МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
Кафедра інформаційних систем

Магістерська робота

РОЗРОБКА АВТОМАТИЗОВАНОГО СЕРВІСУ МОНІТОРИНГУ ЗАБРУДНЕННЯ ПОБУТОВИМИ ВІДХОДАМИ ВУЛИЦЬ ТА ДОРІГ МІСТ

Виконав: студент групи ПМіМ-22
спеціальності
122 "Комп'ютерні науки"
(шифр і назва спеціальності)

О.К. Крикливець О.Є.
(підпис) (прізвище та ініціали)

Керівник Венгерский П.С.
(підпис) (прізвище та ініціали)

Рецензент Трушевський В.М.
(підпис) (прізвище та ініціали)

Львів – 2022

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики
Кафедра інформаційних систем
Спеціальність 122 "Комп'ютерні науки
(шифр і назва)

«ЗАТВЕРДЖУЮ»
Завідувач кафедри проф. Шинкаренко Г.А.
"05" 09 **2022 року**

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Крикливцю Олександру Євгеновичу
(прізвище, ім'я, по батькові)

1 .Тема роботи Розробка автоматизованого сервісу моніторингу забруднення побутовими відходами вулиць та доріг міста

керівник роботи Венгерський Петро Сергійович, доктор фіз-мат. наук, професор,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені Вченою радою факультету від "13" 09 2022 року № 17

2. Строк подання студентом роботи 12 грудня 2022 року

3. Вихідні дані до роботи дослідження нейромереж, їх видів; збір даних для навчання та тестування нейромережі; розробка сервісу для моніторингу забруднення побутовими відходами; застосування нейромережі у розробленій моніторинговій системі.

4. Зміст магістерської роботи (перелік питань, які потрібно розробити)

а) вступ;
б) постановка задачі;
в) технології для вирішення задачі;
г) розробка моніторингового сервісу;
д) висновки та перспективи.

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

а) зображення фрагментів вихідного коду.

6. Консультанти розділів роботи

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

7. Дата видачі завдання 04.09.2022

КАЛЕНДАРНИЙ ПЛАН

з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1.	Аналіз літературних джерел за темою магістерської роботи	вересень	Виконано
2.	Дослідження видів нейромереж і можливостей її навчання	вересень	Виконано
3.	Навчання нейромережі	вересень-жовтень	Виконано
4.	Тестування отриманої моделі після навчання	жовтень	Виконано
5.	Розробка сервісу для моніторингу забруднення	жовтень	Виконано
6.	Об'єднання сервісу та моделі нейромережі	листопад	Виконано
7.	Тестування роботи сервісу	листопад	Виконано
8.	Написання та оформлення тексту магістерської роботи	листопад-грудень	Виконано

Студент _____ Крикливець О.Є.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Венгерський П.С.

ЗМІСТ

ВСТУП

Оцінюючи санітарно-гігієнічну та епідемічну ситуацію на території України можна констатувати, що все населення так чи інакше підлягає впливу шкідливих факторів (фізичної, хімічної та біологічної природи). Нинішню екологічну ситуацію в Україні можна охарактеризувати як кризову. Забруднення довкілля досягло такого рівня, коли воно може негативно впливати на здоров'я населення.

Кожен українець щороку створює близько 330 кг сміття. За рік на полігони і неофіційні звалища вивозять по 11 млн тонн побутових відходів. Загальна площа звалищ становить майже 5% території країни, що порівняно з розмірами Чернівецької області[1]. Станом на 2017 рік на території України було накопичено 36 млрд тон відходів, з яких 1,5 млрд тонн – небезпечні відходи. За різними даними, від 4 до 7% нашої країни завалено сміттям. Ми взяли дані кількості сміття на одного мешканця, в Україні ця цифра сягає 240 кілограмів на особу [2].

На сьогодні в Україні проблема смітників – одна з найважливіших і найактуальніших серед проблем забруднення навколишнього середовища. Ця проблема настільки нагальна не тільки в Україні, а й у всьому світі, що навіть з'явився такий вислів "відходи беруть нас за горло".

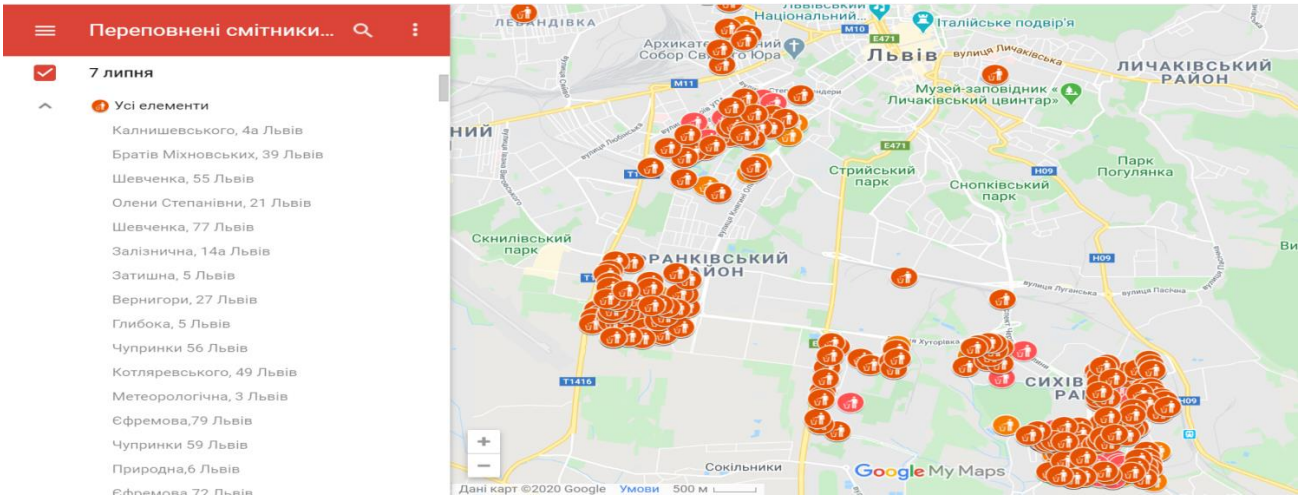

Рис. 1.1 – Знімок екрана з позначеними на карті Львова переповненими смітниками(7.07.2017) [3].

На рис. 1.1 ми можемо побачити справжню трагедію – переповнені смітники. Від невчасного вивозу сміття страждає не лише Львів, а й інші міста України. Переповнені смітники – це екологічна проблема нашої країни, сміття викликає дискомфорт для життя мешканців, в смітті починають жити пацюки, миші та бездомні тварини, які можуть переносити різні важкі захворювання. Смітники в спальних районах міст знаходять близько дитячих майданчиків та місць, де проводять час люди. Це завдає їм шкоди, бо вони дихають забрудненим повітрям, небезпеки через тварин в смітті та незручності через неприємний запах. В літню пору особливо відчутна ця проблема, бо деяким жителям важко відкривати вікна, щоб провітрити оселі, через сильний запах від сміття, яке знаходиться на сонці та гниє.

Рис. 1.2 – Смітники біля ЛНУ ім. Івана Франка(10.04.2017) [4].

Вирішенням цієї проблеми є правильно побудована логістика вивозу сміття та інформування фірм про потребу в цьому.

ПОСТАНОВКА ЗАДАЧІ

В магістерській роботі буде висвітлено можливе вирішення проблеми з переповненими смітниками, забрудненими вулицями та узбіччями з використання побудови нейронної моделі та навчання її для подальшого використання. Для цього необхідно розробити алгоритм навчання моделі, а також зібрати набір даних з фотографій, на яких зображене сміття. Після цього необхідно провести навчання нейромережі та порівняти результати її роботи на різних етапах навчання.

1.1. Маршрут без сміття

Одним із варіантів вирішення цієї проблеми є використання камер відеоспостереження для завантаження фото з різних частин міста та його околиць, на яких будуть зображені точки міста, де розташовані смітники, а також засмічені узбіччя. Ці фото будуть завантажені в програму, у якій нейромережа буде обробляти їх та робити висновок щодо того, чи є зайве сміття, яке може завдати незручності людям при пересуванні містом чи його околицями. Таким чином, людина зможе обрати собі більш чистий маршрут для пересування містом.

1.2. Вивіз сміття

Також ця система може бути використана компаніями, які займаються вивезенням сміття, для маркування проблемних зон або зон, де потрібно частіше вивозити сміття чи, можливо, встановити більшу кількість контейнерів та попрацювати з логістикою вивезення, щоб зменшити забруднення.

Також багато власників комерційних точок будуть зацікавлені, щоб саме маршрут, який проходить повз них, був найбільш чистим і найбільша кількість людей проходила чи проїжджала саме повз них, а тому зможуть посприяти встановленню додаткових контейнерів та очищенню територій.

