумілу структуру
2. Датасети повинні бути об’ємними та містити різні приклади вторсировини для якісного аналізу

3. Помилка про розпізнаванні повинна бути не більше 10%.

1.7 Висновки до першого розділу

Отже, в результаті ретельного огляду літератури за темою «Нейромережева система сортування вторинних відходів» було оцінено сучасний стан та актуальність системи сортування.

Також було проаналізовано основні процеси предметного середовища та побудовано діаграми, які їх відображають. В результаті, ми переконались в актуальності даної системи сортування. Заключним етапом було сформовано задачу на розробку, а саме – визначення всіх функціональних і нефункціональних вимог та їх опис.

РОЗДІЛ 2. РОЗРОБКА АРХІТЕКТУРИ НЕЙРОННОЇ МЕРЕЖІ СОРТУВАННЯ ВТОРИННИХ ВІДХОДІВ

2.1 Розробка архітектури

2.1.1 Функціональний аналіз

Завдяки дереву функцій ми можемо отримати повну картину системи бізнес-процесів. На вершині знаходиться головний модуль для додатку сортування сміття. Сам модуль сортування сміття поділяється на два підмодулі: клієнтські функції та адміністративні функції.

До клієнтських функцій належать модуль завантаження зображень сортування для отримання інформації про сировину та рекомендації для сортування та переробки матеріалу. Модуль перегляди історії сортування для перегляду завантажених зображень та отримання інформації про відсортовані матеріали для перевірки коректності програми. Останній клієнтський модуль – модуль перегляду відсотку сировин які завантажувались до системи та перевірки найпопулярніших вторинних речовин для сортування.

До адміністративного модулю належать модулі роботи з користувачами для формування звітів, завдяки яким можливо перевіряти актуальність системи та удосконавлювати системи завдяки новим користувачам та новим зображення вторинних речовин. Останній адміністративний модуль – модуль формування звітів, який включає в собі модуль формування звітів всіх відсортованих речовин, які взагалі система перевірала та сканувала для коректного сортування.

Рисунок 3 – Дерево функцій.

Опис функціонування додатку – новий етап розробки архітектури системи завдяки якій вже можливо розробляти систему та уявляти її функціонал.

Початком функціонування системи є відкритий додаток, який в свою чергу завантажує зручний інтерфейс для користувача. Користувач в свою чергу завантажує зображення для перевірки та очікує результату від штучного інтелекту. Штучний інтелект тим часом отримує зображення та перевіряє, чи це новий матеріал, який він раніше не бачив ао вже відомий йому матеріал та він може виявити клас предмету для сортування та виявити оптимальний варіант сортування вторинного матеріалу. Останній єтап, штучний інтелект видає інформацію користувачу, який недавно завантажив зображення сировини.

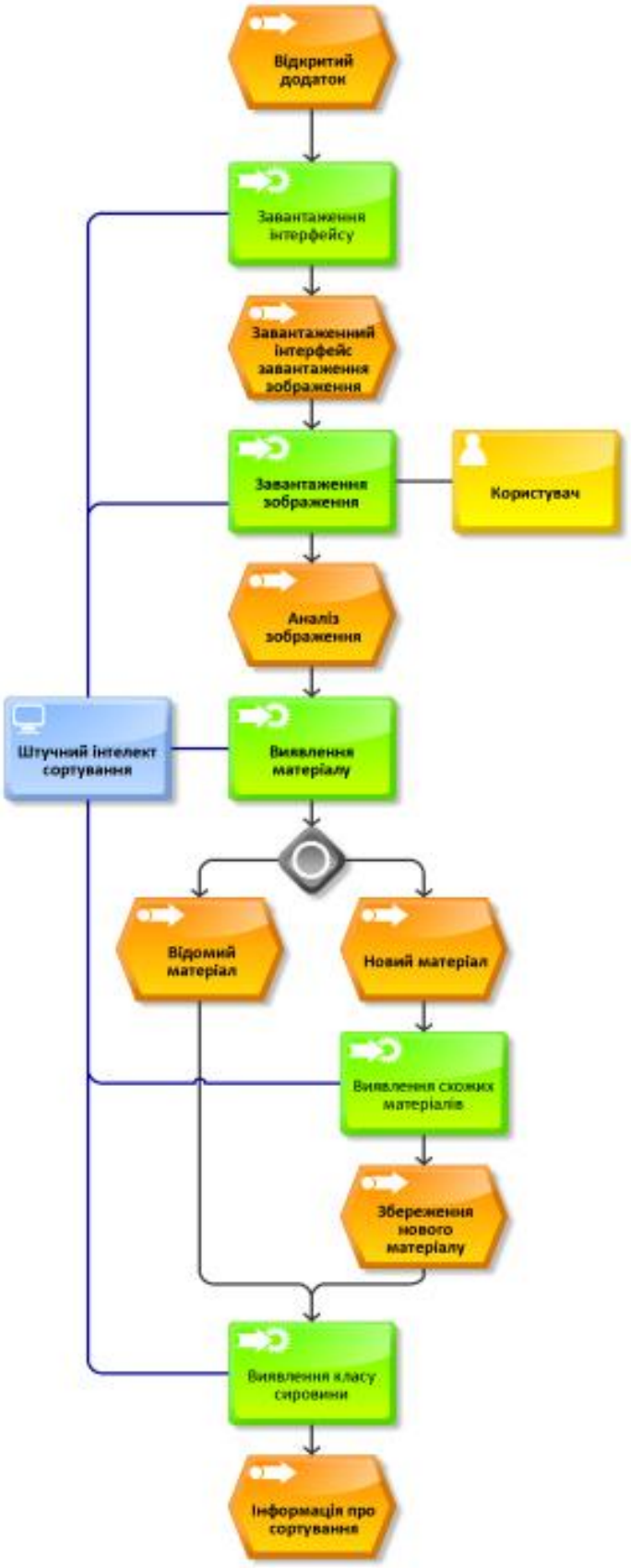

Рисунок 4 – Діаграма Event-Driven Process Chain для функціонування WEB-додатку.

