МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Кафедра прикладної математики

Дипломна робота

Визначення найпопулярніших видів квітів (за допомогою соціальної та нейронних мереж)

Виконала: студентка групи ПМП-42
спеціальності
113 - прикладна математика

_____ Трей А.О. _____
(прізвище та ініціали)

Керівник ас. Борисюк Я.Є. _____
(прізвище та ініціали)

Рецензент_____
(прізвище та ініціали)

Львів - 2023

Анотація

Для цієї дипломної роботи було створено програму, яка завантажує зображення з соціальної мережі, сканує їх на наявність квітів, класифікує та підраховує за допомогою глибокої нейронної мережі. За допомогою цього інструменту ви можете знаходити квіти на фотографіях, ідентифікувати їх види та виконувати обчислення. Крім того, програма допомагає визначити приблизну популярність певних квітів у соціальних мережах.

ЗМІСТ

Вступ

За останні кілька десятиліть розвиток цифрових технологій значно вплинув на наше життя. Для пересічної людини наявність «смартфона» та ноутбука тепер є звичним явищем. Ці пристрої дозволяють нам носити з собою все необхідне, спілкуватися з людьми з усіх куточків світу та отримувати доступ до документів, які раніше займали місце на наших комп'ютерах або навіть у фізичних бібліотеках.

Сьогодні неможливо уявити життя без інтернет-спілкування. Раніше нам доводилося чекати тижнями, щоб отримати листа поштою. Однак сьогодні ми навіть не розплющуємо очей, доки не прочитаємо наші повідомлення в Messenger, WhatsApp і Telegram та не перевіримо стрічки новин у Facebook та Instagram. У найвідоміших соціальних мережах світу мільйони користувачів.

За останні роки багато країн зазнали цифрової трансформації. Завдання, які потребували залучення кількох людей, зараз можливо виконати всього кількома кліками миші на комп'ютері чи смартфоні. Існує багато послуг, які можна отримати, не стоячи в довгих чергах. Ви можете записатися онлайн і прийти в призначений час. А якщо вам потрібно надіслати документ, підписаний вашим електронним підписом, ви можете зробити це за допомогою електронної пошти або сучасної кур'єрської доставки. Багато людей сприймають «розумні» будинки та автомобілі з «автопілотом» як реальність, а не просто мрію — те, що тепер можливо у фізичній формі.

Застосування штучного інтелекту та нейронних мереж у повсякденному житті є окремою сферою впровадження технологій. Вже неможливо уявити собі життя без пошукових систем, чат-ботів, пропозицій щодо продуктів, які автоматично створюються на веб-сайтах онлайн-

магазинів, і списків відтворення музики. У всіх цих прикладах використовується компонент штучного інтелекту.

Останнім часом значно зросли темпи використання «розумних» технологій як у бізнесі, так і в повсякденному житті. Крім того, цей цифровий перехід мав справді позитивні наслідки. Згідно з дослідженнями, понад 75% компаній, які використовують цифрову трансформацію у своїй діяльності, успішніше впроваджують нові товари чи послуги чи впроваджують стартапи; 62% швидше «завойовують» ринок, у тому числі за допомогою цифрового маркетингу та персоналізації; і на 60% краще контролюють витрати в результаті менш дорогих технологічних експериментів[1] .

Флористика – одна з галузей, де цифровізація може знайти свою нішу. Розрахунок актуальності певних квітів займає достатньо багато часу і є неточним.

Метою дипломної роботи є розробка та створення системи нейромережевого виявлення квітів на фотографіях, зібраних за певний період часу із соціальної мережі Instagram, за допомогою згорткової нейронної мережі.

Метою дослідження є створення методу організації нейромережевого детектування об’єктів на зображеннях за допомогою згорткової моделі нейронної мережі.

Вивчення методів навчання нейронної мережі та реалізації згорткової архітектури нейронної мережі є центром цього дослідження.

Практичне значення цього інструменту полягає в його здатності досліджувати та розраховувати популярність від самого початку масового поширення.

Термінологія

Штучний інтелект (ШІ) – це галузь комп'ютерних наук, що займається розробкою алгоритмів та систем, які можуть моделювати імітацію інтелекту та когнітивних функцій людини. ШІ має на меті створення комп'ютерних систем, які здатні розуміти, мислити, вирішувати проблеми, навчатись та адаптуватись до змінного середовища.

Неформалізовані задачі – задачі, які не мають чіткого формалізованого визначення або структури. Вони не піддаються аналізу за допомогою традиційних алгоритмів або методів.

Метод символічних суджень – підхід до розв'язання задач штучного інтелекту, який використовує символічне представлення знань та логічні правила для вирішення завдань розуміння та маніпуляції знаннями.

Machine Learning (Машинне навчання, МН, ML) – галузь штучного інтелекту, що досліджує розвиток алгоритмів та методів, які дозволяють комп'ютерам навчатися на основі даних і виконувати завдання без явного програмування.

Регресія – тип задачі в машинному навчанні, де цільова змінна є неперервною величиною, і завдання полягає у прогнозуванні числового значення на основі вхідних даних.

Кластеризація – метод машинного навчання, який групує схожі об'єкти разом у кластери на основі схожості їх властивостей або характеристик.

Екстраполяція – процес прогнозування значень або властивостей за межами доступного діапазону даних, використовуючи існуючі спостереження або шаблони.

Supervised Learning (Навчання з учителем) – тип машинного навчання, де модель навчається на основі пари вхідних даних та очікуваного вихідного значення для кожного прикладу.

Unsupervised Learning (Навчання без учителя) – тип машинного навчання, де модель навчається на основі невідмічених даних без наявності очікуваних вихідних значень. Модель самостійно виявляє приховані закономірності або структуру даних.

Deep Learning (Глибоке навчання, ГН, DL) – підгалузь машинного навчання, що використовує нейронні мережі з великою кількістю шарів для вирішення завдань навчання і розпізнавання.

Автокодери – тип нейронних мереж, що використовуються для зменшення розмірності даних шляхом стиснення вхідних даних в складний код та їх відновлення назад.

Рекурентні нейронні мережі – тип нейронних мереж, що мають зв'язки зі зворотними зв'язками, що дозволяють передавати інформацію назад у часі. Вони ефективні для моделювання послідовних даних і зв'язаних залежностей.

Навчання з підкріпленням – тип машинного навчання, де агент (модель) навчається взаємодіяти з оточенням, отримуючи винагороду або покарання за свої дії. Агент вчиться приймати рішення, максимізуючи загальну винагороду на протязі часу.

API (Application Programming Interface) – це набір правил та протоколів, які дозволяють різним програмним компонентам взаємодіяти між собою.

Згорткові нейронні мережі (CNN) – тип нейронних мереж, що ефективно застосовуються до обробки зображень. Вони використовують

шари згортки для виявлення локальних функцій і шари агрегації для зменшення розмірності.

Повністю зв’язаний шар (Fully Connected Layer) – шар нейронної мережі, в якому кожен нейрон підключений до всіх нейронів попереднього шару. Цей шар здійснює зв'язок між різними регіонами мережі.

Шари згортки (convolution layers) – шари нейронної мережі, що використовуються в згорткових нейронних мережах для виявлення локальних функцій або шаблонів у вхідних даних.

Шари агрегації (шари підвибірки, pooling) – шари нейронної мережі, які зменшують розмірність даних шляхом об'єднання значень в підвибірках. Вони допомагають зменшити кількість параметрів та ресурсів, що обробляються.

Повністю зв’язані шари (fully-connected) – шари нейронної мережі, в яких кожен нейрон пов'язаний з кожним нейроном попереднього шару. Ці шари використовуються для здійснення остаточних прогнозів або класифікації.

Шар відсіву (dropout) – техніка, що використовується в нейронних мережах для випадкового відключення нейронів з метою запобігання перенавчанню і покращення загальної здатності узагальнення.

Функція активації – функція, яка застосовується до вихідних значень нейронів у шарі нейронної мережі для введення нелінійності та активації нейронів.

Фреймворк – комплексне програмне середовище або платформа, що надає інструменти та бібліотеки для розробки, навчання та експериментування з нейронними мережами.

ImageNet – велика база даних зображень, яка використовується для тренування та оцінки нейронних мереж, особливо в галузі комп'ютерного зору.

Адам (Adaptive Moment Estimation) – алгоритм оптимізації, що використовується для навчання нейронних мереж. Він комбінує метод моменту та адаптивного градієнтного спуску для ефективного оновлення параметрів мережі.

Датасет – набір даних, який використовується для тренування, тестування та оцінки моделей машинного навчання.

Функції втрат (loss function) – функція, яка вимірює різницю між прогнозованими значеннями моделі і фактичними значеннями. Вона використовується під час навчання для налаштування параметрів моделі.

Backpropagation (зворотнє поширення помилки) – алгоритм, що використовується для обчислення градієнтів і оновлення параметрів нейронних мереж під час процесу навчання.

CrossEntropyLoss – функція втрат, яка використовується в задачах класифікації. Вона вимірює відхилення між прогнозованим ймовірнісним розподілом і фактичними категоріями.

Хештег – символ або комбінація символів (#), що використовується в соціальних медіа для позначення теми або ключового слова, яке допомагає згруповувати та знайти пов'язані повідомлення або контент.

Препроцесинг зображень – процес обробки зображень перед використанням їх у моделях машинного навчання, включаючи масштабування, зміну розмірів, обрізку, нормалізацію тощо.

Планувальник швидкості навчання StepLR – метод планування швидкості навчання в нейронних мережах, де швидкість навчання

змінюється на певних епохах згідно з певним розкладом, наприклад, знижується на певну кількість епох.

Матриця плутанини або помилок (confusion matrix) – це інструмент, який використовується для оцінки продуктивності моделі класифікації. Вона відображає кількість правильних і неправильних прогнозів, зроблених моделлю для кожної категорії класифікації. Також вона має форму прямокутної таблиці, де кожен ряд представляє фактичну категорію, а кожний стовпець представляє прогнозовану категорію. Кожна комірка матриці плутанини містить кількість прикладів, які належать до відповідної комбінації фактичної та прогнозованої категорії.

Штучна нейронна мережа (ШНМ) – це комп'ютерна модель, яка намагається моделювати роботу нервової системи людини для вирішення завдань обробки інформації.

SGD (Stochastic Gradient Descent, метод градієнтного спуску) - це алгоритм оптимізації, який використовується для навчання моделей машинного навчання, зокрема штучних нейронних мереж. Він є одним з найпоширеніших методів градієнтного спуску і дозволяє знаходити мінімум функції втрат шляхом ітеративного коригування параметрів моделі.

