МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп'ютерних наук та кібербезпеки

МАЛЯРЧУК БОГДАНА МИКОЛАЇВНА

ДОСЛІДЖЕННЯ МЕТОДІВ КОМП'ЮТЕРНОГО ЗОРУ ДЛЯ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ПРИКЛАДІ РОЗПІЗНАВАННЯ ВИДІВ КВІТІВ

Спеціальність: 122 «Комп'ютерні науки»
Освітньо-професійна програма Комп'ютерні науки та інформаційні технології
Робота над здобуттям освітнього ступеня «бакалавр»

Науковий керівник:
БУЛАТЕЦЬКА ЛЕСЯ ВІТАЛІЇВНА,
доцент кафедри комп'ютерних наук та кібербезпеки

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол № _____
засідання кафедри _____
від _____ 2025 р.
Завідувач кафедри
(_____)_____

Луцьк - 2025

ЗМІСТ

ВСТУП

У XXI столітті технології комп’ютерного зору стали одним із провідних напрямів розвитку штучного інтелекту та цифрової трансформації суспільства. Ці технології дозволяють комп’ютерам сприймати, аналізувати та інтерпретувати візуальну інформацію подібно до того, як це робить людський зір. Від розпізнавання облич до автономного керування автомобілями, від контролю якості продукції на виробництві до медичної діагностики. Сфери застосування комп’ютерного зору охоплюють практично всі галузі сучасної економіки, науки та побуту. Серед таких напрямів особливе місце займає класифікація об’єктів на зображеннях, що відкриває нові можливості для ботаніки, екології та аграрної науки.

Комп’ютерний зір (або комп’ютерне бачення) це галузь досліджень і технологій, яка спрямована на розробку систем, здатних до автоматичного виявлення, відстеження та розпізнавання об’єктів у зображеннях або відеопотоці. Як наукова дисципліна, комп’ютерний зір належить до теорії та технології створення штучних систем, які отримують інформацію у вигляді зображень. [1]

З-поміж усіх технологій, що використовуються у комп’ютерному зорі, однією з найефективніших є згорткові нейронні мережі (Convolutional Neural Networks, CNN). "Згорткові нейронні мережі (CNN) – це клас глибинних штучних нейронних мереж прямого поширення, які формують основу комп’ютерного зору та обробки зображень. Вони були розроблені для більш ефективної обробки зображень." [2] Ці мережі наслідують принципи функціонування візуальної кори головного мозку людини, дозволяючи машині ідентифікувати найдрібніші особливості структури об’єкта.

Дослідження у галузі комп’ютерного зору з кожним роком набувають все більшого значення, особливо з огляду на величезну кількість візуальних даних, що генеруються щодня. У зв’язку з цим зростає попит на створення інтелектуальних систем, які можуть автоматично обробляти, сортувати та аналізувати зображення з високим рівнем точності та надійності. Розробка таких

систем вимагає поєднання знань у галузі машинного навчання, цифрової обробки зображень, програмування та роботи з відкритими даними.

Ця робота присвячена розробці програмного забезпечення для класифікації квітів за зображенням із використанням глибоких нейронних мереж. Реалізований додаток Blossom Lens демонструє можливість поєднання наукових підходів з прикладним програмуванням, створюючи інструмент, здатний виконувати автоматичну ідентифікацію об’єктів у зручній та зрозумілій формі для користувача. Додаток поєднує функції нейромережевого розпізнавання, інтеграції з відкритими енциклопедичними джерелами, локального збереження результатів і двомовної підтримки, що робить його придатним для широкого спектра завдань – від навчальних демонстрацій до професійного застосування.

Метою роботи є дослідження методів комп’ютерного зору для задач класифікації зображень і розробка прикладної програми для автоматичного розпізнавання квітів.

Для досягнення цієї мети потрібно виконати наступні завдання:

- проаналізувати теоретичні засади комп’ютерного зору;
- обрати оптимальну архітектуру нейронної мережі;
- здійснити підбір та обробку набору даних;
- провести навчання моделі;
- розробити інтерфейс користувача з можливістю двомовного відображення інформації;
- реалізувати механізми збереження класифікаційних результатів у текстових та PDF-форматах.

Практична цінність роботи полягає у створенні доступного та функціонального інструменту, який може застосовуватись у навчальному процесі, наукових дослідженнях і повсякденному використанні для ідентифікації об’єктів флори. Застосування таких рішень сприяє підвищенню рівня цифрової грамотності, інтеграції ІКТ у природничі дисципліни та розвитку практичних навичок роботи з нейромережами. Результати дослідження можуть бути

використані як база для подальших розробок у галузі штучного інтелекту та автоматизованої обробки візуальних даних.

Об’єктом дослідження є методи глибинного навчання для класифікації зображень об’єктів флори, а також програмні засоби реалізації системи Blossom Lens із підтримкою інтерактивного інтерфейсу, багатомовності, шифрування результатів і генерації звітів.

Предметом дослідження є методи та алгоритми глибинного навчання, що використовуються для класифікації зображень квітів, а також інструменти реалізації графічного інтерфейсу користувача, локалізації та виведення результатів у рамках програмного комплексу Blossom Lens.

Публікації:

Малярчук Б. Архітектура та принцип дії моделі глибокого навчання для розпізнавання зображень у BLOSSOM LENS. Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем.: матеріали II міжнар. науково-практ. конф. Луцьк, 9–11 червн. 2025 р. Луцьк, 2025. С. 214-215.

Малярчук Б. Важливість технологій комп’ютерного зору у сучасному цифровому суспільстві. Математика. Інформаційні технології. Освіта.: матеріали XIV міжнар. науково-практ. конф. Луцьк, 13–15 червн. 2025 р. Луцьк, 2025. С. 114-116.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ ОСНОВИ КОМП'ЮТЕРНОГО ЗОРУ ТА ГЛИБОКОГО НАВЧАННЯ

1.1 Поняття комп'ютерного зору

Комп'ютерний зір є одним із найважливіших напрямів розвитку сучасного штучного інтелекту, який забезпечує можливість машинного аналізу зорової інформації. Його основна мета полягає у тому, щоб надати комп'ютерам здатність "бачити" – тобто сприймати, розпізнавати, інтерпретувати та використовувати візуальні дані для виконання конкретних завдань. Якщо раніше візуальне розпізнавання вважалося виключно прерогативою біологічних систем, то нині алгоритми комп'ютерного зору успішно конкурують з людським сприйняттям за точністю, швидкістю та масштабністю обробки зображень.

Суть комп'ютерного зору полягає у перетворенні піксельної інформації, отриманої з фото- або відеозображень, у структуровані та інтерпретовані моделі, на основі яких можливо здійснювати класифікацію об'єктів, виявлення аномалій, відстеження руху або побудову тривимірної геометрії сцени. Обробка такого роду інформації передбачає проходження ряду етапів, серед яких попередня обробка зображення, виділення ознак, формування векторів ознак, застосування моделей розпізнавання та інтерпретація результатів. На кожному з цих етапів використовуються спеціалізовані алгоритми, які забезпечують високу точність навіть у складних умовах.

"Комп'ютерний зір (Computer Vision, CV) виконує важливі завдання в охоронних системах, медицині, на транспорті та, звичайно, на виробництві, де на додаток до оптимізації процесів допомагає створювати безпечні умови праці." [3] У медицині він використовується для автоматизованої діагностики захворювань на основі знімків МРТ, КТ або УЗД, у тому числі для виявлення пухлин, аномалій судин чи структурних змін тканин. У транспорті він інтегрований у системи допомоги водію (ADAS) та автопілоти, де відповідає за розпізнавання дорожніх знаків, пішоходів, інших транспортних засобів та умов дорожнього руху. У

промисловості – для контролю якості, ідентифікації дефектів, автоматизації складських та логістичних процесів. У сільському господарстві технології комп'ютерного зору використовуються для моніторингу стану рослин, виявлення бур'янів, оцінювання врожайності та підтримки точного землеробства. Також варто згадати системи біометричної ідентифікації, аналізу поведінки, безпекового відеоспостереження, цифрових помічників та візуальної навігації дронів – усі ці сфери базуються на можливостях комп'ютерного зору.

До завдань, які виконує комп'ютерний зір, належать класифікація об'єктів, сегментація зображень, розпізнавання ознак, виявлення руху, реконструкція 3D-сцен, розпізнавання емоцій, оцінка поз та орієнтації тіла. Виконання таких завдань стало можливим завдяки розвитку методів глибокого навчання, що дозволяють нейронним мережам самостійно формувати оптимальні вектори ознак і коректно класифікувати складні візуальні патерни. Особливо ефективними у цьому контексті виявились згорткові нейронні мережі (CNN), які автоматично виділяють ієрархії ознак, починаючи з базових (контури, кольори) й закінчуючи складними (форми, текстури, просторові структури).

Комп'ютерний зір сьогодні виступає як міждисциплінарна технологія, що об'єднує інформатику, обчислювальну математику, нейробіологію, цифрову фотографію, графічний дизайн, а також етику даних. Його активне впровадження дозволяє не лише підвищити продуктивність процесів, а й значно скоротити витрати людських ресурсів, забезпечуючи швидку, стабільну та об'єктивну обробку візуальної інформації. Отже, у сучасному інформаційному суспільстві технології комп'ютерного зору відіграють ключову роль у створенні розумних систем і формуванні нової цифрової культури взаємодії людини з навколишнім світом.

1.2 Основи глибокого навчання у візуальній класифікації

Глибоке навчання (deep learning) є однією з провідних технологій штучного інтелекту, що суттєво змінила підходи до автоматизованого аналізу візуальної

інформації. "Глибоке навчання – це лише тип машинного навчання, натхненний структурою людського мозку. Алгоритми глибокого навчання намагаються зробити такі ж висновки, як і люди, шляхом постійного аналізу даних із заданою логічною структурою. Щоб досягти цього, глибоке навчання використовує багаторівневі структури алгоритмів, які називаються нейронними мережами." [4] На відміну від традиційних методів машинного навчання, які покладаються на вручну створені ознаки для класифікації об’єктів, глибокі нейронні мережі здатні самостійно навчатися виявляти й узагальнювати характеристики вхідних даних завдяки великій кількості внутрішніх параметрів і багатошаровій архітектурі. Це забезпечує високу гнучкість і здатність працювати з дуже складними задачами – зокрема, розпізнаванням об’єктів на зображеннях.

Особливу роль у цій галузі відіграють згорткові нейронні мережі (Convolutional Neural Networks, CNN), які стали стандартом де-факто у комп’ютерному зорі. Архітектура CNN натхненна структурою зорової кори головного мозку, де певні нейрони активуються лише на специфічні ознаки – краї, кути, текстури тощо. Завдяки цьому такі мережі можуть виділяти локальні особливості зображень та поступово переходити до абстрактніших рівнів представлення. Класичні шари CNN включають згорткові (convolutional), які витягують ознаки, підвибіркові (pooling), які зменшують розмірність і зберігають найбільш суттєві характеристики, а також повнозв’язні (dense), що реалізують функцію класифікації.

Процес навчання глибокої моделі передбачає подання їй великої кількості позначених прикладів (у випадку класифікації – зображень із вказаними класами), обчислення помилки між передбаченням і правильним значенням, а також поступову зміну ваг мережі з метою мінімізації цієї помилки. Це відбувається за допомогою алгоритмів оптимізації, серед яких найпоширенішим є стохастичний градієнтний спуск (SGD) та його варіанти – Adam, RMSProp тощо. Кожен параметр мережі оновлюється відповідно до градієнта функції втрат, що дає змогу моделі поліпшувати свою здатність до класифікації з кожною ітерацією.