РОЗДІЛ 1
ТЕХНОЛОГІЇ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ

1.1 Машинне навчання та нейронні мережі

Машинне навчання (Machine learning, ML) – звід методів в області штучного інтелекту, набір алгоритмів, які застосовують, щоб створити машину, яка вчиться на власному досвіді. В якості навчання машина обробляє величезні масиви вхідних даних і знаходить у них закономірності.

Не варто плутати поняття Data science і Machine learning. Ці інструменти багато в чому перетинаються, але все ж вони різні та кожен зі своїми завданнями.

Штучний інтелект (Artificial intelligence, AI) – різні технологічні та наукові рішення і методи, які допомагають зробити програми за подобою інтелекту людини. Artificial intelligence охоплює безліч інструментів, алгоритмів і систем, серед яких також усі складові Data science і Machine learning.

Data science – наука про методи аналізу даних і вилучення з них цінної інформації та знань. Вона перетинається з такими галузями як машинне навчання і наука про мислення (Cognitive Science), а також із технологіями для роботи з великими масивами даних (Big Data). Результатом роботи Data science є проаналізовані дані та знаходження правильного підходу для подальшої обробки, сортування, вибірки, пошуку даних.

Машинне навчання або Machine learning – один з розділів AI, алгоритми, що дозволяють комп’ютеру робити висновки на підставі даних, не слідуючи

строго заданим правилам. Тобто машина може знайти закономірність у складних і багатопараметричних завданнях (які мозок людини не здатен вирішити), таким чином знаходячи більш точні відповіді. Як результат – правильне прогнозування.

Нейронна мережа за допомогою штучних нейронів моделює роботу людського мозку (нейронів), що вирішує певне завдання, самонавчається з урахуванням попереднього досвіду. І з кожним разом робить усе менше помилок. Нейромережі є одним із видів машинного навчання, а не окремим інструментом.

Ціль машинного навчання – частково або й повністю автоматизувати рішення різних складних аналітичних задач. Тому насамперед машинне навчання покликане давати максимально точні прогнози на підставі вступних даних, щоб власники бізнесів, маркетологи і співробітники могли приймати правильні рішення у своїй роботі. В результаті навчання машина може передбачати результат, запам’ятовувати його, відтворювати за необхідності, вибирати кращий із декількох варіантів.

Машинне навчання будується на трьох китах і максимально ефективній взаємодії з замовником:

А) дані – базова інформація, надати яку ми зазвичай просимо клієнта. Сюди входять будь-які вибірки даних, роботі з якими потрібно навчити систему;

Б) ознаки – ця частина роботи проводиться в тісній співпраці з клієнтом. Ми визначаємо ключові бізнес-потреби і спільно вирішуємо, які саме характеристики та властивості повинна відстежувати система в результаті навчання;

В) алгоритм – вибір методу для вирішення поставленого бізнес-завдання. Це завдання ми вирішуємо без участі клієнта, силами наших співробітників.

За ознакою наявності вчителя, навчання ділиться на навчання з учителем (Supervised Learning), без вчителя (Unsupervised Learning) та з підкріпленням (Reinforcement Learning).

При вмілому підході, комбінуючи різні види машинного навчання, можна домогтися автоматизації всіх рутинних бізнес-процесів. Іншими словами, роботи, підготовлені за допомогою машинного навчання, можуть виконувати всю

рутинну роботу. Людям же залишається вся творча частина: складання стратегій, ведення переговорів, укладення договорів та інше. Це важливий фактор, оскільки машина не може вийти за задані їй рамки, а людський мозок вміє мислити непересічно.

Штучні нейронні мережі (ШНМ) можуть розглядатися як спрямований граф зі зваженими зв'язками, у якому штучні нейрони є вузлами. По архітектурі зв'язків ШНМ можуть бути згруповані у два класи (рис. 1.3): мережі прямого поширення, у яких графи не мають петель, і рекурентні мережі, або мережі зі зворотними зв'язками.

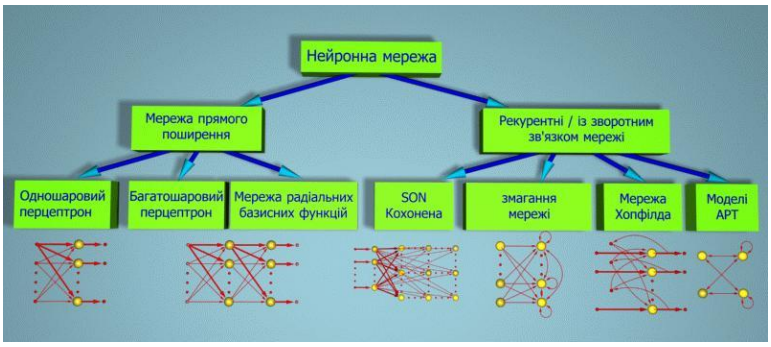

Рис 1.3 – Класифікація нейронних мереж

Класифікація нейронних мереж

За способом подачі інформації до входів нейронної мережі розрізняють:

- подачу сигналів на синапси вхідних нейронів;
- подачу сигналів до виходів вхідних нейронів;
- подачу сигналів у вигляді ваги синапсів вхідних нейронів;
- адитивну подачу на синапси вхідних нейронів.

За способом зняття інформації з виходів нейронної мережі розрізняють:

- зняття з виходів вихідних нейронів;
- зняття з синапсів вихідних нейронів;
- зняття у вигляді значень ваги синапсів вихідних нейронів;
- адитивний зйом із синапсів вихідних нейронів.

За організацією навчання поділяють навчання нейронних мереж з вчителем (supervised neural networks) та без вчителя (nonsupervised). За способом навчання поділяють навчання за входами і за виходами. При навчанні за входами навчальний приклад є тільки вектором вхідних сигналів, а при навчанні за виходами до нього входить і вектор вихідних сигналів, який відповідає вхідному вектору.

За способом подання прикладів розрізняють подання окремих прикладів і “сторінки” прикладів. У першому випадку зміна стану нейронної мережі (навчання) відбувається після подання кожного прикладу. У другому – після подання “сторінки” (множини) прикладів на підставі аналізу одразу усіх їх.

За особливостями моделі нейрона розрізняють нейрони з різними нелінійними функціями.

1.2 Застосування нейронних мереж з учителем

Для вирішення цієї задачі нам необхідно навчити нейромережу розпізнавати на фото переповнені смітники (це ті, в яких сміття виступає за межі країв і не дає можливості його закрити) або сміття, яке лежить біля смітників або на узбіччі.

Нейромережа зможе розпізнавати на зображенні тільки ті об’єкти, які її навчать розпізнавати. Тобто моя нейромережа буде вчитися розпізнавати виключно сміття і не брати до уваги інші об’єкти, включаючи сам смітник.

Один з основних підходів, найбільш широко використовуваних в області розпізнавання зображень, являє собою застосування класичних моделей класифікаторів, що навчаються з учителем. Для навчання таких моделей використовуються маркована вибірка даних, що складається з масиву зображень і відповідного їм масиву міток, що визначають категорію, до якої відноситься зображення. В процесі навчання масив даних розділяється на дві нерівні частини:

- навчальну вибірку і тестову вибірку, а потім за допомогою специфічного для конкретного алгоритму правила навчання параметри моделі налаштовуються

з використанням навчальної вибірки таким чином, щоб, отримавши як вхідні дані зображення, модель на виході виробляла б мітку відповідного класу. Цей підхід представлений безліччю моделей, серед яких найбільш широко використовуваними є регресивна модель, штучна нейронна мережа (багатошаровий перцептрон), метод опорних векторів, а також дерева прийняття рішень і моделі-ансамблі, що являють собою поєднання деяких перерахованих моделей.

Здатність навчатися на базі вибірки робить нейронні мережі та споріднені з ними моделі придатними для розпізнавання природних зображень навколишнього світу, що відрізняються нечіткою структурою і безліччю варіацій в межах класу.

На рис. 1.4 зображена нейронна мережа з прихованими шарами та її принцип дії:

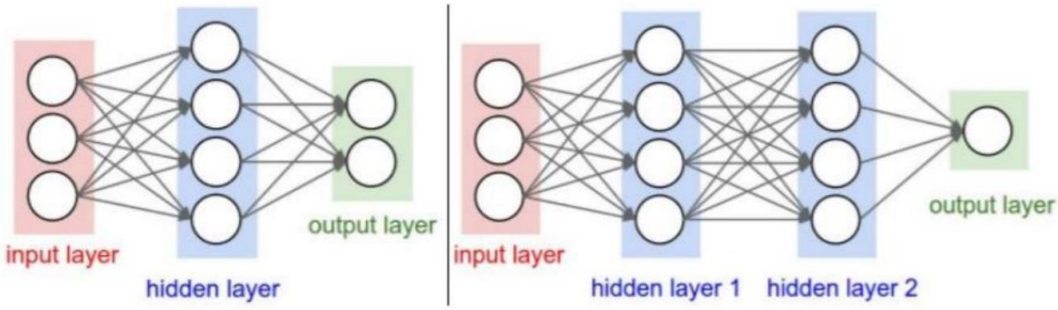

Рис. 1.4 – Ліворуч: двошарова нейронна мережа; праворуч: тришарова нейронна мережа.