Розділ 1. Загальні положення

1.1 Соціальні мережі

Соціальні мережі стали невід'ємною частиною нашого повсякденного життя. З кожним днем ми спостерігаємо збільшення кількості нових користувачів. Це означає, що маркетологи мають велику можливість привернути увагу великої та зацікавленої аудиторії до свого бренду. Соціальними мережами користуються близько 3,5 мільярдів користувачів Інтернету у всьому світі, і ця цифра, ймовірно, зростатиме (рис.1.1).

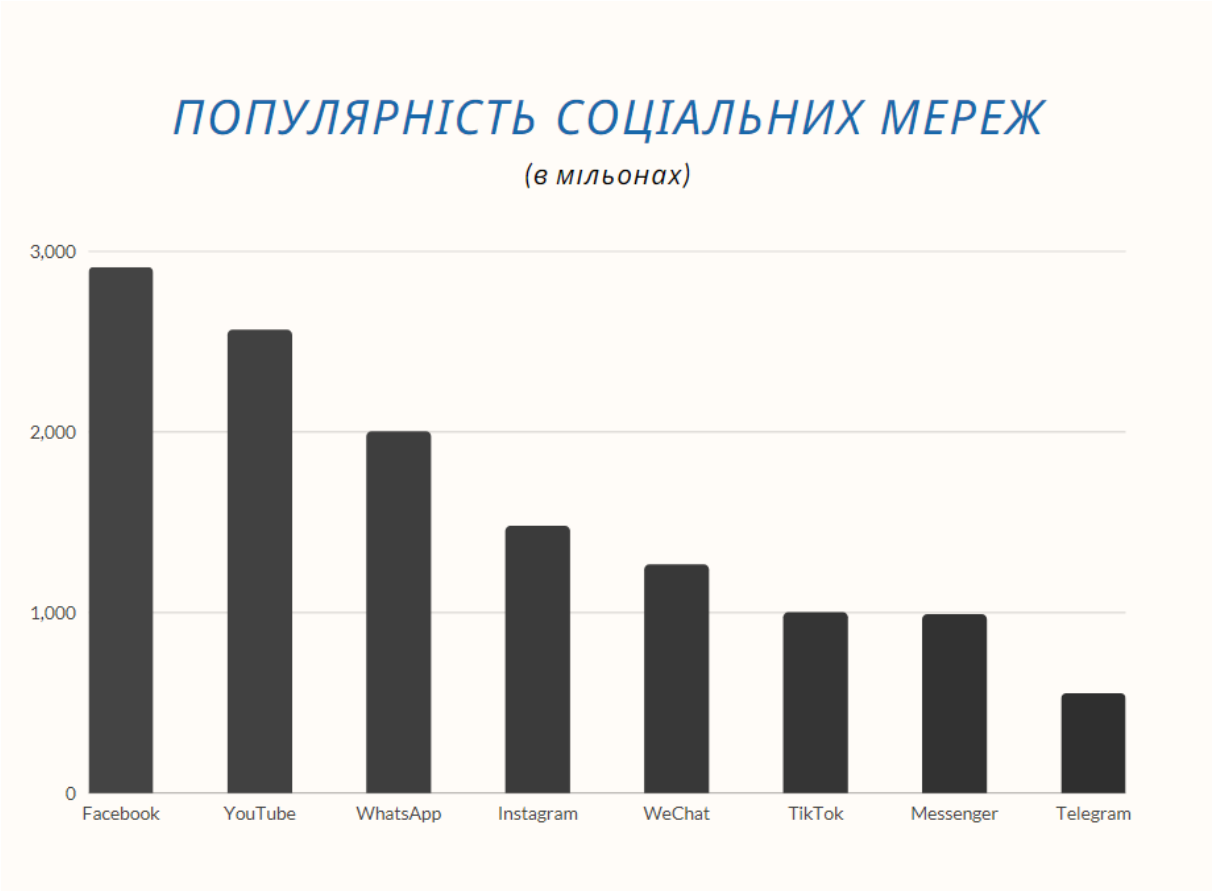

Рис. 1.1. Популярність соціальних мереж

Соціальні медіа-платформи, такі як Facebook та Instagram, здебільшого зосереджені на обміні зображеннями та оновленням статусу між друзями. Тенденції соціальних медіа загалом відрізняються. Мікроблоги – це інші соціальні мережі, такі як Twitter. У той час як деякі соціальні мережі наголошують на спільнотах, інші виділяють і представляють контент, створений користувачами [2]. За даними statista.com, Facebook є найпоширенішою соціальною мережею у світі з приблизно 2,5 мільярдами активних користувачів щомісяця. Мільярд користувачів налічує програма для обміну фотографіями Instagram [3] .

1.2 Штучний інтелект

У сфері штучного інтелекту існує багато програмних застосунків. Отже, важливо спочатку визначити, що означає штучний інтелект (ШІ) у них. Найпростіше визначення програмного забезпечення зі штучним інтелектом – це забезпечення, яке здатне виконувати розумові функції, які зазвичай виконує людина. Ці програми можна розділити на три групи, які пов'язані між собою (рис. 1.2):

1. Системи прийняття рішень
2. Машинне навчання
3. Глибоке навчання

Системи прийняття рішень використовуються для вирішення неформалізованих задач [4] , які мають низку характеристик:

- Задачі не розуміються належним чином або не засвоюються через процес символічних суджень.
- Дослідження питань можна проводити за допомогою методу символічних суджень.

- Неможливо визначити мету в термінах конкретної цільової функції. Проблема не має відомого алгоритмічного вирішення.
- Через брак ресурсів, (час, пам'ять) якщо є алгоритмічне рішення, воно не може бути використане.

Крім того, неформалізовані задачі містять помилки, неповноту, двозначність і конфлікти між оригінальними знаннями та знаннями, що вирішують проблеми.

Рис. 1.2. Зв'язок між галузями ШІ

Досліджуючи вхідні дані, машинне навчання (МН, англ. Machine Learning) прагне вирішити конкретну проблему. Найчастіше зустрічаються такі завдання, як регресія, класифікація, кластеризація, зменшення розмірності, виявлення аномалій тощо. Її часто називають прогностичною аналітикою або статистичним навчанням і є галуззю дослідження, яка поєднує інформатику, штучний інтелект і статистику. Найефективніші алгоритми машинного навчання автоматизують прийняття рішень шляхом екстраполяції відомих прикладів. Алгоритми машинного навчання можна натренувати. Існують такі категорії машинного навчання:

- навчання з інструктором або навчання під наглядом (supervised learning);
- навчання самостійно чи без нагляду (unsupervised learning).

У першому випадку алгоритму надається набір навчальних даних від користувача у формі пар об’єкт-відповідь, і він визначає, як отримати відповідь за об’єктом. Зокрема, система може відповісти на питання про новенький об'єкт без сторонньої допомоги. Оскільки «вчитель» навчає алгоритм правильній відповіді на кожне спостереження, на якому його навчають, алгоритми машинного навчання, які навчаються на парах об’єкт-відповідь, відомі як алгоритми, навчені вчителем.

Алгоритми навчання з викладачем інтерпретуються, а стандарт їхньої роботи легко оцінити, незважаючи на те, що генерація набору з об’єктами та відповідями іноді є виснажливою, ручною операцією. Машинне навчання, швидше за все, зможе вирішити вашу проблему, якщо ваше завдання можна виразити як навчальне завдання з вчителем і ви можете створити набір даних, який містить рішення.

Глибоке навчання (ГН, англ. Deep Learning) – це підгалузь машинного навчання, яка має справу з глибокими нейронними мережами та використовує новий підхід до навчання на основі даних, який робить наголос на збиранні нової інформації з послідовних шарів моделі. Слово «глибоке» в глибокому навчанні стосується поняття роботи з даними шляхом застосування послідовних шарів нейронної мережі, а не будь-якого глибшого розуміння даних, отриманих цим методом. Терміни багаторівневий та ієрархічний також взаємозамінні для цієї назви. Кількість шарів нейронів, що використовуються в моделі, може бути використана для опису її глибини. У сучасному глибокому навчанні часто використовуються десятки або навіть сотні рівнів обробки даних, кожен з яких автоматично

навчається за допомогою вхідних даних. Тоді як просте машинне навчання фокусується на моделях з обмеженою кількістю кроків обробки даних.

1.3 Нейронні мережі

Штучні нейронні мережі (ШНМ) – це клас алгоритмів, які, теоретично, імітують функціонування людського мозку, використовуючи навчання для виявлення кореляцій у наборах даних. Основою для навчання є повторна стимуляція певних зв'язків мозку. Як результат, є більший шанс отримати очікуваний результат при використанні правильних вхідних даних (сигналів). Коли результат є точним, зв'язки мозку, які його створили, стають густішими. Цей вид навчання використовує зворотний зв'язок.

Простіші версії штучних нейронних мереж імітують поведінку мозку. Можливі як контрольовані, так і неконтрольовані методи навчання. У контрольованих ШНМ мережа навчається, отримуючи зразки відповідних вхідних і вихідних даних. Мета неконтрольованого навчання в штучних нейронних мережах (ШНМ) полягає в тому, щоб «змусити» її «розуміти» структуру переданих вхідних даних «самостійно». Нейрон у ШНМ моделюється математично як суматор значень і ваг, які передаються на вхід нейрона. Для нього сума піддається функції активації, що призводить до формування єдиного виходу. У завданнях класифікації функція активації повинна мати властивість «перемикати». Іншими словами, вихідні дані повинні змінити стан, наприклад, з 0 на 1 або -1 на 1, якщо вхідні дані більші за певне число. Це імітує «спрацьовування» біологічного нейрона. Поширеною функцією активації є сигмоподібна функція $f(x) = \frac{1}{1+\exp(-x)}$, яка плавно змінює свої значення (рис. 1.3):