2.1.2 IDEF0 процесу сортування відходів завдяки штучному інтелекту

Діаграма “ЯК БУДЕ” у нотації IDEF0 допоможе представити виконувані процеси сортування системи багатьох зображень для оптимізації процесу сортування.

На вході система потребує набір зображень для сортування матеріалів від користувача, для сканування кожного зображення та кожного предмету на виявлення класу матеріалу та для допомоги користувачу вірно відсортувани речовини за їх класом переробки.

Серед елементів керування для даного процесу зазначено закон України про “регулювання відходів” - який вміщує в собі чітку інформацію, як необхідно сортувати вторинний матеріал за їх класом: скло, пластик, метал та інші.

Механізмами у даному випадку є користувач системи, та розроблений штучний інтелект.

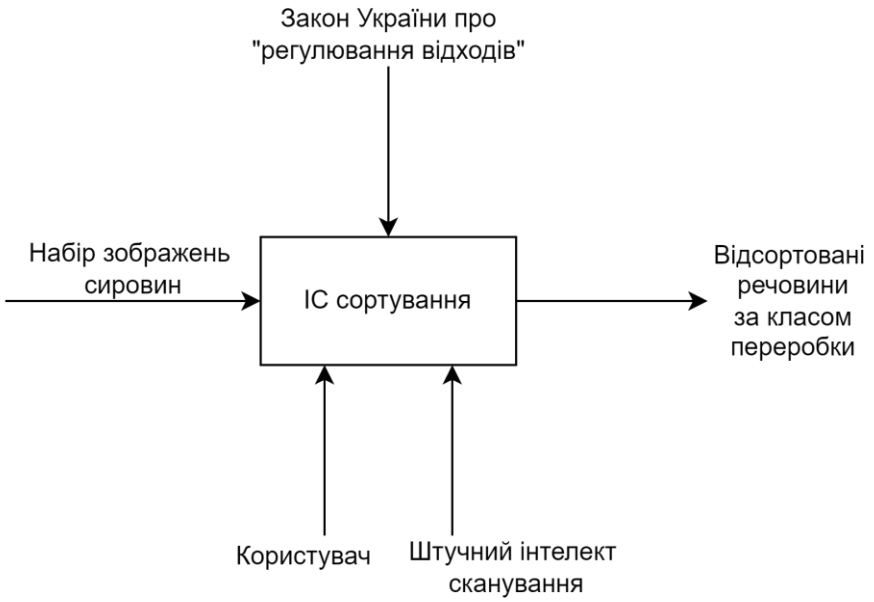

Рисунок 5 - Контекстна діаграма ЯК БУДЕ.

На наступному етапі проведемо декомпозицію основних процесів діаграми ЯК БУДЕ (рис. 6).

Перший процес завантаження користувачем набір зображень сировин для майбутного аналізу та класифікації вторинної речовини. Коли система отримує файли для сортування переходить до етапу аналізу сировини та перевіряє наявність в системі можливих варіантів для класифікації и переходить саме до моменту класифікації при наявності в особистому просторі інформацію про схожі матеріали, після чого на виході видає користувачу відсортовані речовини за класом переробки

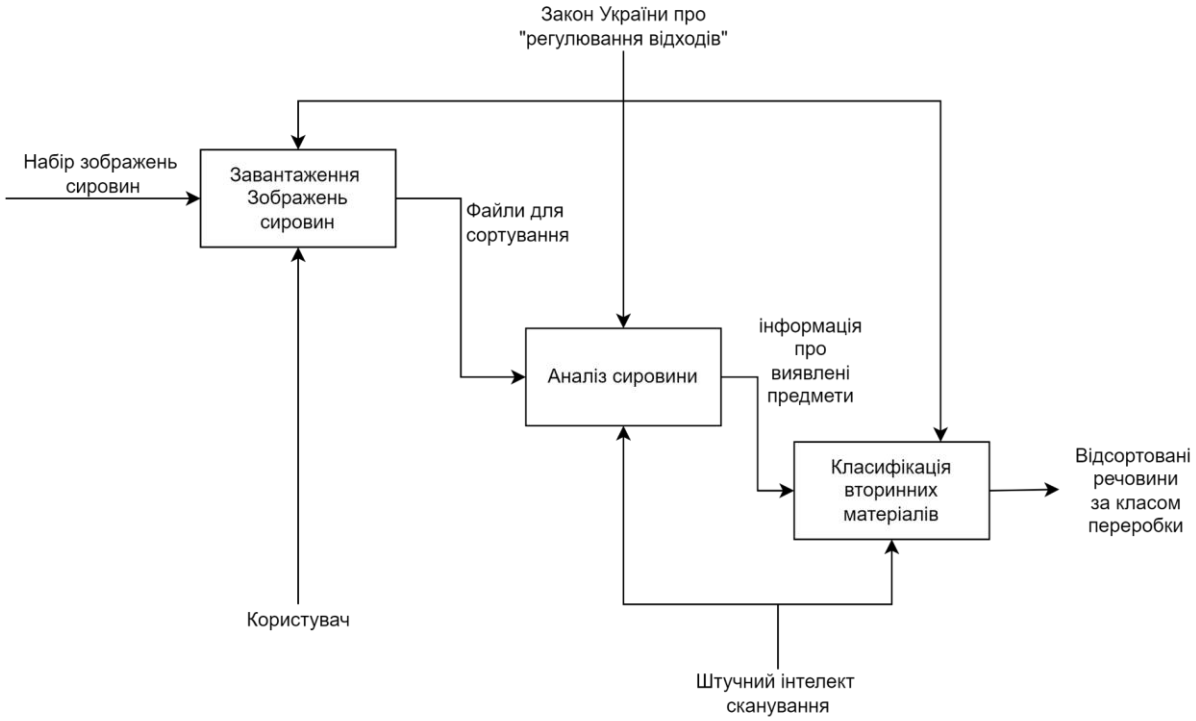

Рисунок 6 - Контекстна діаграма ЯК БУДЕ.

