Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ПІД ОПЕРАЦІЙНУ СИСТЕМУ IOS

Текстова частина до курсової роботи

за спеціальністю 121 «Інженерія програмного забезпечення»

Керівник курсової роботи

ст.в. Борозенний С.О.
(прізвище та ініціали)

_____
(підпис)

“____” _____ 2022 р.

Виконав студент Семенко Е.О.
(прізвище та ініціали)

_____
(підпис)

“____” _____ 2022 р.

Київ 2022

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри мультимедійних систем,
доцент, к.ф-м.н.
_____О. П. Жежерун
(підпис)
“____”_____2022 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студенту 3 курсу факультету інформатики

Семенко Еліні Олександрівні

ТЕМА: Розробка мобільного застосунку під операційну систему iOS

Зміст ТЧ до курсової роботи:

Зміст
Анотація
Вступ
1 Дослідження та аналіз предметної області
2 Аналіз доступних інструментів
3 Проектування архітектури додатку
4 Розробка додатку
Висновки
Список літератури

Дата видачі “____” _____ 2022 р. Керівник _____
(підпис)

Завдання отримав _____
(підпис)

Тема: <u>Розробка мобільного застосунку під операційну систему iOS</u>

Календарний план виконання роботи:

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу	01.10.2021	
2.	Аналіз матеріалів за темою	14.12.2021	
3.	Розробка моделі та мобільного додатку	17.02.2022	
4.	Написання текстової частини до курсової роботи	20.03.2022	
5.	Аналіз та редагування роботи	08.04.2022	
6.	Захист курсової роботи	14.06.2022	

Семенко Е.О. _____

Борозенний О. С. _____

“_____”_____ 2022 р.

Зміст

Анотація

Метою цієї роботи є створення мобільного додатку для людей, що хотіли б займатися сортуванням сміття, проте не мають достатньо інформації про те, як це робити. Розроблений додаток дає змогу визначити, до якого типу сміття належить конкретний об’єкт та надає коротку інформацію про його сортування та переробку. Класифікація відбувається після наведення камери мобільного телефону на предмет.

У роботі розглядаються способи створення та тренування моделей машинного навчання для класифікації зображень, а також їхнє подальше використання. Визначаються переваги тренування моделей безпосередньо на пристрої.

Описується процес створення мобільного додатку, обґрунтовується вибір операційної системи, а також інструментів, що були використані під час розробки.

Вступ

Щорічно в світі утворюється приблизно 2 мільярди тон твердих відходів. Як мінімум 33% з них не обробляються та не зберігаються екологічно безпечними та дружнім до навколишнього середовища способом.[1] Більшість людей, щодня викидаючи сміття, не задумуються про те, куди воно відправляється після домашнього смітника. Проте подібне навіть неумисне ігнорування проблеми, на жаль, не рятує від її наслідків. Відходи залишаються на звалищах, а також перетворюються на небезпечні речовини у навколишньому середовищі.

Тверді відходи – це те, що впливає майже на всі сфери життя суспільства та кожної окремої людини. Непереробленний пластик забруднює Світовий океан. У ґрунтові води та водозабірні озера надходять важкі метали та токсини, в опадах можуть міститись діоксини, а в повітрі сполуки свинцю. Поклади сміття можуть нагріватися до 100°C, а відповідно і спалахувати без додаткових зовнішніх чинників. Таким чином, створюється небезпека пожежі для близьких житлових районів, а також джерело отруйних речовин, що надходять в повітря з процесом горіння.

Хоча і спостерігається вдосконалення та інновації у сфері утилізації та переробки твердих побутових відходів в усьому світі, ця тема все ще залишається критично важливою і потребує змін як на глобальному, так і на локальному рівнях.

З огляду на поточний стан сортування та переробки сміття в світі, за мету даної роботи було поставлено створення мобільного застосунку, за допомогою якого кожна людина може отримати усю потрібну інформацію для правильного сортування сміття вдома. Такий додаток дозволяє визначати тип сміття за допомогою камери телефону, а також дізнатися, який сміттєвий бак найкраще підходить для кожного типу відходів.

Використання такого додатку зможе зберегти час під час розподілення побутових відходів, а також мінімізувати навантаження на співробітників

сортувальних станцій. Також подібний мобільний застосунок робить сортування сміття більш доступним і його розповсюдження може позитивно вплинути на кількість людей, що сортують побутові відходи, адже багато хто нині просто немає багато часу на вивчення усіх типів відходів та правил сортування сміття.

Робота складається з 4-ох розділів:

Перший розділ присвячений дослідженню предметної області.

Другий розділ присвячений опису операційної системи iOS, доступних інструментів та бібліотек.

Третій розділ присвячений проектуванню архітектури додатку.

Четвертий розділ присвячений безпосередньо розробці додатку.

1 Дослідження та аналіз предметної області

1.1 Проблема сортування та переробки сміття

Більше третини відходів у країнах з високим рівнем доходу утилізується шляхом переробки та компостування. Країни з високим та середнім рівнем доходу зазвичай забезпечують універсальний збір відходів. Країни з нижчим рівнем доходу, як правило, збирають близько 48 відсотків відходів у містах, але за межами міст охоплення збором відходів становить близько 26 відсотків.[1]

На міжнародному рівні найбільшою категорією відходів є харчові та зелені відходи, які становлять 44 відсотки світових відходів. Суха вторинна сировина (пластик, папір і картон, метал і скло) становлять ще 38 відсотків відходів.[1] Отже, ці категорії відходів є основними для сортування та переробки.