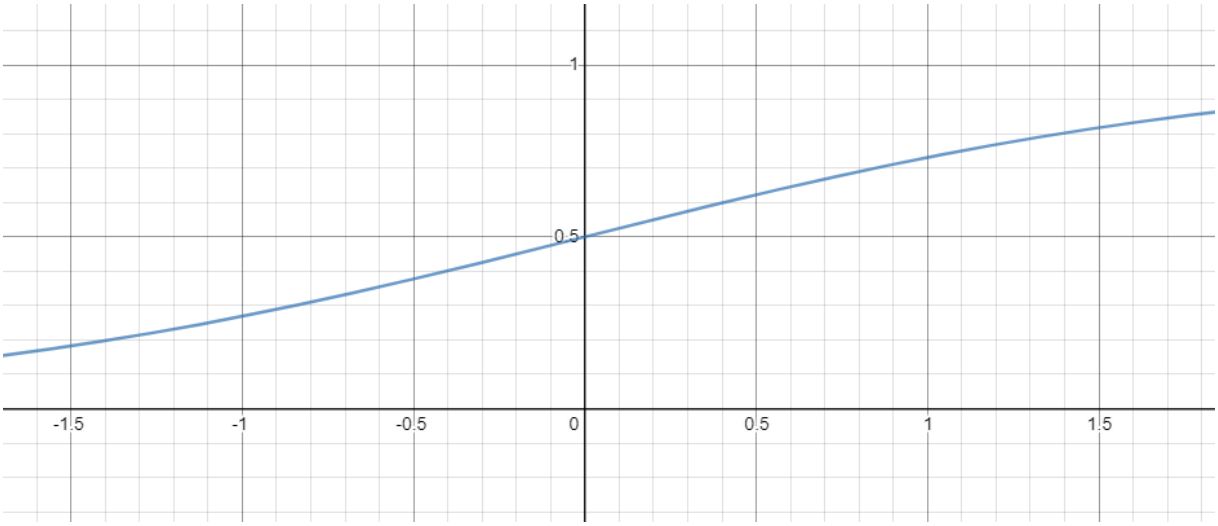

Рис. 1.3. Графік функції

На графіку можна побачити, що функція є «активаційною», оскільки вона зростає від 0 до 1 із кожним зростанням x. Сигмоподібна функція є безперервною і гладкою. Це вказує на те, що можна отримати похідну для цієї функції, яка має вирішальне значення для навчання ШНМ, оскільки значення похідної використовується в обчисленнях протягом усього процесу навчання мережі.

Штучні нейронні мережі, автокодери, рекурентні нейронні мережі та навчання з підкріпленням – це кілька моделей глибокого навчання. Ці структури є основою нейронних мереж.

1.4. Згорткові нейронні мережі

Згорткові нейронні мережі (CNN) [5] є однією з моделей глибокого навчання, яка значно просунула галузі комп’ютерного зору та аналізу зображень. Вони являють собою підмножину нейронних мереж, які розрізняють і класифікують зображення на основі їхніх атрибутів. Основна ідея їхньої роботи полягає в тому, що дві фотографії обробляються та одна

за одною множаться на матриці, щоб отримати результат, який можна використовувати для визначення ключових елементів зображення.

Архітектура CNN складається з двох основних компонентів (рис.1.4):

- Інструмент згортки, який витягує та класифікує інформацію про об’єкт із кожного зображення для аналізу за допомогою процедури, відомої як вилучення функцій.
- Повністю зв’язаний шар (Fully Connected Layer), який виконує прогнозування класу зображення за допомогою вихідних даних етапу згортки.

Як правило, на виході згорткової нейронної мережі чергуються шари згортки (convolution), шари агрегації (шари підвибірки) (pooling) і повністю зв’язані шари (fully-connected). Можливе будь-яке розташування трьох різних типів шарів. Також шар відсіву (dropout) та функція активації є двома додатковими важливими параметрами згорткової нейронної мережі.

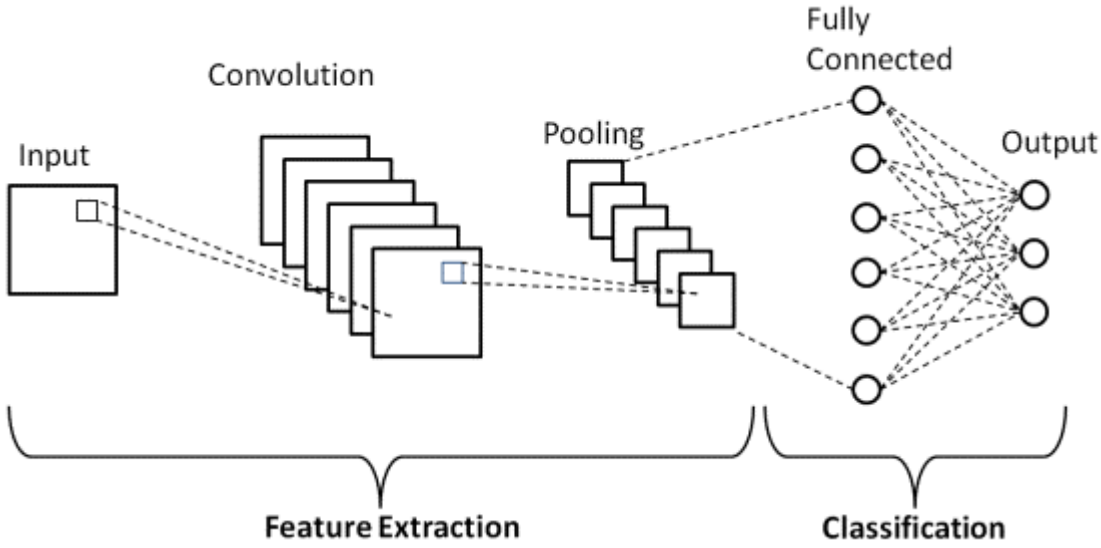

Рис. 1.4 Архітектура CNN [15]

Згортковий шар часто є початковим шаром, який виконує операцію згортки, витягує ключові характеристики із зображень і створює карти функцій. Як правило, рівень агрегації (рівень об’єднання) йде після згорткового рівня. Цей шар зменшує розмір карти об’єктів, щоб скоротити витрати на обчислення. Повністю пов’язаний (Fully Connected), третій рівень, упорядковує відфільтровані фотографії за категоріями [6] .

Щоб запобігти повторному навчанню в нейронних мережах, які часто навчаються шляхом стохастичного градієнтного спуску, випадковим вибором певних елементів із вибірки, використовується рівень Dropout (регуляризація Dropout). Замість навчання однієї нейронної мережі з високим зв’язком Dropout навчає групу з багатьох мереж, усереднюючи результати [7] .

Кажуть, що CNN «перетренована», якщо вона добре працює на тестових даних, але погано на навчальних даних. Це відбувається, якщо навчальний набір даних має недостатню різноманітність або якщо мережа піддається надмірній кількості етапів.

Одним із рішень цієї проблеми є використання шару відсіву (dropout), який видаляє з мережі певну кількість нейронів під час навчання та зменшує розмір моделі. Це видно на малюнку нижче (рис.1.5).

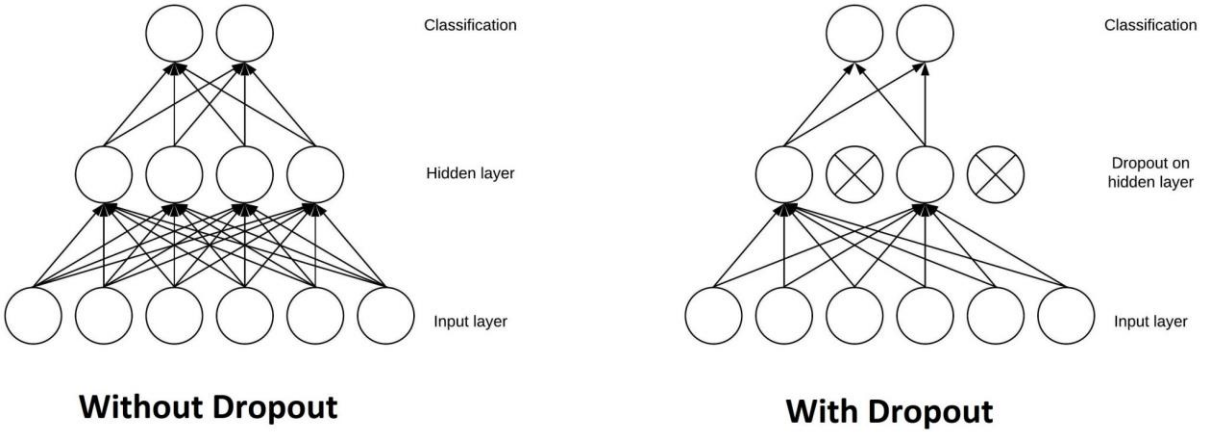

Рис. 1.5 Кількість нейронів з шаром відсіву та без [16]

Згорткові нейронні мережі ідеально підходять для аналізу зображень, тому їх розробка та впровадження в додатки для виявлення та класифікації квітів може мати великий вплив на те, як розвиватиметься індустрія флоризму в майбутньому.

Функція активації – це математична функція, яка застосовується до виходу нейрона в нейронній мережі, включно з згортковою нейронною мережею (CNN). Мета функції активації полягає в тому, щоб ввести нелінійність у вихід нейрона, дозволяючи нейронній мережі вивчати та представляти більш складні моделі даних. Без функцій активації нейронні мережі могли б вивчати лише лінійні моделі, які обмежені у своїй здатності моделювати дані реального світу.

Існує кілька поширених функцій активації в CNN, зокрема наведені на рис. 1.6 [8] :

1. ReLU (Rectified Linear Unit) – Ця функція активації повертає вхідні дані, якщо вони позитивні, і повертає 0, якщо вони негативні. Ця функція ефективна з точки зору обчислень і допомагає пом’якшити проблему зникнення градієнта, яка є загальною проблемою в глибоких нейронних мережах.
2. Sigmoid – ця функція активації відображає вхідні дані на значення від 0 до 1, що корисно для задач двійкової класифікації.
3. Tanh (гіперболічний тангенс) – ця функція активації відображає вхідні дані на значення від -1 до 1, що корисно для проблем класифікації кількох класів.
4. Softmax – ця функція активації зазвичай використовується на вихідному рівні нейронної мережі для задач багатокласової класифікації. Вона відображає вхідні дані на розподіл ймовірностей за класами, причому ймовірність правильного класу близька до 1, а ймовірність неправильних класів близька до 0.

5. Leaky ReLU – схоже на ReLU, але дозволяє пропускати невеликі від’ємні значення через функцію активації замість обнулення.
6. Експонентна лінійна одиниця (ELU) – на відміну від ReLU, ELU мають негативні значення, що дозволяє їм наближати активації середніх одиниць до нуля, подібно до пакетної нормалізації, але з меншою обчислювальною складністю.

Важливо зауважити, що відповідну функцію активації слід вибирати на основі проблеми, яку ви намагаєтеся вирішити, і набору даних, з яким ви працюєте.

Зміна вагових коефіцієнтів окремих нейронів для ідентифікації відповідних елементів у зображеннях є одним із завдань, пов’язаних із створенням CNN. Нейронна мережа «навчається» протягом усього процесу зміни цих ваг. Ваги нейронної мережі спочатку надаються випадковим чином. Нейронна мережа навчається за допомогою великої колекції фотографій, розділених на відповідні класифікації.