2.1.3 Архітектура інтелектуальної системи

Наступний етап для відображення архітектури системи є спроектована діаграма архітектури системи(рис. 7). На вершині якій знаходиться додаток сортування який ділиться на два модулі: модуль роботи з клієнтами та адміністративний модуль.

До адміністративного модулю належить модуль сортування звітності системи для виявленя коректності сортувань системи та перевірки найпопулярніших матеріалів які надходять до системи для виявлення необхідних ресурсів для переробки вторинної сировини.

До клієнтського модулю належить головний модулю роботи з матеріалами які необхідно відсортувати. До модулю роботи з матеріалами належать модулі перегляду історії сортування та модулю завантаження зображень сировин які необхідно проаналізувати та виявити їх клас сортування. Після отримання результатів сортування користувачу відкривається модулю формування особистих звітів для виявлення коректності системи та поліпшення методів сортування для переробки вторинних речовин.

Рисунок 7 – Архітектура системи.

2.2 Загальні характеристики архітектури нейронної мережі

Під час виконання роботи було визначено 12 категорій вторинної сировини, на які ділитимуться зображення відходів. Також була зібрана певна кількість зображень для навчання нейромережевої системи.

Таблиця 1.

Категорії вторинної сировини та кількість зображень для навчання нейромережі

Сировина:	Кількість зображень
1.Картон	891
2.Біологічні відходи	985
3.Одяг	800
4.Зелене скло	629
5.Взуття	977
6.Біле скло	775
7.Метал	769
8.Батареї	945
9.Коричневе скло	607
10.Пластик	865

11.Бумага	1050
12.Побутове сміття (маски, памперси, тощо)	697

Transfer Learning (укр. - трансферне навчання) – це метод машинного навчання, де ми повторно використовуємо попередньо навчену модель як відправну точку для моделі для нового завдання. Простіше кажучи — модель, навчена для однієї задачі, перебудовується на другу пов’язану задачу як оптимізацію, що дозволяє швидко прогресувати при моделюванні другого завдання. Застосовуючи навчання з перенесенням до нового завдання, можна досягти значно вищої результативності, ніж навчання, лише з невеликою кількістю даних.

Модель EfficientNet-b0 — це згортка нейронна мережа, яка навчається на більш ніж мільйоні зображень із бази даних ImageNet [1]. Мережа може класифікувати зображення на 1000 категорій об’єктів, таких як клавіатура, миша, олівець і багато тварин. В результаті мережа навчилася багатим представленням функцій для широкого діапазону зображень.

Image augmentation (укр. - збільшення зображення) – це техніка застосування різних трансформацій до оригінальних зображень, що призводить до множинних перетворених копій одного і того ж зображення. Ці методи збільшення зображень не тільки розширюють розмір вашого набору даних, але й включають рівень варіації набору даних, що дозволяє моделі краще узагальнювати невидимі дані.

Клас Keras *ImageDataGenerator* забезпечує швидкий і простий спосіб доповнювати зображення. Він надає безліч різних методів збільшення, таких як стандартизація, поворот, зсув, перевертання, зміна яскравості та багато

іншого. Він призначений для забезпечення збільшення даних в режимі реального часу. Це означає, що він генерує доповнені зображення на льоту, поки ваша модель ще знаходиться на стадії навчання. Клас ImageDataGenerator гарантує, що модель отримує нові варіації зображень у кожну епоху. Обрана архітектура – глибока згорткова нейронна мережа. Таблиця 2.3 демонструє основні характеристики нейромережі.

Таблиця 2.

Основні характеристики архітектури нейромережі

№	Назва параметру	Характеристика
1	Тип архітектура	Згорткова мережа
2	Вид поширення	Пряме поширення
3	Кількість вхідних нейронів	224*224
4	Кількість вихідних нейронів	12 – за кількістю класів у датасеті
5	Кількість прихованих шарів	237
6	Кількість фільтрів на згорткових шарах	64
7	Розмір ядра згортки	42
8	Активаційна функція згорткових	relu

	шарів	

Продовження таблиці 2.

Таблиця 3.

Основні характеристики архітектури нейромережі

№	Назва параметру	Характеристика
9	Розмір вікна шарів агрегування	2 * 2
10	Функція вибору параметра шарів агрегування	Max()
11	Кількість нейронів на шарах агрегування	1 – 1024, 2 – 512
12	Активаційна функція шарів агрегування	relu
13	Функція для підрахунку помилки	Categorical_Crossentropy
14	Активаційна функція вихідного шару	softmax

Продовження таблиці 3.

2.3 Висновки до другого розділу

В результаті проведення етапу розробки нейромережі для сортування вторинних відходів.

Було проведено функціональний аналіз та побудовано дерево функцій, яке відображає бізнес-процеси інформаційної системи. Також було розроблено діаграми Event-Driven Process Chain.

Також було представлено діаграми IDEF0 "ЯК БУДЕ", які показують процеси видачі списку відсортованих речовин які завантажив користувач.

Було сформовано архітектуру системи для відображення її внутрішніх компонентів та їх взаємозв'язок.

РОЗДІЛ 3. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ НЕЙРОМЕРЕЖІ ДЛЯ СОРТУВАННЯ ВТОРИННИХ РЕЧОВИН

3.1 Обґрунтування вибору програмних засобів

Для розробки застосунку було використано ряд інструментів та методів, за допомогою яких створено нейромережу для сортування матеріалів.

Основна мова програмування Python, версії 3.9 та бібліотеки – Pandas та pd.DataFrame.