Сортування відходів полегшує розуміння того, як зменшити їхній загальний обсяг та допомагає визначити предмети, які можна повторно використати. Нездатність правильно розділити побутові відходи означає, що вони зрештою будуть змішані на звалищах так само, як вони були б змішані у смітнику.

1.2 Огляд потреб користувачів

Нині ринок мобільних додатків є перенасиченим, а користувачі стають все більш вимогливими до деталей. Для того, щоб бути цікавим користувачеві, мобільний додаток має бути зрозумілим, містити корисну інформацію, а також мати функції для постійної потреби у використанні.

Для людини, що хоче отримувати базову інформацію про сортування сміття та мати інструмент, що може з цим допомогти, важливим є привабливий та зрозумілий інтерфейс застосунку, а також простота виконання конкретної задачі – визначення типу побутових відходів.

1.3 Прототипування

Для задоволення основних потреб користувачів варто враховувати усі фактори, що впливають на вибір конкретного додатку. В першу чергу, має бути чітко сформульована його основна функція - визначення типу сміття для сортування. Також, під час реалізації додатку фокусуватись треба саме на розробці головної задачі, додаючи пізніше сторонній функціонал – неосновні функції або корисну інформацію.

На основі цього було прийняте рішення створити додаток, що складається з двох основних екранів:

1. Екран із доступом до камери мобільного телефону, що після наведення на певний предмет показує, до якого типу сміття він належить.
2. Екран з колекцією інформації про сміттєві відходи, способи їхнього сортування та переробки.

1.4 Класифікація зображень

Основною функцією даного мобільного додатку є визначення типу сміття. Для того, щоб додаток міг визначити це за зображенням, потрібно відповідне зображення класифікувати.

Класифікація зображень (інша назва – розпізнавання зображень) – це завдання категоризації та присвоювання міток зображенням згідно з певними правилами. Методи класифікації зображень в основному поділяються на дві категорії: методи класифікації зображень “під наглядом” (supervised) і “без нагляду” (unsupervised).

Техніка класифікації без нагляду означає, що алгоритми машинного навчання використовуються для аналізу та групування довільних наборів даних шляхом виявлення прихованих шаблонів без участі людини.

Метод класифікації зображень під наглядом використовує раніше класифіковані еталонні зразки, щоб навчити класифікатор і згодом класифікувати нові, невідомі дані. У даній роботі використано саме такий тип класифікації зображень.

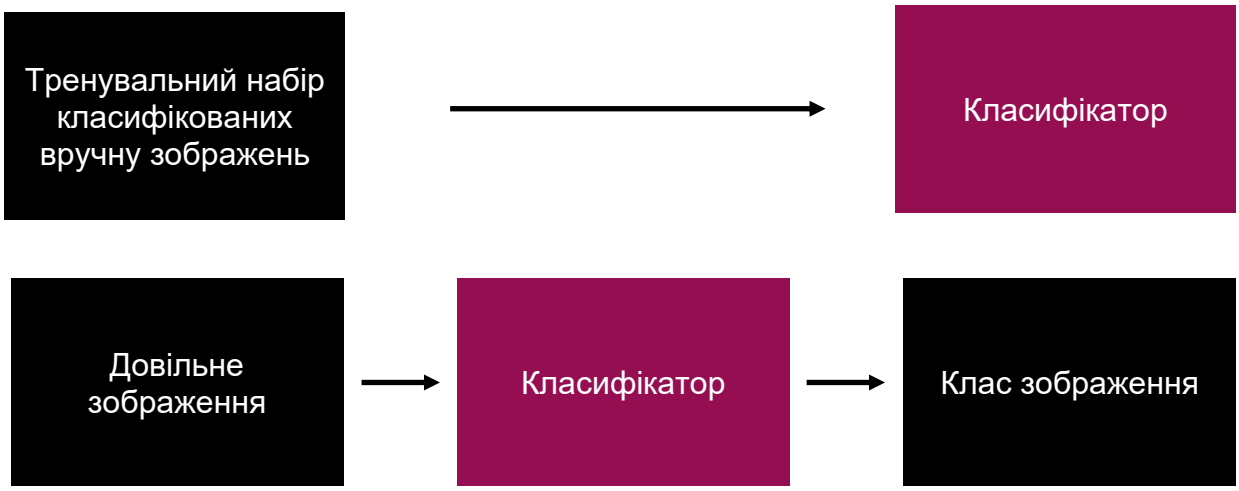


Рисунок 1.1 Діаграма процесу навчання під наглядом та класифікації

Основою роботи з класифікації зображень під наглядом є початковий тренувальний набір даних. Він буде слугувати предметом навчання для моделі (класифікатора). Дані для навчання моделі мають поділятися на кілька категорій:

1. Тренувальні дані (найбільший за обсягом набір) – вже поділені на класи зображення.
2. Дані для оцінювання – набір, що зазвичай створюється із даних для тренування.
3. Дані для тестування – набір довільних зображень для перевірки роботи класифікатора.

Під час навчання та класифікації безпосередньо комп'ютер аналізує зображення у вигляді пікселів, розглядаючи його як масив матриць з розміром, що залежить від роздільної здатності зображення. Алгоритм звертається до тренувальних даних і виявляє схожість між ними та новими вхідними даними.

