Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Розпізнавання порід котів у програмній системі обліку домашніх тварин
(тема)

Виконав:
студент 2 курсу, групи СШІм-21-2
Соловей І.В.
(прізвище, ініціали)

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

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник доц. Золотухін О.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Солов'ю Іллі Владиславовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розпізнавання порід котів у програмній системі обліку домашніх тварин

затверджена наказом університету від 31 березня 20 23 р. № 306Ст

2. Термін подання студентом роботи до екзаменаційної комісії 18 травня 20 23 р.

3. Вихідні дані до роботи macOS Ventura, середа розробки PyCharm та xCode, TensorFlow та Keras документація, дані інтернет-джерел

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Постановка задачі
3) Розробка фунціоналу для сканування порід котів
4) Оптимізація моделі розпізнавання

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Ознайомлення із завданням. Уточнення ТЗ.	03.04.2023	виконано
2	Підбір матеріалів за темою роботи.	04.04.2023	виконано
3	Виконання розділу 1.	05.04 – 09.04.2023	виконано
4	Виконання розділу 2.	10.04 – 16.04.2023	виконано
5	Виконання розділу 3.	17.04 – 23.04.2023	виконано
6	Виконання розділу 4.	24.04 – 30.04.2023	виконано
7	Оформлення пояснювальної записки.	01.05 – 07.05.2023	виконано
8	Оформлення презентації.	08.05 – 12.05.2023	виконано
9	Захист перед ЕК.	18.05.2023	

Дата видачі завдання 3 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Золотухін О.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 74 с., 26 рис., 3 табл., 3 дод., 20 джерел.

АНАЛІЗ, МОБІЛЬНИЙ ДОДАТОК, ПРОГРАМНА СИСТЕМА, ШТУЧНИЙ ІНТЕЛЕКТ, DJANGO, KERAS, SWIFT, TENSORFLOW.

Об’єкт дослідження – розпізнавання зображень за допомогою моделі штучного інтелекту з її інтеграцією у програмну систему домашніх тварин, що складається з серверної частини з REST API та мобільного додатка з можливістю обробки фотографій.

Предмет дослідження – можливості використання штучного інтелекту у розпізнаванні порід котів.

Мета роботи – розробка сервісу з можливістю визначення породи кішки за фотографією.

Методи дослідження – аналіз тематичних матеріалів, реалізація алгоритмів, вивчення документації засобів програмного забезпечення та бібліотек, аналіз ринку програмних систем зі схожим функціоналом, прототипування і практична реалізація.

В ході роботи проведено аналіз наявних технологій розпізнавання зображень, вивчені способи використання штучного інтелекту у взаємодії з мобільним додатком та розроблені вимоги до проєкту.

На базі створених вимог розроблена архітектура програмної системи та створено функціональний зразок.

ABSTRACT

Explanatory note: 74 p., 26 fig., 3 tab., 3 ann., 20 sources.

ANALYSIS, ARTIFICIAL INTELLIGENCE, DJANGO, KERAS, MOBILE APP, SOFTWARE SYSTEM, SWIFT, TENSORFLOW.

The object of research is image recognition using an artificial intelligence model with its integration into a pet software system, consisting of a server-side part with REST API and a mobile application with the ability to process photos.

The subject of research is the possibilities of using artificial intelligence in recognizing cat breeds.

The purpose of the work is to develop a service with the ability to determine the breed of a cat from a photo.

Research methods – analysis of thematic materials, implementation of algorithms, study of documentation of software tools and libraries, analysis of the market of software systems with similar functionality, prototyping, and practical implementation.

In the course of the work, an analysis of existing image recognition technologies was carried out, ways of using artificial intelligence in interaction with a mobile application were studied, and requirements for the project were developed.

Based on the created requirements, the architecture of the software system was developed, and a functional prototype was created.

ЗМІСТ

ВСТУП

XXI століття є часом стрімкого розвитку цифрових технологій. Зберігання паперової документації – пережиток минулого, тому існує потреба у переході на цифрову документацію. Це зручно і екологічно.

У програмній системі, що підтримує створення електронного паспорта тварини було б доцільно використати технології штучного інтелекту.

Штучний інтелект (ШІ) – це галузь комп'ютерної науки, яка зосереджена на розробці комп'ютерних систем, що можуть виконувати завдання, які зазвичай потребують людської інтелектуальної діяльності, такі як розуміння природної мови, розпізнавання зображень, розв'язання складних проблем, прийняття рішень та багато іншого. ШІ використовує методи машинного навчання, нейронні мережі та інші технології для створення імітації людського мислення та діяльності, щоб досягати заданих цілей. Це може бути використано в різних сферах, таких як медицина, транспорт, фінанси, наука та інші, для покращення ефективності та точності роботи та підвищення якості життя людей.

Програмна система для ведення цифрової картки домашньої тварини може бути доповнена модулем розпізнавання порід котів. У подальшому цей модуль можна використати як всередині системи, так і зовні у вигляді окремого сервісу.

Існує безліч різних порід тварин, в особливості кошачих. Штучний Інтелект може допомогти у скануванні тварини та визначенні її породи. Як мінімум, це допоможе заощадити час на пошук конкретної породи у списку із безлічі інших. Сучасне життя вчить нас розумному споживанню, і сфера тварин – не виключення. Все більше людей віддають перевагу тваринам з притулку замість покупки у заводчика. Звісно, визначити породу у такому випадку становиться важче. Тут на допомогу приходить комп’ютерний зір ШІ, який може визначити риси різних порід у однієї тварини.

Власник зможе побачити співвідношення порід у відсотках без проходження дорогого ДНК-тесту та не завдаючи зайвого стресу своєму улюбленцю. Подібні сервіси вже існують для визначення національності людей, використовуються сервісами з аналізу ДНК та побудови сімейного древа, тому було б доцільно реалізувати аналог, але у світі тварин.

Усі сучасні телефони мають камеру, тому доцільно використати сервіс у мобільному додатку програмної системи ведення домашніх тварин. Щоб забезпечити «незалежність» сервісу та забезпечити йому можливість монетизації Штучний Інтелект буде опацьовувати фото на боці серверу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Актуальність сервісу з розпізнавання порід котів

Поява технологій проникла в різні аспекти нашого життя, і сфера догляду за домашніми тваринами та їх ідентифікації не стала винятком. Системи розпізнавання порід котів є корисним інструментом для власників домашніх тварин, ветеринарів і дослідників. Такі системи можуть використовувати розширені алгоритми для точного визначення породи кота, пропонуючи численні переваги перед традиційними методами ідентифікації. Розглянемо актуальність систем розпізнавання порід котів, їх потенційні переваги і застосування.

За оцінками, у світі налічується понад 600 мільйонів домашніх котів, причому приблизно 30-37% американських домогосподарств мають принаймні одну кішку [7]. У Сполученому Королівстві близько 24% [17] домогосподарств мають котячих компаньйонів. Оскільки кількість власників домашніх тварин продовжує зростати, зростає потреба в більш ефективних і менш інвазивних методах ідентифікації та розуміння котів.

Переваги систем розпізнавання порід котів:

– альтернатива ДНК-тесту. ДНК-тест – популярний метод визначення породи кота. Однак це може бути дорогим і вимагає взяття крові або слини, що може бути стресом для тварини. Системи розпізнавання порід котів пропонують менш інвазивну та більш доступну альтернативу, покладаючись на аналіз фізичних характеристик за допомогою фотографій для визначення породи;

– пошук загублених домашніх тварин. Система розпізнавання порід котів може допомогти знайти та возз’єднати загублених домашніх тварин із їхніми власниками. Коли загубленого кота знаходять, його породу можна швидко ідентифікувати, що полегшує пошук кота та його власника через різні бази даних про втрачених домашніх тварин. Це може прискорити

процес повернення кота в його законний дім, мінімізуючи стрес як для кота, так і для його власника;

– допомога в усиновленні домашніх тварин. Системи розпізнавання порід котів можуть бути цінним інструментом для притулків для тварин і центрів усиновлення домашніх тварин. Точна ідентифікація породи може допомогти потенційним власникам вибрати кота, який відповідає їх стилю життя та вподобанням, підвищуючи ймовірність успішного усиновлення. Крім того, це дозволяє притулкам надавати важливу інформацію про специфічні потреби котів, їх здоров'я та темперамент;

– ветеринарна допомога. Ветеринари можуть використовувати системи розпізнавання порід котів, щоб отримати уявлення про генетичне походження кота, інформуючи про свій підхід до діагностики та лікування. Певні породи котів схильні до певних проблем зі здоров'ям, тому знання породи котів може допомогти ветеринару відповідним чином підібрати медичну допомогу та профілактичні заходи;

– дослідження та збереження. Системи розпізнавання порід котів можуть допомогти дослідникам у вивченні генетичних варіацій, проблем зі здоров'ям, характерних для породи, а також у збереженні зникаючих або рідкісних порід котів. Розуміючи ці варіації, вчені можуть розробити цілеспрямовані заходи щодо охорони здоров'я та програми розведення для збереження та захисту вразливих порід;

– розведення котів. Заводчики можуть використовувати системи розпізнавання порід, щоб перевірити родовід котів, забезпечуючи чистоту племінних ліній і зберігаючи цілісність певних порід. Це може бути особливо корисним для заводчиків, які мають справу з рідкісними або цінними породами котів;

– страхування домашніх тварин. Системи розпізнавання порід котів можуть надати цінну інформацію компаніям зі страхування домашніх тварин, дозволяючи їм налаштовувати політику та премії на основі ризиків для здоров'я та очікуваної тривалості життя породи котів.

Отже, система розпізнавання порід котів має великі перспективи для широкого спектру застосувань, від допомоги власникам домашніх тварин у розумінні їхніх котячих товаришів до взаємодії з ветеринарами і дослідниками у покращенні здоров’я та добробуту домашніх улюбленців. Пропонуючи неінвазивну, доступну та точну альтернативу традиційним методам ідентифікації, ці системи можуть революціонізувати спосіб взаємодії з нашими домашніми тваринами та догляду за ними. Оскільки технологія продовжує розвиватися, потенційні переваги та застосування систем розпізнавання порід котів, безсумнівно, розширюватимуться, що ще більше покращить життя як котів, так і їхніх власників.

1.2 Аналіз конкурентів

Останніми роками розробка систем розпізнавання порід котів привернула значну увагу, на ринок з’явилися різноманітні додатки. Як розробнику системи розпізнавання котів, життєво важливо розуміти конкурентне середовище та вирізняти свою пропозицію, підкреслюючи її унікальні особливості та переваги.

Аналіз конкурентів має вирішальне значення для розробки програмної системи, оскільки він допомагає вам зрозуміти ринковий ландшафт і існуючі рішення. Таке розуміння дозволяє визначити прогалини та можливості на ринку, що допоможе вам створити більш цілеспрямований і ефективний продукт. Крім того, аналіз конкурентів дає змогу вчитися на сильних і слабких сторонах конкурентів, даючи можливість перейняти передовий досвід і уникнути поширених помилок. Отримавши уявлення про вподобання та очікування клієнтів, ми зможемо адаптувати сервіс для кращого задоволення їхніх потреб, що зрештою призведе до більш конкурентоспроможного та успішного продукту.

Дослідимо три конкурентні системи та виділимо причини, чому розробляєма система розпізнавання котів виділяється як найкраще рішення:

– Cat Scanner. Мобільний додаток, який використовує штучний інтелект для ідентифікації порід котів за зображеннями. Користувачі можуть сфотографувати свого кота або завантажити наявне зображення, та програма проаналізує фотографію, щоб визначити породу кота. Хоча Cat Scanner пропонує зручний інтерфейс і доступний для пристроїв iOS і Android, деякі користувачі повідомили про неточності в ідентифікації породи (рисунок 1.3);

– Azure Cat Classification. Веб-служба розпізнавання порід котів, яка дозволяє користувачам завантажувати зображення своїх котів для ідентифікації породи. Платформа використовує алгоритми машинного навчання для аналізу зображень і визначення породи кота. Однак вона обмежена лише веб-інтерфейсом, який може бути не таким зручним для користувачів, як мобільний додаток;

– Cat breed identifier. Мобільний додаток, доступний ексклюзивно для пристроїв Android, що обмежує доступ до нього лише користувачам iOS. Додаток використовує штучний інтелект для ідентифікації порід котів та має простий інтерфейс (рисунок 1.1). Врахуючи наявність лише одної функції (сканування) та ексклюзивність для платформи Android, це може бути серйозним недоліком для значної частини потенційних користувачів.

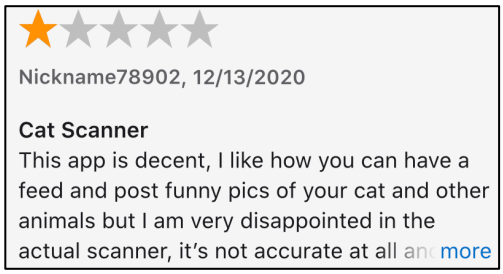


Рисунок 1.1 – Відгук до мобільного додатку «Cat Scanner»

Інтерфейс додатку Cat breed indentifier (рисунок 1.2) дуже простий, він має лише одну функцію – сканування. Також ми можемо побачити, що точність сканування досі невисока, враховуючи те, що розробник

використовує типові фотографії з тестового набору. З плюсів можна зазначити виведення трьох найбільш вірогідних порід у результаті ідентифікації обраної користувачем фотографії. За умови якісно натренованої моделі виведення топ-3 порід допоможе власникам метисів у визначенні можливих предків.

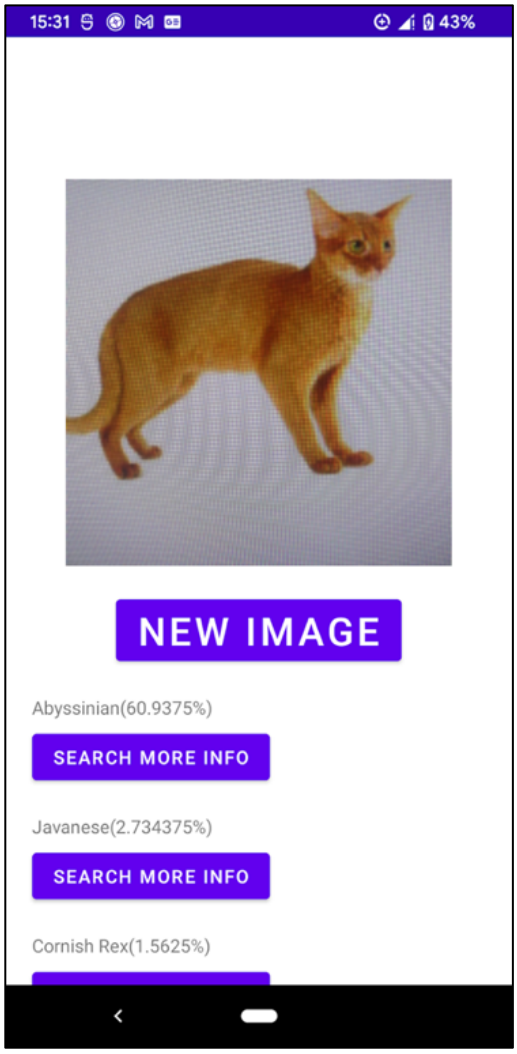


Рисунок 1.2 – Android-додаток «Cat breed indentifier»

Слід також зазначити і сильні сторони конкурентних систем, бо це допоможе у розробці нашого сервісу. Він має можливість як сканування, так і збереження кішки (хоч і у вигляді історії – History). Також додаток містить інформацію за всіма породами, що підтримуються.

Мобільний додаток «Cat Scanner» також має підтримку планшетної орієнтації. Знімок екрану інтерфейсу мобільного застосунку з планшету iPad наведено на рисунку 1.3.

Рисунок 1.3 – Інтерфейс мобільного додатку «Cat Scanner»

Сервіс, який ми розробляємо, має багато переваг у порівнянні з його конкурентами:

– додаток для ведення паспорту домашньої тварини. Система розпізнавання порід котів унікально розроблена для використання у окремому додатку для паспорта тварин, що спрощує процес ідентифікації породи та ведення записів. На відміну від інших програм, таких як Cat Scanner і Azure Cat Classification, розробляєма система призначена не тільки для розпізнавання порід котів, а і для керування записами із своїми улюбленцями, забезпечуючи кращий користувацький досвід;

– автоматичний вибір породи. Вбудована у систему функція розпізнавання фотографій дозволяє автоматично вибирати породу кота, що робить її зручнішою та ефективнішою порівняно з іншими програмами, які можуть вимагати введення вручну. Ця функція економить час і зусилля користувачів, одночасно забезпечуючи точне визначення породи кота;

– загальнодоступний API для сторонніх організацій. Система розпізнавання котів пропонує загальнодоступний API для інтеграції зі сторонніми організаціями, такими як ветеринарні клініки, притулки та компанії зі страхування домашніх тварин. Ця функція відрізняє нашу систему від таких конкурентів, як Cat breed identifier і Perfect Cat, які не надають доступу до API для зовнішніх сторін. Пропонуючи загальнодоступний API, система сприяє безперебійній взаємодії між різними зацікавленими сторонами в галузі догляду за домашніми тваринами, покращуючи загальний досвід;

– підвищена точність і надійність. Система використовує сучасні алгоритми машинного навчання, спеціально оптимізовані для ідентифікації породи котів. Така зосередженість на породах котів дає перевагу системі перед такими програмами, як Cat Scanner, яка отримала відгуки користувачів щодо неточностей у ідентифікації породи. Завдяки нашій системі користувачі можуть очікувати надійних і точних результатів визначення породи котів;

– вичерпна інформація про породу. Система розпізнавання котів надає користувачам докладну інформацію про кожну породу котів. Ця функція пропонує додаткову цінність порівняно з додатками на зразок Perfect Cat, які можуть надавати лише базову інформацію про породу. Надання вичерпної інформації про породу допомагає користувачам приймати обґрунтовані рішення та пропонує краще розуміння та догляд за котами.