Python[12] – це високорівнева мова програмування загального призначення, яка використовується навіть для розробки веб-додатків. Мова орієнтована підвищення продуктивності розробника і читаності коду.

Правильна українська вимова назви мови програмування - Пайтон, але найчастіше використовується спотворене - Пітон.

Python підтримує кілька парадигм програмування: структурне, об'єктно-орієнтоване, функціональне, імперативне та аспектно-орієнтоване. У мові присутня динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки винятків, підтримка багатопоточних обчислень та зручні високорівневі структури даних. Програмний код на Python організовується у функції та класи, які можуть об'єднуватись у модулі, а вони у свою чергу можуть бути об'єднані у пакети. Python зазвичай використовується як інтерпретований, але може бути скомпільований в байт-код Java та в MSIL (в рамках платфори .NET).

Pandas - це бібліотека Python для обробки та аналізу структурованих даних, її назва походить від "panel data" ("панельні дані"). Панельними даними називають інформацію, отриману в результаті досліджень та структуровану у вигляді таблиць. Для роботи з такими масивами даних створено Pandas[13].

DataFrame та Series – щоб аналізувати дані за допомогою Pandas, необхідно зрозуміти, як влаштовані структури цих даних усередині бібліотеки. Насамперед розберемо, що таке DataFrame та Series.

Pandas Series (серія) – це одномірний масив. Візуально він нагадує пронумерований список: ліворуч у колонці знаходяться індекси елементів, а праворуч — самі елементи[13].

3.2 Структура програмного забезпечення

Структура системи – зображення системної архітектури, основних та побічних модулів, їх основних файлів. В наступних пунктах буде описано структуру кожного з вищеописаних модулів.

Таблиця 4.

Специфікація програмних модулів

Модуль	Опис
.py	Головний модулю аналізу та класифікації зображень вторинних сировин для коректної сортировки для переробки матеріалу – Вхідна інформація: Зображення сировини. Вихідна інформація: Відсортована сировина за класом.

3.3 Узагальнена архітектура системи

Архітектура програмного забезпечення системи показує організацію чи структуру системи і дає пояснення того, як вона поводиться. Система являє собою набір компонентів, що виконують певну функцію або набір функцій. Іншими словами, архітектура програмного забезпечення забезпечує міцну основу, на якій може бути побудовано програмне забезпечення.

Система – нейромережева система сортування вторинної сировини.

Підсистеми:

1. Модуль обробки вхідних даних.
2. Модуль створення та навчання архітектури нейромережі.
3. Класифікований датасет з 12 класів зображень.

3.4 Керівництво користувача

Початок користуванням програми починається з відкриттям додатку. Після завантаж

	path	target	filename
0	../input/garbage-classification/garbage_classi...	metal	metal/metal375.jpg
1	../input/garbage-classification/garbage_classi...	metal	metal/metal561.jpg
2	../input/garbage-classification/garbage_classi...	metal	metal/metal341.jpg
3	../input/garbage-classification/garbage_classi...	metal	metal/metal688.jpg
4	../input/garbage-classification/garbage_classi...	metal	metal/metal374.jpg
...	...	...	...
15510	../input/garbage-classification/garbage_classi...	green-glass	green-glass/green-glass127.jpg
15511	../input/garbage-classification/garbage_classi...	green-glass	green-glass/green-glass607.jpg
15512	../input/garbage-classification/garbage_classi...	green-glass	green-glass/green-glass508.jpg
15513	../input/garbage-classification/garbage_classi...	green-glass	green-glass/green-glass61.jpg
15514	../input/garbage-classification/garbage_classi...	green-glass	green-glass/green-glass114.jpg

Рисунок 8 – Завантажений список зображень.

Далі система аналізує завантажені зображення та перевіряє можливі схожі варіанти.

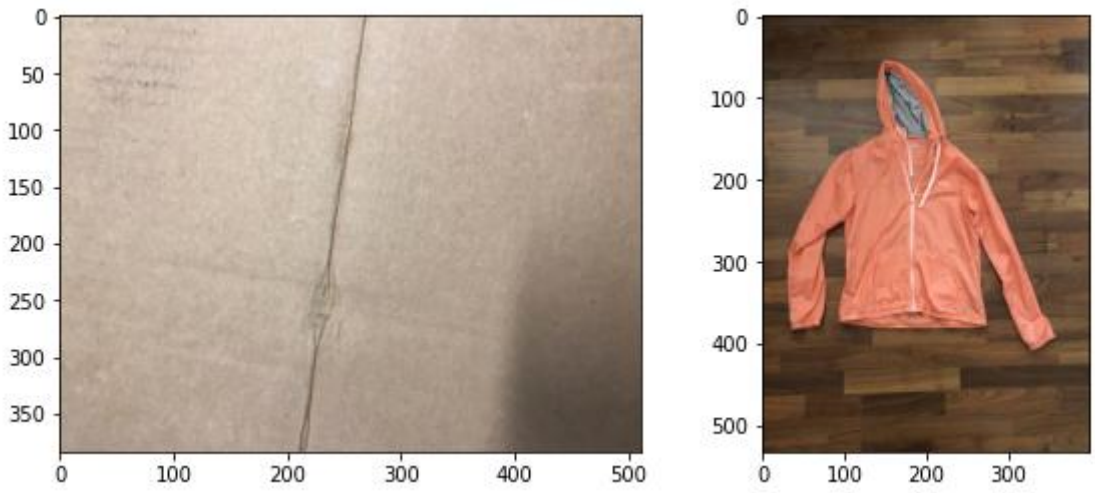

Рисунок 9 – Деякі завантажені зображення.