2 Аналіз доступних інструментів

2.1 Операційна система iOS та її інструменти

2.1.1 Загальні відомості

iOS — це мобільна операційна система, яка працює на мобільних пристроях Apple, включаючи iPhone, iPad та iPod Touch. Це також основа для Apple tvOS, яка успадковує багато функцій від iOS. Перша версія iOS була випущена в червні 2007 року, коли iPhone дебютував на ринку. iOS (абревіатура від iPhone Operating System) є операційною системою на основі Unix, яка працює на всіх мобільних пристроях Apple. Дана операційна система була першою, яка підтримувала технологію Multitouch, що зробило її цікавою для кінцевого користувача одразу після появи на ринку.[2]

Apple iOS — друга за популярністю операційна система для мобільних пристроїв. Станом на квітень 2022 року Apple iOS займає 27,7% ринку мобільних телефонів, поступаючись за поширеністю лише Android, яка займає 71,6% ринку.[4]

Основними перевагами операційної системи iOS є:

1. Зручність у використанні, збереження консистентності дизайну між різними моделями та пристроями.
2. Тривала підтримка застарілих пристроїв, можливість оновлення їх до нових версій операційної системи.
3. Стабільність роботи та багато уваги до безпеки даних на пристроях.
4. Відсутність залежності від програмних компонентів різних виробників. Оскільки компанія Apple розробляє операційну систему виключно під власні пристрої, то вона має повний контроль над якістю роботи системи.
5. Операційна система підтримується лише на пристроях Apple, що спрощує розробку та тестування додатків для неї.

6. Окрім власне операційної системи та її внутрішніх компонентів, Apple надає оновлювані інструменти для розробки, рекомендації для створення користувацького інтерфейсу, навчальні матеріали для розробників тощо.

2.1.2 Мова програмування

Мобільні застосунки бувають двох типів: нативні та гібридні. На перший погляд, обидва типи мають схожі характеристики та дизайн, але основна технологія розробки відрізняється. Як випливає з назви, гібридні програми — це комбінація веб-програм і мобільних додатків. Основою таких програм є інструменти веб-розробки. Їх можна додавати в магазини додатків, встановлювати та використовувати так само, як і будь-які інші мобільні додатки. Основними перевагами гібридних додатків є портативність і простота розробки. Для роботи застосунку на різних платформах достатньо написати одну програму.

Хоча гібридні програми легше та дешевше розробляти, нативні мобільні додатки також мають багато переваг. На відміну від веб-сайтів і гібридних веб-програм, нативні мобільні додатки не запускаються за допомогою браузера або штучних надбудувань. Завдяки цьому вони мають доступ до вбудованих функцій смартфону (камери та мікрофону, наприклад), а також працюють швидше та стабільніше, оскільки не покладаються на сторонні інструменти. Ці переваги є ключовими для розроблення даного додатку, тож у роботі розглядається саме нативна розробка.

Для нативної розробки під операційну систему iOS існує дві основних мови: Objective-C та Swift.[5]

1. Objective-C

Objective-C був розроблений Томом Лавом і Бредом Коксом у 1984 році.[5] До того, як Apple запустила Swift у 2014 році, Objective-C був основною мовою мобільних додатків Apple iOS. Objective-C — це універсальна,

об'єктно-орієнтована мова програмування, яка є поєднанням мови Smalltalk та мови програмування C. Це надійна мова, хоча і застаріла (остання версія була випущена в 2016 році).

Рисунок 2.1 Переваги та недоліки мови Objective-C

2. Swift

На сьогодні Swift є основною мовою програмування для операційної системи iOS. Swift був розроблений та запущений компанією Apple у 2014 році. У грудні 2015 року Swift був випущений із відкритим кодом під ліцензією Apache 2.0. Окрім iOS, Swift також є мовою програмування macOS, watchOS, tvOS, Linux та z/OS.[5] Swift – це нова мова програмування, яка надає такі сучасні мовні функції, як динамічність, безпечність, пізнє зв’язування та розширюваність.

Рисунок 2.2 Переваги та недоліки мови Swift

З огляду на перелік переваг та недоліків двох розглянутих мов, було прийняте рішення використати мову Swift під час розробки додатку. На сьогодні, більшість розробників обирають саме Swift, оскільки ця мова продовжує розвиватися та є значно більш перспективною, ніж Objective-C.

2.2 Машинне навчання

2.2.1 Машинне навчання на пристрої

Машинне навчання – це окрема сфера розробки, яка дозволяє створювати нові функції на основі штучного інтелекту. Процес машинного навчання передбачає використання моделей, які були попередньо підготовані та натреновані. Традиційно моделі машинного навчання працювали лише на

потужних хмарних серверах, однак зараз з’являється все більше можливостей для машинного навчання на власних пристроях.

Машинне навчання на пристрої дає змогу працювати з моделями напряму, наприклад, у мобільному додатку чи веб-браузері. Модель машинного навчання оброблює вхідні дані одразу, а не надсилає ці дані на сервер для обробки там.[6] У такого підходу є багато переваг:

1. Низька затримка (немає очікування даних з сервера, можна працювати з об’єктами в реальному часі);
2. Приватність (обробка відбувається на пристрої користувача, отже є можливість роботи з конфіденційними даними);
3. Робота офлайн (відсутня потреба у мережі Інтернет);
4. Безкоштовність (використання можливостей пристрою замість підтримки додаткових серверів або хмарних обчислень).