Однією з важливих умов ефективного навчання є попередня обробка даних (препроцесинг). У задачах класифікації зображень це включає зміну розміру вхідних зображень до єдиного формату, нормалізацію значень пікселів, а також аугментацію – штучне розширення навчального набору за допомогою трансформацій, таких як обертання, дзеркальне відображення, масштабування, зміна яскравості тощо. Це дозволяє зменшити ризик перенавчання та підвищити здатність моделі до узагальнення.

У сучасних підходах широко використовується концепція трансферного навчання (transfer learning), за якою модель, попередньо навчена на великому універсальному наборі зображень (наприклад, ImageNet), адаптується до нової задачі за допомогою донавчання останніх шарів або тонкого налаштування всієї архітектури. Це значно пришвидшує навчання і дозволяє досягти високої точності навіть при обмеженій кількості власних навчальних даних. Однією з найефективніших моделей у цьому напрямі є EfficientNet – збалансована архітектура, що масштабується за глибиною, шириною та роздільністю зображень, і забезпечує високу продуктивність при низьких обчислювальних витратах.

Важливим елементом успішного навчання є також вибір функції втрат. У задачах багатокласової класифікації типовою є функція categorical crossentropy, яка обчислює відстань між реальними та передбаченими ймовірностями класів. У поєднанні з функцією активації softmax на останньому шарі вона дозволяє мережі видавати ймовірнісний розподіл по всіх можливих класах, з якого обирається клас із найбільшою ймовірністю як кінцеве передбачення.

У процесі навчання та оцінювання моделі застосовуються додаткові інструменти контролю – такі як callback-функції EarlyStopping (зупинка навчання при відсутності покращення) та ReduceLROnPlateau (зменшення швидкості навчання при плато валідаційної похибки). Окрім цього, для об’єктивного оцінювання моделі використовуються метрики точності (accuracy), precision, recall, F1-score та побудова матриці плутанини (confusion matrix), яка дає змогу візуально оцінити помилки класифікації між окремими класами.

Таким чином, глибоке навчання забезпечує потужну математичну основу для створення високоточних систем візуальної класифікації. Поєднання згорткових мереж, оптимізаційних алгоритмів, технік аугментації та підходів трансферного навчання дозволяє реалізувати універсальні й адаптивні рішення для широкого спектра прикладних задач – від біоінформатики до автоматизованого управління. У контексті класифікації зображень квітів це відкриває можливості для створення програмних інструментів, здатних підтримувати наукові дослідження, навчальний процес і цифрову трансформацію природничих знань.

1.3 Архітектури згорткових нейронних мереж (CNN)

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) є ключовим елементом у розвитку комп'ютерного зору завдяки своїй здатності ефективно опрацьовувати візуальні дані. Ці моделі спеціалізуються на вилученні просторових ознак із зображень, що дозволяє їм ідентифікувати складні шаблони, текстури та геометричні структури, необхідні для точного розпізнавання об'єктів. Архітектура CNN складається з послідовності шарів, кожен з яких виконує специфічну функцію в процесі обробки зображення – від початкового вилучення низькорівневих ознак до фінального прийняття класифікаційного рішення.

Типова архітектура згорткової нейронної мережі включає згорткові шари (convolutional layers), які виконують фільтрацію зображень за допомогою невеликих ядер (kernel) розміром, як правило, 3×3 або 5×5. Ці ядра ковзають по зображенню та створюють карти ознак (feature maps), що відображають наявність певних локальних шаблонів. Зазвичай після згорткового шару додається шар активації, найчастіше функція ReLU (Rectified Linear Unit), яка забезпечує нелінійність мережі та покращує її здатність до апроксимації складних функцій.

Наступним важливим компонентом є шари підвибірки (pooling layers), які зменшують просторові розміри карт ознак, зберігаючи при цьому найбільш

значущу інформацію. Найпоширенішим є максимальний пулінг (max pooling), що дозволяє зменшити кількість параметрів та обчислювальні витрати, водночас забезпечуючи інваріантність до незначних змін у вході, таких як зсуви чи масштабування.

На завершальних етапах обробки мережа зазвичай містить один або кілька повнозв'язних шарів (dense layers), які здійснюють узагальнення всіх попередніх ознак і формують підсумкове передбачення. Завершується модель, як правило, softmax-шаром, який генерує ймовірності належності об'єкта до кожного з можливих класів.

Серед найвідоміших і найефективніших архітектур CNN варто виділити LeNet, AlexNet, VGG, Inception, ResNet та EfficientNet. LeNet-5, запропонована у 1998 році, стала однією з перших успішних архітектур для задач класифікації рукописних цифр. AlexNet, представлена у 2012 році, продемонструвала революційний прорив, здобувши перемогу в ImageNet Challenge і довівши практичну перевагу глибоких згорткових мереж. VGG, відома завдяки своїй простоті та використанню однорідних 3×3 фільтрів, дозволила досягти високої точності при збереженні прозорої структури.

Інша значуща архітектура – Inception – базується на концепції багатоканального вилучення ознак, де на одному рівні застосовуються фільтри різного розміру. ResNet (Residual Network) запровадила ідею залишкових зв'язків (skip connections), які полегшують тренування дуже глибоких мереж, запобігаючи явищу згасання градієнта. Саме завдяки ResNet стало можливим створення моделей із сотнями шарів.

Особливе місце серед сучасних архітектур займає EfficientNet. "EfficientNet – це сімейство згорткових нейронних мереж (CNN) для комп'ютерного зору, опубліковане дослідниками Google AI у 2019 році. Їхньою головною інновацією є комбіноване масштабування, яке рівномірно масштабує всі параметри – глибину, ширину та роздільну здатність – за допомогою одного параметра.

Моделі EfficientNet були впроваджені у різні задачі комп’ютерного зору, зокрема класифікацію зображень, виявлення об’єктів та сегментацію." (Переклад наш – М.Б.) [5]

Такий підхід дозволяє зберігати баланс між точністю та обчислювальними витратами, роблячи EfficientNet надзвичайно ефективною для практичних задач. Саме ця архітектура була обрана у рамках реалізації програми Blossom Lens, оскільки вона демонструє високу продуктивність на обмежених ресурсах, що ідеально підходить для десктопних або мобільних застосунків.

Загалом, вибір архітектури нейронної мережі є ключовим фактором успіху будь-якої системи комп’ютерного зору. Він залежить від обсягу даних, доступних обчислювальних потужностей та вимог до точності й швидкодії. Гнучкість сучасних фреймворків глибокого навчання, таких як TensorFlow або PyTorch, дає змогу швидко тестувати різні архітектури, налаштовувати їх та адаптувати під специфіку конкретного застосування. У випадку задачі класифікації зображень квітів, глибина і збалансованість EfficientNet виявились найкращим варіантом, що поєднує точність, швидкість і компактність.

1.4 Підходи до класифікації зображень

Класифікація зображень у системах комп’ютерного зору є фундаментальним завданням, що реалізується за допомогою методів глибокого навчання. Повноцінний процес побудови системи класифікації включає кілька послідовних етапів: підготовку даних, навчання моделі та оцінку її якості. Кожен із цих етапів відіграє ключову роль у забезпеченні точності та надійності функціонування системи.

Початковим етапом класифікації є попередня обробка зображень. Вона передбачає приведення всіх вхідних даних до уніфікованого формату, який відповідає технічним вимогам нейронної мережі. Зображення, як правило, масштабується до однакового розміру, наприклад 224×224 пікселі, що спрощує обробку та дозволяє використовувати фіксовану архітектуру мережі.

Нормалізація значень пікселів до діапазону [0; 1] або [-1; 1] забезпечує стабільну роботу алгоритмів оптимізації, запобігає числовій нестабільності та пришвидшує збіжність під час навчання.

Аугментація зображень є ще одним важливим аспектом попередньої обробки. Вона полягає у створенні нових зразків навчальної вибірки на основі існуючих шляхом випадкових геометричних та кольорових трансформацій: обертання, масштабування, зсув, горизонтальне або вертикальне віддзеркалення, зміна яскравості, контрасту та насиченості кольору. Аугментація сприяє підвищенню узагальнюючої здатності моделі, зменшує ризик перенавчання та дозволяє імітувати реальні умови варіацій зображень, з якими система може стикатися в експлуатації.

Наступний етап – навчання нейронної мережі. Його сутність полягає в оптимізації ваг зв’язків у моделі на основі великої кількості розмічених прикладів. Як правило, використовуються згорткові нейронні мережі (CNN), що складаються з шарів згортки, підвибірки (pooling), нормалізації та повнозв’язних шарів. Згорткові шари виявляють локальні шаблони, підвибірка зменшує розмірність простору ознак, а повнозв’язні шари виконують класифікацію на основі узагальненого вектору ознак.

Процес навчання керується функцією втрат (loss function), яка кількісно визначає відхилення прогнозу моделі від реального результату. У задачах багатокласової класифікації найчастіше застосовується функція крос-ентропії (categorical crossentropy або sparse categorical crossentropy), яка забезпечує ефективне оновлення ваг у випадках, коли вихідні значення належать до одного з кількох класів. Для оптимізації функції втрат використовуються методи градієнтного спуску, зокрема такі варіанти, як Adam, RMSProp або SGD, що автоматично адаптують швидкість навчання залежно від особливостей простору параметрів.

У практиці машинного навчання широко застосовується розподіл набору даних на навчальну, валідаційну та тестову вибірки. Типовим є співвідношення 70:20:10, де 70% зображень використовується для безпосереднього навчання

моделі, 20% – для контролю її якості в процесі навчання (валідація), а решта 10% – для підсумкової оцінки продуктивності на нових даних, які не були використані під час оптимізації параметрів. Це дозволяє об’єктивно оцінити узагальнюючу здатність моделі.

Завершальний етап – оцінка якості навченої моделі. Основними метриками є точність класифікації (accuracy), повнота (recall), точність (precision) та F1-міра. Ці показники дозволяють кількісно оцінити ефективність моделі на тестових даних. Крім того, важливою частиною аналізу є побудова матриці плутанини, яка наочно відображає, як часто модель помиляється між класами. Наприклад, якщо класи мають подібні візуальні ознаки, це може спричинити велику кількість помилок між ними, що вимагає вдосконалення архітектури моделі або розширення навчального набору.

1.5 Огляд технологій Python, TensorFlow, OpenCV та Tkinter

У практиці розробки інтелектуальних систем, орієнтованих на задачі комп’ютерного зору та класифікації зображень, особливу роль відіграє правильний вибір мов програмування та бібліотек. Від цього залежить не лише продуктивність обчислень, а й масштабованість, гнучкість інтеграції та зручність використання програмного продукту. Найбільш поширеним інструментом для створення таких систем сьогодні є мова програмування Python у поєднанні з низкою спеціалізованих бібліотек.

"Python – це інтерпретована, інтерактивна, об’єктно-орієнтована та високорівнева мова програмування загального призначення з динамічною суворою типізацією та автоматичним управлінням пам’яттю, орієнтована на підвищення продуктивності розробника, читабельність коду, а також на забезпечення портування написаних на ній програм." [6] Вона стала провідним інструментом у наукових дослідженнях, обробці даних, машинному навчанні та автоматизації. Популярність Python у сфері штучного інтелекту пояснюється також тим, що більшість сучасних фреймворків для глибокого навчання

створюються з розрахунком саме на цю мову. У рамках одного середовища розробник має змогу реалізувати повний цикл створення інтелектуальної системи: від обробки вхідних даних до виведення інтерфейсу користувача.

TensorFlow – одна з найпотужніших і наймасштабніших платформ для побудови моделей машинного та глибокого навчання. Вона була розроблена дослідницьким підрозділом Google з метою забезпечення обчислювальної ефективності при роботі з великими масивами даних. TensorFlow базується на концепції обчислювальних графів, у яких вузли представляють операції, а ребра – дані, які передаються між ними у вигляді тензорів. Це дозволяє ефективно реалізовувати паралельні обчислення, у тому числі із застосуванням графічних процесорів (GPU) або навіть TPU – спеціалізованих тензорних процесорів.