Для вирішення цієї задачі оптимально використовувати нейронну мережу типу згорткова нейронна мережа (ЗНМ, англ. *convolutional neural network, CNN, ConvNet*). Aruni Roy Chowdhury, Tsung-Yu Lin, Subhranshu Maji та Erik Learned-Miller запропонували метод Bilinear CNN (B-CNN) для ідентифікації особи, який показав різке зростання продуктивності на деяких дрібнозернистих зразках. Модель ліквідує розрив між текстурами і деталями на основі алгоритмів CNN.

Архітектура являє собою орієнтований ациклічний граф (англ. *directed acyclic graph, DAG*), обидві мережі можуть бути навчені одночасно за допомогою градієнтного методу зворотного поширення помилки (англ. *backpropagation*).

Замість того, щоб навчати CNN розпізнавати сміття з нуля, що потребує пошук оптимальної архітектури нейронної мережі та масивні бази даних, можна використати B-CNN, яка може використовувати заздалегідь підготовлені мережі і адаптувати їх до задачі розпізнавання сміття.

1.3 Побудова алгоритму розв’язування задачі

Визначимо алгоритм роботи нашої програми:

Рис. 1.5 – Алгоритм роботи.

Всі ці фактори допоможуть зробити міста та околиці чистішими і сприятливішими для комфортного та чистого життя.

1.4 Розробка програмного забезпечення для реалізації алгоритму

Custom Vision – це сервіс штучного інтелекту і платформа для завершеного застосування комп'ютерного зору для ваших конкретних потреб.

Почати тренувати модель комп’ютерного зору, можна просто завантаживши та позначивши кілька зображень (мінімум 15 зображень). Модель тестує себе на цьому і постійно покращує точність за допомогою циклу зворотного зв’язку під час додавання нових зображень. Щоб прискорити розвиток, можна використати настроювані вбудовані моделі для роздрібної торгівлі, виробництва та харчові продукти. Наприклад, як Minsur, одна з найбільших олов'яних шахт у світі, використовує Custom Vision для сталого видобутку.

Зручний інтерфейс пропонує нам розробляти та розгортати власні моделі комп'ютерного зору. Потім, можна експортувати навчену модель на пристрій або під’єднати за допомогою API, щоб запустити розпізнавання зображень у реальному часі.

1.4.1. Сервіс для навчання нейромережі

Для роботи з нейронною мережею та її навчання я використав сервіс Custom Vision, який надає хмарна платформа “Microsoft Azure”. Custom Vision використовує алгоритм машинного навчання для нанесення міток на зображення. Перед тим як почати тренування мережі, ви повинні завантажити зображення та нанести на них мітки. Потім алгоритм здійснює підготовку за допомогою цих даних і обчислює власну точність, випробовуючи себе на тих самих зображеннях. Згодом, як алгоритм буде навчений, ви можете протестувати, перевчити і в кінцевому підсумку використовувати його для класифікації нових зображень відповідно до потреб вашого додатка. Ви також можете експортувати саму модель для офлайн-використання [5].

Функціональні можливості Custom Vision можна розділити на дві групи: класифікація зображення та виявлення об’єктів на зображенні. Класифікація зображень застосовує одну або кілька міток до зображення. Виявлення об’єктів подібне, але воно також повертає координати на зображенні, де можна знайти застосовані мітки.

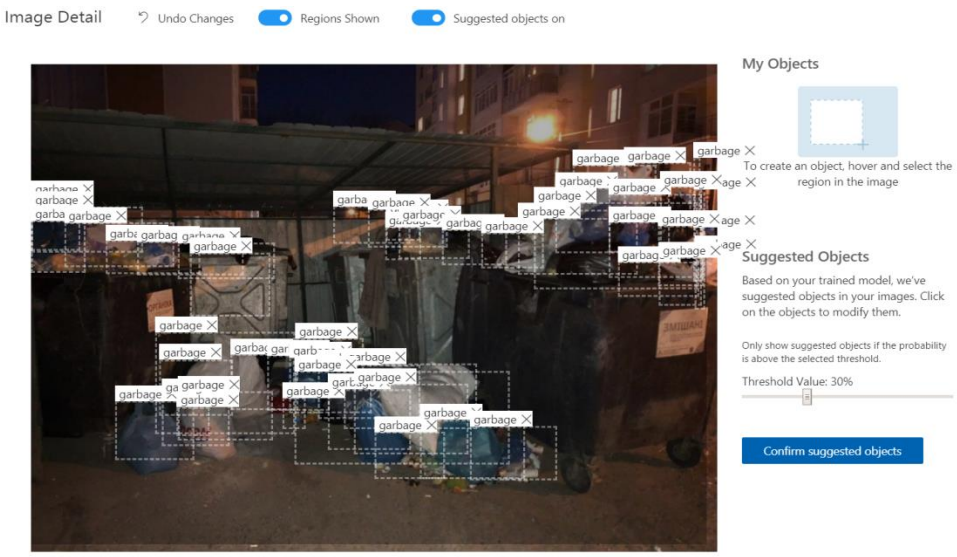

Рис. 1.6 – Нейромережа пропонує, де на фотографії позначити сміття.

На рис. 1.6 можна побачити як нейромережа пропонує нанести мітки на деякі частини фотографії. Ця функція стає доступною після першого тренування мережі і з кожним наступним тренування вона працює краще.

Окрім того, після кожного навчання сервіс надає статистику точності, отже користувач може відстежувати прогрес його нейромережі (див. рис. 1.7).

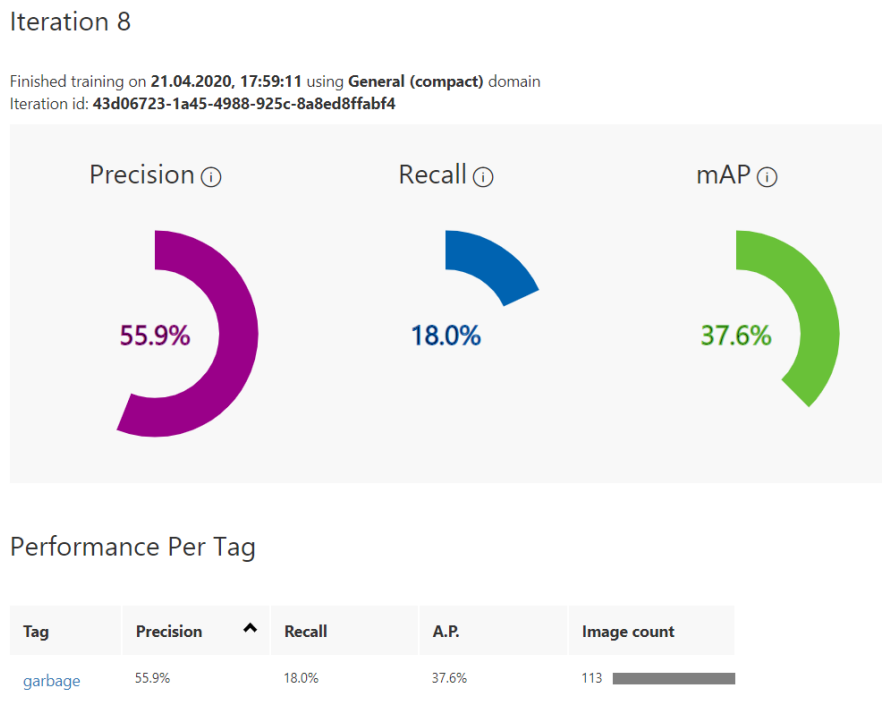

Рис. 1.7 – Приклад статистики після навчання нейромережі.

1.4.2. Вибір моделі глибинного навчання

Глибинне навчання – це підмножина машинного навчання. Для підготовки моделей глибинного навчання потрібна велика кількість даних. Шаблони в даних представлені рядом шарів. Зв'язки в даних кодуються як з'єднання між шарами, що містять ваги. Чим більша вага, тим міцніше відносини. У сукупності ця серія шарів і з'єднань відома як штучні нейронні мережі. Чим більше шарів у мережі, тим вона "глибша", що робить її глибокою нейронною мережею.

Існують різні типи нейронних мереж, найпоширенішими є багатошаровий перцептрон (англ. *multilayer perceptron, MLP*), згорткова нейронна мережа (ЗНМ, англ. *convolutional neural network, CNN, ConvNet*) та рекурентна нейронна мережа (РНМ, англ. *recurrent neural networks, RNN*). Найбільш базовим є MLP, який відображає набір входів на набір виходів. Цей тип нейронної мережі хороший, коли дані не мають просторової чи часової складової. CNN використовує звивисті шари для обробки просторової інформації, що міститься в даних. Хорошим випадком використання для CNN є обробка зображень для виявлення наявності функції в області зображення (наприклад, чи є ніс в центрі зображення?). Нарешті, RNN дозволяють використовувати стійкість стану або пам'яті як вхідні дані. RNN використовуються для аналізу часових рядів, де важливе послідовне впорядкування та контекст подій.