Існує велика кількість інструментів для машинного навчання на пристрої, однак кожен з них має свої особливості та сферу використання. Найбільш відповідними інструментами машинного навчання для додатків під iOS є такі, що створені компанією Apple та є найбільш інтегрованими у середовище розробки та доступні мови програмування.

2.2.2 Інструменти машинного навчання на пристрої від Apple

Apple надає можливість користуватися перевагами машинного навчання за допомогою власних інструментів, створених для простої інтеграції в додатки, без необхідності у глибоких знаннях з машинного навчання.

Create ML – основний фреймворк для роботи із машинним навчанням від Apple. Create ML може працювати з різними типами даних: зображеннями, відео, активністю, звуками, текстом та таблицями. Він дозволяє створювати та тренувати моделі безпосередньо на пристрої, оцінювати їхню роботу, керувати наборами даних та інтегрувати створену модель у додаток за допомогою Core ML.[7]

Core ML – це фреймворк, що дає можливість інтегрувати моделі машинного навчання у існуючі додатки. Він оптимізує роботу на пристрої, використовуючи процесор, графічний процесор і нейронний механізм, мінімізуючи при цьому задіяний обсяг пам’яті та енергоспоживання. Core ML є основою для окремих специфічних фреймворків і функціональних можливостей; підтримує Vision для аналізу зображень, Natural Language для обробки тексту, Speech для перетворення аудіо в текст і Sound Analysis для визначення звуків у аудіозаписах.[3]

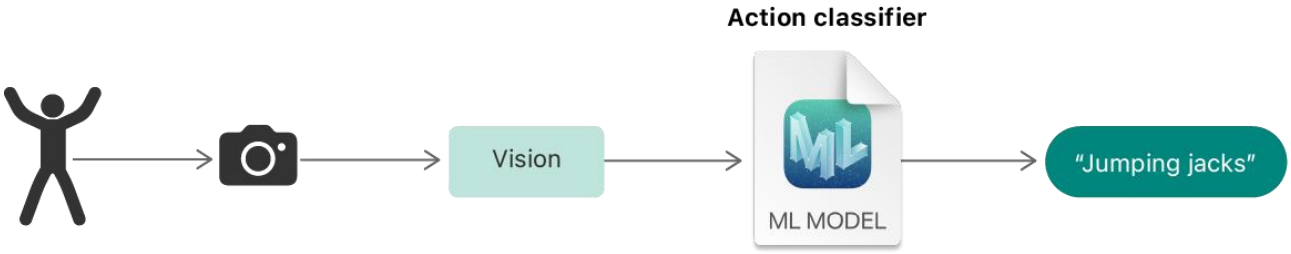


Рисунок 2.3 Робота моделі машинного навчання, створеної за допомогою Create ML

Отже, з урахуванням того, що доступні інструменти від компанії Apple є найбільш оптимізованими та зручними для розробки та інтеграції у додатки під операційну систему iOS, було вирішено використати саме їх. Це дає можливість отримати усі переваги, що надаються даними фреймворками, а також звільняє від потреби у вивченні та інтеграції сторонніх інструментів машинного навчання.

3 Проектування додатку

3.1 Проектування архітектури

Основою створення мобільного застосунку, так само, як і кожної програми, є планування та проектування. Першочерговим етапом є розробка основної ідеї додатку. При цьому варто виокремити основні умови для успішної роботи застосунку, а також виділити ті функції, що є головними для реалізації. Однак не бажано починати розробку без проектування архітектури. Погані архітектурні рішення можуть призвести до ускладненої обробки помилок, неможливості підтримки роботи додатку та додавання нового функціоналу в майбутньому. У той час як хороша архітектура має такі переваги:

1. Слугує основою (шаблоном) для розробки застосунку;
2. Створює можливості для масштабування;
3. Підвищує продуктивність платформи;
4. Дозволяє уникнути дублювання програмного коду;
5. Знижує витрати на розробку.

Отже, архітектура мобільного застосунку має бути детально продуманою та розробленою на початку роботи над проектом. Існує багато способів організації програмних компонентів в архітектурі застосунків. Такі попередньо визначені способи, що допомагають у плануванні, називаються архітектурними шаблонами або паттернами програмування. У кожному шаблоні компоненти організовані по-різному для вирішення певної проблеми або архітектурної задачі.

3.2 Опис обраного шаблону архітектури

Останнім часом у програмуванні додатків під операційну систему iOS набирає популярність такий шаблон як MVVM, що розшифровується, як Model-View-ViewModel. MVVM є певною варіацією шаблону моделі-презентатора Мартіна Фаулера. Цей шаблон складається з чотирьох основних компонентів:

моделі даних, елементів візуального представлення, моделі представлення та контролера:

- Модель даних містить основні структури даних, операції з ними, логіку обробки даних.
- Елементи візуального представлення містять кнопки, зображення, комірки таблиць тощо.
- Модель представлення отримує інформацію про дані, оброблює їх та передає до контролера.
- Контролер перегляду отримує оброблену інформацію від моделі перегляду і передає у відповідні частини користувацького інтерфейсу.