Після аналізу зображень программа видає інформацію про завантажені зображення та сортує по класу кожен матеріал.

```
paper          1050
biological      985
shoes           977
battery         945
cardboard       891
plastic         865
clothes         800
white-glass     775
metal           769
trash           697
green-glass     629
brown-glass     607
Name: target, dtype: int64
```

```
metal has 769 photos
white-glass has 775 photos
biological has 985 photos
paper has 1050 photos
brown-glass has 607 photos
battery has 945 photos
trash has 697 photos
cardboard has 891 photos
shoes has 1977 photos
clothes has 5325 photos
plastic has 865 photos
green-glass has 629 photos
```

Рисунок 10 – Відсортований список всіх предметів.

Precision (укр. - точність) — це один із показників ефективності моделі машинного навчання — якість позитивного прогнозу, зробленого моделлю. Recall (укр. відкликання) — це міра нашої моделі, яка правильно визначає справжні позитиви. F1 Score – це середнє зважене значення точності та запам’ятовування. Таким чином, ця оцінка враховує як помилкові позитивні, так і помилкові негативи.

Classification Report

	precision	recall	f1-score	support
battery	0.93	0.96	0.94	95
biological	0.96	0.98	0.97	109
brown-glass	0.92	0.95	0.93	59
cardboard	0.98	0.90	0.94	90
clothes	0.95	0.99	0.97	73
green-glass	0.98	0.88	0.93	72
metal	0.88	0.80	0.84	70
paper	0.93	0.93	0.93	102
plastic	0.76	0.91	0.83	89
shoes	0.96	0.98	0.97	96
trash	0.98	0.84	0.91	69
white-glass	0.82	0.83	0.82	75
accuracy			0.92	999
macro avg	0.92	0.91	0.91	999
weighted avg	0.92	0.92	0.92	999

Рисунок 11 – Звіт про класифікацію.

Точність — це один із показників продуктивності моделі машинного навчання — якість позитивного прогнозу, зробленого моделлю. Відкликання — це міра нашої моделі, яка правильно визначає справжні позитиви. F1 Score – це середнє зважене значення точності та запам’ятовування. Таким чином, ця оцінка враховує як помилкові позитивні, так і помилкові негативи.

```
array([[ 92,   0,   1,   2,   0,   0,   0,   0,   0,   0,   1,   0],
       [  0, 103,   0,   0,   1,   0,   0,   0,   0,   0,   0,   0],
       [  0,   2,  45,   1,   0,   1,   1,   0,   2,   0,   0,   1],
       [  0,   0,   0,  79,   0,   0,   0,   4,   1,   0,   0,   0],
       [  0,   0,   0,   0,  74,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0,   1,   0,   0,  59,   0,   0,   1,   0,   0,   0],
       [  1,   0,   1,   0,   0,   1,  62,   3,   4,   1,   0,   5],
       [  4,   0,   0,   2,   1,   0,   0,  97,   1,   0,   0,   0],
       [  0,   3,   0,   0,   1,   1,   3,   0,  76,   0,   0,   4],
       [  0,   1,   0,   0,   1,   0,   0,   1,   0, 100,   0,   0],
       [  0,   0,   0,   1,   0,   0,   0,   0,   2,   0,  68,   1],
       [  0,   1,   0,   2,   0,   2,   2,   0,   9,   0,   0,  65]])
```

Рисунок 11 – Точність сортування.

3.5 Огляд процесу тестування

Останнім кроком оглянемо процес тестування нейромережі сортування вторинних сировин(табл. 5).

Таблиця 5.

id	Пріоритет	Модуль	Кроки	Очікувані результати
1	найвищий	.py	Перевірка вторинних матеріалів Кроки: 1. Завантаження зображень матеріалів 2. Почати сортування	1. Завантаження інформації про завантажені зображення 2. Відображення списку відсортованих речовин 3. Отримання інформації точності
2	високий	.py	Перевірка зображень тварин Кроки: 1. Завантаження зображень тварин 2. Почати сортування	1. Завантаження інформації про завантажені зображення 2. Відображення списку невідомих предметів 3. Отримання інформації точності

3.5 Демонстрація результатів

Класс: Біологічні відходи

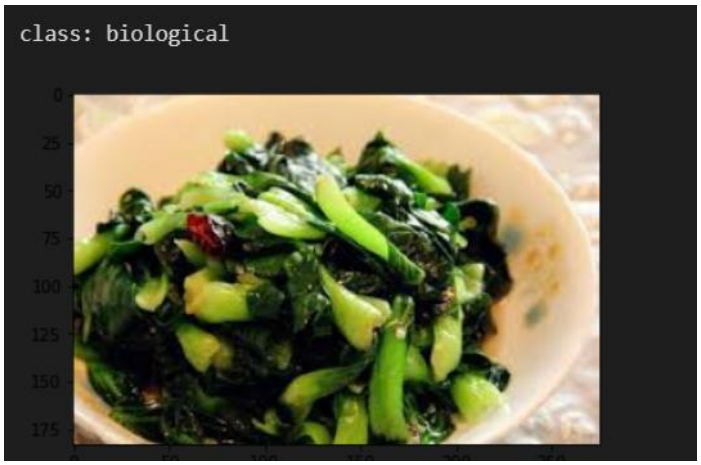

Рисунок 12 - демонстрація результатов.