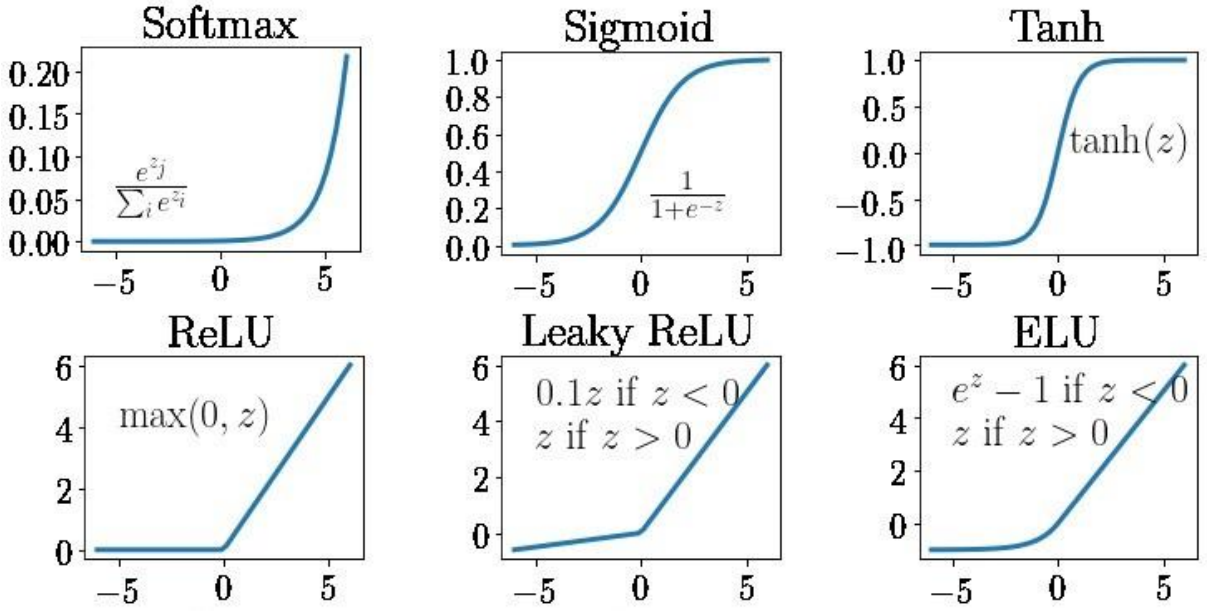

Рис. 1.6 Функції активації

CNN застосовує випадкові значення до кожного зображення під час його обробки, а потім порівнює результати з точною міткою зображення.

Ваги нейронів дещо змінюються, якщо вихідні дані мережі не відповідають мітці, щоб забезпечити точніший результат наступного разу.

«Епоха» – це кожен цикл всього набору даних навчання. Протягом навчання CNN проходить через багато епох і поступово змінює свою вагу. Нейронна мережа трохи покращує класифікацію навчальних зображень після кожної епохи. Модифікації ваги, які CNN робить, стають все меншими, оскільки він стає кращим. Згодом мережа досягає максимальної продуктивності.

Розробники використовують тестовий набір даних після навчання, щоб оцінити точність. Колекція позначених фотографій, які не були включені у фазу навчання, становить цей набір. Кожне зображення проходить процес CNN, і результати порівнюються з точним значенням зображення. Тестовий набір даних по суті оцінює, наскільки нейронна мережа набула досвіду в класифікації зображень, яких вона ніколи раніше не бачила.

Розділ 2. Реалізація нейронних мереж: інструменти та технології

2.1 Інструменти для глибоко навчання

Існує безліч програмних інструментів для впровадження та роботи з машинним навчанням, оскільки його використання стає все більш потрібним у багатьох галузях, від Інтернет-сервісів до мобільних пристроїв і різних вбудованих систем. Кожен із них має різний інтерфейс, іншу мову програмування, іншу робочу платформу та різні вимоги до навичок розробника. Усі ці інструменти можна розділити на дві групи: програмні засоби високого та низького рівня.

Програмні засоби високого рівня надають розробнику готове середовище для створення моделей, їх навчання, виконання та аналізу результатів. Цей тип інструментів також відомий як «Машинне навчання як послуга». Такі програми призначені для програмістів, які планують побудувати модель на основі добре відомих, регулярно використовуваних прикладів для вирішення конкретних завдань, адаптованих до їхніх вимог. Такі рішення пропонують перевагу усунення вимоги щодо значного досвіду програмування або глибокого розуміння математичної сторони побудови моделі. Враховуючи те, що такі системи пропонують користувачеві зручний інтерфейс із кристально зрозумілою, уже розробленою та ретельно перевіреною функціональністю. Робота з такими інструментами часто позбавляє від необхідності писати новий програмний код. Ось кілька прикладів таких програм:

Хмарні платформи: такі компанії, як Amazon Web Services (AWS), Microsoft Azure та Google Cloud Platform (GCP), пропонують різноманітні послуги машинного навчання, зокрема хмарне сховище, обчислення та

інструменти машинного навчання. Ці платформи спрощують створення, навчання та розгортання моделей машинного навчання в масштабі.

AutoML: служби автоматичного машинного навчання (AutoML), такі як Google AutoML, H2O.ai, DataRobot і TPOT, дозволяють користувачам навчати та розгортати моделі без написання будь-якого коду, автоматизуючи процес розробки функцій, вибору моделі та налаштування гіперпараметрів.

Попередньо навчені моделі: Деякі компанії, такі як HuggingFace, OpenAI і TensorFlow Hub, надають попередньо навчені моделі, які можна налаштувати для конкретних завдань.

Прогнозне моделювання: служби прогнозного моделювання, такі як RapidMiner, KNIME та Alteryx, дозволяють користувачам створювати та розгортати прогнозні моделі за допомогою інтерфейсу перетягування.

Інструменти нижчого рівня, які дозволяють працювати з кодом програми, складають другу категорію програмних засобів машинного навчання. Ці різноманітні фреймворки машинного навчання пропонують необхідні можливості для створення, навчання та запуску алгоритмів на рівні коду. Використання такого програмного забезпечення пропонує кілька варіантів для вдосконалення всіх нюансів моделі та розробки нових алгоритмів. Однак використання такого роду технологій вимагає експертних знань певної мови програмування та широкого спектру математичних концепцій. Найпопулярніші фреймворки машинного навчання [9] :

TensorFlow – це фреймворк з відкритим вихідним кодом, розроблена Google для створення, розгортання та ітерації моделей машинного навчання. Він особливо добре підходить для завдань глибокого навчання та підтримує широкий спектр архітектур нейронних мереж, включаючи прямі, рекурентні та згорткові мережі. TensorFlow також включає різноманітні інструменти

для попередньої обробки даних, візуалізації моделі та розгортання, включаючи TensorFlow Lite для мобільних і крайніх пристроїв і TensorFlow.js для Інтернету.

PyTorch – це фреймворк з відкритим вихідним кодом, розроблена Facebook для створення та розгортання моделей машинного навчання. Він відомий своїм динамічним обчислювальним графіком, який забезпечує більшу гнучкість і експерименти, ніж такі фреймворки, як TensorFlow. PyTorch також зосереджений на дослідженнях і експериментах, що робить його популярним серед науковців і дослідників. Він також має зростаючу екосистему інструментів і бібліотек, включаючи torchvision для комп'ютерного зору та torchtext для обробки природної мови.

Caffe – це фреймворк глибокого навчання, розроблений в Центрі бачення та навчання Берклі (BVLC). Він особливо добре підходить для завдань обробки зображень і відео, а також зосереджений на швидкості та ефективності. Caffe також надає різноманітні інструменти для попередньої обробки даних і візуалізації моделі, а також має зростаючу екосистему інструментів і бібліотек, включаючи Caffe2 і CaffeOnSpark.

Keras – це високорівневий API для нейронних мереж, написаний на Python і здатний працювати поверх TensorFlow, CNTK або Theano. Він був розроблений з упором на можливість швидкого експериментування, дозволяє легко та швидко створювати прототипи, підтримує як згорткові мережі, так і повторювані мережі, а також комбінації обох. Він також підтримує кілька серверних програм і може працювати як на процесорах, так і на графічних процесорах.

OpenCV (Open Source Computer Vision) – це бібліотека програмних функцій, призначених головним чином для комп'ютерного бачення в реальному часі. Він має інтерфейси C++, Python і Java і підтримує Windows, Linux, Mac OS, iOS і Android. Він містить широкий спектр інструментів для

аналізу зображень і відео, включаючи виявлення функцій, аналіз руху, виявлення об’єктів і машинне навчання. OpenCV широко використовується в програмах комп’ютерного зору та має все більше можливостей глибокого навчання, включаючи підтримку глибоких нейронних мереж за допомогою TensorFlow, Caffe та інших. За допомогою OpenCV можна легко створювати програми глибокого навчання, він також надає попередньо підготовлені моделі та інструменти для виконання трансферного навчання.

2.2 Технологія передачі навчання

Побудова архітектури моделі [10] та її освоєння займає багато часу та зусиль. У цьому процесі може брати участь одна особа або ціла команда. Через це розробка архітектури з нуля для кожної діяльності виявляється досить дорогим завданням. Ми можемо умовно класифікувати дії, для яких використовуються штучні нейронні мережі, на кілька груп, наприклад, розпізнавання зображень, слів, звуків тощо. Штучні нейронні мережі, навчені на тих самих наборах даних, часто матимуть схожу архітектуру в одній категорії. Тому для зниження вартості створення таких моделей була розроблена техніка трансферного навчання.

Фундаментальний принцип методу передачі моделі навчання полягає в тому, що після того, як нейронна мережа навчена на великому наборі даних, її можна пізніше використовувати для аналізу даних, які є більш спеціалізованими та цілеспрямованими. Оскільки він переносить навчання з однієї колекції даних до іншої, цей метод відомий як перенесення навчання.

Існують різні способи застосувати трансферне навчання (рис 2.1):

1. Використання попередньо створеної мережі та навчання лише останнього рівня класифікації;
2. Використання лише частини готової мережі. У цій ситуації деякі шари переробляються або навіть навчаються з нуля;
3. Використання ваг нової мережі з вагових коефіцієнтів з попередньо навченої мережі.

Важливо змінити останній рівень існуючої мережі, щоб реалізувати перший спосіб передачі моделі навчання. На малюнку зображено остаточний план проектування мережі. Кожен набір даних має різну кількість класів об’єктів, тому зміна останнього рівня є важливою, оскільки кожна мережа потребує окремого вихідного рівня.