1.4.3. Загальні відомості про модель

Виявлення об'єктів – це завдання обробки зображень. Тому більшість моделей глибокого навчання, навчених вирішувати цю проблему, – це CNN. Модель, яку я використовую – це модель Tiny YOLOv2, більш компактна версія моделі YOLOv2 [7]. Tiny YOLOv2 навчали на базі даних Pascal VOC і складається вона з 15 шарів, які можуть передбачати 20 різних класів об'єктів. Оскільки Tiny

YOLOv2 є скороченою версією оригінальної моделі YOLOv2, компроміс здійснюється між швидкістю та точністю. Різні шари, що складають модель, можна візуалізувати за допомогою таких інструментів, як Netron. Вивчення моделі призведе до зіставлення з'єднань між усіма шарами, які складають нейронну мережу, де кожен шар буде містити ім'я шару разом з розмірами відповідних вхідних і вихідних даних. Структури даних, що використовуються для опису входів і виходів моделі, відомі як тензори. Тензорами можна вважати контейнери, які зберігають дані в N-розмірах. У випадку Tiny YOLOv2 назва вхідного шару – *"image"*, і він очікує тензор розмірів *3 x 416 x 416*. Назва вихідного шару – *"grid",* він генерує вихідний тензор розмірами *125 x 13 x 13*.

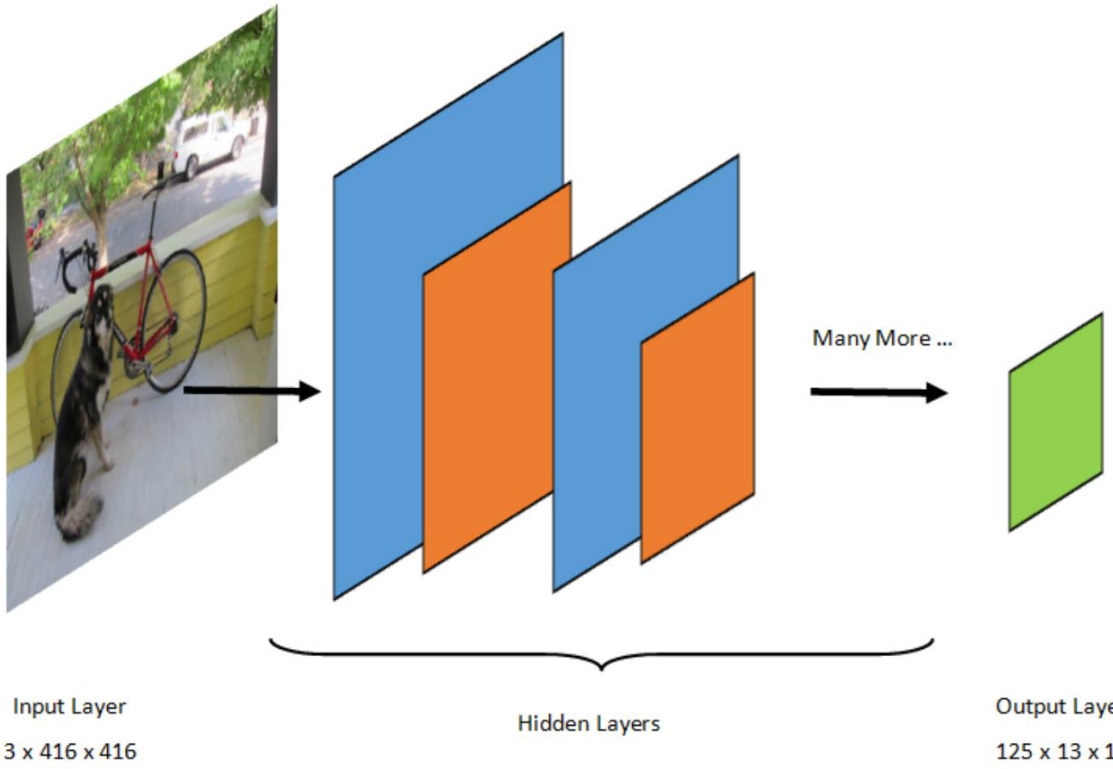

Рис. 1.8 – Приклад обробки зображення.

На рис. 1.8 видно, як модель YOLO приймає зображення *3 (RGB) x 416px x 416px*. Вона обробляє це зображення, передаючи його через різні шари для створення вихідних даних. Вихідні дані ділять дане зображення на сітку *13 x 13*, де кожна клітинка сітки – це *32px x 32px*. Кожна клітинка сітки містить 5

можливих обмежувальних прямокутників. Обмежувальний прямокутник містить 25 елементів (див. рис. 1.9):

- *x* – координата по осі X для центру прямокутника щодо клітинки сітки, з якою він пов’язаний.
- *y* – координата по осі Y для центру прямокутника щодо клітинки сітки, з якою він пов’язаний.
- *w* – ширина прямокутника.
- *h* – висота прямокутника.
- *o* – значення ймовірності того, що об’єкт існує в межах прямокутника, також відомий як оцінка об’єкта.
- *p1-p20* — ймовірності для кожного з 20 класів, прогнозованих моделлю.

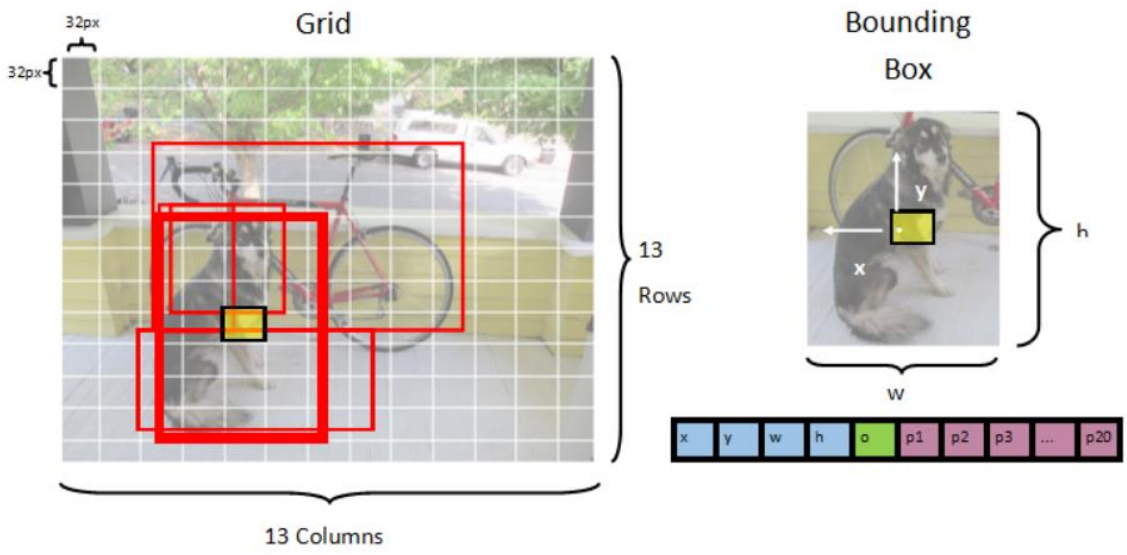

Рис. 1.9 – Приклад обмежувального прямокутника.

В результаті 25 елементів, що описують кожен з п’яти обмежувальних прямокутників, складають 125 елементів, що містяться в кожній клітинці сітки.

Вихідні дані, що формуються попередньо навченою моделлю ONNX, являють собою масив довжини 21125, який представляє тензорні елементи зі *125 x 13 x 13* вимірами.

1.4.4. Експорт моделі

Як було вже згадано вище після того, як модель навчено, її можна експортувати у різних форматах для офлайн-використання:

- CoreML для iOS використання;
- TensorFlow для Android;
- ONNX для Windows ML;
- Dockerfile для Azure IoT Edge, Azure Functions, AzureML.

Оскільки, я обрав ASP.NET Core 3.0 для реалізації своєї програми, то найкращий, для мене, формат – ONNX.

1.4.5. Формат моделі

Open Neural Network Exchange (ONNX) – це формат з відкритим вихідним кодом для моделей штучного інтелекту[8]. ONNX підтримує взаємодію між різними платформами. Це означає, що модель можна навчити в одній з багатьох популярних платформ машинного навчання, таких як PyTorch, перетворити її в формат ONNX і використовувати в іншій інфраструктурі, такій як ML.NET (див. рис. 1.10).

Рис. 1.10 – Приклад експорту і використання ONNX моделі.

1.4.6. ML.NET

ML.NET – це безплатна бібліотека машинного навчання програмного забезпечення для мов програмування C# і F#.