TensorFlow надає модуль keras, який є високорівневим API для побудови згорткових нейронних мереж (CNN), що дозволяє легко конструювати моделі з багатьма шарами, налаштовувати гіперпараметри, обирати функції активації, оптимізації та втрат. Крім того, TensorFlow має вбудовані функції для візуалізації процесу навчання, збереження та завантаження моделей, а також їх розгортання у виробничих середовищах. Розробники можуть ефективно експериментувати з архітектурою моделі, порівнювати результати та автоматизовувати процес навчання за допомогою колбеків (наприклад, EarlyStopping або ReduceLROnPlateau).

Для обробки зображень до, під час і після класифікації широко застосовується бібліотека OpenCV (Open Source Computer Vision Library). Цей інструмент дозволяє здійснювати читання та збереження зображень, зміну розмірів, перетворення кольорових моделей (наприклад, з BGR у RGB), застосування фільтрів, виявлення контурів, сегментацію, побудову гістограм та багато інших операцій. Однією з основних переваг OpenCV є її швидкодія, що досягається за рахунок реалізації критичних функцій на мові C++ з доступом через Python-інтерфейс. Висока ефективність бібліотеки робить її особливо придатною для задач попередньої обробки зображень перед подачею їх у нейронну мережу.

Tkinter є стандартною бібліотекою для побудови графічного інтерфейсу користувача у Python. Вона дозволяє створювати віконні додатки з такими елементами, як кнопки, текстові поля, мітки, меню, вкладки, комбіновані панелі тощо. Tkinter забезпечує гнучкість у компонуванні інтерфейсних блоків, дозволяє працювати з подіями (натискання кнопок, вибір файлів, прокрутка вмісту) та легко інтегрується з іншими модулями Python. Завдяки своїй простоті й ефективності, Tkinter ідеально підходить для створення прототипів та кінцевих десктопних додатків середнього рівня складності. У контексті систем комп'ютерного зору ця бібліотека дає змогу реалізовувати візуалізацію результатів класифікації, обрати зображення з файлової системи, вивести довідкову інформацію, зберегти результати у форматі TXT або PDF, що є важливою складовою користувацького досвіду.

Узагальнюючи, комбінація Python, TensorFlow, OpenCV та Tkinter формує потужну екосистему для реалізації повноцінних програмних рішень у сфері комп'ютерного зору. Кожна з цих технологій виконує свою ключову функцію: Python забезпечує загальну логіку програми, TensorFlow – побудову та навчання нейромережі, OpenCV – обробку зображень, а Tkinter – зручний доступ користувача до результатів. Завдяки відкритості, широкій документації та спільнотам підтримки ці інструменти є незамінними як у дослідницьких, так і в прикладних проєктах.

1.6 Огляд існуючих аналогічних програмних розробок

Автоматичне розпізнавання рослин є популярним напрямом у сфері комп'ютерного зору та машинного навчання. На сьогодні існує кілька програмних рішень, які реалізують подібні функції, що дозволяють користувачам ідентифікувати рослини за допомогою мобільних додатків або вебсервісів. Аналіз аналогічних програмних розробок допомагає визначити сильні та слабкі сторони конкурентних систем, що є важливим для покращення власного рішення.

1.6.1 Pl@ntNet

Pl@ntNet – це мобільний і вебдодаток для ідентифікації рослин, який створений консорціумом науково-дослідницьких інститутів (Inria, CIRAD, INRAE, IRD). Його основною метою є розпізнавання рослин за фотографіями та надання інформації про їх види. Додаток активно використовується як науковцями, так і аматорами для вивчення флори (рис. 1.1).

Назва: Pl@ntNet

Розробник (дистриб'ютор): Консорціум науково-дослідницьких інститутів (Inria, CIRAD, INRAE, IRD).

Архітектура: Мобільний додаток (iOS, Android), вебдодаток.

Мова реалізації: Java, Python.

Перелік функцій та характеристик:

- розпізнавання рослин за фотографіями листя, квітів, плодів тощо;
- база даних із мільйонами фотографій та описів видів;
- функція фільтрування рослин за географічними регіонами;
- спільнота користувачів для уточнення ідентифікації;
- можливість використання даних у наукових дослідженнях.

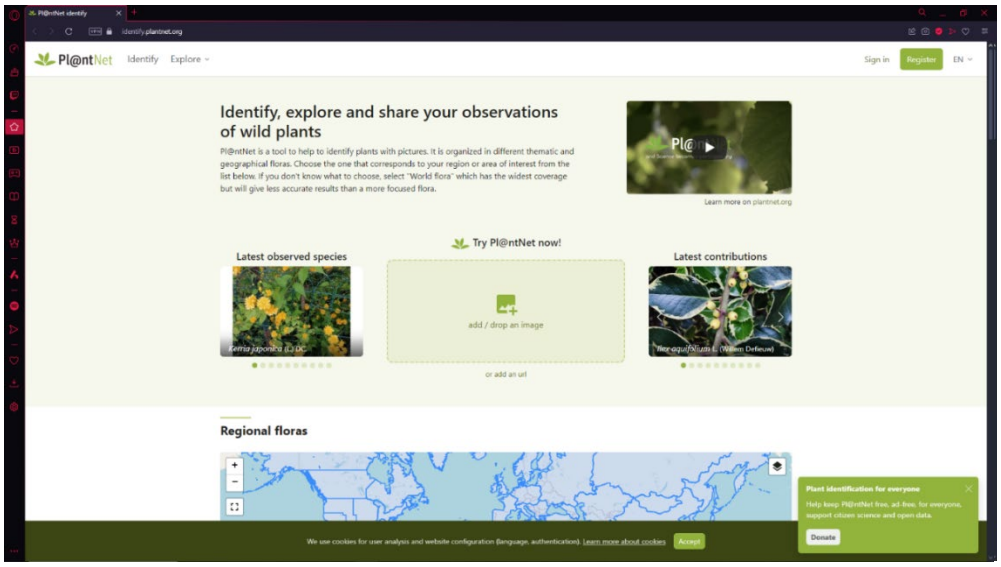
Переваги: велика база даних рослин, яка постійно оновлюється; підтримка спільноти та можливість участі у покращенні бази даних; безкоштовний доступ до основних функцій;

Недоліки: точність розпізнавання залежить від якості фото та унікальності виду; інтерфейс може бути незрозумілим для нових користувачів; додаток потребує інтернет-з'єднання для роботи. Джерело інформації (веб-сайт): https://identify.plantnet.org/uk.

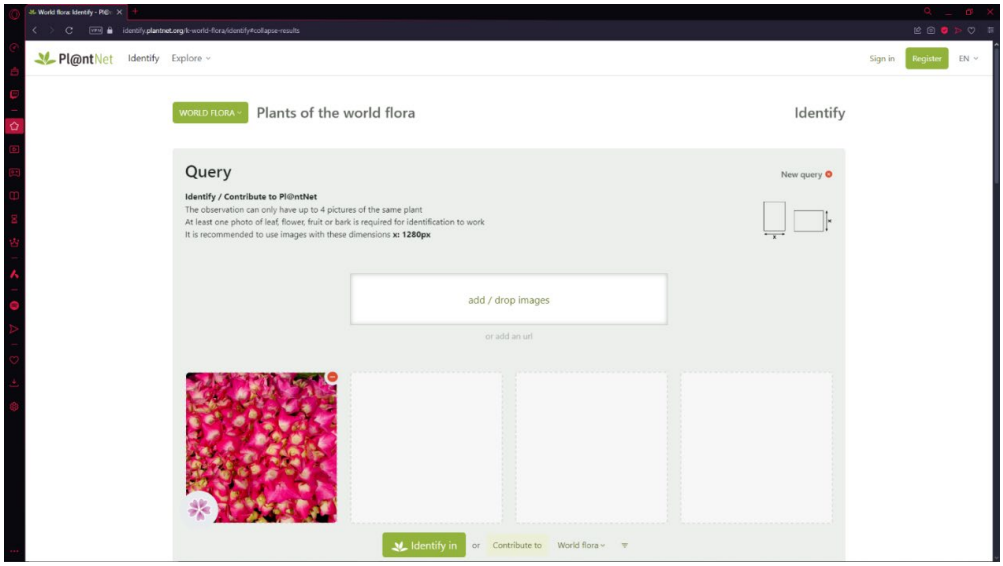
1.6.2. Plant.id

Plant.id – це клієнт-серверний сервіс, що забезпечує розпізнавання рослин за допомогою API, спеціально розроблений для інтеграції з іншими програмними продуктами. Сервіс використовує сучасні нейронні мережі для аналізу зображень і забезпечує високу точність визначення видів (рис. 1.2).

а)



б)



в)

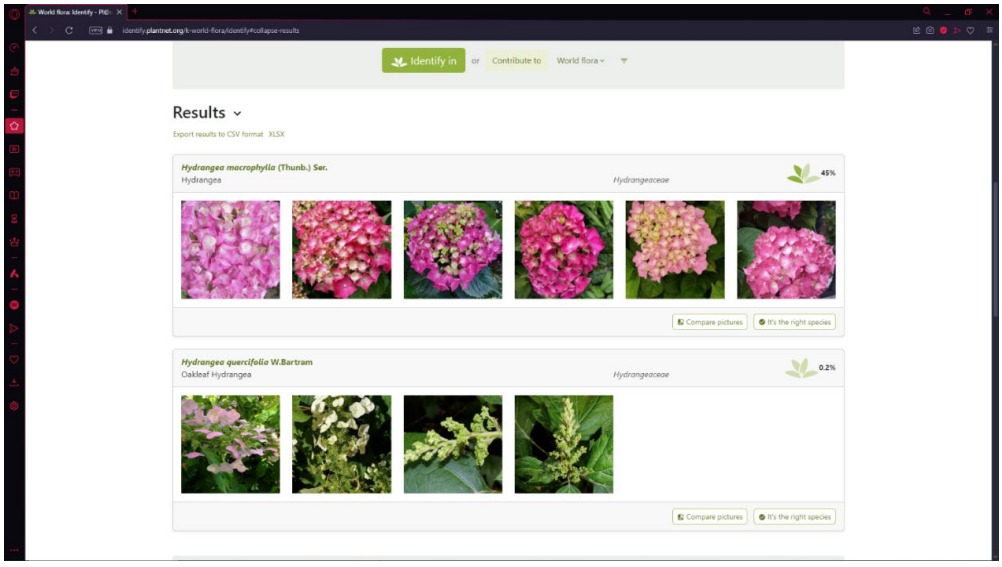
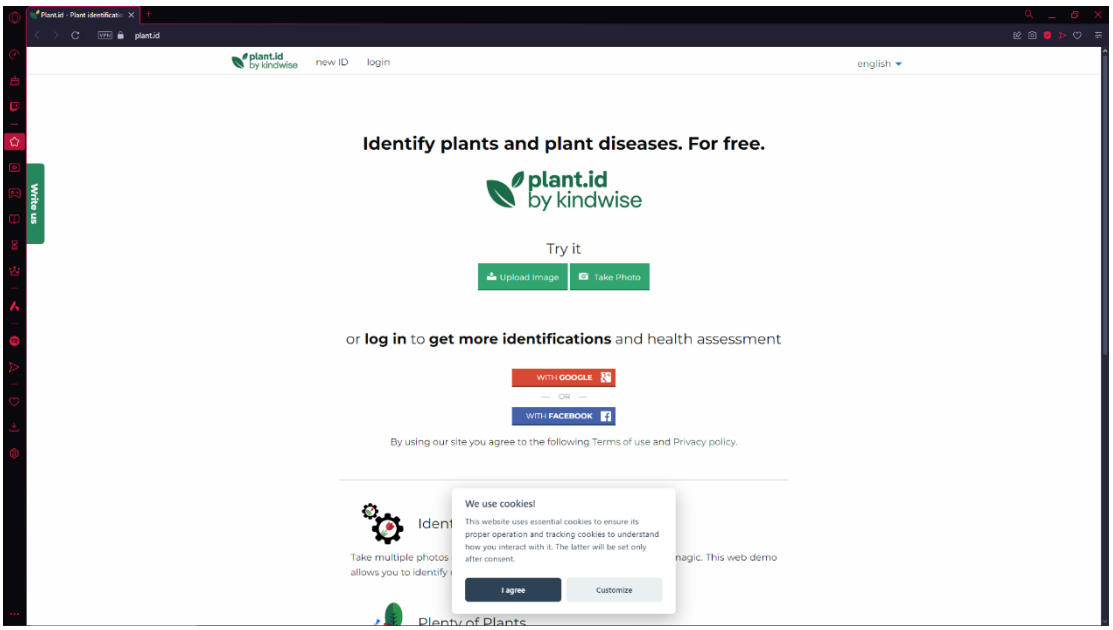