Підсумовуючи, наша система розпізнавання котів пропонує комплексне та спеціалізоване рішення для ідентифікації порід котів та керування ними. Її унікальні функції, такі як окремий додаток для паспорта

тварин, автоматичний вибір породи, публічний доступ до API, підвищена точність і надійність, а також повна інформація про породу, відрізняють її від конкурентів, таких як Cat Scanner, Azure Cat Classification і Cat breed identifier. Зосереджуючись на конкретних потребах власників котів і професіоналів у галузі догляду за домашніми тваринами, сервіс з розпізнавання котів забезпечує хороший досвід користувача, видаючи точні та надійні результати ідентифікації породи. Як наслідок, система розпізнавання котів є найкращим вибором для тих, хто шукає вдосконалене та надійне рішення.

1.3 Інтереси відносно системи

Система зі сканування порід котів може залучити користувачів своїм функціоналом. Визначення груп користувачів [18] для нашого сервісу є важливим, оскільки це допоможе нам адаптувати рішення відповідно до їхніх конкретних вимог, підвищуючи задоволеність і залученість користувачів. Розуміючи потреби та очікування різних груп користувачів, ми зможемо визначити пріоритети та зосередити свої зусилля на розробці функцій, які найбільше відповідають цільовій аудиторії. Цей підхід також допоможе ефективно розподілити ресурси, покращити маркетингові стратегії та, зрештою, підвищити загальний успіх і впровадження системи.

Існує 2 зацікавлені особи у системі: власник тварини та робітник сторонньої організації. Ролі мають свої інтереси та обмеження (таблиця 1.1).

Власник тварини – це користувач програмної системи з ведення цифрової картки тварини. При додаванні тварини у систему, порода кішки автоматично визначається та обирається зі списку запропонованих порід. Користувач може отримувати доступ до ексклюзивних функцій системи.

За допомогою API доступ до ідентифікатора порід може отримати не тільки користувачі нашого додатку, а й робітники публічних установ, які мають інтеграцію з нашим сервісом. Завдяки REST API, ветеринари,

робітники притулків тощо зможуть отримувати доступ до нашої технології, що значно спростить їх роботу.

Таблиця 1.1 – Інтереси відносно системи

Зацікавлена особа	Обмеження	Інтереси
Власник тварини	Не може користуватися програмною системою бе з смартфона	Може додати інформацію про тварину, автоматично визначити породу при виборі фотографії
Робітник сторонньої організації (ветеринар, робітник притулку, …)	Не може користуватися системою без доступу до API	Може визначати породу завдяки інтеграції у систему організації, доповнювати функціонал своїми функціями.

1.4 Технології класифікації зображень

Класифікація зображень – це завдання присвоєння мітки або категорії зображенню на основі його візуального вмісту. Процес передбачає аналіз зображення та виділення відповідних ознак, які можна використовувати, щоб відрізнити його від інших зображень у наборі даних. Існує кілька технологій, які можна використовувати для класифікації зображень, кожна з яких має свої сильні та слабкі сторони.

Вибір вірної технології класифікації зображень для ідентифікації порід котів є суттєвим для загального успіху та ефективності системи. Важливість вибору правильної технології обумовлена декількома факторами:

– точність. Основною метою системи ідентифікації порід котів є точне розпізнавання та класифікація різних порід котів на зображеннях. Вибір технології класифікації зображень значною мірою впливає на здатність системи точно ідентифікувати різні породи, навіть тих, що мають незначні

відмінності. Висока точність необхідна для забезпечення задоволення користувачів та довіри до системи;

– ефективність. Ефективна технологія класифікації зображень дозволяє системі ідентифікації порід котів швидко обробляти зображення без жертвування точністю. Це забезпечує безшовний досвід користувача та реалізацію системи в різних сценаріях, таких як притулки для тварин, ветеринарні клініки або мобільні додатки, де швидкі результати є бажаними;

– стійкість. Надійна технологія класифікації зображень повинна обробляти варіації якості зображень, умов освітлення та поз котів. Це гарантує, що система ідентифікації порід котів залишається точною та надійною, навіть при роботі з неідеальними вхідними зображеннями, що робить її більш універсальною та корисною для користувачів у різних ситуаціях;

– масштабованість. У міру розвитку та росту системи ідентифікації порід котів, можливо, їй доведеться пристосуватися до збільшення кількості порід або додаткових функцій. Обрана технологія класифікації зображень повинна бути масштабованою та адаптивною, щоб відповідати майбутнім вимогам, забезпечуючи.

Отже, ми маємо обрати оптимальну технологію для сканування порід котів. Розглянемо деякі з найпопулярніших технологій класифікації зображень.

1.4.1 Згорткові нейронні мережі (CNN)

Згорткові нейронні мережі (CNN – Convolutional Neural Networks) – це тип глибокої нейронної мережі, спеціально розроблений для класифікації зображень [14]. Вони дуже ефективні в аналізі складних візуальних шаблонів і вилученні відповідних характеристик із зображень. CNN використовують серію згорткових шарів для вилучення функцій із

зображення та об’єднання шарів для зменшення розмірності витягнутих функцій. Вихідні дані останнього рівня об’єднання потім подаються на повністю підключений рівень, який генерує остаточне рішення про класифікацію.

CNN показали високу продуктивність у різних задачах класифікації зображень, таких як розпізнавання об’єктів і обличь. Однак вони вимагають великої кількості навчальних даних і інтенсивні з точки зору обчислень, що ускладнює їх навчання та розгортання в середовищах з обмеженими ресурсами.

1.4.2 Машини підтримки векторів (SVM)

Машини підтримки векторів (SVM – Support Vector Machines) – це тип алгоритму машинного навчання, який можна використовувати для класифікації зображень. Вони працюють, знаходячи гіперплощину, яка розділяє різні класи в просторі ознак. SVM можна навчити на наборі характеристик зображення, вилучених за допомогою різних методів, таких як гістограма орієнтованих градієнтів (HOG) або локальні бінарні шаблони (LBP) [16].

SVM ефективні в обробці просторів функцій великої розмірності та можуть обробляти нелінійні межі рішень. Вони також ефективні з точки зору обчислень і широко використовуються в різних програмах класифікації зображень. Однак SVM вимагають ретельного налаштування своїх гіперпараметрів і можуть бути чутливими до вибору функції ядра.

1.4.3 Випадкові ліси

Випадкові ліси (Random Forests) – це метод ансамблевого навчання, який можна використовувати для класифікації зображень. Він працює шляхом побудови кількох дерев рішень на різних підмножинах навчальних

даних, а потім комбінування передбачень цих дерев для прийняття остаточного рішення щодо класифікації. Випадкові ліси можуть обробляти як безперервні, так і категоричні дані, а також можуть обробляти відсутні дані.

Випадкові ліси менш схильні до перенавчання порівняно з іншими алгоритмами машинного навчання та можуть обробляти шумні або нерелевантні функції. Їх також легко реалізувати та інтерпретувати, що робить їх популярним вибором для програм класифікації зображень. Однак випадкові ліси можуть потребувати інтенсивних обчислень і не підходять для великих наборів даних.

1.4.4 Мережі глибокої віри (DBN)

Глибокі мережі довіри (DBN – Deep Belief Networks) – це тип глибокої нейронної мережі, який можна використовувати для неконтрольованого вивчення функцій [10]. Вони складаються з кількох шарів обмежених машин Больцмана (RBM) і можуть бути навчені за допомогою алгоритму контрастної дивергенції. Функції, отримані DBN, після навчання можна використовувати для класифікації зображень.

DBN можуть вивчати складні ієрархічні представлення вхідних даних і обробляти вхідні простори великої розмірності. Вони також ефективні в обробці шумів або відсутніх даних і можуть використовуватися для різних завдань класифікації зображень, таких як розпізнавання об’єктів і виявлення обличь. Однак DBN можуть бути обчислювально дорогими та вимагати великої кількості навчальних даних.

1.4.5 Трансферне навчання

Трансферне навчання (Transfer Learning) – це техніка, за якої попередньо навчена модель використовується як відправна точка для нового

класифікаційного завдання [19]. Попередньо підготовлена модель зазвичай навчається на великому наборі даних, наприклад ImageNet, і може бути налаштована на меншому наборі для конкретного завдання класифікації.

Трансферне навчання може значно зменшити кількість навчальних даних, необхідних для нового завдання класифікації, і може підвищити точність моделі. Він також є обчислювально ефективним і може бути легко реалізований за допомогою різноманітного глибокого навчання.

1.5 Вибір кращої технології для класифікації зображень котів

Згорткові нейронні мережі (CNN) вважаються найкращою технологією для класифікації порід котів з кількох причин порівняно з іншими наданими варіантами:

– вилучення функцій. CNN можуть автоматично витягувати релевантні ознаки із зображень, не вимагаючи ручного проектування функцій, що робить їх більш ефективними та результативними для завдань класифікації, наприклад, класифікація порід котів. SVM, Random Forests і DBF вимагають певної міри ручної розробки функцій, що може зайняти багато часу та бути менш ефективним у фіксуванні нюансів особливостей породи котів;

– просторові зв’язки. CNN можуть фіксувати просторові співвідношення між пікселями на зображенні, що важливо для класифікації порід котів, оскільки такі ознаки, як розмір і форма вух, очей і тіла, можуть суттєво відрізнятися між породами. SVM, Random Forests і DBF можуть бути не настільки ефективними для захоплення цих просторових зв’язків, оскільки вони не розроблені спеціально для завдань класифікації зображень;

– глибоке навчання. CNN – це тип моделі глибокого навчання, яка дозволяє вивчати складні представлення вхідних даних за допомогою кількох рівнів обробки. Це робить їх добре придатними для класифікації порід котів, оскільки існує багато тонких відмінностей між породами котів,

які можна вловити за допомогою алгоритмів глибокого навчання. SVM, Random Forests і Transfer Learning також можуть використовувати методи глибокого навчання, але для досягнення оптимальної продуктивності може знадобитися більш просунута архітектура та налаштування;

– великі набори даних. CNN потребують великої кількості навчальних даних для досягнення високої точності. На щастя, є багато загальнодоступних наборів даних зображень котів, таких як Oxford-IIIT Pet Dataset, які можна використовувати для навчання CNN класифікації порід котів. SVM, Random Forests, DBF і Transfer Learning також можуть працювати з великими наборами даних, але CNN показали свою ефективність у задачах класифікації зображень завдяки своїй здатності автоматично вивчати відповідні функції та фіксувати просторові зв’язки.

Загалом CNN вважаються найкращою технологією для класифікації порід котів завдяки їхній здатності автоматично отримувати ознаки, фіксувати просторові зв’язки, вивчати складні представлення та обробляти великі набори даних. SVM, Random Forests, DBF і Transfer Learning – це життєздатні варіанти для завдань класифікації зображень, але CNN будуть особливо ефективними в контексті класифікації порід котів.

1.6 Розуміння згорткових нейронних мереж

Згорткові нейронні мережі (CNN) – це тип архітектури глибокого навчання, який здобув значну популярність завдяки успіхам у різних задачах комп'ютерного зору, таких як класифікація зображень, виявлення об'єктів та семантична сегментація. CNN розроблені для автоматичного навчання та видобутку ієрархічних особливостей з вхідних даних [3], завдяки чому вони особливо підходять для завдань, що включають високовимірні входи, такі як зображення. У цьому детальному поясненні ми дослідимо основні компоненти згорткових нейронних мереж та обговоримо, як вони взаємодіють для видобутку особливостей та класифікації.

На рисунку 1.4 наведено схему базової згорткової нейронної мережі.

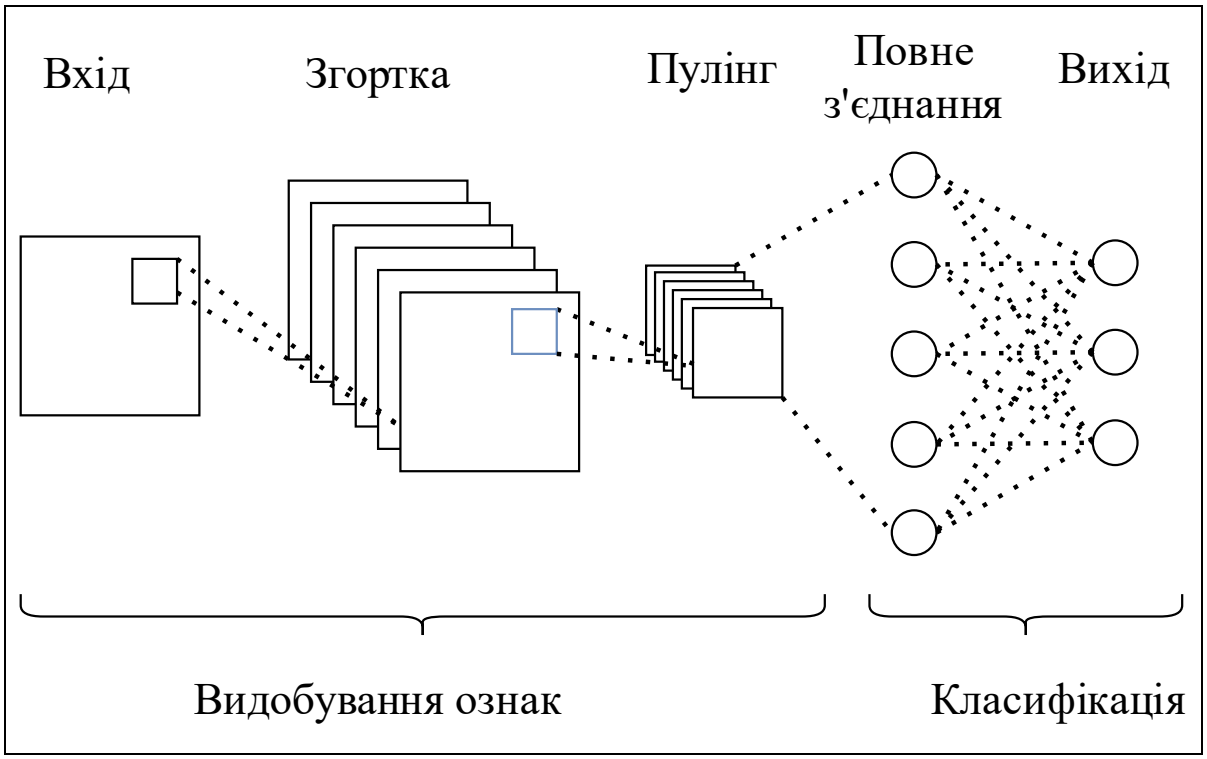


Рисунок 1.4 – Схема базової згорткової нейронної мережі

Розглянемо детально кожний елемент наведеної вище діаграми:

1) видобуток ознак. Процес видобутку ознак у CNN складається з двох основних компонентів: згортки та пулінгу. Ці операції застосовуються послідовно, часто у декількох шарах, для перетворення вхідних даних на набір карт особливостей, які можна використовувати для класифікації:

– згортка. Основна операція у CNN, яка служить для видобутку локальних особливостей з вхідних даних. Під час процесу згортки набір фільтрів (або ядер) застосовується до вхідних даних у вигляді ковзаючого вікна. Кожен фільтр розроблений для виявлення конкретних шаблонів або особливостей, таких як краї, кути або текстури, у вхідних даних. Вихідним результатом операції згортки є карта особливостей – матриця, яка відображає наявність та розташування виявлених особливостей. Оскільки фільтр ковзає над вхідними даними, він обчислює скалярний добуток між

фільтром та відповідною областю у вхідних даних, генеруючи одне значення для кожної позиції на карті особливостей. Цей процес повторюється для кожного фільтра, що призводить до набору карт особливостей, які відображають різні аспекти вхідних даних;

– пулінг. Операція, яка використовується для зменшення просторових розмірів карт особливостей, ефективно відбираючи видобуту інформацію, зберігаючи її основні характеристики. Це зменшення розмірності сприяє зниженню обчислювальної складності мережі [9] та допомагає зменшити ризик перенавчання. Існує декілька стратегій пулінгу, серед яких максимальний та середній пулінг є найпоширенішими. Максимальний пулінг вибирає максимальне значення в локальному регіоні карти особливостей, тоді як середній пулінг обчислює середнє значення в тому ж регіоні. Виконуючи пулінг, мережа стає більш стійкою до невеликих просторових варіацій вхідних даних, що може бути важливим для таких завдань, як розпізнавання зображень;

2) класифікація. Після процесу видобутку особливостей отримані карти особливостей подаються на повністю з'єднаний шар, а потім на вихідний шар для виконання класифікації. Ці шари співпрацюють для відображення високовимірних представлень особливостей на набір ймовірностей або міток класів:

– повністю з'єднаний шар. Служить мостом між видобутком особливостей та класифікацією компонентів CNN. Цей шар приймає вихід з останнього шару пулінгу та перетворює його на одновимірний вектор, який подається на традиційну пряму нейронну мережу. Повністю з'єднаний шар відповідає за інтеграцію навчених особливостей з попередніх шарів та генерацію високорівневого представлення вхідних даних;

– вихідний шар. Є останнім шаром CNN та відповідає за виробництво кінцевих результатів класифікації. Цей шар зазвичай складається з набору нейронів рівних кількості класів у задачі класифікації. Функція активації, що використовується у вихідному шарі, залежить від

характеру задачі. Наприклад, для задач багатокласової класифікації часто використовується функція активації softmax, тоді як для задач бінарної класифікації використовується сигмоїдна функція активації. Вихідний шар призначає ймовірність кожному класу на основі високорівневого представлення, згенерованого повністю з'єднаним шаром. Клас з найвищою ймовірністю обирається як кінцеве передбачення.