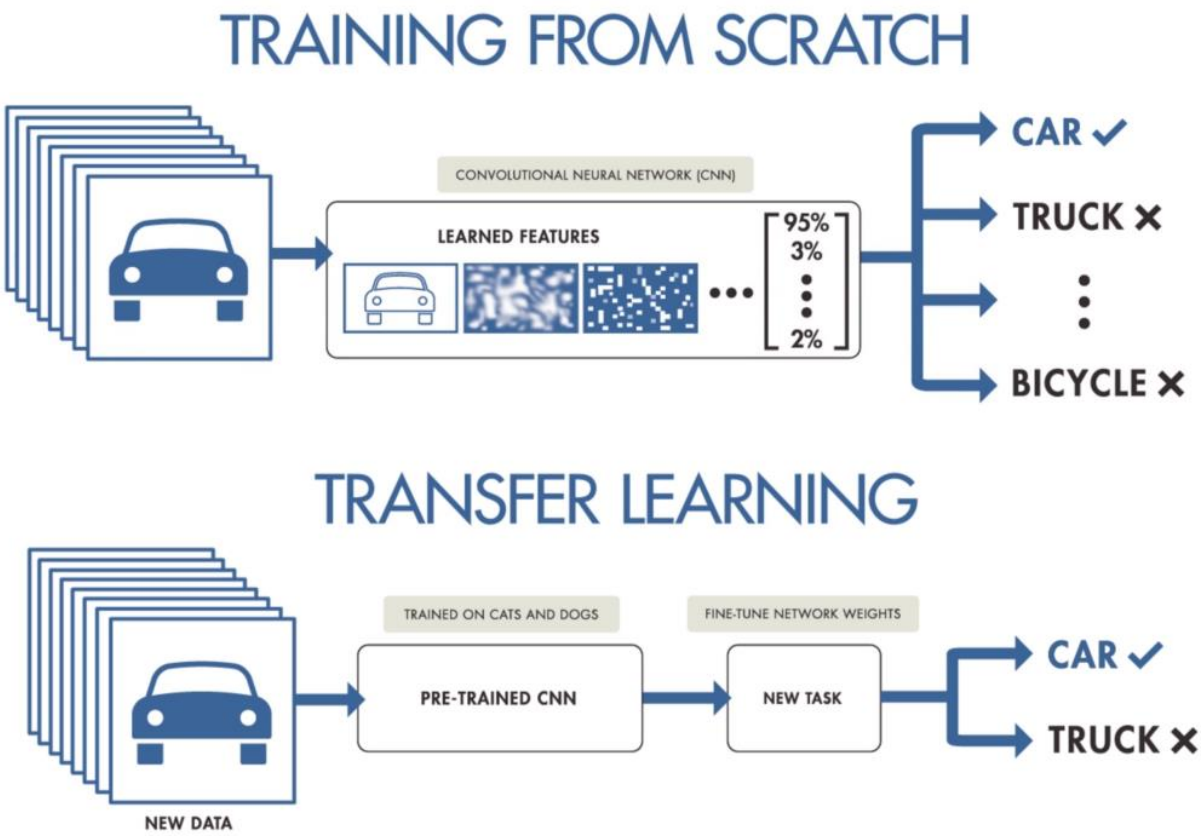

Рис. 2.1 Схема використання трансферного навчання [17]

Наприклад, набори даних ImageNet мають 1000 унікальних класів. Оскільки в новій колекції може бути інша кількість класів, потрібно змінити останній шар згорткової нейронної мережі для включення відповідної кількості виходів з нового набору.

Крім того, важливо гарантувати, що попередньо навчена модель не буде змінена під час процесу навчання. Вимкнення можливості зміни навченої моделі буде найоптимальнішим варіантом. Іншими словами, алгоритму навчання не можна дозволяти змінювати значення під час прямого та зворотного розповсюдження похибок. Під цією процедурою мається на увазі «заморожування моделі» (freezing the model), (рис. 2.2). «Заморожуючи» певні параметри навченої моделі, ми обмежуємо навчання останнім рівнем мережі класифікації, зберігаючи значення змінних навченого рівня. [11]

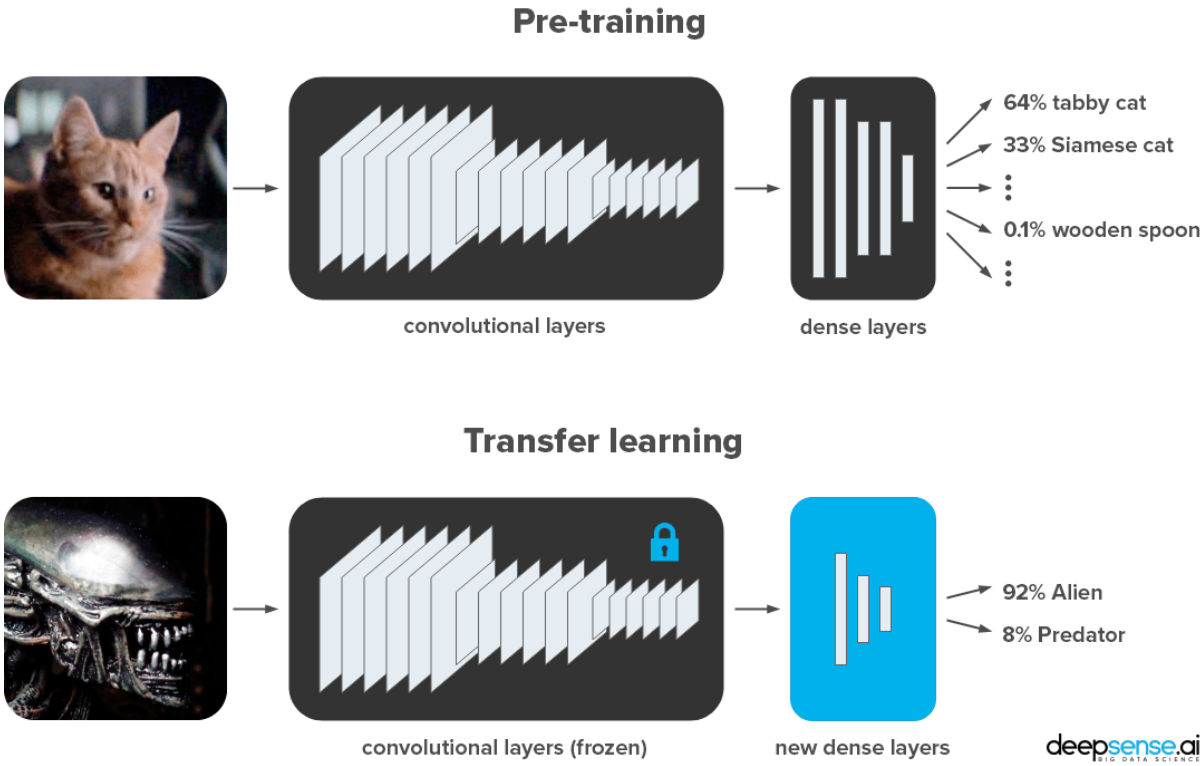

Рис. 2.2 Вигляд моделі з використанням «заморозки» [18]

Застосування другого способу вимагає заморожування лише необхідної кількості шарів, а не всіх, дозволяючи мережі перенавчати деякі важливі рівні. Ця стратегія потребує глибокого розуміння архітектури мережі.

Для застосування третього методу під час процесу ініціалізації мережі необхідно використовувати ваги з попередньо навченої мережі. Крім того, необхідно додати новий шар класифікації.

Перевага використання попередньо підготовлених моделей полягає в тому, що ми можемо навчити меншу частину моделі, скажімо, лише кілька конкретних шарів, за менший проміжок часу, ніж якби ми навчали всю модель з нуля.

2.3 Методи розпізнавання зображень

Збір конкретних метаданих, пов’язаних із зображеннями, відомий як розпізнавання зображень. [12] Існують різні способи вирішення цієї задачі. Загалом їх можна згрупувати в чотири групи (рис. 2.3):

1. Image Classification
2. Object Localization.
3. Object Detection.
4. Instance Segmentation.

Перший метод просто надає деталі про зображення в цілому, відповідаючи на запит "що на зображенні?". Ця стратегія реалізується програмами, які приймають фотографії як вхідні дані та виводять лише значення класу для об’єкта на зображенні

Другий метод пропонує вирішення проблеми "що на зображенні і де це знаходиться?". Іншими словами, він знаходить деталі як об’єкта, так і його положення на зображенні. Цей метод реалізується програмами, які приймають зображення як вхідні дані, виводять значення класу та позначають об’єкт на зображенні обмежувальним прямокутником.

Третій метод відповідає на запит: "Які об'єкти є на зображенні і де вони знаходяться?". Програми, які використовують це, виводять зображення з обмежувальними прямокутниками, що позначають елементи, разом із написами, що вказують, до якого класу належить кожен із позначених об’єктів.