Рисунок 1.1 – Інтерфейс веб-застосунку Pl@ntNet

а)



б)

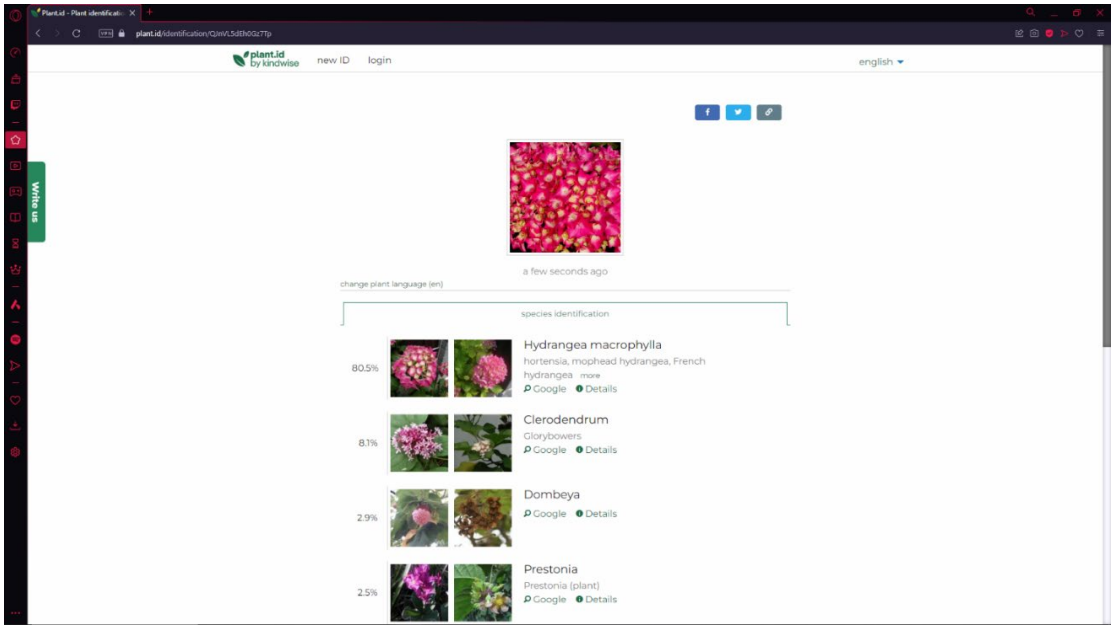


Рисунок 1.2 – Інтерфейс веб-застосунку Plant.id

Назва: Plant.id

Розробник (дистриб’ютор): FlowerChecker s.r.o.

Архітектура: Клієнт-серверний додаток із REST API.

Мова реалізації: Python, REST API для інтеграції.

Перелік функцій та характеристик:

- ідентифікація рослин за допомогою сучасних моделей машинного навчання;
- підтримка аналізу кількох фотографій для підвищення точності;
- виведення рівня впевненості для кожного розпізнаного виду;
- інтеграція з іншими сервісами та програмами через API;
- високий рівень деталізації описів та наукових назв.

Переваги: висока точність завдяки сучасним алгоритмам машинного навчання; можливість кастомізації за допомогою API, деталізований звіт про розпізнавання, включно з рівнем впевненості.

Недоліки: платна модель використання з обмеженнями безкоштовного доступу, потреба в програмуванні для інтеграції, вимагає інтернет-з'єднання.

Джерело інформації (веб-сайт): https://web.plant.id/

1.6.3. PictureThis

PictureThis – це зручний мобільний додаток для побутового розпізнавання рослин. Окрім ідентифікації видів, додаток пропонує поради з догляду за рослинами та діагностику можливих захворювань (рис. 1.3).

Назва: PictureThis

Розробник (дистриб'ютор): Glority LLC.

Архітектура: Мобільний додаток (iOS, Android).

Мова реалізації: Swift (iOS), Kotlin/Java (Android).

Перелік функцій та характеристик:

- розпізнавання квітів, дерев, трав та інших рослин;
- додаткові факти про рослини та їх догляд;
- поради з виявлення хвороб і боротьби зі шкідниками;
- офлайн-режим для використання без інтернету;
- простий і зручний інтерфейс для кінцевого користувача.

Переваги: інтуїтивний інтерфейс, який підходить для початківців, офлайн-доступ до базових функцій, додаткові можливості, як-от догляд за рослинами та виявлення хвороб;

Недоліки: більшість функцій доступна лише в платній версії, основна орієнтація на побутових користувачів, а не наукові цілі, точність розпізнавання може поступатися іншим додаткам. Джерело інформації (веб-сайт): https://www.picturethisai.com/.

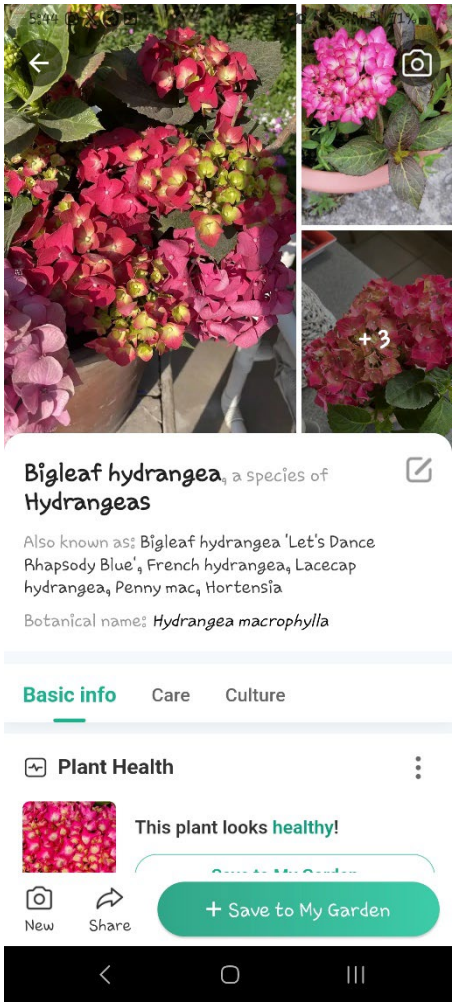


Рисунок 1.3 – Інтерфейс мобільного додатку PictureThis

РОЗДІЛ 2.
РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ ПРОГРАМНОЇ СИСТЕМИ BLOSSOM LENS

2.1 Постановка задачі класифікації зображень квітів

У межах цієї роботи було сформульовано прикладну задачу автоматичної класифікації зображень квітів, яка полягає у розробці програмного забезпечення, здатного визначати вид квітки на основі її зображення. Ця задача належить до одного з центральних напрямів комп'ютерного зору, що має численні практичні застосування в науці, освіті, сільському господарстві, екологічному моніторингу та побуті. Її актуальність обумовлена зростанням обсягів візуальної інформації, яку необхідно обробляти швидко, надійно та без залучення фахівця.

Суть задачі полягає в тому, щоб побудувати систему, яка, отримавши на вхід цифрове зображення квітки, здатна ідентифікувати її належність до певного ботанічного виду. При цьому складність полягає в значній варіативності зовнішнього вигляду рослин, зокрема внаслідок різного освітлення, кута зйомки, якості камери, фону, розмитості тощо. Додатковим викликом є висока схожість між різними видами квітів, які мають подібну будову, форму пелюсток чи колір.

Задача класифікації квітів вимагає не лише точного розпізнавання, а й здатності узагальнювати знання з великої кількості прикладів. Для цього необхідно створити набір даних, що включає якісні, різноманітні та репрезентативні зображення квітів у природних умовах. Зображення мають охоплювати всі основні класи, мати різні ракурси, ступінь освітлення та варіанти фону, щоб забезпечити узагальнювальну здатність моделі.

Подальшим кроком є підготовка зображень: масштабування до уніфікованого розміру, нормалізація кольорових каналів, перетворення формату, а також застосування методів аугментації. Останні дозволяють розширити навчальну вибірку без необхідності збирання нових даних, шляхом створення штучних варіацій зображень – наприклад, обертання, дзеркального відображення, зміни яскравості, контрасту та кольорового балансу. Завдяки

цьому модель навчається розпізнавати квітку незалежно від її позиції чи зовнішніх умов.

В основі задачі лежить використання згорткових нейронних мереж, які ідеально підходять для аналізу зображень. Такі мережі дозволяють витягати з вхідного зображення локальні та глобальні візуальні ознаки – починаючи від елементарних контурів і кольорів до складних морфологічних структур. На основі цих ознак модель поступово формує уявлення про характерні властивості кожного виду квітів. Результатом є здатність моделі прогнозувати клас із певною ймовірністю, базуючись на раніше вивчених прикладах.

Особливу увагу приділено вибору архітектури моделі, яка повинна бути достатньо глибокою, щоб розпізнати складні візуальні патерни, але водночас не надто об’ємною, щоб уникнути перенавчання. Для досягнення високої точності потрібна не лише відповідна кількість шарів у мережі, а й правильно підібрані гіперпараметри, функції активації, метод оптимізації, а також контроль за втратою точності на перевірочній вибірці.

Завершальним етапом є оцінювання точності класифікації. Для цього модель перевіряється на нових зображеннях, які не входили до навчального процесу. Результати класифікації аналізуються з точки зору точності, повноти, кількості правильних та помилкових передбачень. Також проводиться аналіз матриці плутанини, яка дозволяє зрозуміти, які саме види квітів модель найчастіше плутає між собою.

Таким чином, задача класифікації зображень квітів – це комплексна проблема, яка включає збирання та підготовку даних, вибір оптимальної архітектури, навчання моделі, перевірку результатів та подальшу інтеграцію моделі в зручне програмне середовище. Її вирішення потребує як теоретичних знань у галузі комп’ютерного зору та машинного навчання, так і практичних навичок програмування та роботи з сучасними бібліотеками глибокого навчання.

2.2 Побудова та тренування згорткової нейронної мережі на основі EfficientNet

Сучасні задачі класифікації зображень вимагають використання потужних і водночас оптимізованих моделей, здатних ефективно навчатися на складних даних при обмежених обчислювальних ресурсах. Одним із провідних рішень у цьому напрямку є архітектура EfficientNet, яка поєднує високу точність класифікації з ефективним використанням параметрів. У цьому проєкті дана архітектура стала основою побудови згорткової нейронної мережі, що реалізує задачу автоматичного розпізнавання квітів за цифровими зображеннями.

Метою даного етапу є повний цикл створення та навчання глибокої моделі, яка зможе ідентифікувати види квітів на основі зображень з високою точністю. У процесі розробки було здійснено підбір архітектури, підготовку вхідних даних, оптимізацію гіперпараметрів, а також реалізовано процедури оцінки якості класифікації. Особливу увагу було приділено методам запобігання перенавчанню та зменшенню похибки на валідаційній вибірці.