2 ПОСТАНОВКА ЗАДАЧІ

2.1 Постановка задачі дослідження

Головною задачею дослідження є підготовка до розробки сервісу сканування зображень котів та визначення їх породи. Для успішної реалізації треба визначити які технології мають використовуватись для розробки. Сервіс має працювати як усередині програмної системи ведення обліку домашніх тварин, так і зовні, для публічних установ, або як продукт, який компанії матимуть бажання інтегрувати у себе.

Основним завданням магістерської роботи є наступне: розробити алгоритм з використанням технологій штучного інтелекту для сканування зображень котів, який приймає файл-зображення та повертає результат передбачення, який можна отримати за допомогою REST API або мобільного додатку.

Задля реалізації поставленої мети передбачається розглянути наступний набір питань:

1) опрацювати джерела по темі навчання нейронних мереж за технологією Convolutional Neural Networks;

2) опрацювати алгоритм визначення породи кішки за ї фотографією;

3) задокументувати API методи для взаємодії з алгоритмом зовні. Основний метод має приймати зображення та повертати масив з породами та їх вірогідністю у відсотках;

4) розробити інтерфейс мобільного додатку, який дозволить обирати / робити нову фотографію кішки та передавати її на опрацювання до back-end завдяки зазначеному API методу;

5) обрати правильні технології для розробки, а саме:

– для алгоритму ідентифікації котів;

– для серверної частини (back-end) та API методів взаємодії з алгоритмом;

– для мобільного (mobile) застосунку;

6) обрати навчальний набір даних;

7) проаналізувати результати;

8) оптимізувати роботу моделі розпізнавання.

2.2 Навчання нейронної мережі моделлю Convolutional Neural Networks

Згорткові нейронні мережі (CNN) – це тип моделі глибокого навчання, який широко використовується для завдань класифікації зображень, зокрема класифікації порід котів.

Навчання CNN включає кілька етапів, таких як підготовку даних, визначення моделі, навчання моделі та її оцінку. Деталізований опис кожного з цих кроків:

– підготовка даних. Першим кроком у навчанні CNN є підготовка даних для навчання. Це передбачає вибір набору даних із зображеннями котів, які містять мітки із зазначенням породи кожного кішки. Популярним набором даних для класифікації порід котів є Oxford-IIIT Pet Dataset, який містить велику кількість зображень котів різних порід. Набір даних має бути розділений на набори для навчання (навчання моделі), перевірки (налаштування гіперпараметрів) та тестування (оцінка продуктивності моделі);

– визначення моделі. Наступним кроком є визначення архітектури моделі CNN. Це передбачає визначення кількості та типу шарів, кількості фільтрів у кожному шарі, розміру фільтра, функцій активації та інших параметрів. У TensorFlow це можна зробити за допомогою Keras API, який надає інтерфейс високого рівня для визначення нейронних мереж. Загальна архітектура CNN для класифікації порід котів включає кілька згорткових шарів, за якими слідують шари максимального об’єднання, а потім кілька повністю з’єднаних шарів у кінці;

– навчання моделі. Коли модель визначена, її можна навчити за допомогою навчального набору. Це передбачає подачу пакетів зображень і відповідних міток через модель, обчислення втрат (тобто різниці між прогнозованими та фактичними мітками) і оновлення ваг моделі за допомогою зворотного поширення. Мета навчання – мінімізувати втрати та оптимізувати ваги моделі, щоб вона могла точно передбачити породу кота за зображенням;

– оцінка моделі. Після навчання моделі її можна оцінити за допомогою набору перевірки. Це передбачає передачу зображень набору перевірки через модель та обчислення таких показників, як достовірність, точність та повнота, щоб оцінити ефективність моделі. Якщо модель не працює належним чином, гіперпараметри можна налаштувати за допомогою набору перевірки, а модель можна перенавчати, доки не буде досягнуто бажаної продуктивності.

Щоб навчити [1] CNN TensorFlow для класифікації порід котів, ми можемо виконати наступні кроки:

– завантаження набору даних. У TensorFlow ми можемо використовувати клас ImageDataGenerator для завантаження та попередньої обробки зображень із набору даних [6]. Це передбачає перемасштабування значень пікселів до діапазону 0-1, застосування методів збільшення даних, таких як випадкові повороти та перевороти, щоб збільшити розмір навчального набору, і поділ даних на набори для навчання, перевірки та тестування;

– визначення архітектури моделі. У TensorFlow ми можемо використовувати Keras API для визначення архітектури моделі CNN. Це передбачає знаходження кількості та типу шарів, кількості фільтрів у кожному шарі, розміру фільтра, функцій активації та інших параметрів;

– компіляція моделі. Коли модель визначена, ми можемо скомпілювати її за допомогою «compile» методу. Це передбачає визначення

функції втрат (наприклад, категорійної перехресної ентропії), оптимізатора (наприклад, Адама) і метрики оцінки (наприклад, точності);

– навчання моделі. Ми можемо навчити модель за допомогою методу «fit», який приймає навчальні дані як вхідні дані та навчає модель для певної кількості епох (тобто ітерацій по всьому навчальному набору). Під час навчання можна відстежувати продуктивність моделі на перевірочному наборі, щоб переконатися, що вона не перенавчена навчальному набору;

– оцінка моделі. Коли модель навчена, ми можемо оцінити її продуктивність на тестовому наборі за допомогою методу оцінки. Це обчислить такі показники, як достовірність, точність та повнота, які можна використовувати для визначення продуктивності моделі.

2.3 Серверна частина

Django – популярний веб-фреймворк [15] на основі Python, який можна використовувати для розробки потужних і масштабованих внутрішніх програм. У контексті ідентифікації породи котів Django можна використовувати для створення RESTful API, який дозволяє користувачам надсилати зображення та отримувати прогнози щодо породи котів.

Однією з ключових особливостей Django є його здатність забезпечувати високий рівень безпеки для веб-додатків. Фреймворк містить низку вбудованих функцій безпеки, таких як захист від міжсайтової підробки запитів (CSRF) і захист від SQL-ін’єкцій, які допомагають запобігти типовим вразливостям безпеки. Крім того, Django забезпечує безпечний спосіб обробки автентифікації та авторизації користувачів, що полегшує додавання функцій реєстрації та входу до веб-програм.

Ще однією перевагою Django є його модульність і розширюваність. Розробники можуть легко створювати власні модулі та підключати їх до середовища Django, що дає змогу додавати нові функції або змінювати

існуючі за потреби. Цей підхід також полегшує повторне використання коду в різних проектах, заощаджуючи час і зусилля в процесі розробки.

Django також має високу масштабованість, що робить його придатним для створення великомасштабних програм, які потребують високого рівня продуктивності та надійності. Фреймворк включає вбудоване кешування та підтримку розподілених обчислень, що може допомогти покращити продуктивність веб-додатків під великим навантаженням.

Щоб реалізувати API, ми можемо створити додаток Django, який використовує бібліотеку TensorFlow для завантаження та використання навченої моделі CNN для класифікації порід котів. Ось кілька прикладів endpoint-ів (кінцевих точок API), які можна реалізувати:

– /api/predict – цей endpoint може отримувати файл-зображення і повертати об’єкт JSON із 3 найпопулярнішими прогнозованими породами та їхніми ймовірностями;

– /api/train – цей endpoint можна використовувати для перенавчання моделі CNN за допомогою нових даних. Кінцева точка може отримати набір навчальних зображень і міток і відповідно оновити ваги моделі;

– /api/status – цей endpoint можна використовувати для отримання інформації про поточний стан моделі, наприклад її точність, кількість навчальних зразків тощо.

Щоб реалізувати кінцеву точку /api/predict, ми можемо визначити функцію перегляду, яка приймає запит POST із зображенням у кодуванні base64. Потім функція може декодувати зображення, попередньо обробити його (наприклад, змінити розмір, нормалізувати тощо) і пропустити через модель CNN, щоб отримати прогнозовані ймовірності породи. Після цього функція може повернути об’єкт JSON із 3 найпопулярнішими прогнозованими породами та їхніми ймовірностями.

2.4 Мобільний додаток

Щоб доповнити серверне рішення, ми можемо створити мобільний додаток, який дозволить користувачам отримувати доступ до API та виконувати класифікацію порід котів на власних зображеннях. Мобільний додаток можна розробити за допомогою технології, такої як Swift або Kotlin.

Swift – потужна та інтуїтивно зрозуміла мова програмування [4], розроблена Apple для створення додатків на платформах iOS, macOS, watchOS і tvOS. Представлений у 2014 році як наступник Objective-C, Swift стає все більш популярним завдяки своєму сучасному синтаксису, продуктивності та функціям безпеки.

Серед ключових переваг Swift:

– зручність читання. Swift має чистий, виразний синтаксис, який легко читати та писати. Така читабельність робить його більш доступним для нових розробників і спрощує обслуговування коду для досвідчених програмістів;

– продуктивність. Swift розроблений як швидкий і ефективний, з оптимізацією продуктивності, що робить його конкурентоспроможним з іншими мовами програмування;

– безпека. Swift містить кілька мовних функцій, які сприяють безпечнішому кодуванню, наприклад визначення типу, додаткові параметри та обробку помилок, що допомагає запобігти поширеним помилкам програмування;

– взаємодія з Objective-C. Swift розроблено для бездоганної роботи з кодом Objective-C в рамках одного проекту, дозволяючи розробникам поступово переводити свої існуючі програми на Swift або використовувати існуючі бібліотеки та фреймворки Objective-C;

– відкритий вихідний код. Swift є мовою з відкритим кодом, що означає, що розробники можуть брати участь у його розробці, повідомляти про проблеми та отримувати доступ до вихідного коду.

Таким чином, Swift – це сучасна, високопродуктивна мова програмування, яка забезпечує читабельність, безпеку та взаємодію з Objective-C. Також його використання дозволить перетворити iOS додаток у повноцінну macOS програму практично не змінюючи код, так як у нових ноутбуків компанії Apple використовуються ARM процесори, здатні нативно її запускати. Саме тому Swift є оптимальним вибором для розробки мобільного додатку для програмної системи сканування порід котів.

Мобільний додаток для ведення цифрової картки домашньої кішки з можливістю визначення її породи повинен мати наступні функції:

– автентифікація. Користувач повинен мати можливість увійти або зареєструватися, щоб використовувати додаток;

– камера / галерея. Користувач повинен мати можливість зробити знімок або вибрати зображення зі своєї галереї. Приклад наведено на рисунку 2.1;

– обробка зображення. Зображення слід обробити та змінити його розмір, щоб відповідати вхідному розміру моделі CNN;

– прогнозування. Зображення має бути надіслано у внутрішній API для прогнозування, а результат має відобразитися користувачеві;

– результати. Користувач повинен мати можливість переглянути прогнозовану породу та оцінку ймовірності.

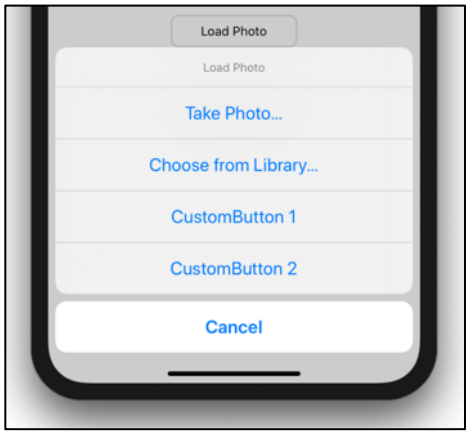


Рисунок 2.1 – Вибір зображення у мобільному додатку

2.5 Навчальний набір даних

Oxford-IIIT Pet Dataset – це велика колекція зображень котів і собак, створена Visual Geometry Group Оксфордського університету та Індійського технологічного інституту. Набір даних містить понад 7000 зображень 37 різних порід котів [11], що робить його цінним ресурсом для навчання та тестування алгоритмів комп'ютерного зору, які класифікують і розпізнають породи домашніх тварин.

Зображення в наборі даних мають різні розміри та якість, і багато з них містять кілька тварин або об'єктів. Щоб полегшити навчання та тестування алгоритмів комп'ютерного зору, кожне зображення в наборі даних було позначено рамкою, яка ідентифікує тварину на зображенні. Крім того, кожне зображення позначено породою домашньої тварини, забезпечуючи базову істину для оцінки точності алгоритмів класифікації.

Oxford-IIIT Pet Dataset використовується як еталонний набір даних для оцінки продуктивності моделей глибокого навчання, таких як згорткові нейронні мережі (CNN).

Набір даних є у вільному доступі для завантаження, і до нього можна отримати доступ через веб-сайт групи Visual Geometry Group. Крім того, набір даних було ліцензовано за міжнародною ліцензією Creative Commons Attribution-ShareAlike 4.0, яка дозволяє обмінюватися та адаптувати набір даних із належним зазначенням авторства.

Підсумовуючи, Oxford-IIIT Pet Dataset є цінним ресурсом для дослідників і розробників, які працюють у сфері комп'ютерного зору та машинного навчання. Завдяки великій колекції зображень домашніх тварин і ґрунтовних анотацій, він надає багатий набір даних для навчання та тестування алгоритмів класифікації порід домашніх тварин.

3 РОЗРОБКА ФУНКЦІОНАЛУ ДЛЯ СКАНУВАННЯ ПОРІД КОТІВ

3.1 Модель розпізнавання порід котів

Першим етапом є створення моделі, яка використовуватиметься для сканування зображень порід котів. У попередніх розділах ми вже визначились з технологію, яку будемо для цього використовувати – Convolutional Neural Networks (Згорткові Нейронні Мережі, CNN).

Треба створити модель, яка зможе прогнозувати породу кішки за її фотографією. Для розробки ми будемо використовувати мову програмування Python та бібліотеку Keras.

Python – це популярна мова програмування, відома своєю читабельністю, простотою та легкістю використання. Вона особливо підходить для машинного навчання з кількох причин:

– у Python є велика кількість бібліотек та фреймворків, спеціально розроблених для машинного навчання. Ці бібліотеки спрощують процес реалізації, тренування та оцінювання моделей машинного навчання;

– чіткий та лаконічний синтаксис мови дозволяє розробникам писати та підтримувати код машинного навчання з мінімальними зусиллями, скорочуючи криву навчання для новачків та спрощуючи реалізацію алгоритмів для досвідчених програмістів;

– Python сумісний з різними платформами, що забезпечує розробку та розгортання додатків машинного навчання на різних операційних системах без значних змін;

– Python має велику активну спільноту розробників, які сприяють розвитку мови, підтримують бібліотеки та надають підтримку. Ця жива спільнота допомагає тримати Python актуальним щодо останніх досягнень у машинному навчанні.

Універсальність Python у різних галузях, включаючи веб-розробку та науку про дані, дозволяє розробникам працювати над різними аспектами проекту, використовуючи одну мову, що спрощує процес розробки.

Keras – це бібліотека нейронних мереж з відкритим кодом, написана на Python, яка спрощує процес створення, навчання та оцінки моделей глибокого навчання. Вона набула широкої популярності серед фахівців з машинного навчання завдяки зручному високорівневому API та здатності працювати з кількома бекендами глибокого навчання, такими як TensorFlow, Microsoft Cognitive Toolkit та Theano.