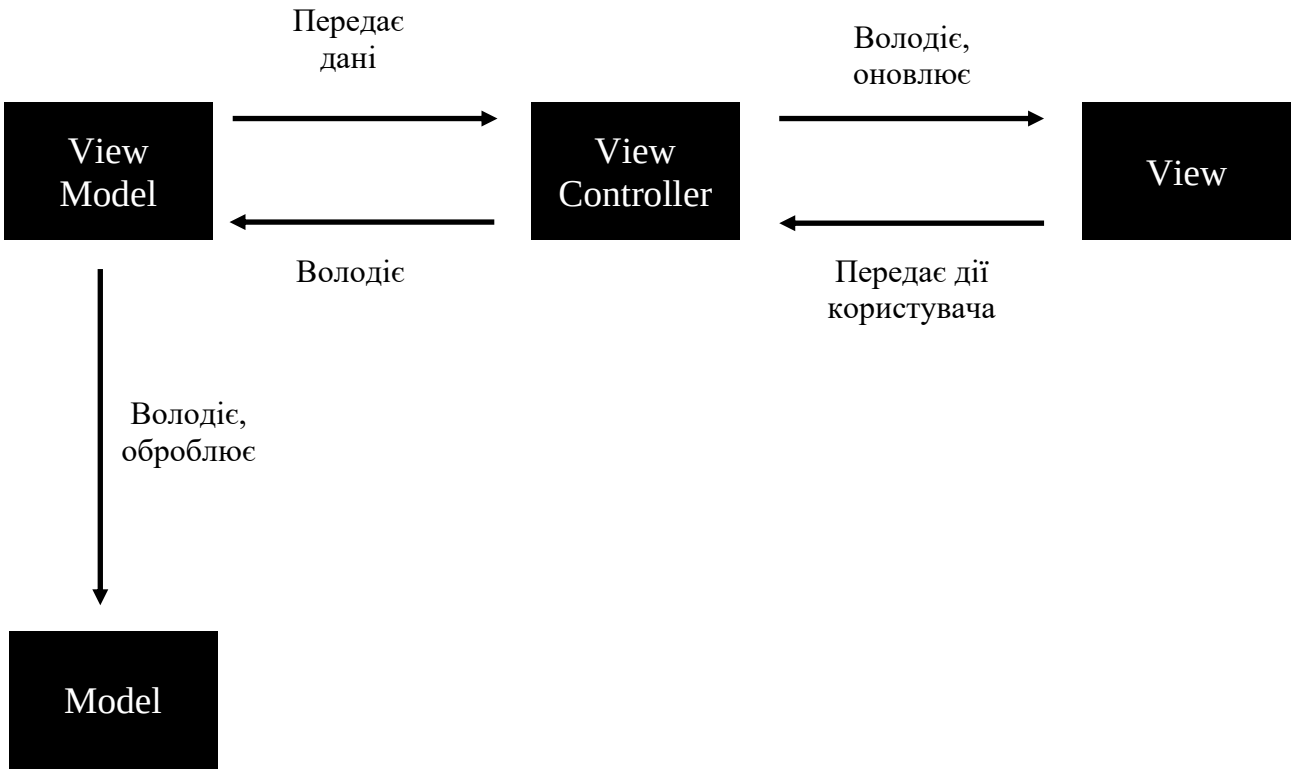


Рисунок 3.1 Схема взаємодії компонентів шаблону MVVM

Окрім прозорого розподілу обов’язків між компонентами, MVVM має також інші переваги. Переміщуючи обробку даних до ViewModel, MVVM значно полегшує тестування та розділює відповідальність окремих елементів.[8]

Swift UIKit не підтримує прямого прив’язування даних, але цього можна досягти за допомогою певних розподілів архітектурних компонентів та правильної обробки змінних.

На жаль, Apple не використовує шаблон MVVM напряму, але він стає все більш популярним та легким для розуміння, тож з урахуванням цього було надано перевагу саме MVVM шаблону в процесі розробки додатку.

Таким чином усі окремі логічні частини додатку були розподілені пошарово та виокремлені в окремі файли і папки:

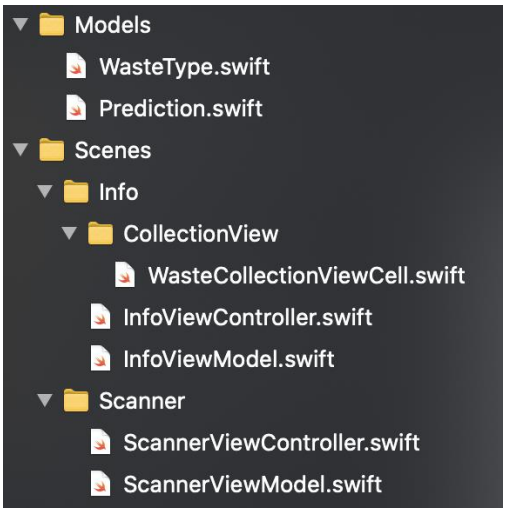


Рисунок 3.2 Розподіл основних класів програми згідно з шаблоном MVVM

- Основною моделлю даних є WasteType, що декодується з JSON файлу та містить інформацію про конкретні типи сміття та їхнє сортування. Також використовується Prediction як модель передбачення класифікатора.
- Для зображення користувацького інтерфейсу створені окремі підкласи UIView або UITableviewCell.
- Контролерами для кожного екрану виступають підкласи UIViewController.
- Класи ViewModel також створені для кожного екрану і виконують роботу, пов’язану з даними, їхньою обробкою та передачею до контролера.