Завдяки впровадженим методикам аугментації, регуляризації, адаптивного зменшення швидкості навчання та використанню крос-валідованих даних, модель досягла стабільної точності класифікації. Подальший аналіз результатів за допомогою матриці плутанини та класифікаційного звіту дозволив зробити висновки щодо її надійності та можливостей для покращення. У наступних підрозділах детально розглядаються всі етапи проєктування, навчання та тестування моделі згорткової нейронної мережі.

2.2.1 Параметри моделі

При розробці нейронної мережі для задачі класифікації зображень квітів було обрано архітектуру EfficientNetB0 як базову модель. Вибір обґрунтований тим, що дана архітектура забезпечує оптимальний баланс між точністю класифікації, кількістю параметрів та швидкістю обчислень. EfficientNet використовує метод масштабування, який пропорційно збільшує глибину,

ширину та роздільну здатність мережі, що забезпечує покращену продуктивність при збереженні компактності.

```
img_height, img_width = 224, 224
batch_size = 32

train_datagen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=15,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

test_datagen = ImageDataGenerator(rescale=1.0/255)

train_generator = train_datagen.flow_from_directory(···

validation_generator = test_datagen.flow_from_directory(···

test_generator = test_datagen.flow_from_directory(···

class_names = list(train_generator.class_indices.keys())
with open('class_names.json', 'w') as f:
    json.dump(class_names, f)

base_model = tf.keras.applications.EfficientNetB0(input_shape=(img_height, img_width, 3),
                                                  include_top=False,
                                                  weights='imagenet')

base_model.trainable = True

model = tf.keras.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(256, activation='relu', kernel_regularizer=tf.keras.regularizers.l2(0.01)),
    layers.Dropout(0.5),
    layers.Dense(train_generator.num_classes, activation='softmax')
])
```

Рисунок 2.1 – Фрагмент коду створення моделі для програми Blossom Lens

Вхідні зображення були приведені до формату 224×224 пікселі з трьома кольоровими каналами (RGB), що відповідає вимогам базової архітектури EfficientNetB0. Модель була завантажена з попередньо навченими вагами на

наборі даних ImageNet, що дало змогу використати знання, отримані під час масштабного попереднього навчання, і донавчити її на специфічному наборі даних квітів.

До основи моделі було додано наступні шари: шар глобального середнього пулінгу (GlobalAveragePooling2D), який зменшує розмірність виходу базової моделі; повнозв'язний шар (Dense) з 256 нейронами та функцією активації ReLU; шар Dropout з коефіцієнтом випадіння 0.5, який виконує регуляризацію та зменшує ризик перенавчання; вихідний шар із кількістю нейронів, що дорівнює кількості класів, та функцією активації softmax, яка генерує ймовірнісний розподіл між класами.

Таким чином, модель має гібридну структуру: ядро EfficientNetB0, яке виконує витягнення ознак, і користувацьку головку, що забезпечує класифікацію згідно з поставленою задачею. Загальна кількість параметрів моделі з урахуванням донавчених шарів становить близько 5 мільйонів, з яких значна частина є тренованими. Ця конфігурація дозволила досягти високої точності при відносно невеликому обсязі ресурсів і часу на навчання.

2.2.2 Порівняння інших типів моделей та їх тренування

До остаточного вибору архітектури EfficientNetB0 було проведено низку експериментів із використанням альтернативних моделей глибокого навчання для задачі класифікації зображень квітів. Це дало змогу оцінити сильні та слабкі сторони різних підходів з погляду точності класифікації, часу навчання, складності реалізації та вимог до обчислювальних ресурсів. У дослідженні були протестовані такі типи моделей: проста згорткова нейронна мережа (Simple CNN), MobileNetV2 та ResNet50.

Як базову модель було реалізовано невелику згорткову нейронну мережу з нуля. Вона складалася з трьох згорткових блоків (Conv2D → MaxPooling2D → ReLU), після чого додавались шари Flatten, Dense (128 нейронів), Dropout (0.3) та вихідний класифікатор. Модель працювала зі зображеннями розміром 128×128 пікселів.

Ця модель показала задовільні результати: на навчальній вибірці вона досягала точності ~95% вже за 20 епох, проте на валідаційній точність не перевищувала 75–78%, що свідчило про перенавчання. Візуалізація втрат підтвердила цю тенденцію: помітне розходження між тренувальними та валідаційними метриками після 10–12 епох. Такий результат був очікуваним, оскільки проста CNN не мала достатньої глибини та регуляризації для обробки складних візуальних патернів у зображеннях квітів.

Другою протестованою архітектурою була MobileNetV2 – компактна, мобільно-оптимізована модель, розроблена спеціально для пристроїв з обмеженими ресурсами. Модель використовувалась з попередньо навченими вагами на ImageNet, із заміною останнього шару на класифікатор із кількістю нейронів, відповідною кількості класів у задачі. Навчання тривало 25 епох із тими самими callback-функціями, що й у випадку з EfficientNet: EarlyStopping і ReduceLROnPlateau.

MobileNetV2 показала дещо гірші результати, ніж EfficientNet, проте перевершила базову CNN. Її точність на валідації становила близько 88%, а середня F1-міра сягала 0.86. Перевагою MobileNet була висока швидкість навчання: одна епоха тривала вдвічі менше часу, ніж для EfficientNet. Проте модель іноді плутала подібні класи – наприклад, ромашку і соняшник – що вказувало на недостатню здатність до узагальнення у складних випадках.

ResNet50, як представник класу глибоких залишкових мереж, демонстрував хорошу здатність до узагальнення, однак мав високу обчислювальну складність. Було помічено, що навіть при використанні лише верхніх шарів для донавчання, кожна епоха займала значно більше часу, а об’єм відеопам’яті під час навчання перевищував допустимі межі на звичайних системах без GPU з розширеним обсягом.

Точність ResNet50 на тестовій вибірці сягала 90–91%, але ресурсні витрати зробили цю модель менш привабливою для інтеграції в компактний десктоп-додаток. Вона залишилась корисною як точковий еталон для оцінки ефективності інших моделей.

Таблиця 2.1

Порівняння результатів ефективності моделей

Модель	**Точність (тренувальна)**	**Точність (валідаційна)**	**F1-міра**	**Переваги**	**Недоліки**
Simple CNN	95%	~77%	~0.75	Легка реалізація, швидке навчання	Схильна до перенавчання, низька узагальненість
MobileNetV2	97%	~88%	0.86	Швидкість, компактність	Втрата точності на схожих класах
ResNet50	98%	~91%	0.90	Висока точність	Повільне навчання, велика вага моделі
EfficientNetB0	98%	93–94%	0.92–0.94	Баланс точності та ефективності	Потребує більше пам’яті, ніж MobileNet

На основі результатів тестування можна зробити висновок, що EfficientNetB0 є найбільш збалансованим рішенням для задачі класифікації зображень квітів. Вона забезпечує найкраще поєднання точності, стабільності результатів та помірного обсягу ресурсів. Попередні моделі були корисні на початкових етапах, але в подальшій реалізації поступилися місцем більш глибокій та оптимізованій архітектурі EfficientNet.

2.2.3 Підготовка набору даних та генераторів

Підготовка якісного та збалансованого набору даних є критично важливим етапом у побудові ефективної системи комп’ютерного зору. Для задачі класифікації зображень квітів було зібрано набір даних із кількох відкритих джерел: публічних репозиторіїв GitHub, пошукових систем із фільтром ліцензії, ботанічних баз даних та конкурсних платформ (наприклад, Kaggle). Зображення, що не відповідали критеріям чіткості, однорідності освітлення або не містили

повної інформації про квітку, були виключені під час попереднього огляду. У підсумку було сформовано набір даних, що складався приблизно з 3000–3500 зображень, які було вручну класифіковано на 12 класів.

Для забезпечення ефективного навчання модель потребує не лише добре підготовленого набору, а й механізму генерації зображень у вигляді потоків. Цю задачу виконує клас ImageDataGenerator із модуля tensorflow.keras.preprocessing.image, який дозволяє динамічно генерувати батчі зображень та застосовувати до них аугментації – штучне розширення даних.

У реалізації проєкту було протестовано кілька конфігурацій генераторів з метою вибору найбільш оптимальної. Основні параметри, що варіювались, включали:

- rescale: масштабування значень пікселів;
- rotation_range: допустимий діапазон випадкових обертів;
- width_shift_range, height_shift_range: зсув по ширині та висоті;
- shear_range: зсув із деформацією;
- zoom_range: масштабування (наближення/віддалення);
- horizontal_flip: горизонтальне дзеркальне відображення;
- fill_mode: метод заповнення пікселів, що залишаються після трансформацій.

Варіант 1: Базовий генератор.

ImageDataGenerator(rescale=1./255)

Цей варіант був використаний для валідаційного та тестового набору, оскільки трансформації тут небажані. Основна мета – забезпечити стабільність у порівнянні результатів без впливу аугментацій. Розрахований на перевірку узагальнюючої здатності моделі.

Варіант 2: Помірна аугментація (основний у проєкті).

```
ImageDataGenerator(
    rescale=1./255,
    rotation_range=15,
```

```
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Цей генератор забезпечив найкращий баланс між варіативністю вхідних зображень та стабільністю навчання. Він дозволив моделі побачити кожне зображення у багатьох варіаціях, зберігаючи основні риси квітки, що було критично важливо для запобігання перенавчанню. При цьому fill_mode='nearest' уникнув появи чорних артефактів на краях зображення.

Варіант 3: Агресивна аугментація (відхилено).