Keras зосереджений на тому, щоб бути зручним, модульним та розширюваним [5], що робить його чудовим вибором як для початківців, так і для досвідчених практиків глибокого навчання. Ключові особливості Keras:

– високорівневе API. Keras надає високорівневе, інтуїтивне API, яке значно спрощує створення моделей глибокого навчання. Це дозволяє користувачам легко створювати, навчати та оцінювати нейронні мережі лише за допомогою кількох рядків коду, прискорюючи процес розробки та скорочуючи криву навчання для початківців;

– модульність та розширюваність. Keras побудований навколо концепції модульності, що дозволяє користувачам комбінувати різні шари, оптимізатори, функції активації та функції втрат для створення власних архітектур глибокого навчання. Ця гнучкість дозволяє користувачам експериментувати з різними конфігураціями та швидко ітерувати свої моделі;

– попередня обробка та аугментація даних. Keras пропонує вбудовані утиліти для попередньої обробки та аугментації даних, що допомагає користувачам легко підготувати свої набори даних для навчання та поліпшення продуктивності моделі, вводячи варіативність у вхідні дані;

– навчені моделі. Keras надає колекцію навчених моделей для загальних завдань глибокого навчання, таких як класифікація зображень та

обробка природної мови. Ці моделі можна налаштувати для конкретних застосувань, економлячи час та ресурси під час розробки.

Отже, використаємо мову програмування Python та бібліотеку Keras для реалізації нашого проєкту.

Першим кроком напишемо функцію ініціалізації набору даних для навчання та валідації. Спочатку створимо екземпляр ImageDataGenerator із зазначеними доповненнями даних (рисунок 3.1).

```
def init_dataset(data_dir):
    datagen = ImageDataGenerator(
        rescale=1. / 255,
        shear_range=0.2,
        zoom_range=0.2,
        rotation_range=ROTATION_RANGE,
        horizontal_flip=True,
        validation_split=0.2
    )

    train_generator = datagen.flow_from_directory(
        data_dir,
        target_size=(image_size, image_size),
        batch_size=batch_size,
        class_mode='categorical',
        subset='training'
    )

    validation_generator = datagen.flow_from_directory(
        data_dir,
        target_size=(image_size, image_size),
        batch_size=batch_size,
        class_mode='categorical',
        subset='validation'
    )

    return train_generator, validation_generator
```

Рисунок 3.1 – Код функції init_dataset

Створимо генератор – екземпляр ImageDataGenerator з наступними параметрами:

– rescale. Використовується для нормалізації значень пікселів на зображенні. Якщо кожне значення пікселя поділити на 255 (максимальне значення пікселя для 8-бітного зображення), значення пікселів змінюються

в діапазоні від 0 до 1. Ця нормалізація допомагає моделі збігатися швидше під час навчання та зазвичай призводить до кращої продуктивності;

– shear_range. Керує діапазоном кутів зсуву (у градусах) для трансформації зсуву, який є типом геометричної трансформації, що спотворює зображення шляхом зміщення точок уздовж осі. shear_range 0,2 означає, що кут зсуву буде вибрано випадковим чином із діапазону від -20 градусів до 20 градусів. Трансформація зсуву може допомогти моделі навчитися розпізнавати котів, навіть якщо вони трохи спотворені;

– zoom_range. Керує діапазоном рівнів масштабування для трансформації масштабу. zoom_range 0,2 означає, що коефіцієнт масштабування буде вибрано випадковим чином із діапазону від 0,8 до 1,2. Це може допомогти моделі навчитися розпізнавати котів різних масштабів;

– rotation_range. Керує діапазоном кутів повороту (у градусах) для здійснення обертання. Наприклад, якщо встановити rotation_range на 30, зображення будуть випадковим чином повертатися на кут від -30 до 30 градусів. Це допомагає моделі навчитися розпізнавати котів з зображень різної орієнтації;

– horizontal_flip. Якщо встановлено значення True, цей параметр випадково перевертає зображення по горизонталі. Це допомагає моделі навчитися розпізнавати котів, які дивляться вліво або вправо;

– validation_split. Використовується для поділу набору даних на підмножини навчання та перевірки. Значення 0,2 означає, що 20% набору даних буде використано для перевірки, а решта (80%) – для навчання.

Далі йде створення генератора даних для тренування:

– data_dir. Визначає шлях до каталогу, який містить зображення, організовані у підкаталогах для кожного класу (в даному випадку породи котів);

– target_size. Представляє бажаний розмір зображень після зміни розміру. У цьому випадку встановлено значення (image_size, image_size),

що становить 224 x 224 пікселів. Це змінення розміру необхідне, оскільки модель MobileNetV2 очікує вхідних зображень такого розміру [13];

– batch_size. Встановлює кількість зображень, які будуть включені в кожну партію під час навчання. Менші розміри пакетів можуть допомогти моделі збігатися швидше, але також можуть вимагати більше пам’яті;

– class_mode. Визначає тип масивів міток, які мають бути повернуті. У цьому випадку для нього встановлюється значення «categorical», що означає, що мітки будуть закодовані одним оперативним кодуванням, тобто представлені як двійкові вектори з 1 в індексі, що відповідає справжньому класу, і нулями в інших місцях;

– subset. Для цього параметра встановлено значення «training», що вказує на те, що генератор має використовувати лише частину набору даних, указану для навчання (у цьому випадку 80%).

Параметри для генератора перевірки такі ж, як і для генератора навчання, за винятком параметра subset, для якого встановлено значення «validation». Це вказує на те, що генератор повинен використовувати частину набору даних, указану для перевірки (у даному випадку – 20%).

Далі напишемо функцію створення моделі із вказаною кількістю шарів тонкого налаштування. Fine-tuning [2] – це підхід до перенесення навчання, за якого ваги попередньо навченої моделі навчаються на нових даних. Тонке налаштування можна виконати на всій нейронній мережі або лише на підмножині її шарів, у цьому випадку шари, які не налаштовані, «заморожуються» (не оновлюються під час кроку зворотного поширення). Функція виконує наступні дії:

1) завантажує модель MobileNetV2 без верхнього шару та з попередньо підготовленими вагами;

2) здійснює заморозку шарів перед їх тонким налаштуванням;

3) додає власні шари для класифікації:

– витягує вихідний тензор попередньо навченої моделі MobileNetV2 (base_model) і призначає його змінній x. Вихідний тензор містить функції, витягнуті з вхідного зображення базовою моделлю;

– застосовує двовимірну операцію Global Average Pooling (GAP) до тензора x. GAP зменшує просторові розміри [8] (висоту та ширину) тензора, беручи середнє значення кожного каналу по всій карті функцій. Це допомагає зменшити кількість параметрів і обчислень, роблячи модель більш ефективною та менш схильною до перенавчання;

– створює повністю пов’язаний (щільний) шар із 1024 нейронами та застосовує функцію активації ReLU (Rectified Linear Unit). Він з’єднує вихід шару GAP із щільним шаром, який може вивчати функції високого рівня та виконувати нелінійні перетворення даних;

– створює ще один повністю зв’язаний (щільний) шар із стільки нейронів, скільки є класів у наборі даних (кількість порід котів). Застосовується функція активації softmax, яка перетворює логіти (необроблені оцінки) у ймовірності для кожного класу. Результатом цього шару будуть остаточні прогнози моделі;

4) створює фінальну модель.

Функція активації softmax використовується на вихідному рівні нейронної мережі для задач багатокласової класифікації. Вона перетворює необроблені оцінки (також звані логітами), створені мережею, у ймовірності, сума яких дорівнює 1, що полегшує інтерпретацію результату.

Код функції наведений на рисунку 3.2.

```
def create_model(train_generator, fine_tune_layers=FINE_TUNE_LAYERS):
    base_model = MobileNetV2(input_shape=(image_size, image_size, 3), include_top=False, weights='imagenet')
    for layer in base_model.layers[:-fine_tune_layers]:
        layer.trainable = False
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dense(1024, activation='relu')(x)
    predictions = Dense(len(train_generator.class_indices),activation='softmax')(x)
    model = Model(inputs=base_model.input,outputs=predictions)
    return model
```

Рисунок 3.2 – Код функції create_model

Наступним етапом є створення функцій тренування на збереження моделі (рисунок 3.3).

У функції тренування ми можемо змінювати кількість епох (epochs) та швидкість навчання (learning_rate). У ній ми:

1) компілюємо модель із наступними конфігураціями:

– optimizer. Алгоритм оптимізації, який використовується для оновлення ваг моделі. У цьому випадку використовується оптимізатор Adam із заданою швидкістю навчання;

– loss. Функція втрат, яка використовується для вимірювання різниці між прогнозованими та фактичними мітками;

– metrics. Показники оцінки, які будуть розраховані під час навчання та перевірки. У цьому випадку використовується метрика точності;

2) навчаємо модель за допомогою наданих генераторів даних навчання та перевірки. Процес навчання повторює весь набір даних для вказаної кількості епох;

3) повертаємо навчену модель.

```
def train_model(
        model,
        train_generator,
        validation_generator,
        epochs=EPOCHS,
        learning_rate=0.0001
    ):
    model.compile(
        optimizer=Adam(learning_rate=learning_rate),
        loss='categorical_crossentropy',
        metrics=['accuracy']
    )

    model.fit(train_generator, epochs=epochs, validation_data=validation_generator)

    return model


def save_model(model, model_path):
    model.save(model_path)
```

Рисунок 3.3 – Код функцій train_model та save_model

Далі напишемо функцію передбачення 3х найбільш вірогідніших порід (рисунок 3.4). У якості параметрів вона приймає: model_path – шлях до файлу навченої моделі, image_path – до файлу зображення, для якого слід передбачити три найвірогідніші породи, а train_generator – генератор навчальних даних.

```
def predict_top_3_breeds(model_path, image_path, train_generator):
    img = image.load_img(
        image_path,
        target_size=(image_size, image_size)
    )
    img_array = image.img_to_array(img)

    model = load_model(model_path)
    img_array /= 255.
    img_array = np.expand_dims(img_array, axis=0)

    predictions = model.predict(img_array)[0]
    top_3_idx = np.argsort(predictions)[-3:][::-1]
    top_3_prob = predictions[top_3_idx]
    top_3_breeds = [list(train_generator.class_indices.keys())[i] for i in top_3_idx]

    return list(
        zip(top_3_breeds, top_3_prob)
    )
```

Рисунок 3.4 – Код функції predict_top_3_breeds

Функція виконує наступні дії:

– завантажує зображення за вказаним шляхом і змінює його розмір до вказаного;

– перетворює зображення в масив;

– завантажує навчену модель із заданого шляху;

– нормалізує масив зображень, розділивши його на 255;

– додає додатковий вимір до масиву зображень, щоб відповідати очікуваній вхідній формі моделі;

– використовує навчену модель, щоб передбачити ймовірність породи для заданого зображення;

– отримує індекси трьох найкращих прогнозованих порід;

– отримує ймовірності, що відповідають 3 найкращим прогнозованим породам;

– зіставляє 3 найкращі індекси порід із відповідними назвами порід;

– об’єднує 3 найпопулярніші назви порід і їхні відповідні ймовірності в список кортежів і повертає цей список.

Завершуючим етапом є створення функції тестування передбачень моделі (рисунок 3.5). В якості параметрів вона приймає: data_dir – директорію до папки з зображеннями, model_path – шлях до моделі та train_generator – генератор навчальних даних.

```
def test_random_image(data_dir, model_path, train_generator):
    breed_dirs = [d for d in os.listdir(data_dir) if os.path.isdir(os.path.join(data_dir, d))]
    random_breed = random.choice(breed_dirs)
    breed_images = [f for f in os.listdir(os.path.join(data_dir, random_breed)) if
                    f.endswith(('.jpg', '.jpeg', '.png'))]
    random_image = random.choice(breed_images)
    image_path = os.path.join(data_dir, random_breed, random_image)

    print(f"SELECTED RANDOM IMAGE: {random_image}")

    top_3_breeds = predict_top_3_breeds(model_path, image_path, train_generator=train_generator)

    return top_3_breeds
```

Рисунок 3.5 – Код функції test_random_image

Підсумовуючи, ми успішно розробили код передбачення породи котів, який використовує спеціально навчену модель. Цей код можна використовувати як основу для створення REST API та мобільного додатку, що дозволяє користувачам легко ідентифікувати породи котів із зображень.

3.2 Серверна частина

Ми маємо створити серверну частину для взаємодії з моделлю мобільного додатку, а також для доступу до неї користувачів публічного API. У якості технології для розробки REST API [12] було обрано

фреймворк Django. Взаємодія із моделлю спрощується завдяки використання ним такої ж самої мови програмування – Python, що дозволяє нам вбудувати логіку сканування порід котів у back-end.

Django – це високорівневий веб-фреймворк із відкритим вихідним кодом, який забезпечує швидку розробку безпечних веб-додатків, які зручно підтримувати. Він дотримується архітектурного шаблону Model-View-Template (MVT) і заохочує чистий, багаторазово використовуваний код [20], надаючи набір інструментів, утиліт і передових практик. Django спрощує звичайні завдання веб-розробки, такі як маршрутизація URL-адрес, керування базами даних і обробка форм, дозволяючи розробникам зосередитися на створенні основних функцій своїх програм.

MVT, або Model-View-Template, – це архітектурний шаблон, який використовується у фреймворках веб-розробки, таких як Django. Він розділяє логіку програми на три взаємопов’язані компоненти:

– Model. Представляє структуру даних і забезпечує зберігання, пошук і маніпулювання даними. Зазвичай він взаємодіє з базою даних і визначає схему, зв’язки та правила перевірки даних програми;

– View. Керує обробкою та представленням даних користувачеві. Представлення отримує введені користувачем дані, отримує дані з моделі та відображає відповідний шаблон із даними. По суті, він керує потоком даних між моделлю та шаблоном;

– Template. Визначає структуру та макет HTML-сторінок, представлених користувачеві. Він відповідає за відображення даних, наданих представленням, у візуально привабливий і зручний спосіб.

Патерн MVT сприяє поділу проблем у веб-розробці, роблячи код більш модульним, придатним для обслуговування та масштабованим.

Першим чином створимо файл views.py та імпортуємо залежності, шляхи до моделі та папок для збереження зображень. Код наведено на рисунку 3.6.

```
import uuid
import os
from pathlib import Path
from . import ai
from django.conf import settings
from django.core.files.storage import FileSystemStorage
from rest_framework.views import APIView
from rest_framework.response import Response
from rest_framework import status
from .serializers import BreedSerializer

temp_images_storage = FileSystemStorage(location=os.path.join(settings.MEDIA_ROOT, 'temp_images'))
moderation_storage = FileSystemStorage(location=os.path.join(settings.MEDIA_ROOT, 'moderation'))
data_dir = Path('classifier/dataset').absolute()
model_path = Path('classifier/model/model_name.h5')

train_generator, validation_generator = ai.init_dataset(data_dir)
model = ai.create_model(train_generator)
```

Рисунок 3.6 – Імпорт залежностей

Реалізуємо GET запит /supported_breeds – Supported Breeds (рисунок 3.7). Він повертає масив назв класів порід, на яких навчалась наша модель зі сканування зображень.

```
class SupportedBreeds(APIView):
    def get(self, request):
        breeds = list(train_generator.class_indices.keys())
        return Response(breeds, status=status.HTTP_200_OK)
```

Рисунок 3.7– API запит /supported_breeds

Далі створимо POST запит для передбачення породи /predict_image (PredictImage). Окрім надання результатів передбачення, метод зберігає зображення у тимчасову директорію та присвоює йому унікальний id. Це дозволить у майбутньому звертатись до цього зображення без необхідності повторного завантаження на сервер. Користувач API має надати зображення параметром «image» у форматі відправки FormData.

У результаті метод повертає масив з трьох найвірогідніших передбачень порід котів та «image_id», який є тимчасовим ідентифікатором завантаженого зображення. На рисунку 3.8 наведено код цього POST запиту.

```python
class PredictImage(APIView):
    def post(self, request):
        image = request.FILES.get('image')
        if image:
            image_id = str(uuid.uuid4())
            temp_images_storage.save(f"{image_id}.{image.name.split('.')[-1]}", image)
            image_path = temp_images_storage.path(f"{image_id}.{image.name.split('.')[-1]}")
            prediction_result = ai.predict_top_3_breeds(model_path, image_path, train_generator=train_generator)
            response_data = {
                "predictions": prediction_result,
                "image_id": image_id,
            }
            return Response(response_data, status=status.HTTP_200_OK)
        return Response({"error": "No image provided"}, status=status.HTTP_400_BAD_REQUEST)
```

Рисунок 3.8 – API запит /predict_image

Результат запиту /predict_image можна побачити нижче (рисунок 3.9).


```
{
    "predictions": [
        [
            "Bengal",
            0.9999951124191284
        ],
        [
            "Siamese",
            2.45209275817615e-06
        ],
        [
            "Abyssinian",
            1.3288170066516614e-06
        ]
    ],
    "image_id": "73cf5589-773a-4e9b-805e-6f9608d558fa"
}
```


Рисунок 3.9 – Результат API запиту /predict_image

Останнім кроком є створення методу корекції результату. У якості параметру він приймає назву породи та тимчасовий ідентифікатор, який мав бути отриманий після завантаження зображення та першого результату. Ціль функції – перемістити зображення кішки, порода якої була передбачена некоректно, для перевірки, щоб фахівці могли звернути увагу на конкретне зображення та відкоригувати роботу моделі для досягнення кращих результатів.