У ML.NET взаємодія з ONNX досягається за допомогою пакетів NuGet *ImageAnalytics*[9] і *OnnxTransformer* [10]. Пакет ImageAnalytics містить ряд перетворень, які приймають зображення і кодують його в числові значення, які можна використовувати як вхідні дані в конвеєрі прогнозування або навчання. Пакет OnnxTransformer використовує середовище виконання ONNX для завантаження моделі ONNX і використовує її для створення прогнозів на основі наданих вхідних даних (див. рис. 1.11).

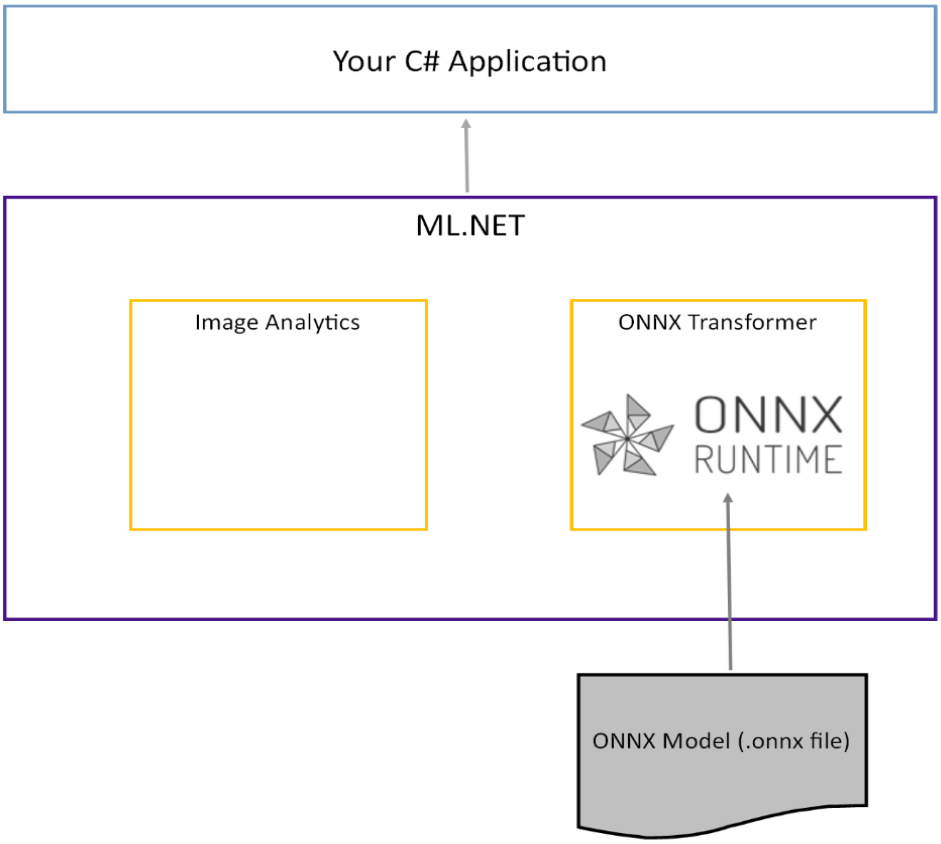

Рис. 1.11 – Приклад взаємодії C#, ML.NET та ONNX моделі.

1.4.7. Приклад роботи мережі та коду

Для демонстрації роботи нейромережі взято фотографію (див. рис. 1.12), яка не використовувалася для її навчання. Можемо бачити, що мережа виділяє прямокутниками ділянки зі сміттям, при цьому уникаючи виділення самого сміттєвого бака.

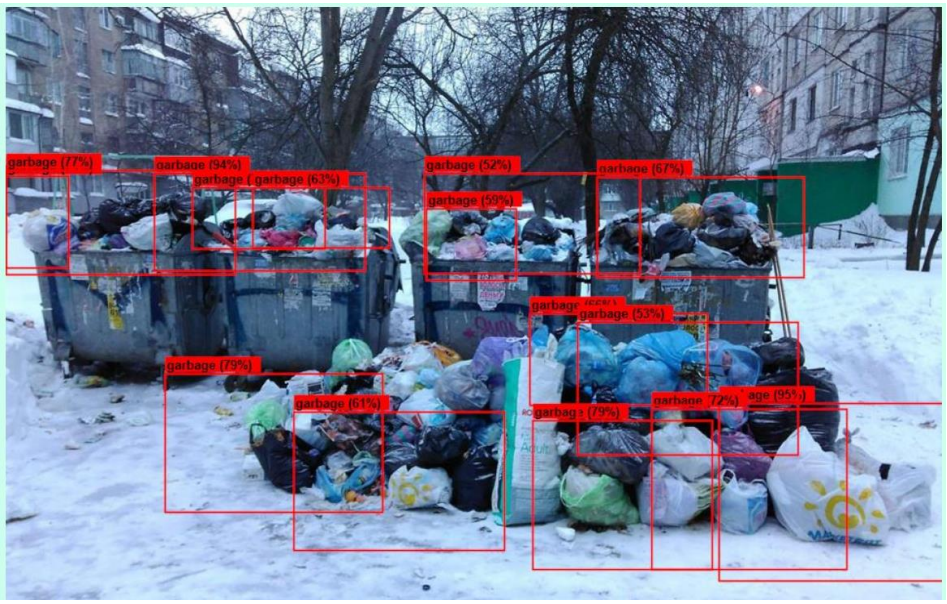

Рис. 1.12 – Приклад роботи нейромережі.

Приклад коду для роботи з моделлю і визначення сміття на зображеннях.

Обробка фотографії починається із методу *DetectAndPaintImage*, який у параметрах отримує три параметри (див. рис. 1.13):

1. *imageInputData* – сама фотографія, яка буде оброблятися;
2. *minProbabilityToShow* – щоб зменшити кількість визначених ділянок на вихідному зображенні, вказуємо мінімальну ймовірність для ділянки;
3. *minThresholdToShow* – також, щоб зменшити кількість ділянок і отримати точніші результати вказуємо мінімальне значення **IoU** (Intersection over union [11]).

```
1 reference
private string DetectAndPaintImage(ImageInputData imageInputData, float minProbabilityToShow, float minThresholdToShow)
{
    //Predict the objects in the image
    _objectDetectionService.DetectObjectsUsingModel(imageInputData, minProbabilityToShow, minThresholdToShow);
    var img = _objectDetectionService.DrawBoundingBox(imageInputData.Image);

    using (MemoryStream m = new MemoryStream())
    {
        img.Save(m, img.RawFormat);
        byte[] imageBytes = m.ToArray();

        // Convert byte[] to Base64 String
        return Convert.ToBase64String(imageBytes);
    }
}
```

Рис. 1.13 – Початковий метод для обробки зображення

Далі у методі *DetectObjectsUsingModel* (див. рис. 1.14) спочатку визначаються ділянки, які модель вважає сміттям, а після того відсіюються усі, які не відповідають заданим критеріям. Отримуємо масив ділянок, які потрібно відзначити на фотографії.

```
2 references
public void DetectObjectsUsingModel(ImageInputData imageInputData, float minProbabilityToShow, float minThresholdToShow)
{
    var probs = _predictionEngine.Predict(imageInputData).PredictedLabels;
    var boundingBoxes = _outputParser.ParseOutputs(probs, (minProbabilityToShow / 100));
    _filteredBoxes = _outputParser.FilterBoundingBoxes(boundingBoxes, 50, (minThresholdToShow / 100));
}
```

Рис. 1.14 – Обробка зображення моделлю

Останнім виконується метод *DrawBoundingBox* (див. рис. 1.15), який відзначить червоними прямокутниками усі ділянки на фото, які модель визначила, що це ділянки зі сміттям.

```
2 references
public Image DrawBoundingBox(Image image)
{
    //Image image = Image.FromFile(imageFilePath);
    var originalHeight = image.Height;
    var originalWidth = image.Width;
    if (_filteredBoxes != null)
    {
        foreach (var box in _filteredBoxes)
        {
            // Process output boxes
            var x = (uint) Math.Max(box.Dimensions.X, 0);
            var y = (uint) Math.Max(box.Dimensions.Y, 0);
            var width = (uint) Math.Min(originalWidth - x, box.Dimensions.Width);
            var height = (uint) Math.Min(originalHeight - y, box.Dimensions.Height);

            // Fit to current image size
            x = (uint) originalWidth * x / ImageSettings.ImageWidth;
            y = (uint) originalHeight * y / ImageSettings.ImageHeight;
            width = (uint) originalWidth * width / ImageSettings.ImageWidth;
            height = (uint) originalHeight * height / ImageSettings.ImageHeight;

            using (var thumbnailGraphic = Graphics.FromImage(image))
            {
                thumbnailGraphic.CompositingQuality = CompositingQuality.HighQuality;
                thumbnailGraphic.SmoothingMode = SmoothingMode.HighQuality;
                thumbnailGraphic.InterpolationMode = InterpolationMode.HighQualityBicubic;

                // Define Text Options
                var drawFont = new Font("Arial", 10, FontStyle.Bold);
                var size = thumbnailGraphic.MeasureString(box.Description, drawFont);
                var fontBrush = new SolidBrush(Color.Black);
                var atPoint = new Point((int) x, (int) y - (int) size.Height - 1);

                // Define BoundingBox options
                var pen = new Pen(box.BoxColor, 2.2f);
                var colorBrush = new SolidBrush(box.BoxColor);

                // Draw text on image
                thumbnailGraphic.FillRectangle(colorBrush, (int) x, (int) (y - size.Height - 1),
                    (int) size.Width, (int) size.Height);
                thumbnailGraphic.DrawString(box.Description, drawFont, fontBrush, atPoint);

                // Draw bounding box on image
                thumbnailGraphic.DrawRectangle(pen, x, y, width, height);
            }
        }
    }

    return image;
}
```

Рис. 1.15 – Позначення ділянок на зображенні

РОЗДІЛ 2
ОПИС ІНТЕРФЕЙСУ СЕРВІСУ ДЛЯ МОНІТОРИНГУ ЗАБРУДНЕННЯ

У програмі існує два типи користувачів: корпоративний акаунт для компанії, яка займається вивозом сміття та акаунт адміністратора. Основний функціонал розроблений для компаній, які займаються збором сміття, а адміністратор створює особисті акаунти для таких компаній.