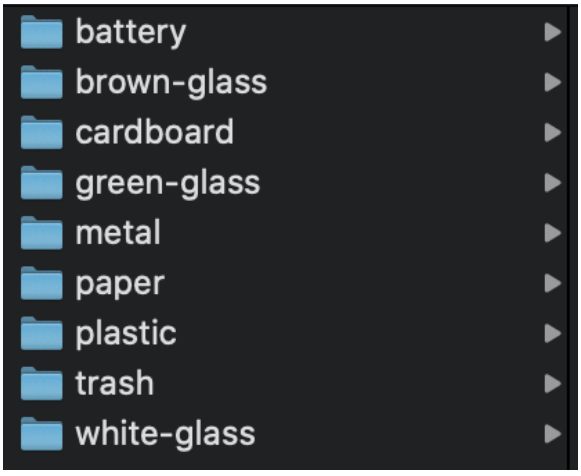
4 Розробка додатку

4.1 Розробка та тренування моделі

Класифікатор зображень - це модель машинного навчання, яка розпізнає зображення різних типів. Після того, як класифікатор отримує певне зображення, він повертає назву категорії для цього зображення.

Для створення такої моделі машинного навчання за допомогою Core ML необхідно мати оброблений набір початкових даних. Ці дані використовуються для тренування моделі. Для кращої роботи моделі варто мати велику кількість тренувальних зображень різної якості, зроблених під різним кутом та з різним освітленням. Кількість зображень також має бути збалансованою, варто застосовувати приблизно однакову кількість для кожної категорії. Зображення можуть бути збережені в будь-якому форматі, наприклад, JPEG або PNG. Вони не повинні мати конкретний розмір та не повинні бути виключно однакового розміру між собою. Підготований набір тренувальних даних має бути відсортованим та розподіленим на папки, названі за категоріями зображень, що містяться в них.[9]

Для розробки потрібної моделі було створено 9 папок з короткими описовими назвами. Кожна папка містить зображення певної категорії сміття, що може бути ідентифіковане в додатку.



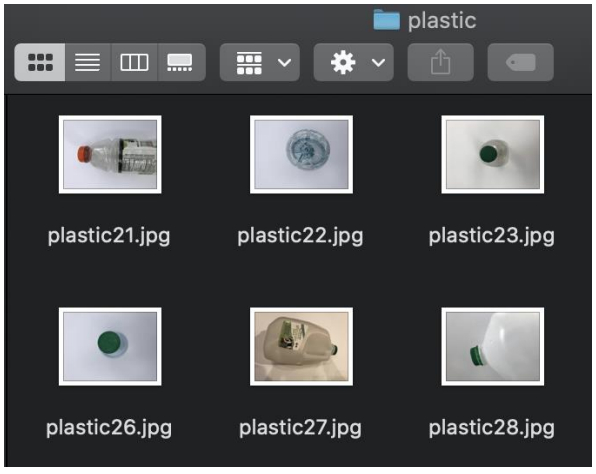


Рисунок 4.1, Рисунок 4.2 Тренувальні дані моделі

В роботі було використано два датасети зображень сміття з відкритих джерел, які були поєднані, проаналізовані та розподілені на потрібні категорії. Загальна кількість використаних зображень для тренування складає 6203, кожна з категорій містить близько 600-800 зображень.

Перед тренуванням моделі можна також додати певні параметри для покращення якості створеної моделі. Регульованою є максимальна кількість тренувальних ітерацій, а також зміни наданих зображень, такі як: додавання шумів, замилювання зображення, повороти та віддзеркалювання. Кожне таке розширення копіює зображення з набору даних і застосовує перетворення або фільтр, що ефективно надає набору даних більше різноманітності, не вимагаючи додаткових зображень. Під час тренування моделі для класифікації сміття було пройдено 30 ітерацій з накладанням шуму на зображення та його поворотами.

4.2 Тестування моделі

Для отримання правильної точності роботи моделі, перш ніж інтегрувати її в мобільний додаток, варто провести тестування та оцінку моделі. Тестування моделі за допомогою тестового набору даних – це найшвидший спосіб побачити, наскільки добре модель може працювати в реальних умовах. Отже, потрібно створити тестовий датасет, що організований так само, як і тренувальний: класифіковані та розподілені на потрібні папки зображення. Для кожної категорії кількість тестових зображень має складати 20-40% від кількості тренувальних зображень.

Create ML дає можливість без зайвих зусиль протестувати власну модель. Для цього потрібно просто завантажити тестовий датасет у відповідну частину програми та запустити тестування.

Для тестування моделі було використано 2650 зображень різних категорій. Завдяки цій перевірці було визначено точність роботи моделі на реальних даних. Також перевірити роботу моделі можна завантажуючи зображення безпосередньо в проект Create ML: у вкладці preview можна додати

будь-яке зображення чи зробити його напряму з телефону та отримати результат класифікації. Після тренування та тестування була отримана робоча модель класифікації зображень сміття з такими результатами роботи: тренування – 87%, валідація – 82%, тестування – 74%.

4.3 Інтеграція моделі в мобільний додаток

У разі задоволеності якістю роботи отриманої моделі її можна інтегрувати в мобільний додаток. Для цього потрібно завантажити її до файлової системи у форматі Core ML та додати в існуючий проект Xcode.