Якщо користувач подав невірний ідентифікатор тимчасового зображення або його термін вже минув, запит поверне помилку.

Нижче, на рисунку 3.10, наведено код методу POST /correct_result (CorrectResult).

```
class CorrectResult(APIView):
    def post(self, request):
        serializer = BreedSerializer(data=request.data)
        if serializer.is_valid():
            breed = serializer.data['breed']
            image_id = request.data.get('image_id')
            if breed not in train_generator.class_indices.keys():
                return Response({"error": "Invalid breed"}, status=status.HTTP_400_BAD_REQUEST)
            if temp_images_storage.exists(f"{image_id}.jpg"):
                image_ext = '.jpg'
            elif temp_images_storage.exists(f"{image_id}.jpeg"):
                image_ext = '.jpeg'
            elif temp_images_storage.exists(f"{image_id}.png"):
                image_ext = '.png'
            else:
                return Response({"error": "Invalid image_id"}, status=status.HTTP_400_BAD_REQUEST)
            image_path = temp_images_storage.path(f"{image_id}{image_ext}")
            moderation_storage.save(f"{breed}/{image_id}{image_ext}",
                                    temp_images_storage.open(f"{image_id}{image_ext}"))
            temp_images_storage.delete(f"{image_id}{image_ext}")
            return Response({"status": "success"}, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
```

Рисунок 3.10 – Результат API запиту /correct_result

Отже, було створено серверну частину програмної системи, яка дозволить мобільному додатку взаємодіяти з моделлю, а також отримувати доступ до неї стороннім організаціям, які є нашими клієнтами.

3.3 Мобільний додаток

Завершуючим етапом розробки є створення мобільного Swift додатку. Swift є ідеальним рішенням для розробки мобільного застосунку для ведення обліку домашніх тварин завдяки своїй високій продуктивності, яка забезпечує зручну роботу користувача та розпізнавання породи в реальному часі. Його зручний синтаксис дозволяє створювати візуально привабливі та інтуїтивно зрозумілі інтерфейси, тоді як нативна інтеграція з iOS гарантує безперебійну роботу на пристроях Apple. Також, функції безпеки Swift допомагають захистити конфіденційні дані користувачів.

Було реалізовано мобільний додаток для ведення обліку домашніх тварин. На головній (Home) сторінці відображені усі тварини, яких користувач додав раніше (рисунок 3.11).

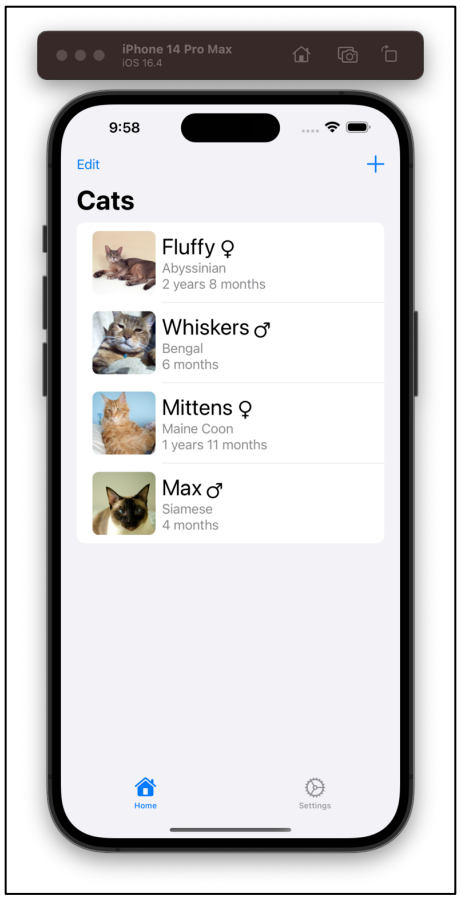


Рисунок 3.11 – Головна сторінка

Сторінка відображає наступні компоненти:

1) список котів. Кожен елемент списку містить властивості:

– ім’я тварини;

– символ статі тварини;

– порода тварини;

– вік тварини;

– фотографія тварини;

2) кнопка «+» додавання нової тварини;

3) кнопка редагування, запускає відповідний режим (рисунок 3.12);

4) перемикач сторінок.

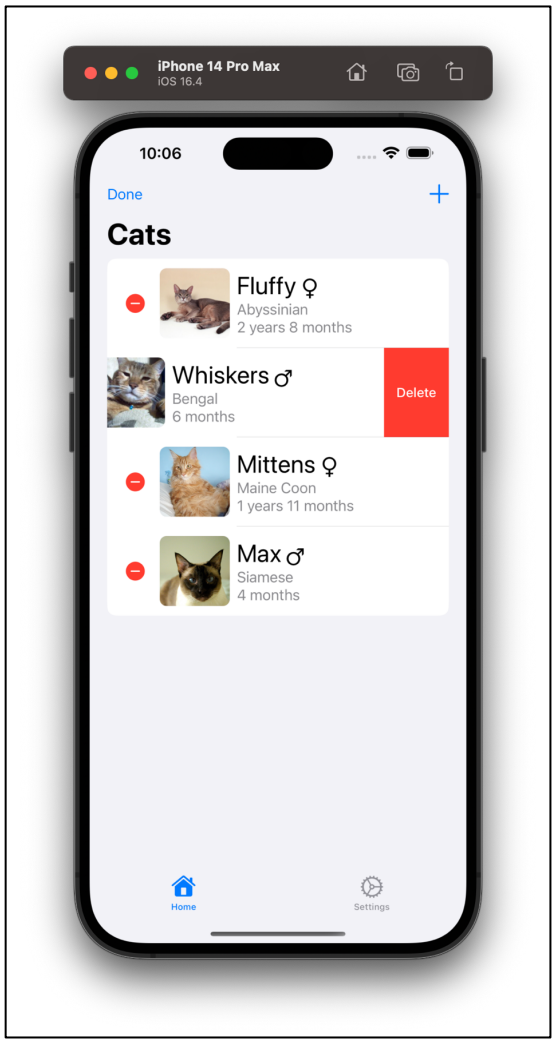


Рисунок 3.12 – Режим редагування

При натисканні на кнопку «+» з’являється сторінка додавання нової тварини. Інтерфейс зображено на рисунку 3.13.

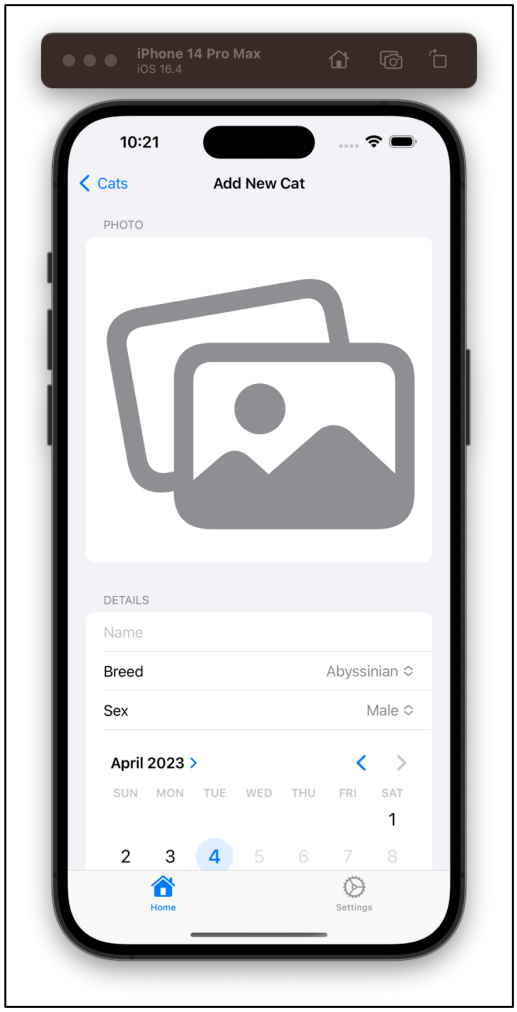


Рисунок 3.13 – Додавання нової тварини

Сторінка містить:

– засіб вибору зображень;

– поле для введення імені тварини;

– список для вибору статі тварини;

– засіб вибору дати народження тварини. Додаток не може додавати немовлят, тому дата народження має бути як мінімум місяць тому;

– кнопка збереження.

Елемент попереднього перегляду при натисканні запускає вибір зображення з фотоплівки користувача (рисунок 3.14) У додатку присутня валідація, кнопка збереження не буде активною поки користувач не вибере фотографію тварини та не введе її ім’я, стать та дату народження.

Рисунок 3.14 – Вибір зображення кішки з фотоплівки

Після вибору зображення з фотоплівки смартфону на екрані сторінки додавання нової тварини відобразиться відповідний індикатор завантаження (рисунок 3.15), поки серверна частина не визначить породу кішки та не надійшле JSON результат з ймовірностями, який буде опрацьований додатком.

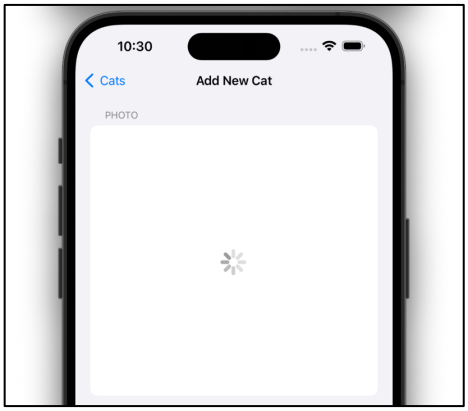


Рисунок 3.15 – Індикатор обробки зображення

Мобільний додаток обробив запит із сервера та обрав правильну породу – Британську короткошерсту. Нижче на рисунку 3.16 наведено код обробки передбачень.

```
func predictBreed() {
    guard let catImageData = catImage?.jpegData(compressionQuality: 0.5) else { return }
    DispatchQueue.main.async {
        isLoading = true
    }
    let url = URL(string: "https://127.0.0.1:8000/api/predict_image/")!
    var request = URLRequest(url: url)
    request.httpMethod = "POST"
    request.setValue("multipart/form-data; boundary=\(boundary)", forHTTPHeaderField: "Content-Type")
    request.httpBody = createBody(with: catImageData, boundary: boundary)
    let sessionDelegate = SessionDelegate()
    let session = URLSession(configuration: .default, delegate: sessionDelegate, delegateQueue: nil)
    session.dataTask(with: request) { data, response, error in
        guard let data = data, error == nil else { return }
        do {
            guard let jsonData = try JSONSerialization.jsonObject(with: data, options: []) as? [String: Any] else {…
            }
            guard let predictionsArray = jsonData["predictions"] as? [[Any]], !predictionsArray.isEmpty else {…
            }
            guard let imageId = jsonData["image_id"] as? String else {…
            }
            let predictions = predictionsArray.compactMap { entry -> (String, Double)? in…
            }
            let firstPrediction = predictions[0]
            DispatchQueue.main.async {
                isLoading = false
            }
            if (firstPrediction.0 == "No cat detected") {…
            }
            validCatImage = catImage
            DispatchQueue.main.async {
                if let index = CatBreed.allCases.firstIndex(where: { $0.id == firstPrediction.0 }) {
                    catBreed = CatBreed.allCases[index]
                }
            }
        } catch {
            print("Error decoding JSON: \(error.localizedDescription)")
        }
    }.resume()
}
```

Рисунок 3.16 – Індикатор обробки зображення

Код обробляє запит /predict_image, видає помилку, якщо сервер надіслав некоректний результат, масив передбачень порожній, у запиті відсутній параметр image_id, або, найголовніше, жодної кішки не було ідентифіковано. Якщо все добре з фотографією, додаток візьме найвірогідніше передбачення та обере відповідну породу зі списку. Мобільний додаток стискає зображення, що призводить до пришвидшення запиту. Результат успішної ідентифікації наведено на рисунку 3.17.

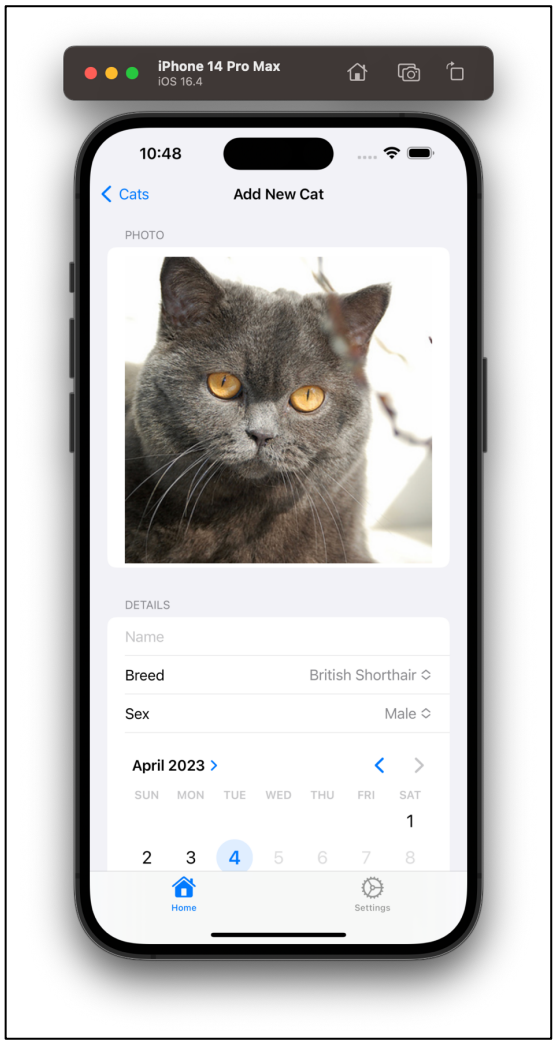


Рисунок 3.17 – Результат успішної ідентифікації зображення

Мобільний додаток також має можливість виведення помилки у разі, якщо обрана фотографія не містить жодної кішки (рисунок 3.18). Серверна частина робить відповідну перевірку та передає її мобільному додатку у

вигляді нульової вірогідності `[("No cat detected", 0)]`. У такому випадку порода не буде обрана, та, відповідно, кнопка збереження залишиться неактивною, і користувач не зможе зберегти неправильний запис у систему.

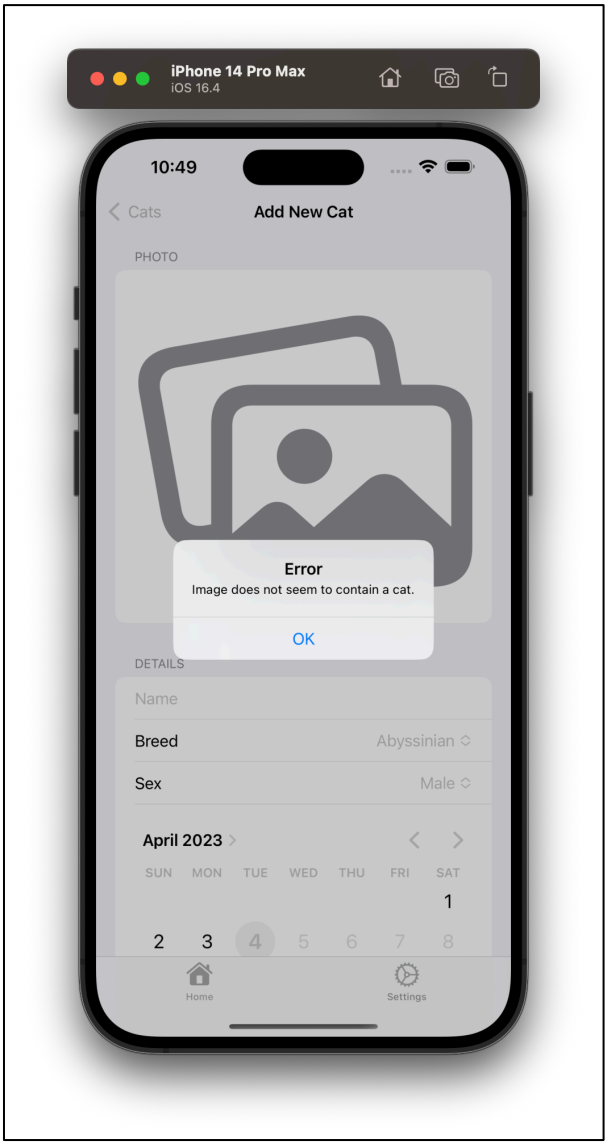


Рисунок 3.18– Результат успішної ідентифікації зображення

Після введення усіх даних про тварину, а саме: фотографії, імені, статі та дати народження британський короткошерстий кіт був успішно доданий у систему. Оновлений список можна побачити на рисунку 3.19. Після цього електронний паспорт тварини можна вважати зареєстрованим.