2.1 Опис інтерфейсу програми для корпоративного користувача

Після запуску програми користувач попадає на сторінку логування (див. рис. 2.1).

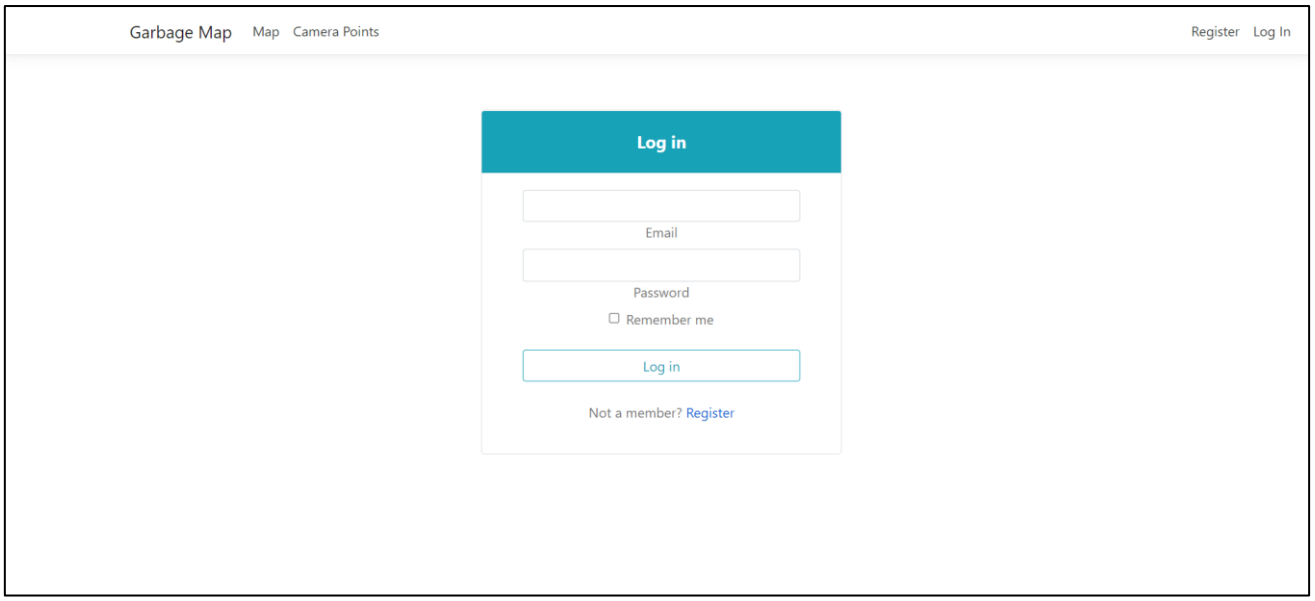

Рис. 2.1 – Сторінка логування

Увійшовши в програму, користувач опиняється на головній сторінці. Тут він може подивитися на карту міста, а також на точки, де знаходяться сміттєві баки, які обслуговуються цією компанією (див. рис. 2.2).

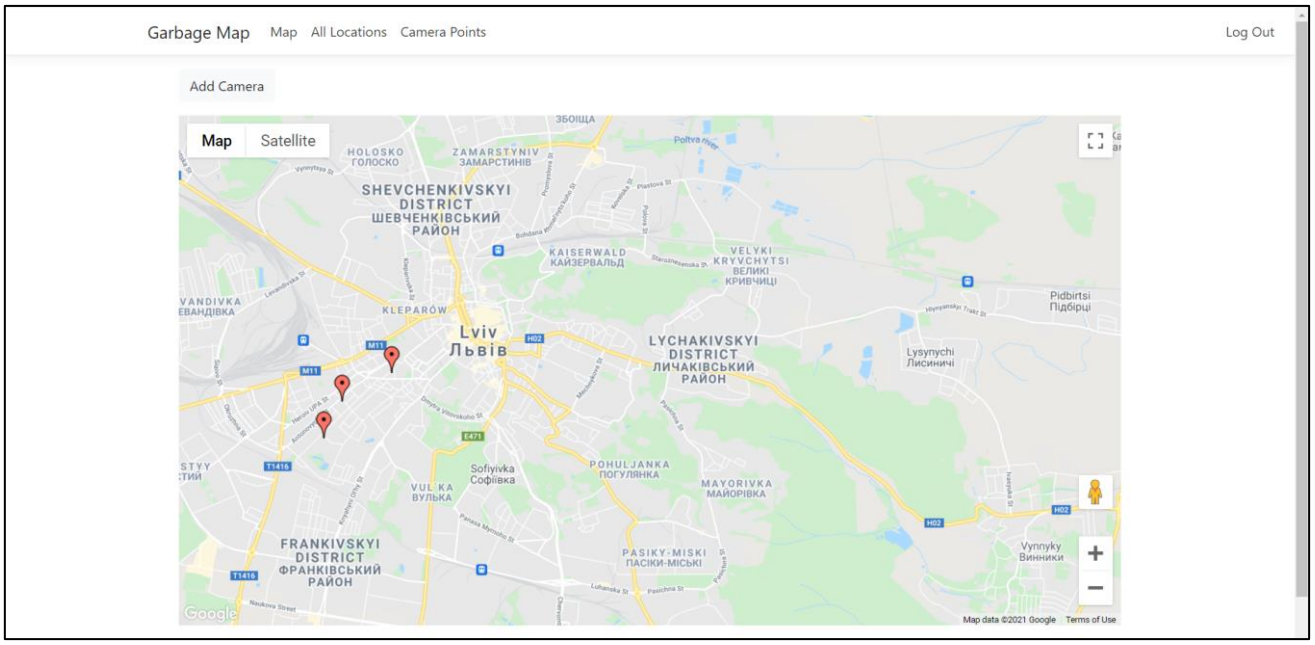

Рис. 2.2 – Карта міста зі сміттєвими баками.

Також на цій сторінці є можливість додати нову точку зі сміттєвими баками через діалогове вікно, яке появиться після натискання кнопки *“Add Camera”* (див. рис. 2.3).

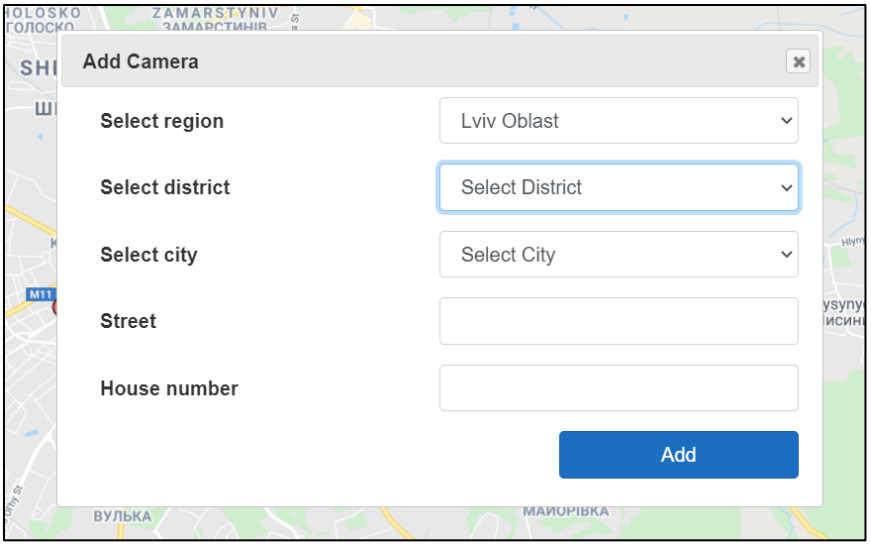

Рис. 2.3 – Діалогове вікно для додавання нової точки на карту.