Для того, щоб модель працювала та її можна було вільно використовувати, потрібно також провести налаштування в коді програми. Варто створити екземпляр класу обгортки моделі, який Xcode автоматично генерує під час компіляції, отримати базову змінну MLModel екземпляра класу обгортки та передати цю модель до ініціалізатору VNCoreMLModel. За допомогою цієї моделі потрібно створити об’єкт запиту на класифікацію зображення та обробник його результату. Коли запит завершує свою роботу, фреймворк Vision сповіщає про це викликом обробника результату, в який можна додати будь-які бажані функції.

4.4 Зчитування зображення

Для того, щоб користувач міг отримати результат класифікації зображення з камери його телефону, це зображення потрібно зчитати та передати до класифікатора. Існує кілька способів це зробити:

- Зробити фото за допомогою камери;
- Обрати попередньо зроблене фото з галереї;
- Зчитувати дані з камери напряму в додатку.

Кожен з цих способів є досить зручним для користувача, але перші два передбачають створення фотознімків та/або зберігання їх до галереї. Відповідно, користувачеві потрібно витратити час, щоб зробити фотографії

кожної одиниці сміття, яку він хотів би класифікувати та сортувати, а також зберегти ці фотографії у галереї мобільного телефону. Таким чином, використання подібного додатку перетворюється на довгий та складний процес, що потребує багато додаткових дій з боку користувача, а також використовує зайву пам’ять мобільного пристрою. Для того, щоб класифікація побутових відходів була найбільш зручною та ефективною, найкращим способом передачі зображення до класифікатора є безпосередньо зчитування цього зображення з камери телефону.

Для того, щоб отримувати зображення з камери телефону, можна використати вбудовану бібліотеку від Apple – AVKit (AVFoundation). Підсистема AVFoundation Capture забезпечує загальну архітектуру високого рівня для захоплення відео, фотографій та аудіо в iOS. За допомогою неї можна інтегрувати зйомку фотографій або відео в додаток, редагувати метадані фотографій або параметри зйомки, а також отримати прямий доступ до потокової передачі піксельних або аудіо даних безпосередньо з пристрою.

Основою Capture є сесія, вхідні дані з пристрою та вихідні дані. Сесія має бути створена, пристрій має бути доданий в ініціалізатор, як джерело вхідних даних. Для того, щоб камера почала фіксувати вхідні дані, потрібно запустити сесію. У разі потреби її можна призупиняти та запускати знову. Цей підхід використаний в додатку для зручності роботи із зчитувачем. Після зчитування вхідні дані передаються до функції класифікатора, що оброблює їх та повертає текстовий результат класифікації, а також відсоток «впевненості», що зображуються на екрані користувача в реальному часі.

4.5 Створення інформаційного розділу

Окрім конкретного функціонального навантаження, мобільний додаток має бути також цікавим для користувача візуально та інформативно. Це досягається продуманим користувацьким інтерфейсом, а також розширенням можливостей мобільного застосунку. Одним з можливих розширень є інформаційне наповнення – розділ з короткими фактами про різні типи

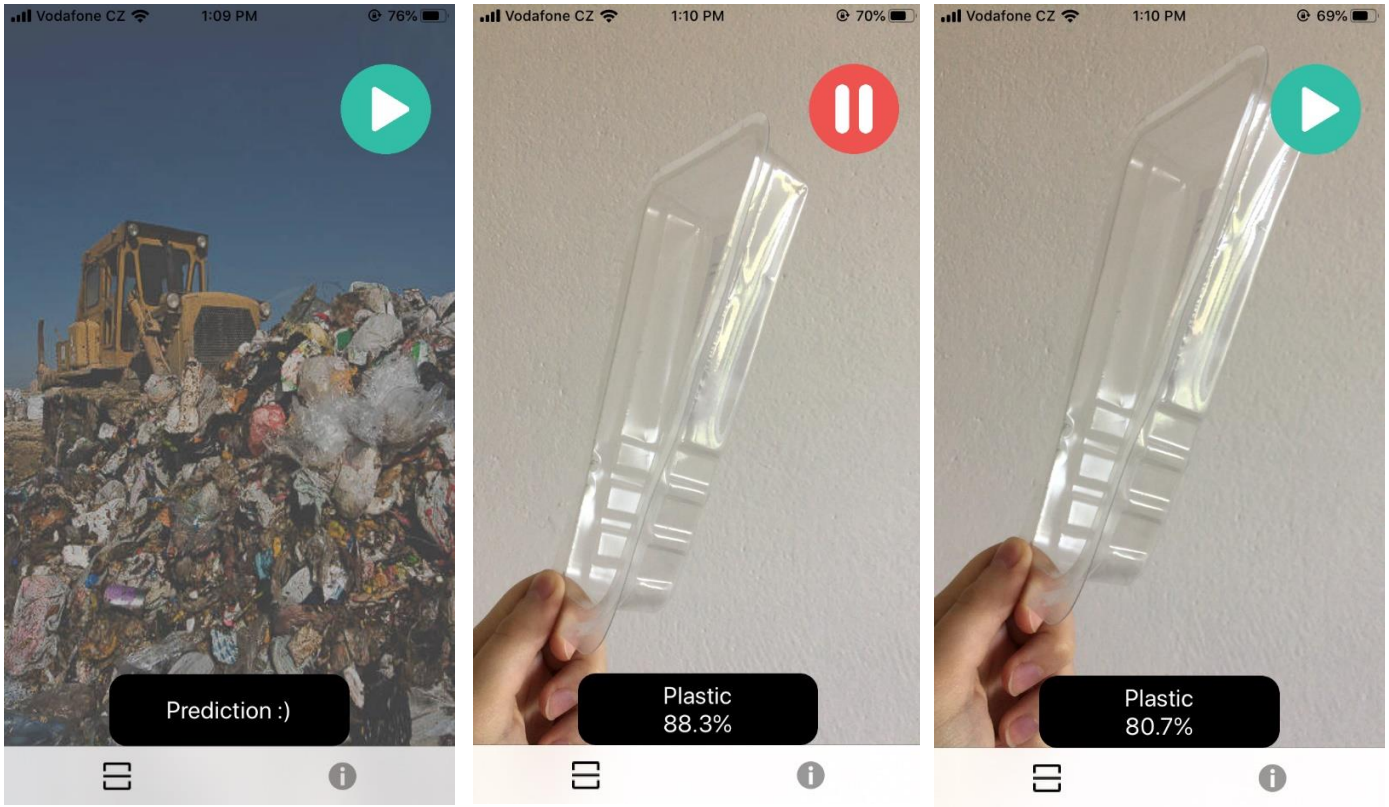
побутових відходів, а також їхнє сортування. Замість того, щоб перевантажувати користувача та давати детальну інформацію про кожен тип, що він класифікує за допомогою камери, було вирішено винести це в окрему частину додатку, яку користувач зможе відкрити та проглянути в будь-який зручний для нього час.