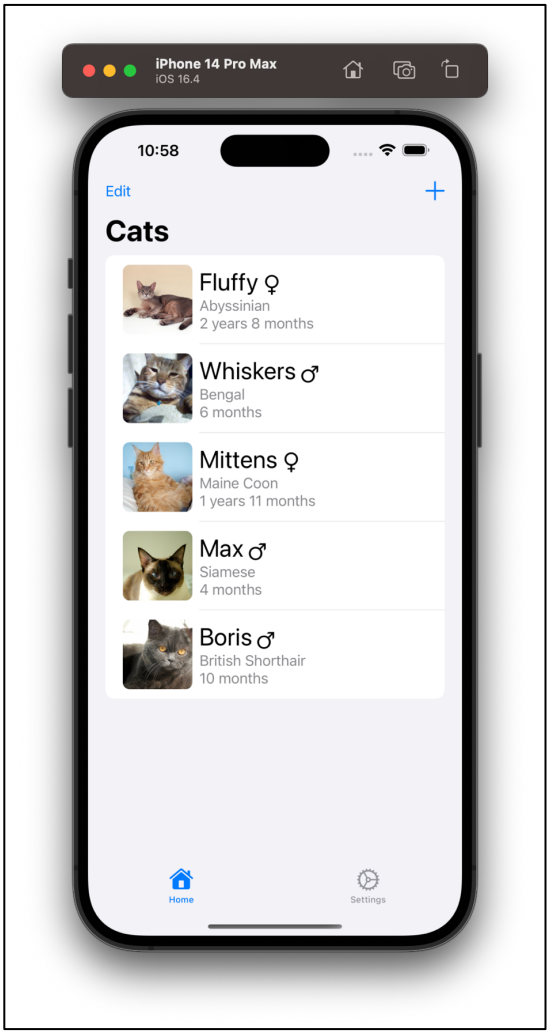


Рисунок 3.19 – Оновлений список тварин

Підсумовуючи, було успішно розроблено комплексну систему розпізнавання порід котів, яка інтегрує модель розпізнавання порід, серверну частину та зручний мобільний додаток. Ця система дозволяє користувачам легко ідентифікувати породи котів, просто завантажуючи фото. Серверна частина дозволяє взаємодіяти мобільному додатку із моделлю, а також надає доступ до неї клієнтам програмної системи. Комбінація цих компонентів надає надійне та стабільне рішення для визначення породи котів у програмній системі обліку домашніх тварин.

4 ОПТИМІЗАЦІЯ МОДЕЛІ РОЗПІЗНАВАННЯ

Після розробки моделі, її коду, серверної та мобільної частин, необхідно оптимізувати систему, щоб покращити точність прогнозування порід котів і забезпечити вищі відсотки правильних передбачень.

Щоб досягти кращих результатів, ми маємо внести наступні зміни:

– кількість епох. Кількість разів, коли весь набір даних передається через модель під час процесу навчання. Збільшуючи кількість епох, ми можемо дати моделі більше можливостей навчитися від даних та налаштувати її параметри. Однак важливо знайти рівновагу, оскільки занадто велика кількість епох може призвести до перенавчання, а занадто мала кількість епох може призвести до недонавчання;

– діапазон обертання. Техніки збільшення даних, такі як налаштування діапазону обертання вхідних зображень, можуть допомогти моделі краще узагальнювати, навчаючи її на різноманітних варіаціях даних. Експериментуючи з різними діапазонами обертання, ми можемо забезпечити, щоб модель була достатньо стійкою для розпізнавання порід котів на зображеннях з різними орієнтаціями, що призводить до покращення точності прогнозування;

– кількість шарів для тонкого налаштування. Тонке налаштування передбачає вибіркове навчання конкретних шарів передтренованої моделі, щоб адаптувати її до виконання конкретного завдання. Змінюючи кількість шарів, які ми точно налаштовуємо, ми можемо оптимізувати рівновагу між використанням знань передтренованої моделі та налаштуванням моделі для нашого конкретного випадку використання. Тонке налаштування більшої кількості шарів може потенційно покращити роботу моделі на нашому наборі даних, але також збільшити ризик перенавчання.

Ретельно налаштовуючи ці параметри, ми прагнемо оптимізувати нашу систему розпізнавання порід котів, покращуючи її точність прогнозування та гарантуючи надійні та точні результати для користувачів.

Під час вибору навчального та валідаційного наборів даних важливо забезпечити репрезентативність та різноманітність зображень, що дозволить моделі навчитися ідентифікувати породи тварин з різних кутів, світлових умов та композицій. Для навчання використаємо обраний раніше Oxford-IIIT Pet Dataset.

Для тестування використаємо фотографії з Instagram, оскільки ця соціальна мережа може надати унікальні зображення, які є новими порівняно з навчальним набором. Це гарантує, що навчальний та перевірочний набори не перетнуться. Instagram дозволяє легко шукати породу котів, використовуючи хештеги.

У ході тестування було обрано 4 зображення (рисунок 4.1), які виявилися найпроблемнішими для розпізнавання моделлю:

– licks.png. Кіт дивиться вліво, язик закриває праву частину обличчя, видно вуха (частково) та передні лапи;

– licks_2.png. Кіт дивиться у камеру, язик закриває проміжок між губою та носом, повністю видно обличчя, яке переходить у тулуб (знімок – портретний);

– paws.png. Кіт лежить на лівому боці, очі закриті, витягує лапи у напрямку камери, видно половину тулуба;

– young_boy_stretching.png. Кошеня витягується, тварина повністю у кадрі, але стегна закривають задні лапи та хвіст, очі спрямовані у камеру.

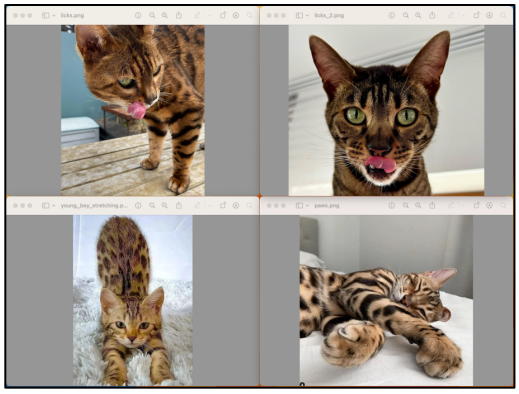


Рисунок 4.1 – Зображення для тестування

Всі подальші порівняння та дослідження ґрунтуватимуться на перелічених вище фотографіях.

Під час першого експерименту було змінено кількість епох та діапазон обертання. Ви можете бачити, що використання діапазону обертання призвело до гірших результатів у більшості випадків. 90 епох з 0 діапазоном обертання показали найкращі результати за сумою прогнозів, але фото «young_boy_stretching.png» показало дуже низький результат прогнозування. Результати експерименту наведені на таблиці 4.1.

Таблиця 4.1 – Змінення кількості епох та діапазону обертання

Епохи	**Діапазон обертання**	**licks.png**	**licks_2.png**	**paws.png**	**young_boy_stretching.png**	**Сума передбачень**
20	0	0,2468581	0,46475583	0,50478816	0,0012653596	0,0012653596
	20	0,037327435	0,5265063	0,89545536	0	1,459289095
	40	0,7512714	0,116529614	0,30538774	0	1,173188754
	90	0,54677147	0,09970146	0,84583753	0	1,49231046
40	0	0,104850814	0,7268813	0,16578248	0	0,997514594
	20	0,72506565	0,5364206	0,9851682	0,13368464	2,24665445
	40	0,28071627	0,001479856	0,3997336	0,276307	0,681929726
	90	0,9930642	0,019279541	0,6081584	0,0024454203	1,620502141
90	0	0,9881932	0,98907566	0,99566287	0,09193284	2,97293173
	20	0,18356118	0,03234846	0,9981996	0,08763256	1,21410924
	40	0,8249214	0,17083511	0,14421256	0,0036268656	1,13996907
	90	0,99901116	0,0050652316	0,9498225	0	1,953898892

На рисунку 4.2 наведено графік, який виводить передбачення для кожного зображення відносно кількості епох. Для кожного діапазону обертання було обрано свій колір: 0 – червоний, 20 – блакитний, 40 – зелений, 90 – жовтий.

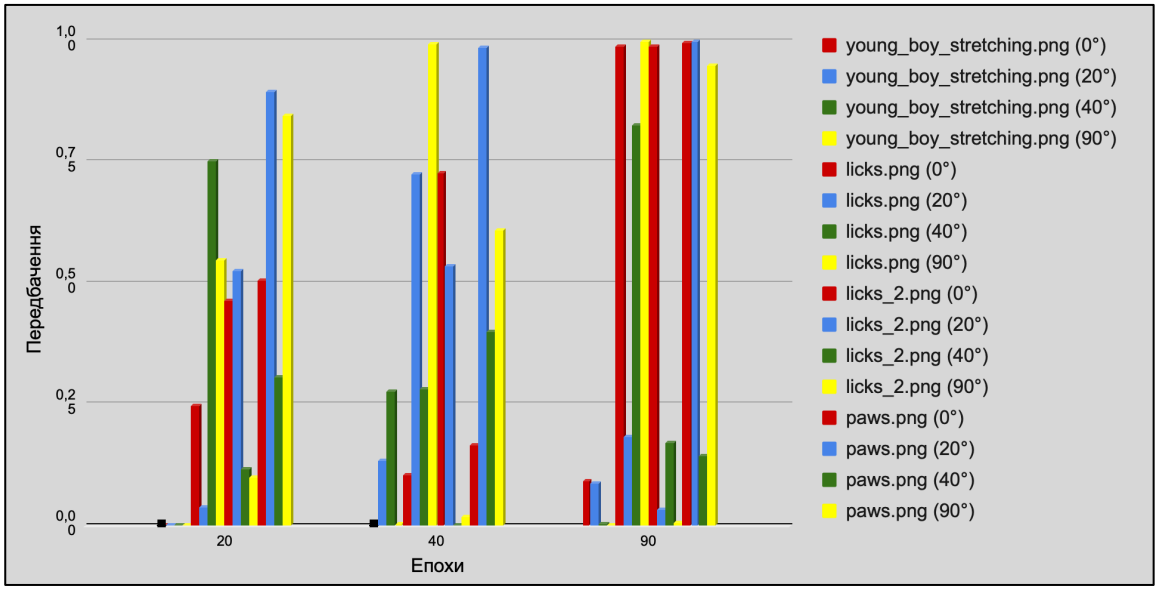


Рисунок 4.2 – Графік змінення кількості епох та діапазону обертання

Продовжуючи наш експеримент з 90 епохами, на наступному кроці ми будемо поступово збільшувати кількість шарів для тонкого налаштування поки не досягнемо бажаного результату. Тонке налаштування дозволяє моделі отримати більше знань, необхідних для нашої задачі розпізнавання порід котів.

Починаючи з 10 шарів для тонкого налаштування, ми спостерігатимемо за змінами в точності класифікації моделі на валідаційному наборі даних. З кожним кроком, коли додаємо додаткові шари, ми слідкуємо за тим, як це впливає на точність моделі та розпізнавання порід котів. Результати тренувань наведені на таблиці 4.2.

Таблиця 4.2 – Змінення кількості шарів тонкого налаштування

Кількість шарів тонкого налаштування	**young_boy_stretching.png**	**licks_2.png**	**licks.png**	**paws.png**	**Сума передбачень**
10	0	0,9495698	0,9985551	0,17236744	2,12049234
20	0	0,0006462419	0,07196111	0,626319	0,6989263519
30	0,019149607	1	0,99662966	0,9999877	3,015766967
40	0	0,06370371	0,9993318	0,99840134	2,06143685
50	0,9913385	0,9999951	0,99996936	1	3,99130296

На рисунку 4.3 наведено дані таблиці у візуальному вигляді.

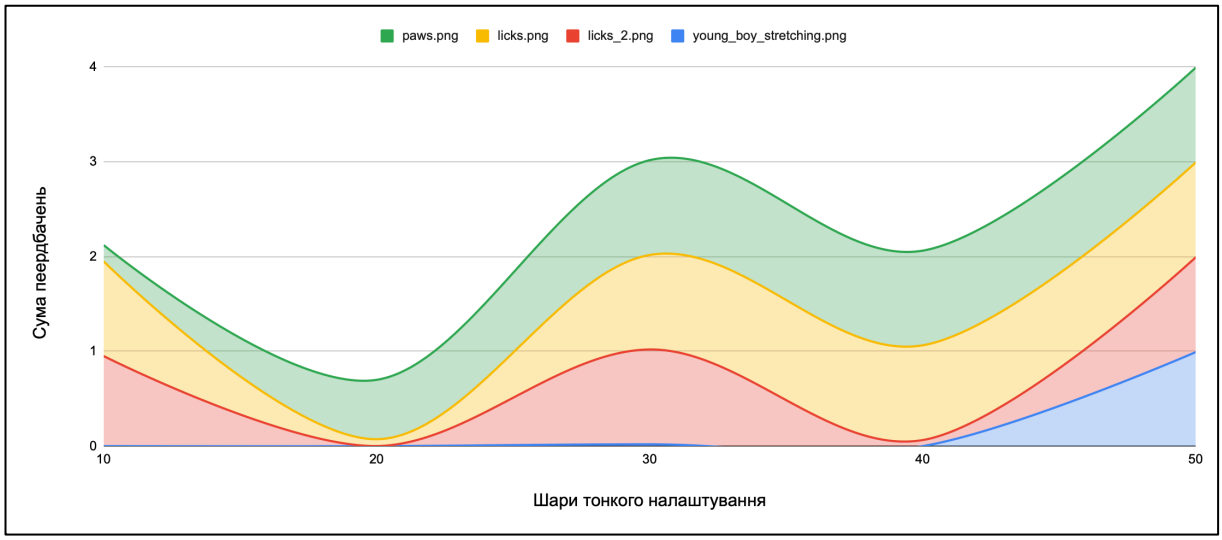


Рисунок 4.3 – Графік змінення кількості шарів тонкого налаштування

Ви можете побачити на графіку, що збільшення кількості шарів для тонкого налаштування до 50 дозволило моделі розпізнати проблемне зображення «young_boy_stretching.png». Сума передбачень практично дорівнюється 4, що є ідеальним результатом.

Підбір оптимальних параметрів допоміг нам досягти кращих результатів у системі розпізнавання порід котів. Експерименти з різними гіперпараметрами, такими як кількість епох, діапазон обертання та кількість шарів для тонкого налаштування, дозволили нам знайти оптимальний баланс між точністю та стабільністю моделі. Завдяки цим оптимізаціям, наша модель може забезпечити точні та надійні результати розпізнавання зображень.

ВИСНОВКИ

Після ретельного аналізу предметної галузі та вивчення актуальності сервісу з розпізнавання порід котів ми провели дослідження, щоб забезпечити найкращі результати для нашої системи. Враховуючи конкурентів, наша система розроблена відповідно до потреб різних груп користувачів, із використанням передових технологій класифікації зображень.

Ми обрали згорткові нейронні мережі (CNN) як основу нашої моделі, через їх високу точність та ефективність у класифікації зображень. В процесі розробки ми використали цей підхід, що дозволило нам скоротити час навчання та збільшити точність.

Серверна частина розроблена з використанням мови програмування Python та фреймворку Django, що гарантує надійність та швидкість роботи системи, дозволяє користуватися моделлю за допомогою REST API. Мобільний додаток розроблено на мові програмування Swift, що забезпечує високу продуктивність та зрозумілий користувацький інтерфейс для користувачів iOS.

Для тренування нашої моделі ми використали Oxford-IIIT Pet Dataset та фотографії з Instagram, що допомогло нам створити репрезентативний набір даних для навчання та валідації. В ході оптимізації моделі ми провели експерименти з різними гіперпараметрами, такими як кількість епох, діапазон обертання та кількість слоїв для тонкого налаштування.

Результати досліджень показали, що підбір оптимальних параметрів допоміг досягти більшої точності та стабільності нашої моделі. Зокрема, збільшення кількості шарів для тонкого налаштування допомогло покращити точність розпізнавання на складних зображеннях. Таким чином, наша система може забезпечити високі результати передбачень порід котів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Atienza R. Advanced Deep Learning with TensorFlow 2 and Keras. Apply DL, GANs, VAEs, Deep RL, Unsupervised Learning, Object Detection and Segmentation, and More, 2nd Edition. UK : Packt Publishing, 2020. 512 p.

2. Azunre P. Transfer Learning for Natural Language Processing. US : Manning, 2021. 272 p.

3. Chen S., Pedrycz W. Development and Analysis of Deep Learning Architectures. CH : Springer International Publishing, 2019. 292 p.

4. Grummitt C. IOS Development with Swift. US : Manning, 2017. 568 p.

5. Gulli A., Pal S. Deep Learning with Keras. UK : Packt Publishing, 2017. 318 p.

6. Image data preprocessing. *Keras* : веб-сайт. URL: https://keras.io/api/preprocessing/image/ (дата звернення: 14.04.2023).