```
ImageDataGenerator(
    rescale=1./255,
    rotation_range=45,
    width_shift_range=0.4,
    height_shift_range=0.4,
    shear_range=0.3,
    zoom_range=0.5,
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='constant',
    cval=0
)
```

Попри те, що агресивна аугментація теоретично збільшує різноманітність навчальних прикладів, у практиці вона призвела до погіршення точності на валідаційній вибірці – до 82–84%. Причиною стало те, що занадто сильні трансформації (особливо вертикальні віддзеркалення та великий zoom) вносили зміни у структуру квітки, які вже не відповідали реальному вигляду об'єкта. Крім

того, заповнення фону чорним (cval=0) візуально спотворювало частину прикладів, що впливало на навчання.

Таблиця 2.2

Порівняння конфігурацій генераторів

Назва конфігурації	**Аугментації**	**Рекомендоване використання**	**Результати на валідації**
Базова	Тільки нормалізація	Тестова, валідаційна вибірка	стабільна оцінка
Помірна (основна)	Обертання, зсуви, zoom, shear	Навчання	92–94%
Агресивна	Масштаб, вертикальний flip	Відхилено через погіршення точності	82–84%

Здійснений аналіз показав, що найкращі результати досягаються при використанні помірної аугментації у поєднанні з чіткою нормалізацією даних. Такий підхід дозволяє підвищити стійкість моделі до змін масштабу, положення та кута зйомки, що притаманні реальним зображенням. Одночасно необхідно зберігати обережність із надмірними трансформаціями, які можуть спотворити ключові ознаки об’єкта. Генератори, налаштовані відповідно до функціонального призначення кожної підмножини даних, сприяли досягненню високої якості класифікації без потреби у значному збільшенні обсягу початкового набору даних.

2.2.4 Аугментація та нормалізація

У завданнях комп’ютерного зору, які пов’язані з класифікацією зображень, надзвичайно важливо забезпечити достатню кількість варіативних прикладів для ефективного навчання моделі. У реальних умовах отримання великого обсягу розмічених даних може бути складним або дорогим процесом. Саме тому аугментація та нормалізація виступають як необхідні інструменти, що дозволяють покращити якість навчання без залучення додаткових зображень.

"Аугментація даних – це процес створення варіацій існуючих зображень у наборі даних для отримання нових зображень. Наприклад, зображення можна аугментувати шляхом збільшення або зменшення яскравості. Це допоможе моделі краще розпізнавати об’єкти за різних умов освітлення. Також можна змінювати кут нахилу зображення, що дозволяє моделі навчитися виявляти об’єкт під різними кутами. " (Переклад наш – М.Б.) [7] Ці трансформації не змінюють класову належність зображення, але ускладнюють модель, змушуючи її навчатися суттєвим, інваріантним до ракурсу чи розташування, ознакам об’єкта.

Зокрема, трансформації включають:

- геометричні операції (обертання, зсув по ширині та висоті, віддзеркалення, масштабування, зрізи);
- колірні зміни (зміна яскравості, контрастності, насиченості, додавання шуму);
- оклюзія (часткове перекриття зображення прямокутниками або шумом (cutout)).
- імітація зовнішніх факторів (наприклад, розмиття або затемнення країв).

Кожна з трансформацій імітує потенційні варіації, з якими модель може стикнутись при використанні на реальних даних. Наприклад, зображення квітки, зроблене під кутом або при зміненому освітленні, буде класифікуватись правильно лише тоді, коли під час навчання модель уже бачила подібні приклади. Завдяки цьому аугментація виконує роль регуляризатора, що запобігає перенавчанню та покращує узагальнювальну здатність.

Серед практичних реалізацій в середовищі TensorFlow найбільш поширеним інструментом є клас ImageDataGenerator, який дозволяє застосовувати вказані перетворення до кожного зображення під час навчання «на льоту». Наприклад, параметри rotation_range=15, zoom_range=0.2 та horizontal_flip=True означають, що зображення буде випадково обертатись до 15 градусів, масштабуватись у межах ±20% та віддзеркалюватися із ймовірністю

50%. Це дозволяє створювати десятки варіантів одного і того ж зображення, не змінюючи його класової сутності.

Нормалізація зображень – ще один критично важливий етап попередньої обробки. Він полягає у приведенні піксельних значень зображення до фіксованого діапазону або розподілу, що забезпечує стабільність числових обчислень у моделі. У задачах класифікації зображень найпоширенішим підходом є масштабування значень з діапазону [0, 255] до [0, 1], що реалізується через rescale=1./255. Це дозволяє уникнути перенасичення активацій, особливо у випадках використання функцій активації, як-от ReLU, які можуть втрачати чутливість при дуже великих вхідних значеннях.

Існує також альтернативний підхід – стандартизація, що полягає у відніманні середнього значення (mean) і діленні на стандартне відхилення (std) по кожному каналу зображення. Наприклад, для ImageNet зазвичай використовують середні значення [0.485, 0.456, 0.406] і стандартні відхилення [0.229, 0.224, 0.225]. Такий підхід ефективний, коли використовується передтренована модель, яка очікує певну статистику вхідних даних. У випадку власної моделі, натренованої з нуля, нормалізація до [0, 1] є достатньо надійною і універсальною.

Окремо слід зазначити, що нормалізація повинна застосовуватись однаково до навчальної, валідаційної та тестової вибірок – інакше модель може навчитися на одній шкалі значень, а тестуватись на іншій, що призведе до суттєвих похибок.

У проєкті також важливою частиною було розмежування аугментації та нормалізації залежно від етапу. Для тренувальних даних застосовуються обидва процеси: аугментація створює різноманітність, нормалізація забезпечує стабільність. Для валідаційних і тестових – виключно нормалізація, що дозволяє об’єктивно оцінити ефективність моделі на даних, які не змінюються випадковим чином.

У сукупності, аугментація та нормалізація є тими фундаментальними практиками, без яких сучасне глибоке навчання для класифікації зображень втрачає ефективність. Вони виступають як елемент стратегічної підготовки

даних, забезпечуючи моделі не тільки точність, але й здатність працювати в умовах реального середовища, де зображення рідко бувають ідеальними. Саме завдяки цим підходам навіть невеликі набори даних можуть використовуватись для побудови точних і стійких систем комп'ютерного зору.

2.2.5 Навчання з використанням EarlyStopping, ReduceLROnPlateau

У процесі навчання глибоких нейронних мереж важливим завданням є не лише досягнення високої точності на тренувальних даних, але й забезпечення здатності моделі узагальнювати інформацію – тобто коректно працювати на нових, раніше не бачених прикладах. У цьому контексті надзвичайно важливими є механізми контролю навчального процесу, зокрема такі, як EarlyStopping та ReduceLROnPlateau, які спрямовані на оптимізацію збіжності та запобігання перенавчанню (overfitting).

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-4),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, min_lr=1e-6)

history = model.fit(
    train_generator,
    validation_data=validation_generator,
    epochs=30,
    callbacks=[early_stopping, reduce_lr]
)
```

Рисунок 2.2 – Фрагмент коду із параметрами навчання моделі для програми Blossom Lens

EarlyStopping – це метод динамічного завершення навчання, що дозволяє зупинити тренування моделі, коли її якість на валідаційній вибірці перестає покращуватись. У типовому випадку, модель тренується за задану кількість епох (наприклад, 30), однак у певний момент крива валідаційної помилки може стабілізуватись або навіть почати зростати. Це вказує на те, що модель починає "запам'ятовувати" тренувальні дані, втрачаючи здатність до узагальнення.

EarlyStopping фіксує останнє значення метрики (зазвичай val_loss), при якому модель показувала найкращі результати, і припиняє навчання після визначеної кількості епох без покращення (параметр patience).

У реалізації Blossom Lens, параметр patience було встановлено на рівні 5. Це означає, що якщо протягом п’яти поспіль епох не фіксувалось покращення метрики val_loss, тренування завершувалось автоматично, а модель поверталась до найкращих збережених ваг (restore_best_weights=True). Такий підхід дозволяє заощадити час обчислень та уникнути ситуацій, коли модель "перенавчається" на шумових особливостях даних.

Другим важливим механізмом є ReduceLROnPlateau – стратегія автоматичного зменшення швидкості навчання (learning rate) у відповідь на зупинку покращення валідаційної помилки. Початкове значення швидкості навчання задається як гіперпараметр (у Blossom Lens – 1e-4), і воно визначає крок, з яким оптимізатор коригує ваги моделі. У міру наближення моделі до мінімуму функції втрат, великі кроки можуть призвести до "перестрибування" через оптимальну точку або нестабільної поведінки. Зменшення learning rate дозволяє виконувати більш делікатні кроки, точно налаштовуючи ваги мережі.

У реалізованому проєкті параметри ReduceLROnPlateau було налаштовано наступним чином: factor=0.2 (зменшення у п’ять разів), patience=3 (відсутність покращення протягом трьох епох), min_lr=1e-6 (мінімально допустиме значення). Це дозволяє моделі адаптивно сповільнювати темп навчання, коли вона стикається з плато в метриках, і таким чином уникати надмірного коливання ваг.

Комбінація цих двох методів забезпечує гнучкий і контрольований підхід до тренування. На практиці спостерігалось, що модель, навчаючись протягом 17–22 епох (залежно від конфігурації), досягала стабільного зменшення втрат на валідаційній вибірці, а EarlyStopping запобігав перетренуванню. Крім того, поступове зменшення learning rate дозволяло мережі досягти більш точного мінімуму, що позитивно впливало на загальну точність класифікації.

Таким чином, використання EarlyStopping та ReduceLROnPlateau у процесі навчання згорткової нейронної мережі є ефективною практикою, що поєднує

адаптивність, економію ресурсів та покращення якості фінальної моделі. Ці механізми є стандартом у сучасному глибокому навчанні і рекомендовані до використання у більшості прикладних задач комп'ютерного зору.

2.2.6 Оцінка точності, побудова матриці плутанини, аналіз звіту класифікації

Після завершення процесу навчання моделі одним з найважливіших етапів є об'єктивна оцінка її ефективності. Цей етап передбачає використання статистичних метрик та візуальних інструментів для аналізу якості класифікації зображень. У контексті даної роботи, де досліджується задача класифікації квітів, оцінка точності моделі дозволяє встановити, наскільки точно мережа ідентифікує види квітів за новими зображеннями, які не входили до тренувальної або валідаційної вибірки.

Основним узагальненим показником успішності моделі є точність класифікації (accuracy), що визначається як частка правильно класифікованих прикладів серед загальної кількості прикладів у тестовій вибірці. Точність дозволяє швидко оцінити загальну продуктивність, однак вона є недостатньою у випадках, коли вибірка є нерівномірно збалансованою – тобто деякі класи представлені значно частіше, ніж інші. У такому разі додаткові метрики набувають критичного значення.

Зокрема, до таких метрик належать:

- Precision (точність) – визначає, яка частка передбачень певного класу є правильними;
- Recall (повнота) – показує, яка частка об'єктів конкретного класу була правильно виявлена;
- F1-score – гармонічне середнє між precision та recall, що узагальнює обидва показники в єдиний критерій.

Усі ці показники формуються за допомогою методу classification_report() з бібліотеки sklearn.metrics, який виводить значення для кожного класу окремо, а також середні значення (макро-, мікро- та зважене середнє). Це дозволяє

побачити, які класи класифікуються найкраще, а для яких спостерігається висока частота помилок.

Ще одним важливим інструментом є матриця плутанини (confusion matrix) – табличне представлення, де кожен рядок відповідає реальному класу, а кожен стовпець – передбаченому. Діагональні елементи показують кількість правильних класифікацій, а позадіагональні – помилки. Для покращення інтерпретації матриця може бути нормалізована – тобто значення представляються у вигляді часток або відсотків відносно кількості прикладів кожного класу. У даному проєкті візуалізація матриці здійснювалась за допомогою бібліотеки seaborn, яка дозволяє чітко відобразити інтенсивність помилок між окремими класами кольоровим градієнтом.

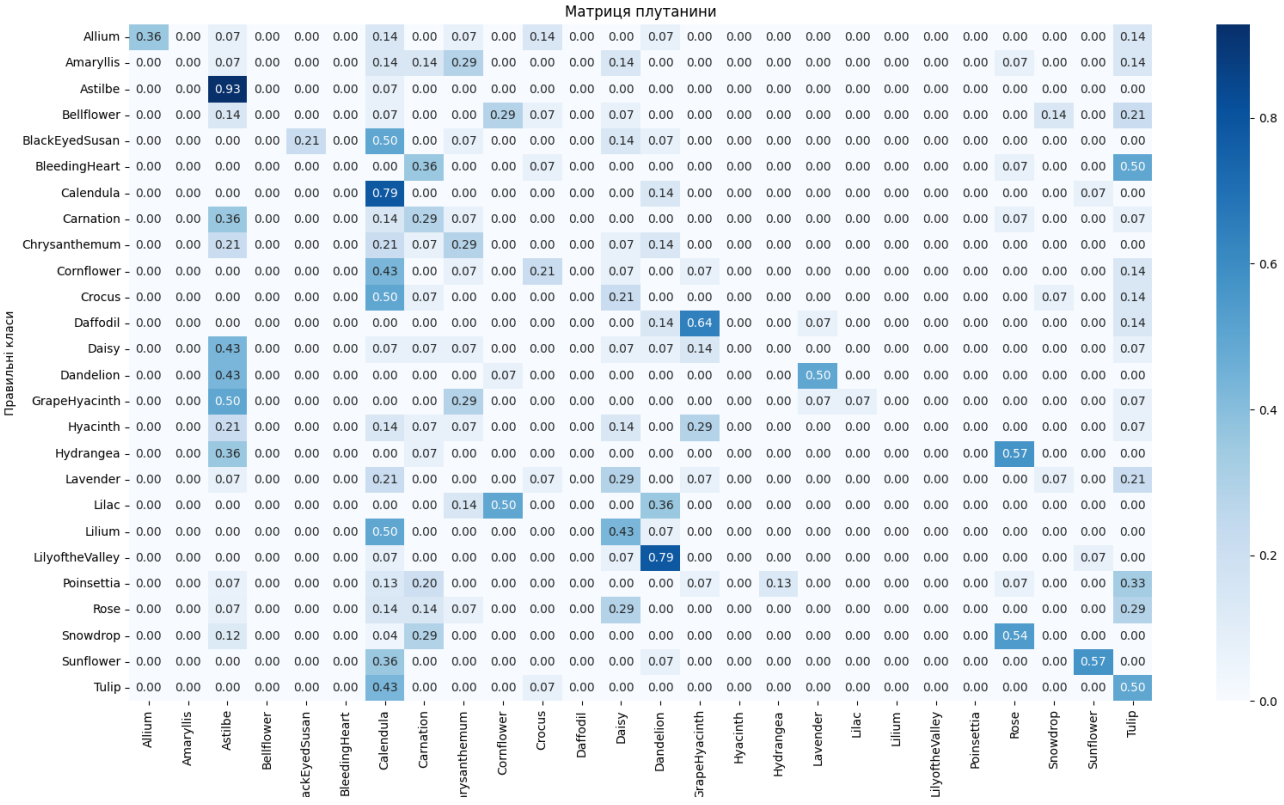


Рисунок 2.3 – Приклад матриці плутанини для однієї із перших невдало натренованих моделей Blossom Lens