Четвертий метод порівнюється з третім, за винятком того, що він пропонує більш точні дані про місцезнаходження об'єкта. На відміну від обмежувального прямокутника, цей підхід надає більш точну інформацію про границі та форму об’єкта, оскільки програми малюють пікселі, пов’язані з певним зображенням, певним кольором. [13]

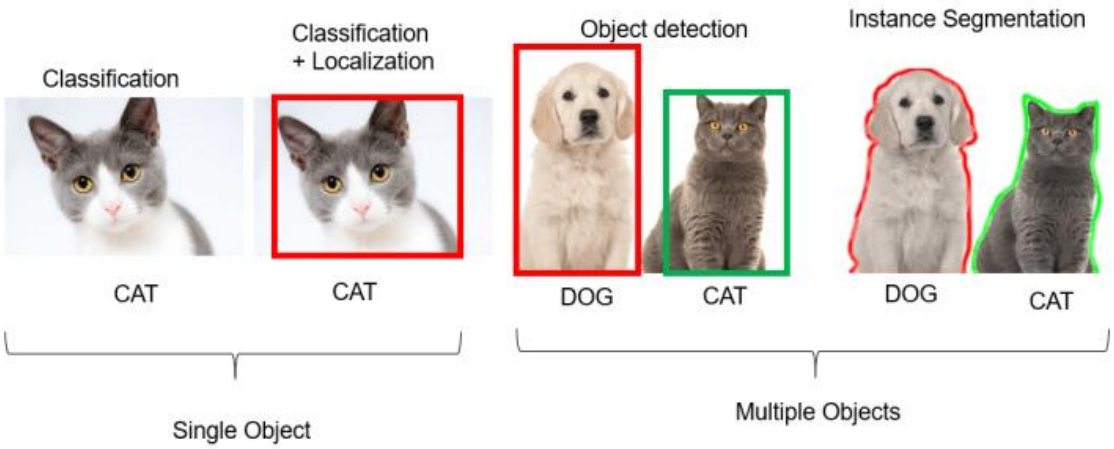

Рис. 2.3 Відмінності у підходах розпізнавання зображень [19]

2.4 Метод оптимізації Adam

Моделі глибокого навчання потребують великої кількості даних і обчислювальних ресурсів для навчання. Одним із важливих факторів, який може вплинути на їх продуктивність, є вибір оптимізатора. Оптимізатор Адама є одним із найпопулярніших алгоритмів оптимізації, який використовується в глибокому навчанні (рис. 2.4). [14]

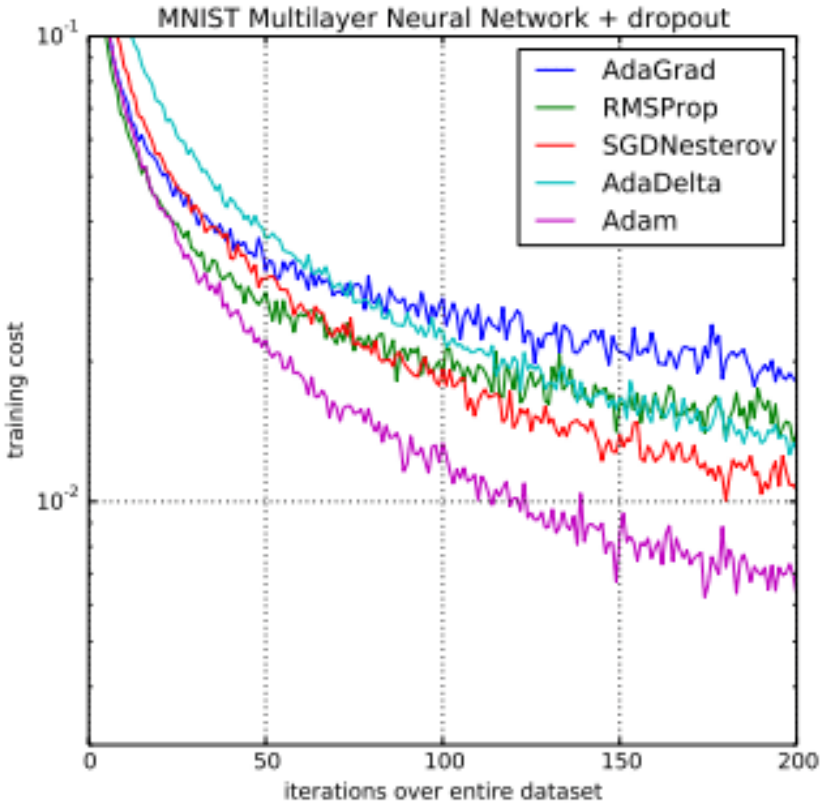

Рис. 2.4 Порівняння методів оптимізації для нейронних мереж [20]

Адам (Adaptive Moment Estimation) – це алгоритм оптимізації, який використовується для оновлення вагових коефіцієнтів нейронної мережі під час навчання. Оптимізатор Адама є розширенням оптимізатора стохастичного градієнтного спуску (SGD), який використовує комбінацію імпульсу та адаптивної швидкості навчання для швидшої конвергенції та

досягнення глобального мінімуму функції втрат. Алгоритм був запропонований у 2014 році Дідеріком П. Кінгмою та Джиммі Ба.

Оптимізатор Адама використовує комбінацію першого та другого моментів градієнтів для оновлення вагових коефіцієнтів нейронної мережі. Перший момент – це середнє значення градієнтів, а другий момент – це дисперсія градієнтів. Оптимізатор Адама підтримує два вектори ковзного середнього, щоб відстежувати ці моменти. Ковзні середні розраховуються таким чином:

1. Перший момент (середнє): $m_t = beta1 * m_{t-1} + (1 - beta1) * g_t$
2. Другий момент (дисперсія): $v_t = beta2 * v_{t-1} + (1 - beta2) * g_t^2$

Де m_t і v_t — перший і другий моменти в момент часу t, beta1 і beta2 – це швидкості спаду для ковзних середніх, а g_t – градієнт у момент часу t.

Оптимізатор Adam об’єднує ці ковзні середні, щоб оновити вагові коефіцієнти нейронної мережі за допомогою такого рівняння:

$w_{t+1} = w_t - alpha * m_t / (sqrt(v_t) + epsilon)$

Де w_t і w_{t+1} – ваги в момент часу t і t+1, альфа – швидкість навчання, а епсилон – мала константа для запобігання ділення на нуль.

Оптимізатор Adam має кілька переваг перед іншими алгоритмами оптимізації, такими як SGD. Деякі з цих переваг:

- Швидша конвергенція: оптимізатор Adam сходиться швидше, ніж інші алгоритми оптимізації, що означає, що для досягнення глобального мінімуму функції втрат потрібно менше ітерацій.
- Адаптивна швидкість навчання: оптимізатор Adam адаптує швидкість навчання на основі першого та другого моментів градієнтів, що робить його більш придатним для моделей глибокого навчання.

- Стійкість до зашумлених даних: оптимізатор Adam стійкий до зашумлених даних, що означає, що він може обробляти дані з великим шумом і все одно збігатися до глобального мінімуму функції втрат.

У висновку можна сказати, що Adam optimizer – це ефективний алгоритм оптимізації, який може покращити продуктивність моделей глибокого навчання. Він використовує комбінацію імпульсу та адаптивної швидкості навчання для швидшої конвергенції та досягнення глобального мінімуму функції втрат.

Розділ 3. Реалізація програми

3.1 Розробка програми

Для навчання нейронної мережі був використаний датасет від оксфордського університету, який містить 102 класи квітів. Набір даних був підготовлений для навчання, валідації та тестування моделі. Дані були підготовлені за допомогою бібліотеки torchvision, яка надає корисні інструменти для завантаження та обробки даних.

Створення програмного додатку, який мав би досягти поставленої в цій роботі мети, а саме: завантаження даних з соціальної мережі Інстаграм, аналіз зображення, розпізнавання та підрахунок зображень з різними видами квітів, було розбито на декілька основні етапів:

1. Програмна реалізація архітектури з використанням фреймворків PyTorch та TensorFlow;
2. Навчання нейронної мережі на обраному наборі даних;
3. Використання на верхніх лінійних шарах вже натренованої мережі Resnet18 на багатомільйонному наборі даних ImageNet ;
4. Використання архітектури YOLOv3 для фільтрування зображень з квітами і без;
5. Завантаження даних з Інстаграм за допомогою API;
6. Сортування фотографій за наявністю на них квітів;
7. Вивід кінцевого графіку з результатами по кожному класу.

Для класифікації зображень була використана попередньо натренована модель ResNet-18: ResNet (Residual Network), доступна в бібліотеці torchvision. Архітектура ResNet-18 є потужною архітектурою

глибоких нейронних мереж, яка дозволяє ефективно навчати глибокі моделі. Ця модель є однією з версій ResNet, яка складається з 18 шарів, включаючи згорткові, пулінгові та повнозв'язні шари. Ця архітектура попередньо навчена на великому наборі зображень, відомому як ImageNet.

У попередньо навченій моделі ResNet-18 останній лінійний шар призначений для 1000 класів з набору даних ImageNet. Оскільки нам потрібно розпізнавати класи квітів, яких є 102, я замінила останній лінійний шар моделі на новий лінійний шар з кількістю вихідних нейронів, відповідних кількості класів квітів. Попередньо навчені ваги решти моделі залишаються незмінними. Таким чином, під час тренування нового останнього лінійного шару, попередньо навчена модель передає свої знання про загальні ознаки зображень до нового завдання розпізнавання квітів. Це дозволяє ефективно використовувати знання, набуті на великому наборі даних, для покращення результатів навчання на меншому наборі даних квітів. Такий підхід особливо корисний, коли доступний обмежений набір навчальних даних. Він дозволяє швидко та ефективно навчити модель розпізнавати нові класи на основі знань, набутих на попередньому етапі навчання.

Для класифікації та виявлення об'єктів на зображеннях використовується архітектура YOLOv3 (You Only Look Once) – одна з популярних моделей для об'єктного виявлення. Для цієї моделі був використаний попередньо навчений набір вагів, який може бути завантажений з файлу "yolov3.pt".

3.2 Тренування нейронної мережі

Для тренування нейронної мережі був підготовлений окремий набір даних розбитих на класи. Він містить безпосередньо самі зображення та словник формату «зображення – номер класу» до них (рис. 3.1):

```
"image_00001.jpg",77
"image_00002.jpg",77
"image_00003.jpg",77
"image_00004.jpg",77
"image_00005.jpg",77
"image_00006.jpg",77
"image_00007.jpg",77
"image_00008.jpg",77
"image_00009.jpg",77
"image_00010.jpg",77
"image_00011.jpg",77
"image_00012.jpg",77
```

Рис. 3.1 Приклад файлу словника

Основні кроки тренування включають в себе:

- Визначення функцій перетворення даних для тренування, валідації та тестування. Для тренування використовуються перетворення, такі як випадкове обрізання і дзеркальне відображення, щоб збільшити різноманіття даних. Для валідації та тестування застосовується лише нормалізація зображень.
- Визначення функції train_model, яка виконує тренування моделі за заданою кількістю епох. Вона приймає модель, критерій (функцію втрат), оптимізатор, планувальник швидкості навчання та кількість епох як вхідні параметри. У першій ітерації модель переключається в режим тренування, а у наступних ітераціях – у режим оцінки (evaluation mode). Відбуваються ітерації по партіях

даних (batch) – кожна партія містить певну кількість зображень та їх міток класів. Вхідні зображення та мітки класів переносяться на пристрій для обчислення та обнуляються градієнти параметрів моделі. Розраховуються функції втрат (loss function), тобто порівнюються прогнозовані значення з мітками класів, що дає значення втрат (помилок) та зворотні проходження (backpropagation), тобто обчислюються градієнти відносно параметрів моделі, що дозволяє оновити їх за допомогою оптимізатора. Прохід назад і вперед робить разом одну «ітерацію». Під час однієї ітерації передається підмножина набору даних (batch). Сама «Епоха» означає передачу всього набору даних пакетами. Оптимізатор оновлює ваги моделі, використовуючи обчислені градієнти та метод оптимізації Adam. В кінці розраховується значення функції втрат за допомогою критерію CrossEntropyLoss та точності для моніторингу тренування.