Після переходу на сторінку *“Camera Points”*, користувач побачить список усіх точок (див. рис. 2.4).

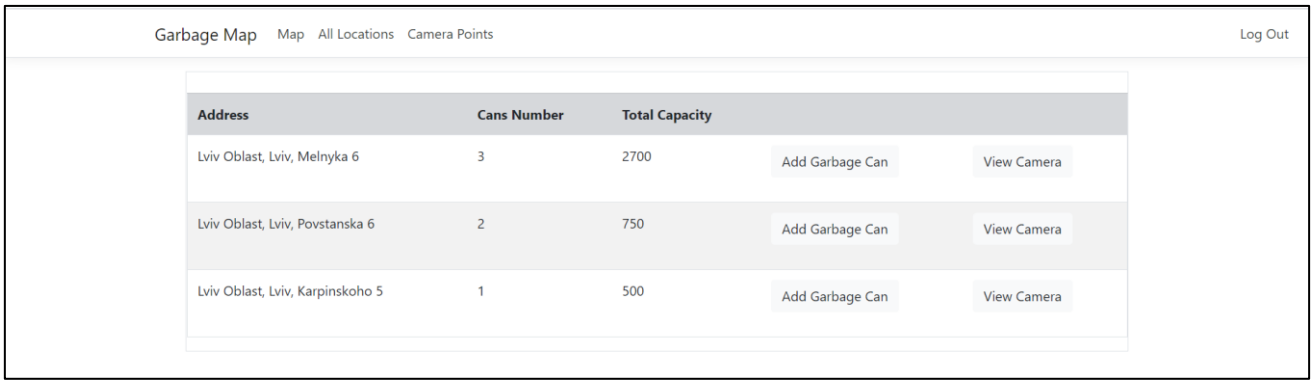

Рис. 2.4 – Список усіх точок.

Також, як видно на фотографії вище, доступна інформація про кількість сміттєвих баків *(“Cans Number”)* для кожної точки, а також сумарний літраж для всіх баків *(“Total Capacity”)*.

Натиснувши кнопку *“View Camera”*, користувач переходить на сторінку, де в режимі реального часу, він отримує зображення смітників. Це зображення вже оброблене нейромережею для виявлення сміття і допомагає оцінити наскільки сильно смітники заповнені (див. рис. 2.5).

Рис. 2.5 – Виявлення сміття на зображенні.

Натиснувши кнопку *“Add Garbage Can”*, в користувача відкривається діалогове вікно, яке дає можливість додати сміттєві баки для місця зі смітниками (див. рис. 2.6).

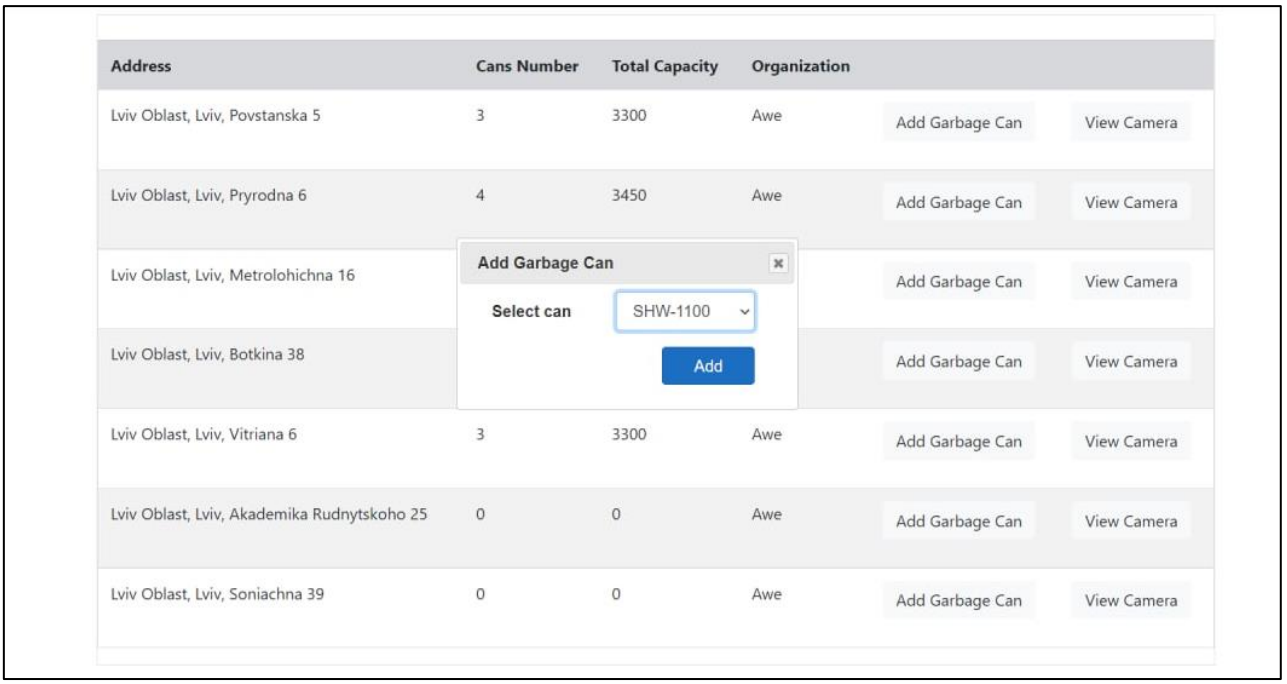

Рис. 2.6 – Діалог “Add Garbage Can”.

2.2 Опис інтерфейсу програми для адміністратора

Для адміністратора програма не відрізняється, але дає розширені можливості у перегляді точок зі сміттєвими баками для кожної з компаній, а також доступ до камер.

Порівняємо інтерфейс для корпоративного користувача (див. рис. 2.7) і адміністратора (див. рис. 2.8).

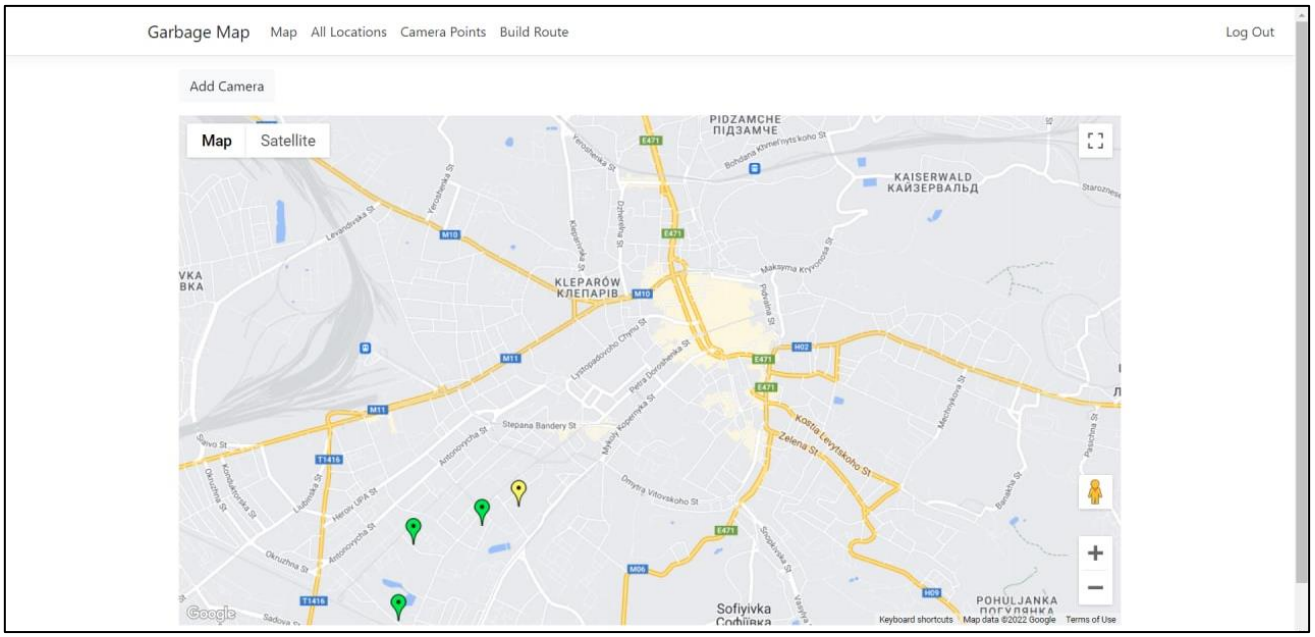

Рис. 2.7 – Інтерфейс корпоративного користувача.