7. Interesting Pet Statistics. *MaCorr Research Blog* : веб-сайт. URL: https://www.macorr.com/blog/?p=310 (дата звернення: 05.04.2023).

8. Li K., Wang L., Liao L., Li M., Xiong N. Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery. US : Springer International Publishing, 2023. 1508 p.

9. Murphy K. Machine learning: a probabilistic perspective. US : MIT Press, 2012. 1104 p.

10. Nakamoto P. Neural Networks and Deep Learning. US : CreateSpace Independent Publishing Platform, 2018. 148 p.

11. Oxford-IIIT Pet Dataset. *TensorFlow* : веб-сайт. URL: https://www.tensorflow.org/datasets/catalog/oxford_iiit_pet (дата звернення: 12.04.2023).

12. Richardson L., Amundsen M., Ruby S. RESTful Web APIs. Services for a Changing World. CA : O`Reilly, 2013. 406 p.

13. Sandler M., Howard A. MobileNetV2: The Next Generation of On-Device Computer Vision Networks. *Google Research* : веб-сайт. URL: https://ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html (дата звернення: 21.04.2023).

14. Sewak M., Karim R., Pujari P. Practical Convolutional Neural Networks. Implement Advanced Deep Learning Models Using Python. UK : Packt Publishing, 2018. 218 p.

15. Shaw B., Badhwar S., Bird A., Guest C. Web Development with Django. Learn to Build Modern Web Applications with a Python-based Framework. UK : Packt Publishing, 2021. 826 p.

16. Szeliski R. Computer Vision. Algorithms and Applications. US : Springer International Publishing, 2022. 925 p.

17. UK cat statistics. UK Cat Statistics. *Tippaws* : веб-сайт. URL: https://www.tippaws.com/blogs/news/uk-cat-statistics (дата звернення: 05.04.2023).

18. User groups & scenarios. *Medium* : веб-сайт. URL: https://medium.com/@Kristijan197/hci-method-user-groups-and-scenarios-8904ad706e93 (дата звернення: 06.04.2023).

19. Yang Q., Zhang Y., Dai W. Transfer Learning. US : Cambridge University Press, 2020. 390 p.

20. Швець О. Патерни проектування. Кам'янець-Подільський : Refactoring.Guru, 2018. 393 с.

ДОДАТОК А

Слайди презентації

Рисунок А.1 – Титульний аркуш

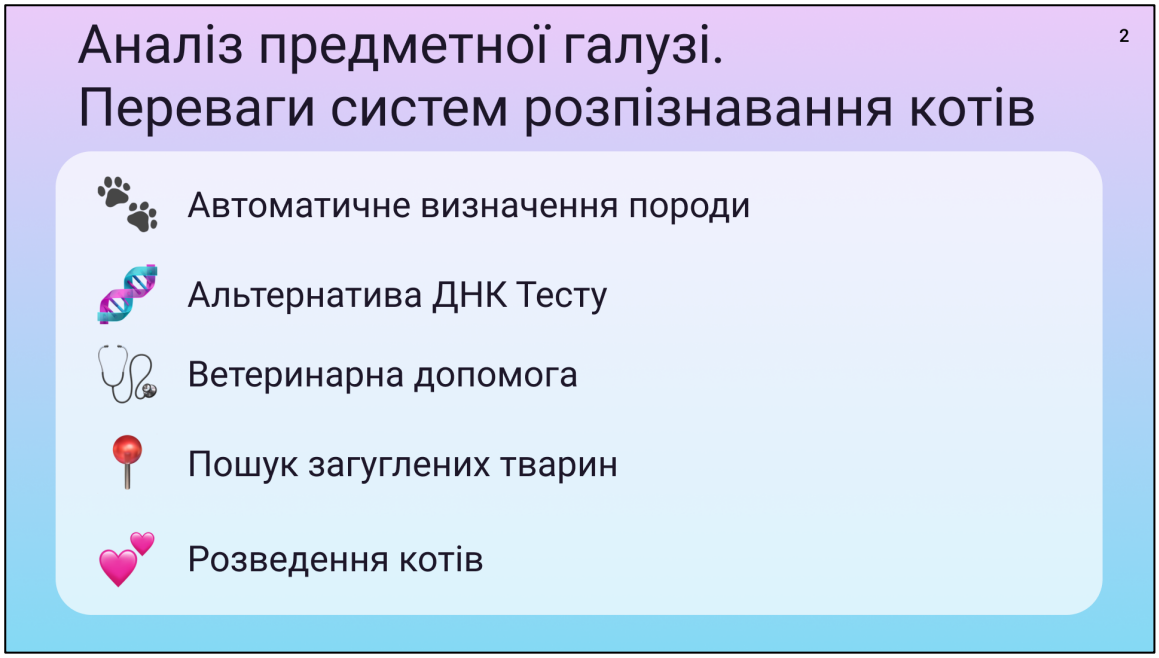


Рисунок А.2 – Аналіз предметної галузі. Переваги систем розпізнавання котів

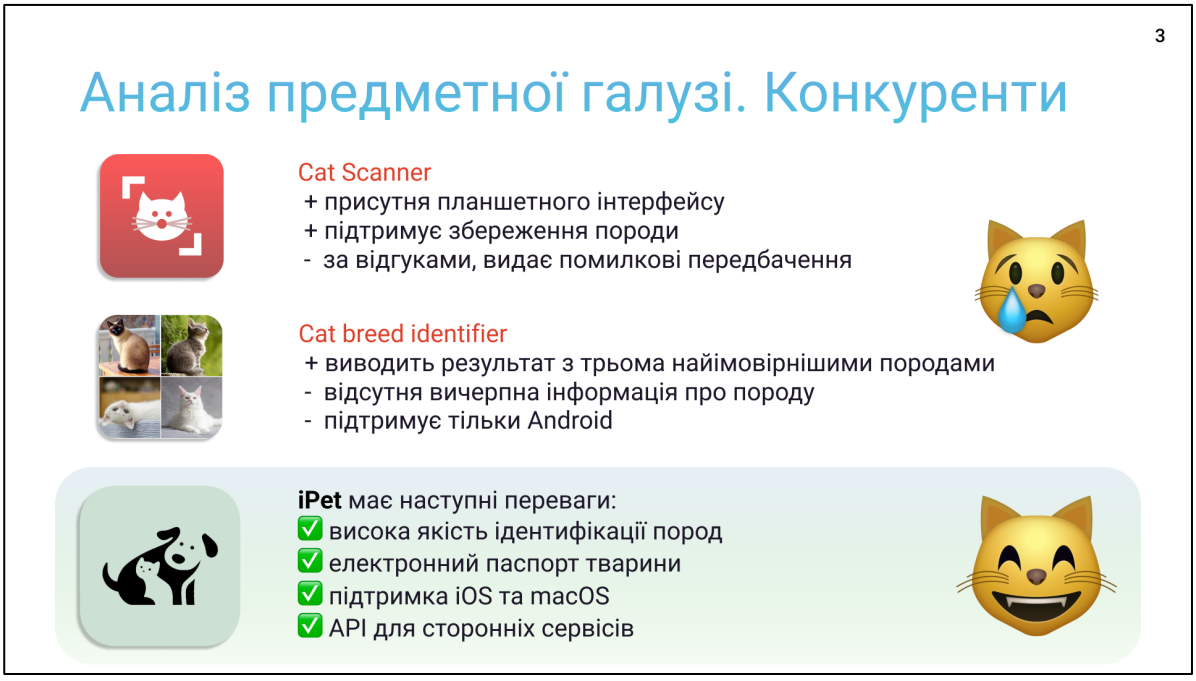


Рисунок А.3 – Аналіз предметної галузі. Конкуренти

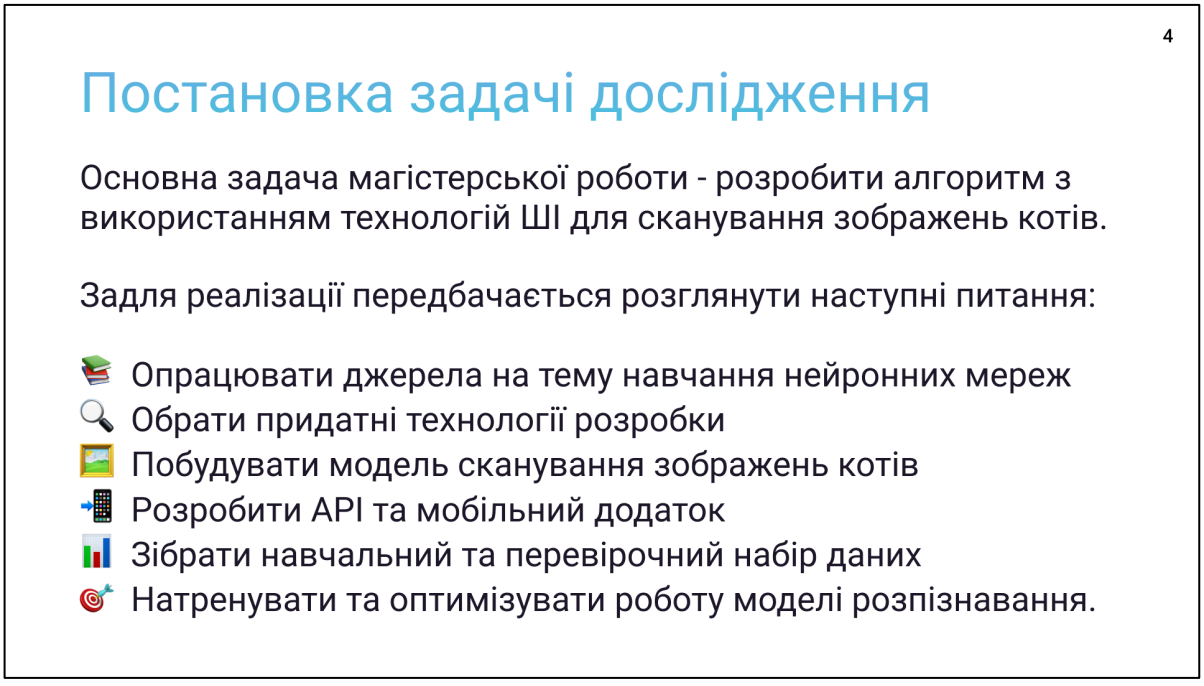


Рисунок А.4 – Постановка задачі дослідження

5

Вибір технології класифікації зображень

Згорткові нейронні мережі (Convolution Neural Networks, CNN) є найкращою технологією для класифікації порід котів:

Вилучення функцій. Автоматичне витягування ознак із зображень без необхідності ручного проєктування функцій.

Просторові зв'язки. Фіксація ознак допомагає класифікувати розмір і форму вух, очей і тіла, які суттєво відрізняються між породами.

Глибоке навчання. Є типом глибокого навчання, що дозволяє вивчати складні представлення вхідних даних за допомогою кількох рівнів обробки.

Великі набори даних. Можуть навчатися на великій кількості даних для досягнення високої точності. Наприклад, Oxford-IIIT Pet Dataset.

Рисунок А.5 – Вибір технології класифікації зображень

6

Архітектура CNN

Згортка. Основна операція видобутку локальних особливостей з вхідних даних. Під час процесу згортки до вхідних даних застосовується набір фільтрів.

Пулінг. Використовується для зменшення просторових розмірів карт особливостей, відбираючи інформацію та зберігаючи її основні характеристики.

Повністю з'єднаний шар. Служить мостом між видобутком особливостей та класифікацією компонентів. Приймає вихід з останнього шару пулінгу та перетворює його на одновимірний вектор, що подається на традиційну пряму нейронну мережу.

Вихідний шар. Є останнім та відповідає за виробництво кінцевих результатів класифікації. Зазвичай складається з набору нейронів рівних кількості класів у задачі класифікації.

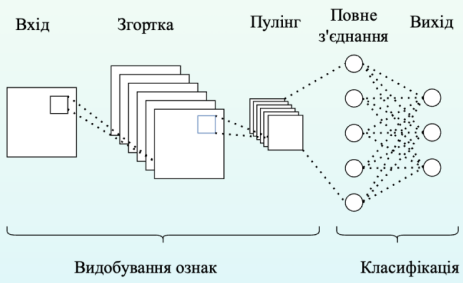


Схема базової згорткової нейронної мережі

Рисунок А.6 – Архітектура Convolution Neural Networks

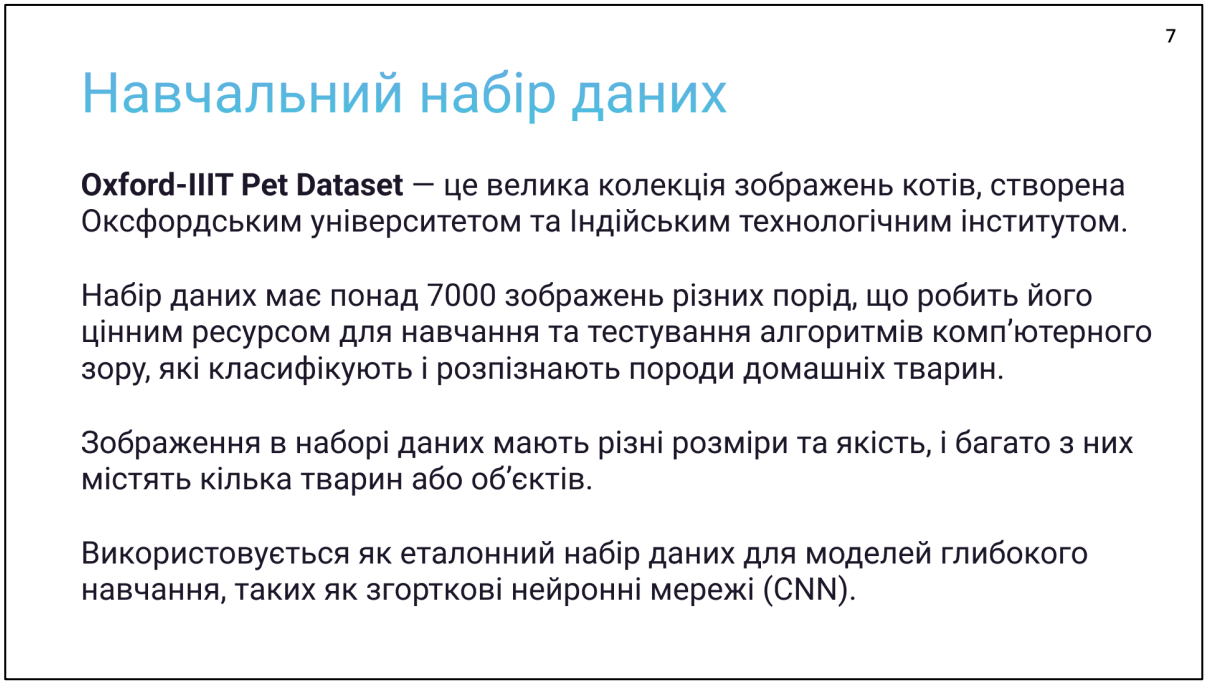


Рисунок А.7 – Навчальний набір даних

Рисунок А.8 – Розробка моделі

9

Серверна частина

API запити:

GET: /supported_breeds - повертає масив назв класів порід, на яких навчалась модель зі сканування зображень.

POST: /predict_image - повертає масив з топ-3 передбаченнями порід. Присвоює тимчасовому файлу зображення унікальний ID, що дозволяє звертатись до нього у наступному запиті.

POST: /correct_result - метод корекції передбачення. Переміщає зображення кішки, порода якої була видана некоректно, для перевірки фахівцями сервісу.