Класс Картон:

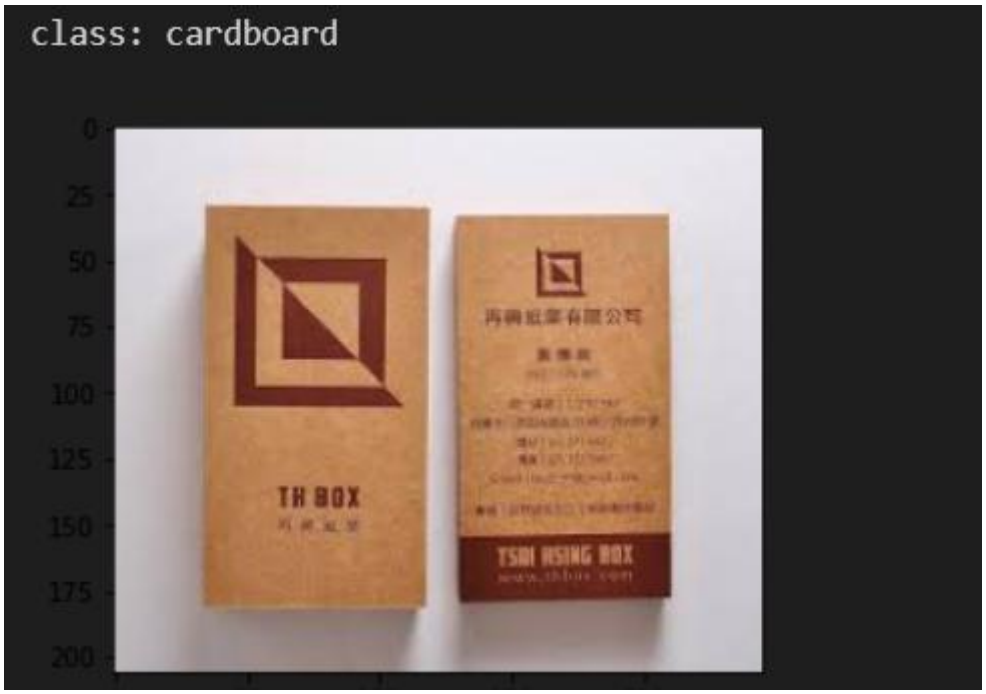

Рисунок 13 - демонстрація результатів.

Класс Біологічні відходи:

Рис. 14 - демонстрація результатів.

Класс Пластик

Рис. 15 - демонстрація результатів.

3.6 Висновки до третього розділу

Для безпосередньої реалізації нейромережі для сортування вторинних матеріалів.

Початок шляху вибір мови програмування та застосунків реалізації матеріалу.

Було визначено структуру програмного застосунку, описано специфікацію програмних модулів у вигляді таблиці.

Заключним етапом був огляд процесу тестування створення штучного інтелекту сортування, а саме – було створено ряд тест кейсів для перевірки функціоналу.

ВИСНОВОК

В ході виконання випускної кваліфікаційної роботи, було спроектовано та навчено глибоку згорткову нейронну мережу, що здатна розпізнавати 12 категорій вторинних відходів, точність на тестовій вибірці – 92%.

Під час проведення досліджень було виконано аналіз методів машинного навчання в сфері розпізнавання відходів, існуючих систем, що допомогло сформулювати функціональні та нефункціональні вимоги, виділити головні інструменти, описати системні вимоги. На основі вищеописаних пунктів був проведений структурний аналіз системи та побудована загальна архітектура системи.

- Запропонована інтерпретація задачі сортування вторинних відходів як задачі розпізнавання та
- Класифікації образів;
- Проаналізовано можливі технології які дозволяють вирішити задачу сортування відходів;
- Виконана розробка непромереженої системи для розпізнавання вторинних відходів;
- Доведення нейромережевої системи до оптимальних показників точності класифікації вхідних зображень – 92% точності.

Розроблену нейромережеву систему можна покращувати в такому напрямку:

•Збільшення об’єму даних для навчання, покращення їх якості – для збільшення обсягу розпізнавання, його якості.

•Доповнити нейромережу функцією сприйняття зображень, щоб наглядно продемонструвати роботу системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Снитюк В.Є. Прогнозування. - К.: Маклаут, 2008. - 364 с
2. Уільям Гібсон. Розпізнавання образів – 2015 р.
3. Крістофер Бішоп. Розпізнавання зразків та машинне навчання (інформатика та статистика) – 2006 р.
4. Гудфеллоу Я. Глибоке навчання. Гудфеллоу Я., Бенджио И., Курвилль А. – 2018 р.
5. J.Pares. The true cost of fast fashion model/ Barcelona - 2020, 61 с.
6. Франсуа Шолле. Глибоке навчання за допомогою Python – 2018 р.
7. Петер Фланх. Машинне навчання. Наука і мистецтво побудови алгоритмів, які витягують знання з даних – 2015 р.
8. Орельєн Жеронімі. Прикладне машинне навчання за допомогою Scikit-Learn і TensorFlow – 2018 р
9. Шалев–Шварц Шай. Ідеї машинного навчання/ Шалев–Шварц Шай, Бен–Давид Шай – 2019 р.
10. Роберт Мартін. Чиста архітектура – 2019 р.
11. Майкл Нейгард. Проектування ПЗ – 2016 р.
12. 12 sustainable goals - https://www.un.org/sustainabledevelopment/sustainable-consumption-production/
13. Мова програмування Python – https://web-creator.ru/articles/python
14. Pandas – https://blog.skillfactory.ru/glossary/pandas/
15. Why clothes are so hard to recycle - https://www.bbc.com/future/article/20200710-why-clothes-are-so-hard-to-recycle
16. Waste statistics Eurostat - https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Waste_statistics#Total_waste_generation
17. EU waste management - https://www.europarl.europa.eu/news/en/headlines/society/20180328STO00751/eu-waste-management-infographic-with-facts-and-figures

18. Отброси та суспільство: як Україна загинається від мусора- https://platfor.ma/topic/otbrosy-y-obshhestvo-kak-ukrayna-zagybaetsya-ot-musora/

Додатки

Програмна частина:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
import tensorflow as tf
```

In [2]:

```
os.listdir("../input/garbage-classification/garbage_classification")
```

Out[2]:

```
['metal',
 'white-glass',
 'biological',
 'paper',
 'brown-glass',
 'battery',
 'trash',
 'cardboard',
 'shoes',
 'clothes',
 'plastic',
 'green-glass']
```

Creating a DataFrame

In [3]:

```
data = pd.DataFrame()
path="../input/garbage-classification/garbage_classification/"
for category in os.listdir(path):
    temp = pd.DataFrame()
    temp['path'] = np.nan
    temp['target'] = category
    i = 0
    for photo in os.listdir(path+category):
        temp.loc[i, 'path'] = path+category+ "/" + photo
        temp.loc[i, 'filename'] = category+ "/" + photo
        temp.loc[i, 'target'] = category
        i += 1
```

```
    data = pd.concat([data, temp], ignore_index=True)
    del temp
```

In [4]:

```
data
```

Out[4]:

	path	target	filename
0	../input/garbage-classification/garbage_classi...	metal	metal/metal375.jpg
1	../input/garbage-classification/garbage_classi...	metal	metal/metal561.jpg
..15515	../input/garbage-classification/garbage_classi...	green-glass	green-glass/green-glass114.jpg

15515 rows × 3 columns

Displaying random images

In [5]:

```
import random
import matplotlib.image as mpimg

for i in range (10):
    random_row = random.randint(0, len(data)-1)
    sample = data.iloc[random_row]
    image = mpimg.imread(sample['path'])
    plt.imshow(image)
    print(sample['path'])
    plt.show()
```

../input/garbage-classification/garbage_classification/cardboard/cardboard221.jpg

../input/garbage-classification/garbage_classification/clothes/clothes2127.jpg

Preprocessing the dataset

In [6]:

```
total_counts = 0

for category in os.listdir(path):
    count_class = 0
    for photo in os.listdir(path + category):
        count_class += 1
        total_counts += 1
    print(str(category) + " has " + str(count_class) + " photos")
```

metal has 769 photos

white-glass has 775 photos

biological has 985 photos

paper has 1050 photos

brown-glass has 607 photos

battery has 945 photos

trash has 697 photos

cardboard has 891 photos

shoes has 1977 photos

clothes has 5325 photos

plastic has 865 photos

green-glass has 629 photos

In [7]:

```
clothesDrop=data[data['target']=='clothes'].sample(n=4525)
shoesDrop=data[data['target']=='shoes'].sample(n=1000)
```

In [8]:

```
data.drop(labels=clothesDrop.index.values, inplace=True)
data.drop(labels=shoesDrop.index.values, inplace=True)
data['target'].value_counts()
```

Out[8]:

```
paper          1050
biological      985
shoes           977
battery         945
cardboard       891
plastic         865
clothes         800
white-glass     775
metal           769
trash           697
green-glass     629
brown-glass     607
Name: target, dtype: int64
```

Model

In [9]:

```
# Shape of EfficientNetB0
im_shape = (224, 224)
batch_size = 64
seed = 42
```

In [10]:

```
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.efficientnet import preprocess_input, EfficientNetB0

data_generator = ImageDataGenerator(preprocessing_function=preprocess_input, validation_split=0.2)
```

Splitting the dataset into training, testing and validation.

In [11]:

```
from sklearn.model_selection import train_test_split

train_df, val_df = train_test_split(data, test_size=0.2, random_state=42)
val_df, test_df = train_test_split(val_df, test_size=0.5, random_state=42)

train_df = train_df.reset_index(drop=True)
val_df = val_df.reset_index(drop=True)
test_df = test_df.reset_index(drop=True)

len(train_df), len(val_df), len(test_df)
```

Out[11]:

(7992, 999, 999)

Using ImageDataGenerator on training, testing and validation dataset.

In [12]:

```
train_generator = data_generator.flow_from_dataframe(
    dataframe=train_df,
    directory=path,
    x_col='filename',
    y_col='target',
    target_size=im_shape,
    class_mode='categorical',
    batch_size=batch_size,
    seed=seed)
```

Found 7992 validated image filenames belonging to 12 classes.

In [13]:

```
val_generator = data_generator.flow_from_dataframe(
    dataframe=val_df,
    directory=path,
    x_col='filename',
    y_col='target',
```

```
    target_size=im_shape,
    class_mode='categorical',
    batch_size=batch_size,
    seed=seed)
Found 999 validated image filenames belonging to 12 classes.
In [14]:
test_generator = data_generator.flow_from_dataframe(
    dataframe=test_df,
    directory=path,
    x_col='filename',
    y_col='target',
    target_size=im_shape,
    color_mode="rgb",
    class_mode="categorical",
    batch_size=1,
    shuffle=False,
    seed=seed)
Found 999 validated image filenames belonging to 12 classes.
In [15]:
nb_train_samples = train_generator.samples
nb_validation_samples = val_generator.samples
nb_test_samples = test_generator.samples
classes = list(train_generator.class_indices.keys())
print('Classes: '+str(classes))
num_classes  = len(classes)
Classes: ['battery', 'biological', 'brown-glass', 'cardboard', 'clothes', 'green-glass', 'metal', 'paper', 'plastic', 'shoes', 'trash', 'white-glass']
In [16]:
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Flatten, Dense



base_model = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(im_shape[0], im_shape[1], 3))


x = base_model.output
x = Flatten()(x)
'''
Flattening is converting the data into a 1-dimensional array for inputting it to the next layer.
```

We flatten the output of the convolutional layers to create a single long feature vector.

And it is connected to the final classification model, which is called a fully-connected layer.

'''

```
x = Dense(100, activation='relu')(x)

predictions = Dense(num_classes, activation='softmax', kernel_initializer='random_uniform')(x)


model = Model(inputs=base_model.input, outputs=predictions)


# Freezing pretrained layers

for layer in base_model.layers:

    layer.trainable=False


# model.summary()


optimizer = Adam()
```

'''

'''

```
model.compile(optimizer=optimizer,loss='categorical_crossentropy',metrics=['accuracy'])
```

2022-04-23 17:46:39.018344: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:39.114693: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:39.115531: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:39.116681: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX2 AVX512F FMA

To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.

2022-04-23 17:46:39.117008: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:39.117716: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:39.118363: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:40.792682: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:40.793553: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:40.794257: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero

2022-04-23 17:46:40.795561: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1510] Created device /job:localhost/replica:0/task:0/device:GPU:0 with 15403 MB memory: -> device: 0, name: Tesla P100-PCIE-16GB, pci bus id: 0000:00:04.0, compute capability: 6.0

Downloading data from https://storage.googleapis.com/keras-applications/efficientnetb0_notop.h5

16711680/16705208 [==============================] - 0s 0us/step

16719872/16705208 [==============================] - 0s 0us/step