Найзручнішим програмним компонентом для відображення великої кількості категорій є колекція – UICollectionView. Дані в колекції мають бути організовані в окремі елементи. Елемент – це найменша одиниця даних, яку потрібно представити. Для заповнення комірки колекції було взято зображення та назву типу сміття. Після натискання на комірку користувач бачить окремий екран із зображенням потрібного сортувального баку, а також інформацію про даний тип сміття та вимоги до його сортування.

4.6 Результат розробки

В результаті розробки було створено мобільний застосунок, що містить два розділи: зчитування та класифікацію зображення, а також інформацію про типи сміття та їх сортування. Після створення та тренування моделі було досягнуто точності класифікації зображень 74% на тестових даних.

Після відкриття додатку, користувач потрапляє на екран зчитування. Він бачить тематичне фонове зображення та може запустити класифікацію за допомогою кнопки в правому верхньому куті екрану. Під час роботи зчитувача кожен об’єкт, на який наведено камеру мобільного пристрою, класифікується, а результат класифікації та його відсоткова точність зображуються на екрані користувача. Зчитування зображень можна призупинити за допомогою відповідної кнопки, а потім запустити знову.



Початковий стан зчитування | Зчитування | Зупинка процесу зчитування

Рисунок 4.3 Демонстрація станів зчитування та класифікації

Для перемикання між різними розділами додатку використовуються дві кнопки внизу екрану. У вікні з інформацією знаходяться комірки із зображеннями та назвами типів сміття. При натисканні на кожну з них, користувач бачить екран з детальною інформацією про сортування конкретного типу.

Інформаційний розділ

Детальна інформація

Рисунок 4.4 Демонстрація інформаційного розділу додатку

Висновки

У даній роботі була розглянута технологія машинного навчання безпосередньо на мобільному пристрої. Під час виконання роботи була створена модель машинного навчання, досягнуто точності класифікації зображень у 74%, а також досліджено способи інтеграції даної моделі у мобільний додаток. Було розглянуто різні способи отримання вхідних зображень від користувача, їхні переваги та недоліки.

Було проаналізовано сучасні методи та підходи до проектування та розробки архітектури мобільних додатків.

Результатом виконаної роботи є мобільний додаток для платформи iOS, що дозволяє класифікувати зображення певних типів побутових відходів. Програма була розроблена мовою програмування Swift із використанням вбудованих бібліотек та інструментів для роботи з машинним навчанням від компанії Apple.

Список використаних джерел

1. Trends in Solid Waste Management[Електронний ресурс] – Режим доступу до ресурсу: https://datatopics.worldbank.org/what-a-waste/trends_in_solid_waste_management.html
2. Advantages and disadvantages of iOS operating system[Електронний ресурс] – Режим доступу до ресурсу: https://www.itrelease.com/2020/09/advantages-and-disadvantages-of-ios-operating-system/
3. Core ML Overview[Електронний ресурс] – Режим доступу до ресурсу: https://developer.apple.com/documentation/coreml/
4. Mobile Operating System Market Share Worldwide[Електронний ресурс] – Режим доступу до ресурсу: https://gs.statcounter.com/os-market-share/mobile/worldwide
5. Picking The Best Language For iOS App Development In 2022[Електронний ресурс] – Режим доступу до ресурсу: https://www.ideamotive.co/blog/picking-the-best-language-for-ios-app-development
6. Why On-Device Machine Learning?[Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/learn/topics/on-device-ml/learn-more
7. Machine Learning[Електронний ресурс] – Режим доступу до ресурсу: https://developer.apple.com/machine-learning/create-ml/
8. MVVM in iOS – A Quick Walkthrough[Електронний ресурс] – Режим доступу до ресурсу: https://www.clariontech.com/blog/mvvm-in-ios-a-quick-walkthrough
9. Creating an Image Classifier Model[Електронний ресурс] – Режим доступу до ресурсу: https://developer.apple.com/documentation/createml/creating_an_image_classifier_model