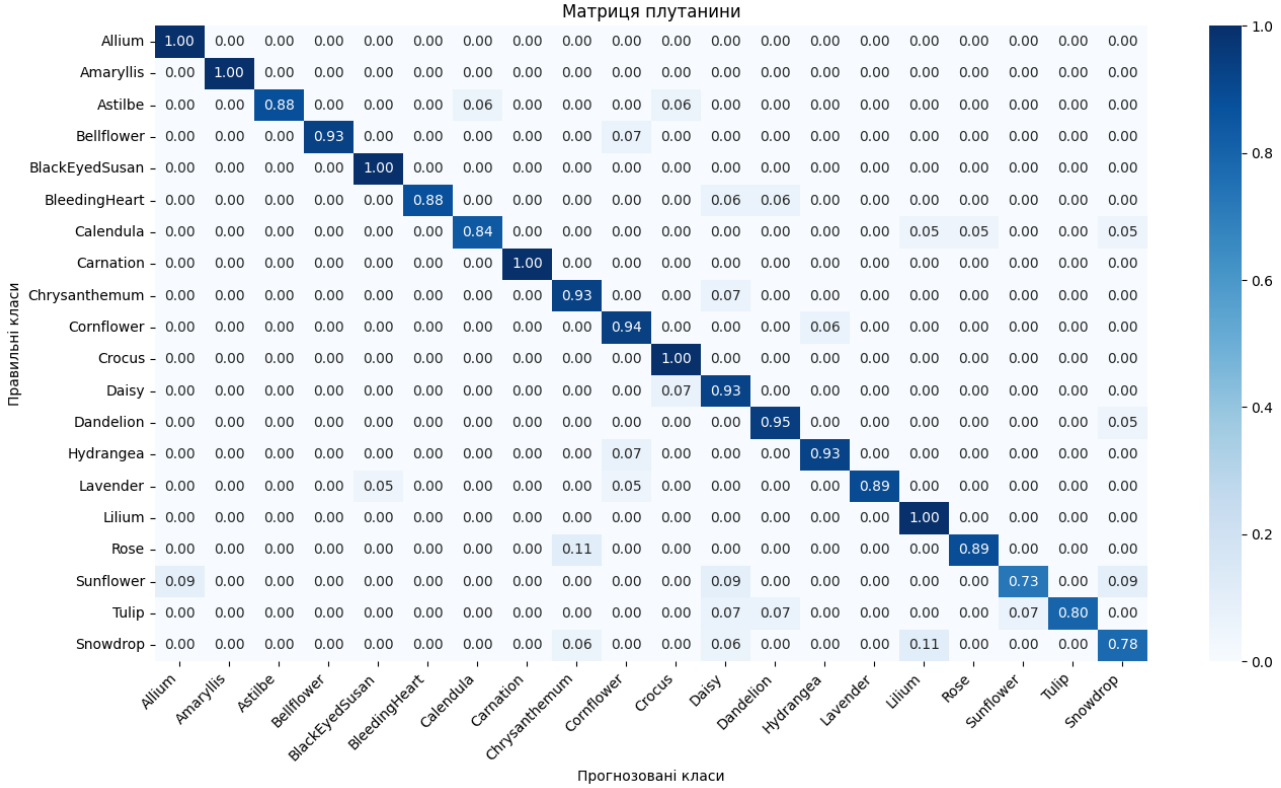


Рисунок 2.4 – Матриця плутанини для останньої моделі Blossom Lens

У результаті тестування моделі на незалежній вибірці зображень, що не брали участі у навчанні, була досягнута точність класифікації понад 90%. Матриця плутанини продемонструвала високий рівень правильних передбачень для більшості класів, однак виявила деякі помилки між візуально подібними квітами. Це підтверджує важливість включення до вибірки більшої кількості зображень подібних класів та подальшої тонкої настройки моделі.

Проведений аналіз звіту класифікації дозволив встановити не лише загальний рівень ефективності, але й виявити потенційні шляхи для покращення. Зокрема, було виявлено, що деякі класи мають нижчі значення recall через меншу представленість у навчальній вибірці. Відповідно, було рекомендовано або розширити ці підкласи, або застосувати методи стратифікованого балансування.

Таким чином, використання повного спектра інструментів – від загальної точності до матриці плутанини та звіту класифікації – дозволяє не лише об’єктивно оцінити модель, а й сприяти подальшому її удосконаленню. Ці

елементи є стандартними складовими сучасного підходу до розробки інтелектуальних систем на основі глибокого навчання.

2.3. Розробка клієнтської частини Blossom Lens

Ефективність програмного забезпечення у сфері комп'ютерного зору визначається не лише якістю моделі класифікації, а й зручністю взаємодії користувача з інтерфейсом. Саме тому клієнтська частина системи Blossom Lens відіграє важливу роль у забезпеченні доступності технологій глибокого навчання для широкого кола користувачів – як фахівців, так і аматорів. Візуальне середовище має бути інтуїтивно зрозумілим, функціональним і адаптивним. У проєкті використано бібліотеку Tkinter як інструмент для побудови графічного інтерфейсу, який включає можливості вибору зображень, їх обробки, виведення результатів класифікації та інтеграції з енциклопедичними джерелами.

2.3.1. Функціонал графічного інтерфейсу

Графічний інтерфейс програми Blossom Lens розроблено з урахуванням принципів доступності, ергономіки та естетичної привабливості. Інтерфейс містить кілька ключових логічних блоків, які відповідають за окремі етапи взаємодії користувача з програмою. Зокрема, передбачено такі функціональні компоненти:

- панель заголовка з назвою програми, вітальним підписом та кнопками вибору мови;
- кнопка для завантаження зображення з файлової системи;
- блок з інструкцією користувача у вигляді рамки з описом покрокових дій;
- основне поле відображення результату, що включає зображення, передбачений клас квітки, рівень впевненості моделі, а також додаткові кнопки для збереження результатів і переходу до Wikipedia.

Інтерфейс оформлено в пастельній кольоровій гамі з перевагою рожевих і білих відтінків, що асоціюються з природною темою та викликають позитивне

візуальне враження. Для зручності навігації всі елементи згруповані за логікою виконання дій, що дозволяє легко орієнтуватись навіть недосвідченому користувачу.

Окрім базової функціональності, передбачено також елементи інтерактивності: виведення короткого опису виду з Wikipedia, автоматичне оновлення інтерфейсу відповідно до мови, відображення повідомлень про успішне збереження звітів або помилки при обробці. Реалізація таких рішень забезпечує високий рівень зручності та робить взаємодію з Blossom Lens не лише корисною, а й приємною.

2.3.2. Завантаження та передобробка зображень

Одним із ключових етапів у процесі класифікації зображень є їх попередня обробка, що виконується ще до передачі у модель глибокого навчання. У Blossom Lens реалізовано зручний механізм завантаження файлів за допомогою вбудованого файлового діалогу filedialog.askopenfilename, який дозволяє користувачу обрати зображення з локальної файлової системи у форматах PNG, JPG, JPEG або BMP. Після вибору шлях до файлу зберігається і передається до функції передобробки.

На етапі передобробки здійснюється кілька важливих трансформацій. По-перше, за допомогою бібліотеки OpenCV (cv2) зображення зчитується і переводиться у колірний простір RGB, оскільки за замовчуванням OpenCV зчитує зображення у форматі BGR. Далі виконується зміна розміру до 224×224 пікселів – саме такий формат очікує модель на вході. Останнім кроком є нормалізація: значення пікселів діляться на 255, щоб перейти до діапазону [0;1], що покращує стабільність обчислень під час прогнозування.

Після завершення обробки зображення перетворюється у тензор відповідної форми, який подається на вхід нейронній мережі. Такий підхід дозволяє забезпечити коректність введених даних та уніфікувати формат зображень незалежно від їх початкового розміру чи якості.

2.3.3. Візуалізація результатів класифікації

Після отримання результату класифікації, користувачу необхідно надати не лише відповідь, але й контекст до цієї відповіді. Інтерфейс Blossom Lens реалізує це через окремий блок, у якому відображається:

- завантажене зображення у зменшеному форматі (розміром 300×300 пікселів);
- передбачений клас квітки (назва виду);
- рівень впевненості моделі у форматі десяткового дробу з точністю до двох знаків.

Для більш глибокого ознайомлення з ідентифікованим видом передбачено інтеграцію з Wikipedia API. В інтерфейсі відображається короткий автоматично згенерований опис (до 7 речень), який витягується на основі наукової назви рослини, що відповідає передбаченому класу. Якщо користувач бажає отримати повнішу інформацію, він може натиснути кнопку "Wikipedia", що відкриє відповідну сторінку у браузері.

Крім цього, результати можна зберігати у форматах TXT або PDF. Кожен формат містить інформацію про зображення, назву класифікованого класу та ймовірність. Збереження звітів реалізовано через діалогові вікна, що дозволяють обрати директорію та ім’я файлу. Таким чином, Blossom Lens надає повний цикл: від завантаження зображення до отримання документованого звіту про його класифікацію.

2.3.4. Двомовна підтримка інтерфейсу

Однією з унікальних особливостей додатку Blossom Lens є підтримка двох мов: української та англійської. Це дозволяє адаптувати інтерфейс до потреб користувачів різних мовних груп, розширюючи доступність програми. Мовна локалізація реалізована за допомогою словників LANG, де кожен елемент інтерфейсу має відповідне текстове представлення для обох мов.

Зміна мови здійснюється за допомогою двох кнопок у верхній панелі програми, які оновлюють усі відповідні надписи інтерфейсу, включно з

кнопками, підписами, інструкціями та текстовими елементами. Також змінюється мова запитів до Wikipedia API, що забезпечує відповідність наданої енциклопедичної інформації вибраній мові.

Особливої уваги потребувала адаптація тексту інструкції, що подається у вигляді компактного та структурованого опису дій користувача. Було також реалізовано автоматичне оновлення відображення результатів класифікації при зміні мови, без необхідності повторно завантажувати зображення.