- Валідація моделі. Після кожної епохи проводиться оцінка моделі на валідаційному наборі даних. Модель переключається у режим оцінки (evaluation mode). Обчислюються значення функції втрат та точності на валідаційних даних без здійснення зворотного поширення (backpropagation) та оновлення параметрів моделі. Отримані значення використовуються для моніторингу та порівняння з тренувальними результатами.
- Збереження найкращої моделі. Після кожної епохи перевіряється, чи досягнуто кращої точності на валідаційному наборі даних. Якщо так, то параметри моделі зберігаються для подальшого використання.

На наступному малюнку (рис 3.2) будуть зображені дані функцій втрат та точності на тренувальному та валідаційному наборах даних, при проходження останньої епохи:

```
Epoch 25/25
----------
train Loss: 0.2617 Acc: 0.9283
val Loss: 0.1854 Acc: 0.9535

Training complete in 206m 58s
Best val Acc: 0.953548
```

Рис. 3.2 Результати останньої епохи та закінчення тренування

Результати нейронної мережі після закінчення тренування, а саме побудова графіків, які візуалізують залежність функції втрат (рис. 3.3) та точності (рис. 3.4) від кількості пройдених епох .

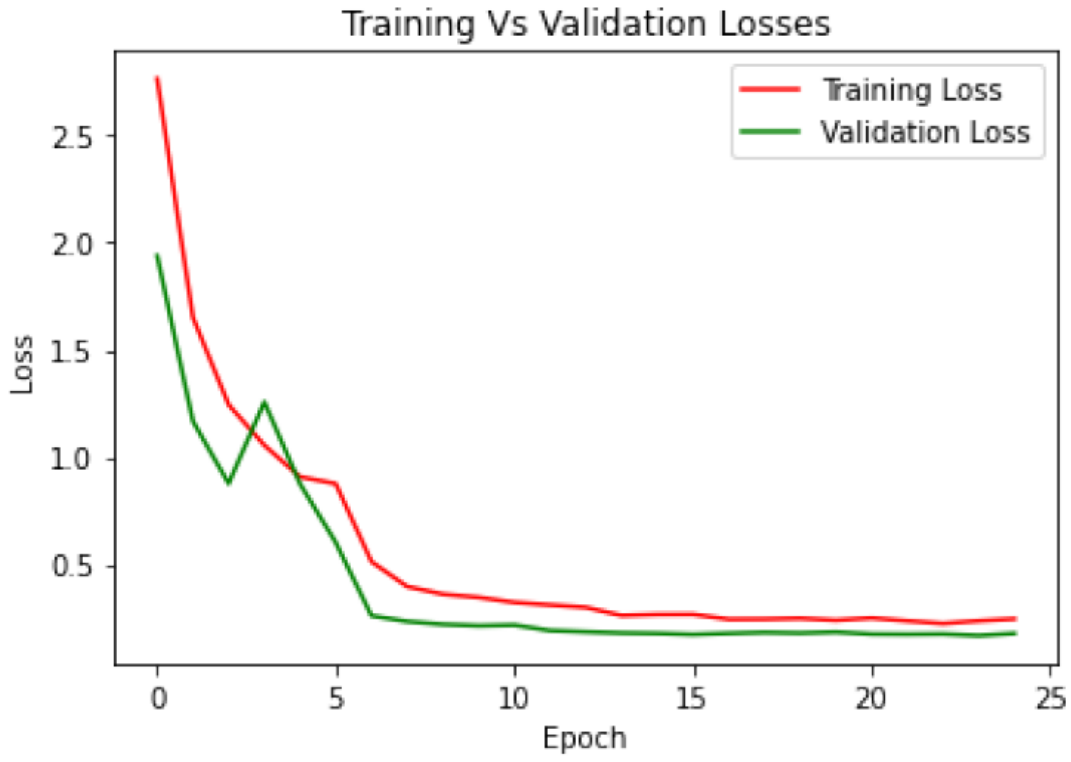

Рис. 3.3 Залежність функції втрат

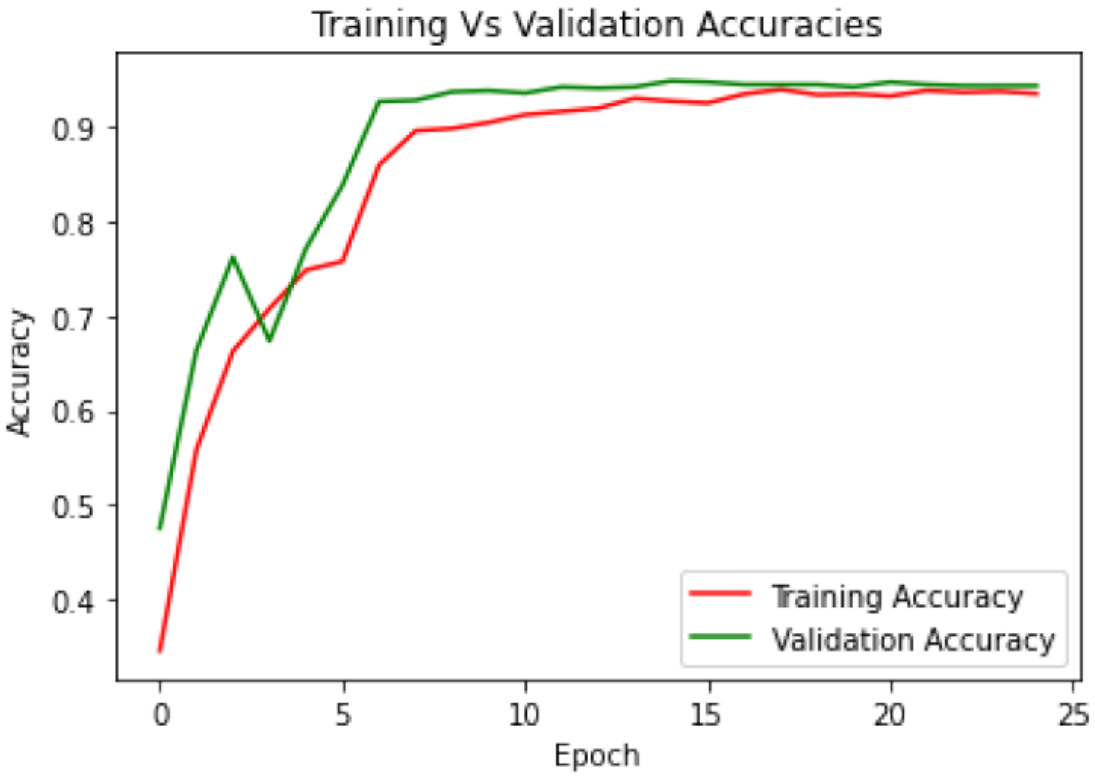

Рис. 3.4 Залежність функції точності

Останнім етапом тренування є тестування моделі. Після завершення тренування модель оцінюється на тестовому наборі даних. Для кожного зображення знаходиться прогнозований клас та порівнюється з дійсним класом для обчислення точності. Обчислюється загальна точність мережі на тестовому наборі даних. Для оцінки результатів класифікації побудована матриця плутанини або помилок (confusion matrix) (рис. 3.5). Ця матриця показує кількість зображень, які були правильно або неправильно класифіковані для кожної пари класів. Матриця плутанини візуалізується у вигляді теплової карти, де кольори відображають кількість зображень.

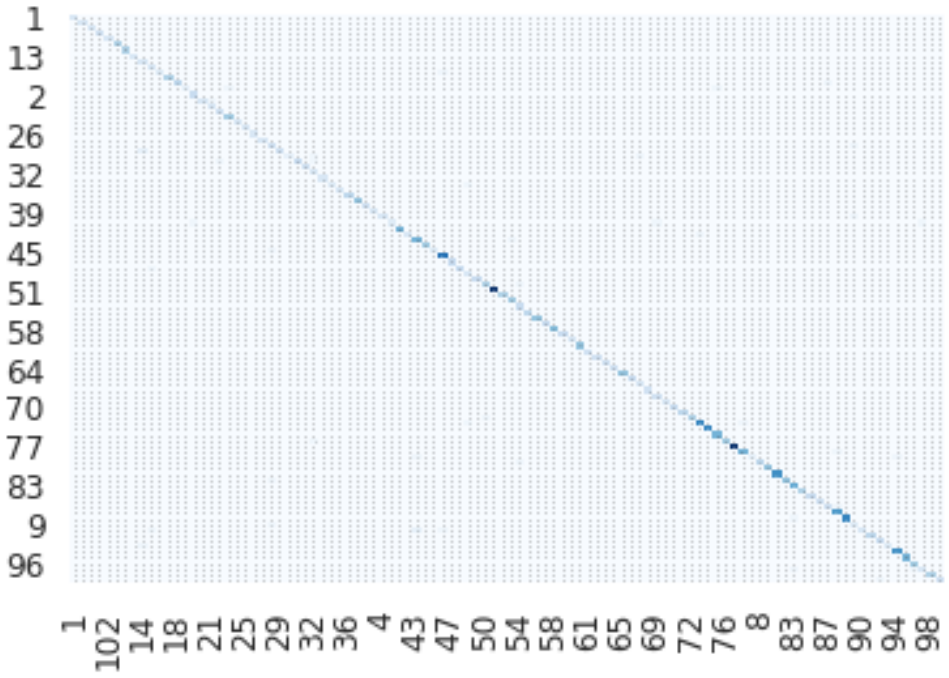

Рис. 3.5 Матриця помилок (плутанини)

Таке тренування мережі дозволяє оновлювати ваги моделі за допомогою градієнтного спуску і покращувати її здатність до класифікації зображень квітів з кожною епохою тренування. Результати тренування моніторяться за допомогою функції втрат та точності, а також візуалізуються за допомогою графіків.

3.3 Використання архітектури YOLOv3

Щоб передбачити коректну роботу нейронної мережі слід врахувати, що при завантаженні даних з соціальної мережі Instagram з хетштегом «Квіти» можуть попадатись зображення не тільки самих квітів, а й інших об’єктів (рис. 3.6). Для цього в роботі буде використана архітектура YOLOv3. У даному розділі виконується класифікація та виявлення об'єктів на наборі вхідних зображень. По черзі обробляються всі зображення зі

списку source_image_names. Для кожного зображення викликається метод detectObjectsFromImage об'єкту obj_detect, передаючи шлях до вхідного зображення та шлях для збереження вихідного зображення з позначеними об'єктами. Результати класифікації та виявлення об'єктів виводяться на екран. Якщо на зображенні не було знайдено жодного об'єкту, то вихідне зображення просто копіюється у вихідну директорію sample_destination. У протилежному випадку, для кожного знайденого об'єкту виводиться його назва, ймовірність (у відсотках) та координати рамки, яка охоплює об'єкт на зображенні (рис. 3.7).