```
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

from tensorflow.keras.models import load_model


epochs = 30


#Callback to save the best model

callbacks_list = [

    ModelCheckpoint(filepath='model_EfficientnetB0.h5',monitor='val_loss', save_best_only=True, verbose=1),

    EarlyStopping(monitor='val_loss', patience=10,verbose=1)

]


#Training

history = model.fit(

        train_generator,

        steps_per_epoch=nb_train_samples // batch_size,

        epochs=epochs,

        callbacks = callbacks_list,

        validation_data=val_generator,

        verbose = 1,

        validation_steps=nb_validation_samples // batch_size)
```

2022-04-23 17:46:43.609861: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR Optimization Passes are enabled (registered 2)

Epoch 1/30 124/124 [==============================] - ETA: 0s - loss: 0.4161 - accuracy: 0.8693 Epoch 1: val_loss improved from inf to 0.25518, saving model to model_EfficientnetB0.h5 124/124 [==============================] - 487s 4s/step - loss: 0.4161 - accuracy: 0.8693 - val_loss: 0.2552 - val_accuracy: 0.9167 Epoch 2/30 124/124 [==============================] - ETA: 0s - loss: 0.0486 - accuracy: 0.9842 Epoch 2: val_loss did not improve from 0.25518 124/124 [==============================] - 468s 4s/step - loss: 0.0486 - accuracy: 0.9842 - val_loss: 0.2758 - val_accuracy: 0.9240 Epoch 3/30 124/124 [==============================] - ETA: 0s - loss: 0.0209 - accuracy: 0.9942 Epoch 3: val_loss did not improve from 0.25518 124/124 [==============================] - 457s 4s/step - loss: 0.0209 - accuracy: 0.9942 - val_loss: 0.3015 - val_accuracy: 0.9271 Epoch 4/30 124/124 [==============================] - ETA: 0s - loss: 0.0117 - accuracy: 0.9967 Epoch 4: val_loss did not improve from 0.25518 124/124 [==============================] - 464s 4s/step - loss: 0.0117 - accuracy: 0.9967 - val_loss: 0.2765 - val_accuracy: 0.9375 Epoch 5/30 124/124 [==============================] - ETA: 0s - loss: 0.0065 - accuracy: 0.9987 Epoch 5: val_loss did not improve from 0.25518 124/124 [==============================] - 443s 4s/step - loss: 0.0065 - accuracy: 0.9987 - val_loss: 0.2785 - val_accuracy: 0.9344 Epoch 6/30 124/124 [==============================] - ETA: 0s - loss: 0.0103 - accuracy: 0.9979 Epoch 6: val_loss did not improve from 0.25518 124/124 [==============================] - 451s 4s/step - loss: 0.0103 - accuracy: 0.9979 - val_loss: 0.3432 - val_accuracy: 0.9250 Epoch 7/30

...

124/124 [==============================] - ETA: 0s - loss: 0.0394 - accuracy: 0.9902 Epoch 11: val_loss did not improve from 0.25518 124/124 [==============================] - 421s 3s/step - loss: 0.0394 - accuracy: 0.9902 - val_loss: 0.4870 - val_accuracy: 0.9302 Epoch 11: early stopping

Prediction and Accuracy

In [18]:

```
from sklearn.metrics import classification_report, confusion_matrix
```

```
Y_pred = model.predict(test_generator)
y_pred = np.argmax(Y_pred, axis=1)
target_names = classes

# Confusion Matrix
cm = confusion_matrix(test_generator.classes, y_pred)
cm

# Classification Report
print('Classification Report')
print(classification_report(test_generator.classes, y_pred, target_names=target_names))
```

Classification Report

	precision	recall	f1-score	support
battery	0.95	0.96	0.95	96
biological	0.94	0.99	0.96	104
brown-glass	0.94	0.85	0.89	53
cardboard	0.91	0.94	0.92	84
clothes	0.95	1.00	0.97	74
green-glass	0.92	0.97	0.94	61
metal	0.91	0.79	0.85	78
paper	0.92	0.92	0.92	105
plastic	0.79	0.86	0.83	88
shoes	0.99	0.97	0.98	103
trash	0.99	0.94	0.96	72
white-glass	0.86	0.80	0.83	81
accuracy			0.92	999
macro avg	0.92	0.92	0.92	999
weighted avg	0.92	0.92	0.92	999

In [19]:

```
cm
```

Out[19]:

```
array([[ 92,   0,   1,   2,   0,   0,   0,   0,   0,   0,   1,   0],
       [  0, 103,   0,   0,   1,   0,   0,   0,   0,   0,   0,   0],
       [  0,   2,  45,   1,   0,   1,   1,   0,   2,   0,   0,   1],
```

[0, 0, 0, 79, 0, 0, 0, 4, 1, 0, 0, 0],

[0, 0, 0, 0, 74, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 1, 0, 0, 59, 0, 0, 1, 0, 0, 0],

[1, 0, 1, 0, 0, 1, 62, 3, 4, 1, 0, 5],

[4, 0, 0, 2, 1, 0, 0, 97, 1, 0, 0, 0],

[0, 3, 0, 0, 1, 1, 3, 0, 76, 0, 0, 4],

[0, 1, 0, 0, 1, 0, 0, 1, 0, 100, 0, 0],

[0, 0, 0, 1, 0, 0, 0, 0, 2, 0, 68, 1],

[0, 1, 0, 2, 0, 2, 2, 0, 9, 0, 0, 65]])