Поділяється на **рівні**:
- сервісів (settings.py)
- бізнес-логіки (views.py)
- доступу до даних (models.py)

Основні переваги **django** :
- ORM
- міграції бази даних
- аутентифікація користувача
- швидке розгортання
- форми

Рисунок А.9 – Серверна частина. Опис

10

Серверна частина. Результат запиту

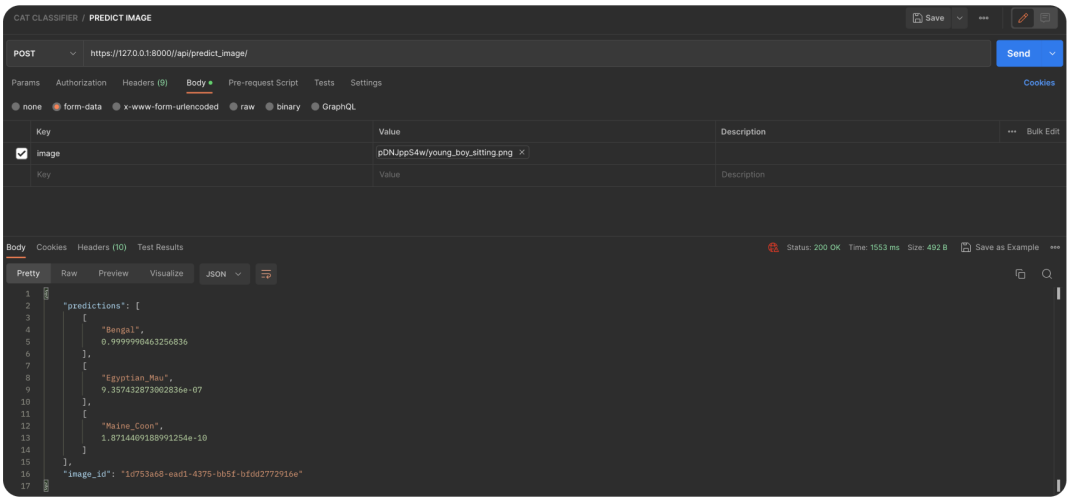


Рисунок А.10 – Серверна частина. Результат запиту

Рисунок А.11 – Мобільний застосунок. Опис

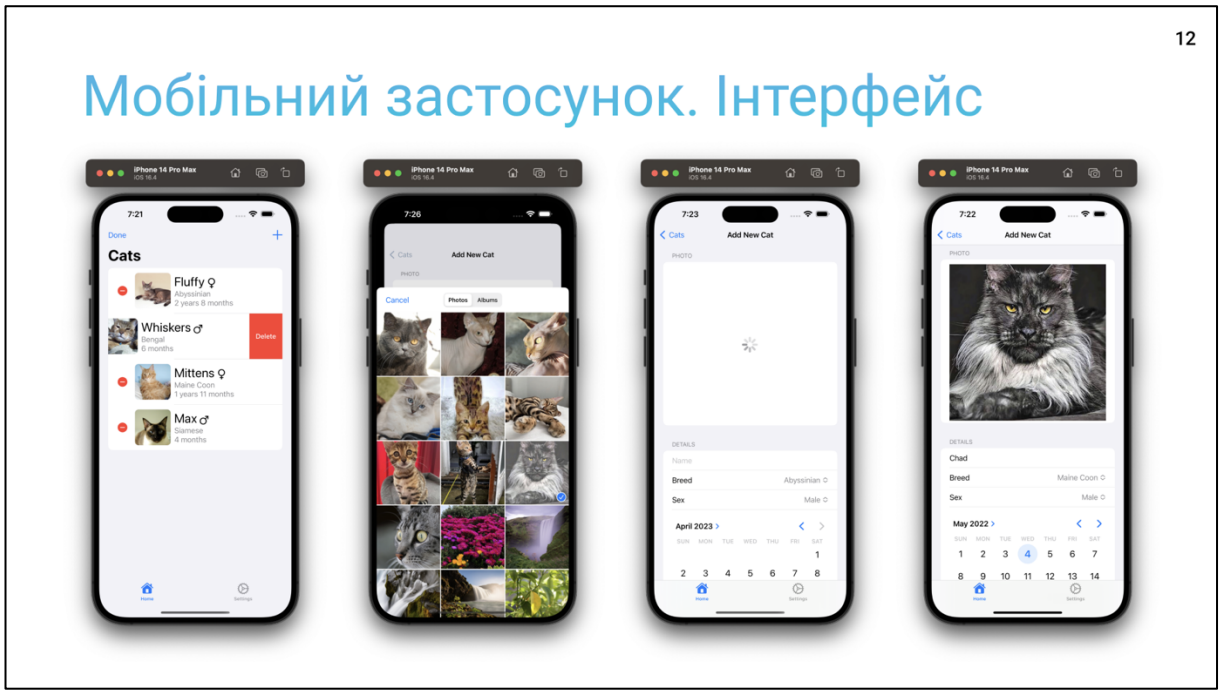


Рисунок А.12 – Мобільний застосунок. Інтерфейс

13

Оптимізація моделі

У ході тестування було обрано 4 зображення, які виявилися найпроблемнішими для розпізнавання:

licks.png. Кіт дивиться вліво, язик закриває праву частину обличчя, частково видно вуха та передні лапи.

licks_2.png. Кіт дивиться у камеру, язик закриває проміжок між губою та носом, повністю видно обличчя, яке переходить у тулуб.

paws.png. Кіт лежить на лівому боці, очі закриті, витягує лапи до камери, видно половину тулуба.

young_boy_stretching.png. Кошеня витягується, тварина повністю у кадрі, але стегна закривають задні лапи та хвіст, очі спрямовані у камеру.

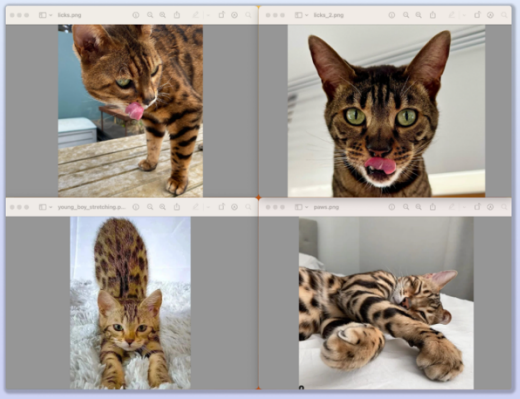


Рисунок А.13 – Оптимізація моделі. Проблемні зображення

14

Оптимізація моделі

Під час першого експерименту було змінено:

- **кількість епох.** Кількість разів, коли весь набір даних передається через модель під час процесу навчання.

- **діапазон обертання.** Розгортання зображення від -x° до x°. Забезпечує стійкість роботи з різними орієнтаціями зображень.

Можна побачити, що застосування діапазону обертання призвело до гірших результатів.

90 епох з 0 діапазоном обертання показали найкращі результати за сумою прогнозів, але фото *young_boy_stretching.png* залишилось з низьким результатом передбачення.

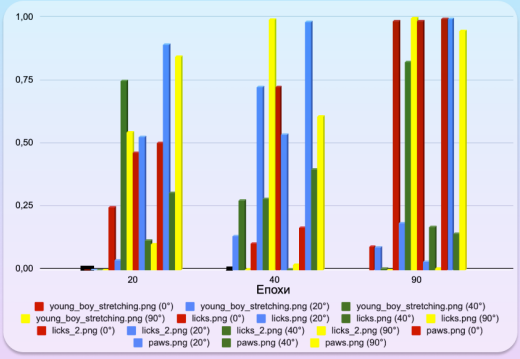


Рисунок А.14 – Оптимізація моделі. Перший експеримент

15

Оптимізація моделі

Продовжуючи експеримент з 90 епохами, поступово збільшимо кількість шарів для тонкого налаштування, поки не досягнемо бажаного результату.

Тонке налаштування (Fine-tuning) дозволяє моделі отримати більше знань, необхідних для нашої задачі розпізнавання порід котів.

Збільшення шарів для тонкого налаштування до 50 **дозволило** моделі **розпізнати проблемне зображення**.

Сума передбачень практично дорівнюється 4, що є ідеальним результатом.

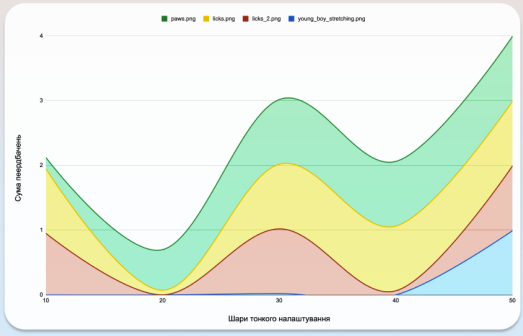


Рисунок А.15 – Оптимізація моделі. Другий експеримент

16

Висновки

У результаті написання кваліфікаційної роботи було:

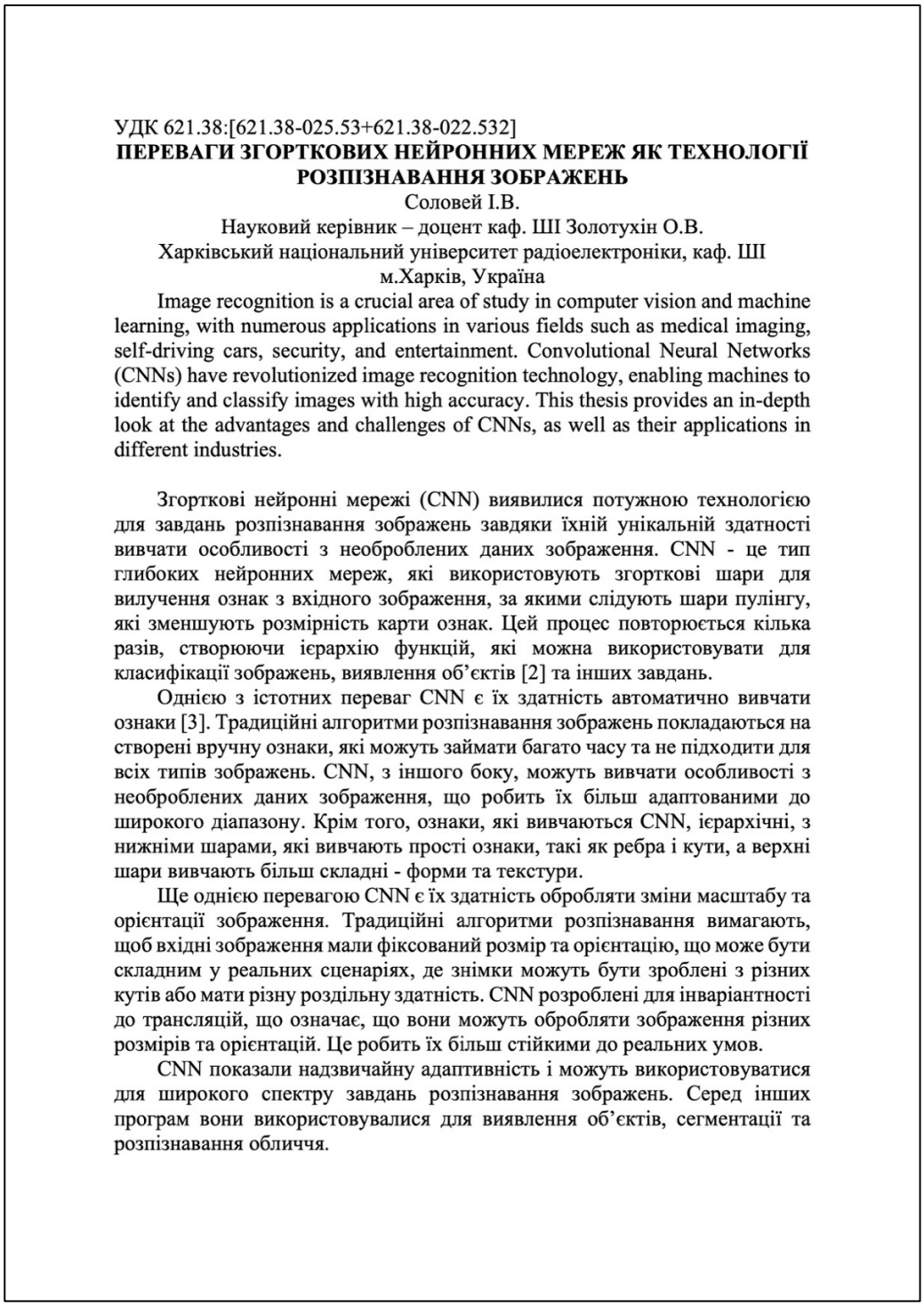
- проаналізовано предметну галузь;
- проведено аналіз конкурентів;
- сформовано вимоги;
- спроєктовано та розроблено програмну систему: модель, серверну та мобільну частину;
- підібрано оптимальні параметри та оптимізовано роботу моделі.

Результатом є функціонуюча модель розпізнавання порід котів, інтегрована у мобільний додаток.

Рисунок А.16 – Висновки

ДОДАТОК Б

Тези доповідей до XXVII Міжнародного молодіжного форуму
«РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ»
(ХНУРЕ, Харків, 10-12 травня 2023 року)



УДК 621.38:[621.38-025.53+621.38-022.532]

ПЕРЕВАГИ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ЯК ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ

Соловей І.В.
Науковий керівник – доцент каф. ШІ Золотухін О.В.
Харківський національний університет радіоелектроніки, каф. ШІ
м.Харків, Україна

Image recognition is a crucial area of study in computer vision and machine learning, with numerous applications in various fields such as medical imaging, self-driving cars, security, and entertainment. Convolutional Neural Networks (CNNs) have revolutionized image recognition technology, enabling machines to identify and classify images with high accuracy. This thesis provides an in-depth look at the advantages and challenges of CNNs, as well as their applications in different industries.

Згорткові нейронні мережі (CNN) виявилися потужною технологією для завдань розпізнавання зображень завдяки їхній унікальній здатності вивчати особливості з необроблених даних зображення. CNN - це тип глибоких нейронних мереж, які використовують згорткові шари для вилучення ознак з вхідного зображення, за якими слідують шари пулінгу, які зменшують розмірність карти ознак. Цей процес повторюється кілька разів, створюючи ієрархію функцій, які можна використовувати для класифікації зображень, виявлення об'єктів [2] та інших завдань.

Однією з істотних переваг CNN є їх здатність автоматично вивчати ознаки [3]. Традиційні алгоритми розпізнавання зображень покладаються на створені вручну ознаки, які можуть займати багато часу та не підходити для всіх типів зображень. CNN, з іншого боку, можуть вивчати особливості з необроблених даних зображення, що робить їх більш адаптованими до широкого діапазону. Крім того, ознаки, які вивчаються CNN, ієрархічні, з нижніми шарами, які вивчають прості ознаки, такі як ребра і кути, а верхні шари вивчають більш складні - форми та текстури.

Ще однією перевагою CNN є їх здатність обробляти зміни масштабу та орієнтації зображення. Традиційні алгоритми розпізнавання вимагають, щоб вхідні зображення мали фіксований розмір та орієнтацію, що може бути складним у реальних сценаріях, де знімки можуть бути зроблені з різних кутів або мати різну роздільну здатність. CNN розроблені для інваріантності до трансляцій, що означає, що вони можуть обробляти зображення різних розмірів та орієнтацій. Це робить їх більш стійкими до реальних умов.

CNN показали надзвичайну адаптивність і можуть використовуватися для широкого спектру завдань розпізнавання зображень. Серед інших програм вони використовувалися для виявлення об'єктів, сегментації та розпізнавання обличчя.

Рисунок Б.1 – Тези доповідей. Перша сторінка

CNN можна навчати на великих наборах даних за допомогою сучасних графічних процесорів, що робить їх масштабованою технологією. Це дозволило дослідникам навчати CNN на великих наборах даних, таких як ImageNet, який містить понад 14 мільйонів зображень і 20 000 категорій, що призвело до значного прогресу у розпізнаванні зображень. Можливість навчання на великих наборах даних дозволила розробити високоточні моделі, які можуть розпізнавати зображення з надзвичайною точністю.

CNN мають потенціал розвинути різні сфери, від охорони здоров'я до транспорту. В охороні здоров'я CNN є корисною для аналізу медичних зображень [4], таких як рентгенівські знімки та магнітно-резонансна томографія, що сприяє точності діагностики та лікування. Технологію можна зустріти в безпілотних автомобілях для виявлення та класифікації об'єктів на дорозі [1]. Крім того, CNN можна використовувати в сільському господарстві для виявлення шкідників у культурах, що призводить до підвищення врожайності та зменшення використання пестицидів.

Незважаючи на вражаючу продуктивність CNN у задачах розпізнавання зображень, вони не позбавлені своїх обмежень. Однією з головних проблем є потреба у великих обсягах позначених даних для ефективного навчання моделей. Це може бути особливо складно в таких програмах, як аналіз медичних зображень, де отримання мічених даних може бути важким і трудомістким.

Підсумовуючи, CNN — це потужна технологія для розпізнавання зображень, яка пропонує кілька переваг перед традиційними алгоритмами розпізнавання зображень. Вони можуть вивчати особливості з необроблених даних зображення, обробляти варіації в масштабі та орієнтації, є адаптованими і масштабованими та мають потенціал для революції в різних сферах. Зі збільшенням наборів даних і потужних процесорів використання CNN стане більш поширеним у найближчі роки.

Список використаних джерел:

1. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., ... & Zhang, X. (2016). End to end learning for self-driving cars. https://doi.org/10.48550/arXiv.1604.07316.

[2] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

[3] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958.

[4] Wang, S., Wang, R., Xu, X., Yang, S., Zhang, B., & Shao, L. (2021). Medical image analysis with convolutional neural networks. IEEE Transactions on Medical Imaging, 40(4), 998-1014.

Рисунок Б.2 – Тези доповідей. Друга сторінка

ДОДАТОК В

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	74 с.
	Інші документи	
2.	Презентаційні матеріали	16 плакатів

.	.	Прізвище та ініціали.	Підп.	Дата	
Розробив		Соловей І.В.			Розпізнавання порід котів у програмній системі обліку домашніх тварин
Перевірив		Золотухін О.В.			
Н.контр.		Малєєва І.А.			
Затв.		Філатов В.О.			

Шифр групи	Код напр./спец.
СШІм-21-2	122
ХНУРЕ кафедра ШІ	