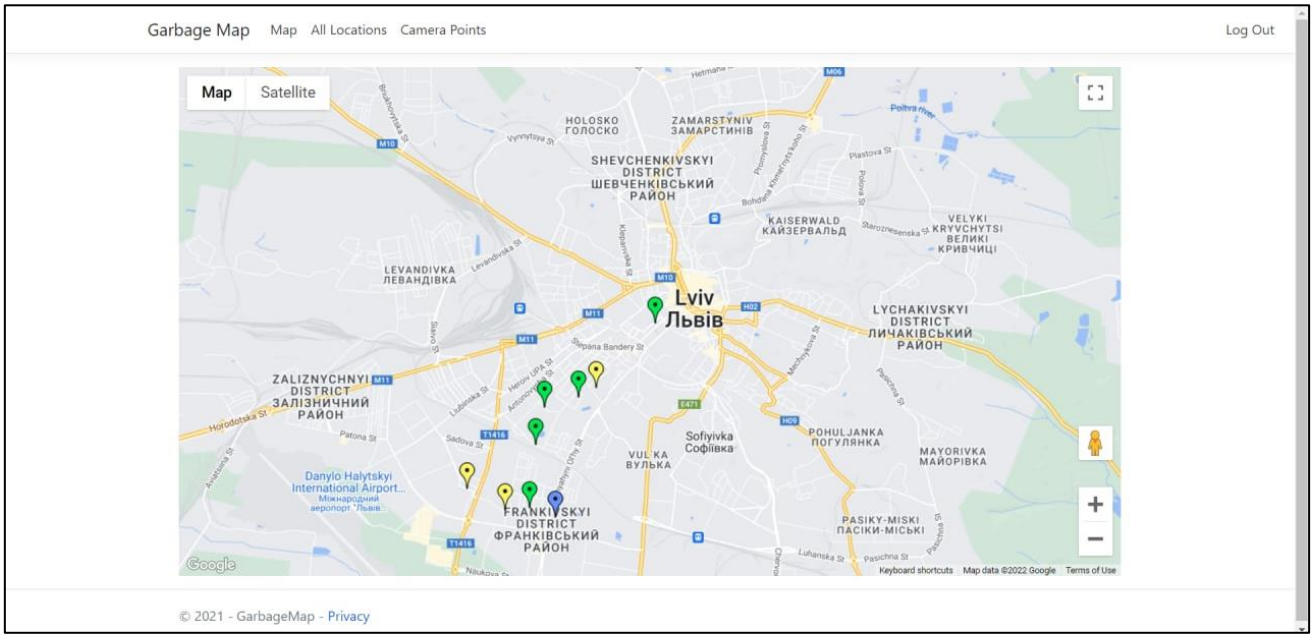

Рис. 2.8 – Інтерфейс адміністратора.

2.3 Опис позначок на карті

У вкладці “Map” знаходиться карта, де вказані місця зі смітниками. Ці місця позначаються різними кольорами і символізують приблизну наповненість смітників у певному місці.

Типи міток:

- Синя мітка – вказує на місце, де знаходяться машини певної компанії і звідки вони будуть виїжджати для збору сміття.
- Зелена, жовта та червона мітки символізують приблизну наповненість смітників (зелений колір – смітники майже порожні, жовтий колір – заповнені більше ніж половина і червоний колір символізує, що смітник переповнений і потребує негайного очищення).
- Прапор – мітка, яка показує кінцеву точку машини для вивозу сміття (у нашому випадку це “Львівський полігон твердих побутових відходів”[12]).

2.4 Побудова маршруту для збору сміття

Коли корпоративний працівник хоче побудувати маршрут для збору сміття, він переходить у меню “*Build Route*” і автоматично починає працювати алгоритм прокладання маршруту.

Етапи прокладання маршруту:

1. Спочатку виконується запит до бази даних (SQL Server) і витягуються усі місця зі смітниками для конкретної організації.
2. Після того будується повний неорієнтований граф. Кожне місце зі смітником з’єднане з усіма іншими місцями, а також з місцем старту і кінця. Ще на етапі створення нової точки, я за допомогою Google API визначаю координати точки, тому можу порахувати відстань між двома точками на карті використовуючи широту і довготу. В результаті я отримую матрицю, діагональ якої нулі (оскільки, відстань від точки до самої себе дорівнює нуль).
3. На останньому етапі я використовую метод найближчого сусіда і таким чином прокладаю маршрут від початкового місця до сміттєзвалища.

РОЗДІЛ 3
ОПИС МОЖЛИВОГО ЗАСТОСУВАННЯ РОЗРОБЛЕНОЇ ПРОГРАМИ

З кожним роком стає все більше і більше камер у містах. Багато з них застосовуються міськими службами в межах проєкту “Безпечне місто” для спостереження за діяльністю в місті.

Розроблену програму можна використати для об’єднання усіх служб вивозу сміття. Ідея полягає в тому, що оскільки камер стає більше, то велика ймовірність що місця, де розташовані смітники, будуть попадати на камеру. Це дасть можливість службам в режимі реального часу оцінити, які смітники переповнені, а які ще порожні.

Кожна служба може зареєструватися в програмі і розмістити на карті смітники, які вона обслуговує. І якщо поблизу є камера, то вона за допомогою описаної вище нейронної мережі буде оцінювати завантаженість смітників і посилати сигнал в службу вивозу сміття, якщо смітники переповнені.

Це дозволить службам ефективніше використовувати їхні ресурси для вивозу сміття. Тобто, не буде єдиного маршруту, а маршрут буде будуватися на основі того, де потрібно забрати сміття. Це дозволить уникнути ситуацій, коли в одних точках смітники постійно заповнені, а в інших – порожні.

ВИСНОВКИ

В цій магістерській роботі була проведена розробка програми та застосування нейронної мережі для розпізнавання сміття на узбіччях та в переповнених сміттєвих баках.

Оскільки наші міста стають технологічно розвинуті і запроваджується все більше різних систем для моніторингу ситуації у місті, а також збільшується кількість камер, то велика ймовірність того, що підхід, який описаний в магістерській роботі, щодо моніторингу завантаженості смітників, реальний для впровадження у містах України. Запровадження такої системи полегшить збір сміття для служб, які вивозять сміття, а також для людей, які на карті можуть знайти найближчий до них смітник.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сміттєве майбутнє. Коли і як буде вирішена проблема утилізації сміття в Україні [Електронний ресурс]. – Режим доступу: https://news.finance.ua/ua/news/-/419120/smittyeve-majbutnye-koly-i-yak-bude-vyrishena-problema-utylizatsiyi-smittya-v-ukrayini.

2. Надія на очищення. Як Україна може подолати сміття [Електронний ресурс]. – Режим доступу: https://ukr.lb.ua/society/2018/04/28/395698_nadiya_ochishchennya_yak_ukraina_mozhe.html.

3. Переповнені смітники Львова [Електронний ресурс]. – Режим доступу: https://www.google.com/maps/d/u/0/viewer?mid=18dKse0w8qfZDDEd5wp6s6OvizvU&ll=49.81271806353455%2C24.03422163597463&z=13.

4. Біля Університету у Львові страшно переповнені смітники [Електронний ресурс]. – Режим доступу: https://galinfo.com.ua/news/bilya_universytetu_u_lvovi_strashno_perepovneni_smitnyky_257190.html.

5. Tutorial: Detect objects using ONNX in ML.NET [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/dotnet/machine-learning/tutorials/object-detection-onnx#onnx-object-detection-sample-overview.

6. What is Custom Vision? [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/en-us/azure/cognitive-services/custom-vision-service/home.

7. Redmon J., Farhadi A. YOLO9000: Better, Faster, Stronger. *Cornell University*. 2016. URL: https://arxiv.org/abs/1612.08242v1 (дата звернення: 05.05.2020).

8. YOLO: Real-Time Object Detection [Електронний ресурс]. – Режим доступу: https://pjreddie.com/darknet/yolov2/.

9. Onnx.ai [Електронний ресурс]. – Режим доступу: https://onnx.ai/get-started.html.

10. YOLO: Real-Time Object Detection [Електронний ресурс]. – Режим доступу: https://pjreddie.com/darknet/yolov2/.

11. Intersection over Union (IoU) [Електронний ресурс]. – Режим доступу: https://hasty.ai/docs/mp-wiki/metrics/iou-intersection-over-union.

12. Львівський полігон твердих побутових відходів [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Львівський_полігон_твердих_побутових_відходів

13. Microsoft.ML.Transforms.Image Namespace [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/ru-ru/dotnet/api/microsoft.ml.transforms.image?view=ml-dotnet.

14. OnnxTransformer Class [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/ru-ru/dotnet/api/microsoft.ml.transforms.onnx.onnxtransformer?view=ml-dotnet.

15. Subramanian D. A Simple Introduction to K-Nearest Neighbors Algorithm. Towards Data Science. URL: https://towardsdatascience.com/a-simple-introduction-to-k-nearest-neighbors-algorithm-b3519ed98e.

16. K-Nearest Neighbor(KNN) Algorithm for Machine Learning. JavaTpoint. URL: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning.

17. Papadopoulos A. N., Manolopoulos Y. Nearest Neighbor Search: A Database Perspective. New York : Springer Science+Business Media, Inc., 2005. 168 p.