Рис. 3.6 Приклад вхідних даних для обробки та фільтрації фотографій

Рис. 3.7 Приклад вихідних даних для обробки та фільтрації фотографій

Цей розділ показує, як використати модель YOLOv3 для класифікації та виявлення об'єктів на зображеннях. Результатом є фільтрація фотографій, що містять зображення квітів, що дозволяє зосередитися на подальшому аналізі. Використання попередньо навчених моделей дозволяє отримати швидкі та точні результати з обмеженого набору даних.

3.4 Результат виконання програми

Загалом, в набор даних було завантажено 7029 зображень, з яких 6639 – зображення квітів та 390 – зображень не квітів. Нейронна мережа

продемонструвала здатність правильно класифікувати різні види квітів, див. рис. 3.8. Дані з соціальної мережі були прив'язані до регіону Великої Британії, а саме Англії. Найпопулярнішою квіткою з завантаженого набору фотографій є квітка «morning glory». Нейронна мережа виявила 1899 фотографій з нею. Також на другому та третьому місці є квітки «canna lily» з результатом 1103 зображень та «globe-flower» з результатом 531 зображення.

У деяких випадках модель може бути схильною до помилок. Наприклад, для деяких видів квітів, які мають схожі особливості або вигляд, модель може робити неправильні прогнози. Однак загальна точність моделі залишається високою.

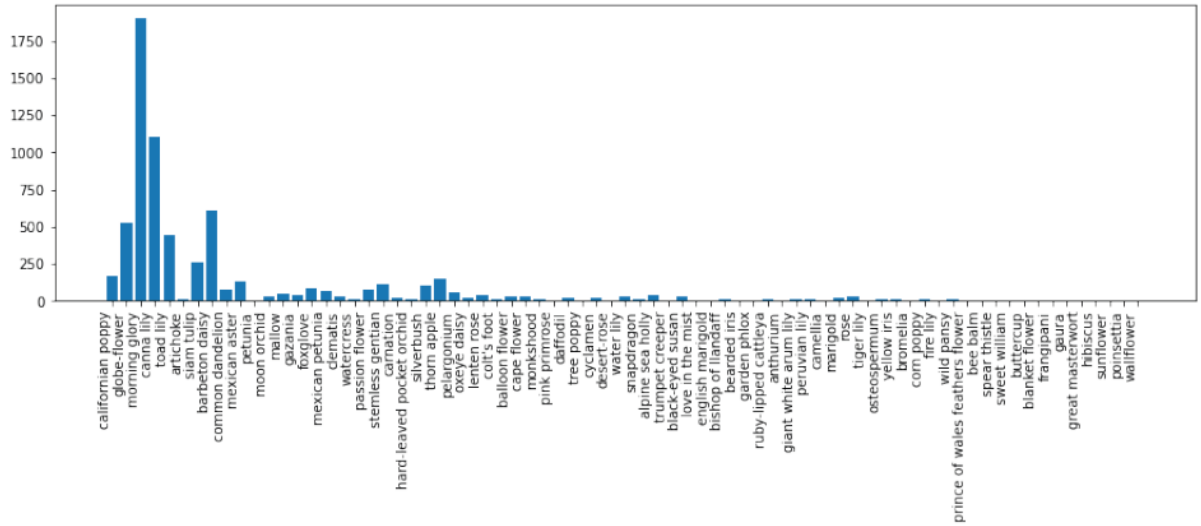

Рис. 3.8 Результати популярності квітів

Варто зауважити, що під час обробки фотографій з Instagram було здійснено певну підготовку даних, включаючи попередню обробку та препроцесинг зображень. Для покращення якості вхідних даних для нейронної мережі використовувалися такі інструменти, як нормалізація та змінення розміру зображень.

Отримані результати підтверджують ефективність розробленої нейронної мережі в розпізнаванні та класифікації квітів. Використання соціальної мережі Instagram як джерела даних дозволило отримати значну кількість зображень квітів для тренування моделі. Щоб покращити точність і загальну продуктивність нейронної мережі, подальші дослідження можуть включати розширення набору даних за допомогою залучення інших джерел і більшої різноманітності квітів.

Висновки

У цій роботі використано модель глибокого навчання ResNet-18, попередньо навчену на великому наборі даних ImageNet. Модель була додатково оптимізована та переконфігурована для вирішення проблеми класифікації зображень для нового набору, що містить 102 класи. Для підвищення точності моделі було застосовано функцію втрати CrossEntropyLoss і оптимізатор Adam, який мав параметри навчання швидкості = 0.001 і бета-коефіцієнти 0,9 і 0,999. Крім того, застосовано планувальник швидкості навчання StepLR, який зменшував швидкість навчання в 10 разів кожні сім епох, щоб покращити точність моделі та запобігти перенавчанню.

За допомогою нейронної мережі та даних з соціальної мережі Instagram вдалось з’ясувати, що найпопулярнішою квіткою є "morning glory". Вона становить 27% від загальної кількості зображень.

Після тренування моделі на наборі даних отримано дуже задовільні результати. Найкраща точність досягнута на валідаційному наборі даних і становить 94%. Це демонструє ефективність моделі в класифікації зображень на новому наборі даних із 102 класів.

Загалом результати цієї дипломної роботи підтверджують ефективність використання глибокого навчання та попередньо натренованих моделей для завдань класифікації зображень. Отримана модель може бути використана для автоматичного розпізнавання об’єктів на зображеннях з високою точністю та надійністю в багатьох галузях, таких як медицина, флористика та сільське господарство.

Список літератури

1. Terrasoft. "Digital Transformation." [Онлайн]. Доступно: https://www.terrasoft.ua/page/digital-transformation. (Дата звернення 10.10.2022)
2. Uaspectr. "Найпопулярніші соціальні мережі в Україні та країнах світу 2020." [Онлайн]. Доступно: https://uaspectr.com/2020/06/23/najpopulyarnishi-sotsialni-merezhi-v-ukrayini-ta-krayinah-svitu-2020/. (Дата звернення 10.10.2022)
3. Marketer. "The Most Popular Social Networks in the World as of January 2022." [Онлайн]. Доступно: https://marketer.ua/ua/the-most-popular-social-networks-in-the-world-as-of-january-2022/. (Дата звернення 10.10.2022)
4. Stud.com.ua. "Особливості призначення експертних систем." [Онлайн]. Доступно: https://stud.com.ua/158232/informatika/osoblivosti_priznachennya_ekspertnih_sistem. (Дата звернення 15.10.2022)
5. Insights Imaging. "Radiomics: the bridge between medical imaging and personalized medicine." [Онлайн]. Доступно: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9. (Дата звернення 08.11.2022)
6. Wikiwand. "Згорткова нейронна мережа." [Онлайн]. Доступно: https://www.wikiwand.com/uk/%D0%97%D0%B3%D0%BE%D1%80%D1%82%D0%BA%D0%BE%D0%B2%D0%B0_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0. (Дата звернення 11.11.2022)
7. TechUkraine. "Згорткові нейронні мережі (CNN): вступ." [Онлайн]. Доступно: https://techukraine.net/%D0%B7%D0%B3%D0%BE%D1%80%D1%82%D0%BA%D0%BE%D0%B2%D1%96-

%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D1%96-%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D1%96-cnn-%D0%B2%D1%81%D1%82%D1%83%D0%BF/Wikipedia. (Дата звернення 11.11.2022)

8. "Activation function." [Онлайн]. Доступно: https://en.wikipedia.org/wiki/Activation_function. (Дата звернення 17.11.2022)
9. Poduska J. "Top Open Source Tools for Deep Learning." [Онлайн]. Доступно: https://www.rtinsights.com/top-deep-learning-tools/. (Дата звернення 18.11.2022)
10. Data Driven Investor. "Introducing Transfer Learning as Your Next Engine to Drive Future Innovations." [Онлайн]. Доступно: https://medium.datadriveninvestor.com/introducing-transfer-learning-as-your-next-engine-to-drive-future-innovations-5e81a15bb567. (Дата звернення 14.12.2022)
11. deepsense.ai. "Keras vs. PyTorch – AVP Transfer Learning." [Онлайн]. Доступно: https://deepsense.ai/keras-vs-pytorch-avp-transfer-learning/. (Дата звернення 16.01.2023)
12. Machine Learning Mastery. "Object Recognition with Deep Learning." [Онлайн]. Доступно: https://machinelearningmastery.com/object-recognition-with-deep-learning/. (Дата звернення 16.01.2023)
13. anten475. "Image Classification vs Object Detection." [Онлайн]. Доступно: https://anten475.blogspot.com/2021/03/get-43-image-classification-vs-object.html. (Дата звернення 16.01.2023)
14. Machine Learning Mastery. "Adam Optimization Algorithm for Deep Learning." [Онлайн]. Доступно: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. (Дата звернення 27.02.2023)

15. ResearchGate. "Principial architecture of CNN." [Онлайн]. Доступно: https://www.researchgate.net/figure/Principial-architecture-of-CNN-3_fig1_366289843. (Дата звернення 08.11.2023)
16. "Principial architecture of CNN" - ResearchGate [Онлайн]. Доступно: https://www.researchgate.net/figure/Principial-architecture-of-CNN-3_fig1_366289843. (Дата звернення 11.11.2022)
17. "Transfer Learning with VGG16 Architecture" - Medium [Онлайн]. Доступно: https://1197.medium.com/transfer-learning-with-vgg16-architecture-6685431218dd. (Дата звернення 14.12.2022)
18. "Keras vs PyTorch: AVP Transfer Learning" - freeCodeCamp [Онлайн]. Доступно: https://www.freecodecamp.org/news/keras-vs-pytorch-avp-transfer-learning-c8b852c31f02/. (Дата звернення 14.12.2022)
19. "A Gentle Introduction to YOLO v4 for Object Detection in Ubuntu 20.04" - Robocademy [Онлайн]. Доступно: https://robocademy.com/2020/05/01/a-gentle-introduction-to-yolo-v4-for-object-detection-in-ubuntu-20-04/. (Дата звернення 16.01.2023)
20. "Papers with Code: Adam Optimization Algorithm" - Papers with Code [Онлайн]. Доступно: https://paperswithcode.com/method/adam. (Дата звернення 27.02.2023)