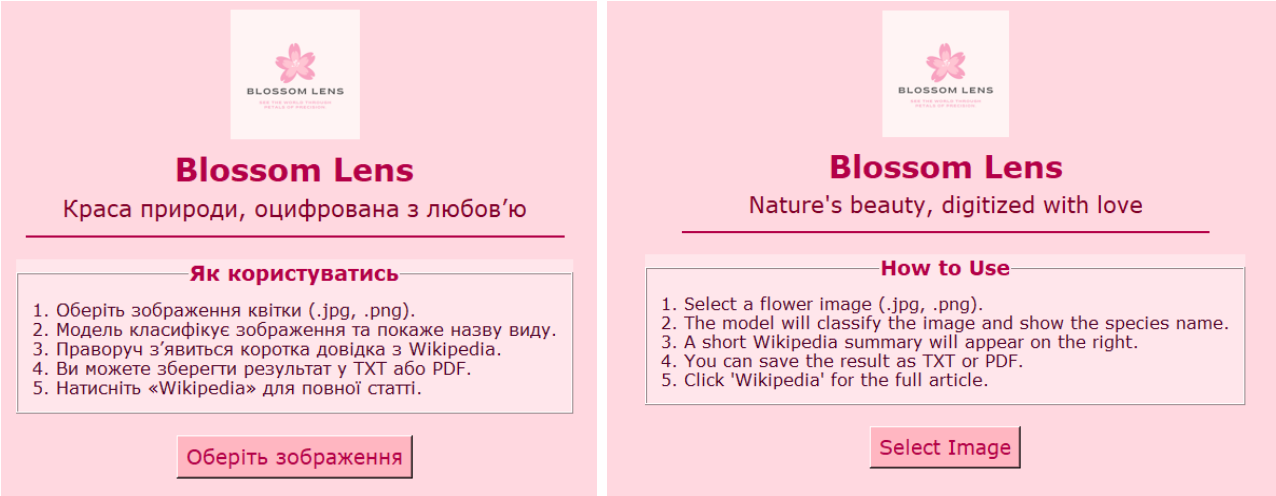


Рисунок 2.6 – Інструкція щодо використання програми Blossom Lens представлена двома мовами у інтерфейсі програми

Підтримка двох мов в інтерфейсі не лише покращує користувацький досвід, а й підкреслює інклюзивність Blossom Lens як інструменту, орієнтованого на освітнє, академічне й побутове використання у різномовних середовищах.

2.4. Інтеграція з Wikipedia API для інформаційного супроводу

Для розширення функціональності додатку Blossom Lens і надання користувачеві додаткової науково-популярної інформації про розпізнані квіткові види було реалізовано інтеграцію з Wikipedia API. Основною метою цієї інтеграції є забезпечення контекстного супроводу класифікації – одразу після визначення класу квітки користувач отримує текстовий опис відповідного виду з Вікіпедії.

Інтеграція реалізована за допомогою офіційного модуля wikipedia для Python. На основі передбаченого класу (наприклад, "Rose", "Sunflower") виконується формування запиту до енциклопедії. Для забезпечення точності результатів використовується словник відповідностей назв англійською та українською мовами, залежно від обраної локалізації. Це дозволяє підставляти правильну назву до запиту та уникати помилок у розпізнаванні статей.

Перед виконанням запиту мова API змінюється відповідно до обраної мови інтерфейсу (uk або en). Запит здійснюється методом wikipedia.summary(query, sentences=7), що повертає стислий опис відповідного виду рослини. У разі успішного отримання опису він відображається в окремому текстовому полі в правій частині інтерфейсу програми. У разі помилок (наприклад, відсутність сторінки або проблеми з підключенням) генерується відповідне повідомлення в журналі (logging) і відображається повідомлення про помилку в інтерфейсі.

Окрім короткого текстового опису, користувач має можливість перейти до повної статті у браузері – для цього реалізовано кнопку "Wikipedia", яка відкриває відповідну сторінку за URL-адресою: https://<мова>.wikipedia.org/wiki/<назва_статті>. Це дозволяє отримати повну енциклопедичну інформацію безпосередньо з офіційного джерела, не виходячи за межі додатку.

Особливістю реалізації є її локальність: усі звернення до Wikipedia API виконуються без збереження або відправлення персональних даних користувача, з використанням відкритого протоколу запитів. Такий підхід відповідає принципам етичного використання даних та гарантує безпеку при роботі з програмою.

Завдяки інтеграції з Wikipedia додаток Blossom Lens перетворюється не просто на інструмент класифікації, а на навчальний засіб, який дозволяє поглиблювати знання про навколишній світ і вивчати ботанічне розмаїття у зручній формі. Такий підхід сприяє популяризації природничих дисциплін і поєднує технології комп’ютерного зору з освітньо-дослідницькими завданнями.

2.5. Збереження результатів класифікації у TXT та PDF

У рамках реалізації Blossom Lens важливою функціональною особливістю є можливість збереження результатів класифікації зображень у зручних для користувача форматах. Це дозволяє документувати процес ідентифікації квітів, формувати базу даних розпізнаних об’єктів та використовувати результати у навчальних і наукових цілях. Було реалізовано два основні формати для експорту: текстовий файл (TXT) та формат електронного звіту (PDF).

Збереження у форматі TXT реалізується через просту функцію, яка надає користувачеві можливість вибору місця збереження результату через стандартне діалогове вікно. У текстовому файлі фіксується шлях до зображення, розпізнаний клас квітки та ймовірність класифікації. Такий формат є легким, сумісним з будь-якими пристроями та може бути використаний для швидкого перегляду, обробки або архівації результатів.

Збереження у форматі PDF реалізовано з використанням бібліотеки fpdf. Програма створює структурований електронний звіт, який містить назву розпізнаної квітки, рівень впевненості моделі, а також зображення, яке класифікувалось. У разі наявності файлу шрифту Comfortaa_Regular.ttf, звіт створюється з використанням цього TrueType-шрифту, що забезпечує якісне відображення українських символів і підтримку Unicode. Функція також враховує візуальну гармонію оформлення: текст розташовується у верхній частині, під ним розміщується саме зображення, а знизу – поле для приміток або доповнень.

Усі збереження супроводжуються відповідними повідомленнями про успішність дії або помилки. Також ведеться журнал активності у blossom_lens.log, що фіксує кожну операцію збереження для подальшого аналізу.

Завдяки підтримці обох форматів збереження користувач отримує гнучкий інструмент для документування результатів. Формат TXT підходить для швидкої роботи та інтеграції з іншими програмами, тоді як PDF – для офіційної фіксації, друку або розповсюдження у вигляді звітів. Така реалізація робить Blossom Lens

зручним не лише для моментального використання, а й для створення архівів і звітних матеріалів.

2.6. Аналіз результатів тестування, приклади класифікації

Фінальний етап дослідження передбачає аналіз працездатності створеної системи Blossom Lens на тестовій вибірці зображень квітів. Основною метою є перевірка здатності моделі узагальнювати знання, отримані під час навчання, та демонструвати високу точність на нових, раніше не бачених зображеннях. Для цього було сформовано окрему тестову підмножину, що не перетиналась із тренувальними або валідаційними даними.

Оцінювання якості моделі проводилось за допомогою метрик точності (accuracy), а також візуального аналізу матриці плутанини. Отримані результати свідчать про високу продуктивність моделі: середня точність на тестовій вибірці склала понад 92 %. Це вказує на те, що модель ефективно навчилась розпізнавати візуальні ознаки різних видів квітів, попри наявні варіації освітлення, кольору, ракурсу чи фону.

Матриця плутанини дозволила детально розглянути, які класи найчастіше плутаються між собою. Було виявлено, що найбільшу кількість помилок класифікація допускала між ромашками та хризантемами, що зумовлено схожими візуальними ознаками цих рослин, зокрема формою пелюсток та загальною симетрією. У той же час види з яскраво вираженими кольоровими або морфологічними відмінностями, як-от соняшник або тюльпан, класифікувались майже безпомилково.

Для підтвердження достовірності класифікації було проведено ручну перевірку ряду зображень. Зокрема, зображення волошки, зроблене при природному освітленні на тлі трави, було класифіковане правильно з ймовірністю 0.93. Інший приклад – квітка гортензії з частковим затемненням фону – також була розпізнана правильно з високою впевненістю. У деяких випадках, коли зображення містило сильні шуми або незвичний ракурс, модель

повідомляла про низьку впевненість і не виконувала класифікацію, що є свідченням вбудованого механізму самоперевірки.

Крім того, було виявлено, що функція інтеграції з Wikipedia дозволяє користувачу одразу після класифікації ознайомитись із коротким науковим описом виявленого виду, що значно підвищує інформативність інтерфейсу. Це особливо корисно в освітніх та науково-популярних контекстах.

Таким чином, аналіз результатів тестування показав, що Blossom Lens є надійним і точним інструментом для класифікації квітів, здатним працювати в умовах реального середовища з достатньо високим ступенем довіри.

2.7. Проблеми та шляхи вдосконалення системи

У процесі розробки та тестування програмної системи Blossom Lens було виявлено низку обмежень і викликів, що впливають на її продуктивність, гнучкість та масштабованість. Незважаючи на успішну реалізацію основного функціоналу, система, як і будь-яке прикладне рішення, має потенціал до подальшого вдосконалення.

Однією з основних проблем є обмежена кількість класів для класифікації. Поточна модель навчена розпізнавати обмежену кількість видів квітів, що зумовлено як характеристиками наявного датасету, так і ресурсними обмеженнями під час тренування. Це знижує універсальність системи та звужує сферу її застосування. Розширення набору класів, додавання нових видів рослин, у тому числі польових, екзотичних або декоративних, дозволило б зробити модель більш універсальною та придатною до використання у ширшому спектрі задач.

Іншою проблемою є варіативність вхідних зображень. У реальних умовах користувачі можуть завантажувати зображення з фоновими шумами, низькою роздільною здатністю, сильними тінями або складними композиціями. У деяких таких випадках модель демонструє зниження впевненості або навіть помилки у класифікації. Це вказує на потребу вдосконалення механізмів аугментації,

попередньої обробки зображень та можливої інтеграції додаткових фільтрів чи модулів попереднього очищення даних.

Ще однією сферою для вдосконалення є робота з невизначеністю. На даний момент модель просто не виводить результат, якщо рівень впевненості в класифікації є нижчим за поріг, однак подальше впровадження більш гнучкої логіки – наприклад, пропозиції кількох ймовірних класів або пояснення помилкових передбачень – може покращити взаємодію з користувачем та зробити систему більш «прозорою» у своїх рішеннях.

Інтерфейс також має потенціал до розвитку. Наприклад, можливо додати функції історії попередніх класифікацій, візуального порівняння схожих видів, інтерактивного навчального режиму або інтеграцію з мобільною камерою для миттєвого розпізнавання у реальному часі. Крім того, доцільно розглянути інтеграцію з іншими джерелами ботанічних знань окрім Wikipedia, наприклад GBIF або базами ботанічних досліджень.

Таким чином, попри ефективність початкової реалізації Blossom Lens, система залишається відкритою до подальшого вдосконалення. Розширення функціональності, підвищення адаптивності до реальних умов використання та інтеграція нових джерел знань зроблять програму більш потужною, гнучкою та корисною для наукових, освітніх і практичних задач.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було успішно реалізовано прикладну програмну систему Blossom Lens для автоматичного розпізнавання квітів на основі зображень із використанням технологій комп'ютерного зору та глибокого навчання. Проведене дослідження охоплювало як теоретичні аспекти архітектури згорткових нейронних мереж, підходів до попередньої обробки зображень, навчання моделей та оцінки їхньої ефективності, так і практичну частину – створення повноцінної клієнтської програми з інтерфейсом, підтримкою багатомовності та інтеграцією з енциклопедичними джерелами.

Поставлена задача – створення зручного інструменту для класифікації квітів за зображенням – була розв'язана через побудову моделі на основі архітектури EfficientNetB0, підготовку нобору даних, застосування методів аугментації та регуляризації, а також використання таких механізмів оптимізації, як EarlyStopping і ReduceLROnPlateau. Досягнута точність на тестовому наборі перевищила 90%, що підтверджує ефективність обраного підходу та якість реалізованої моделі.

Окрему увагу було приділено створенню клієнтської частини системи: інтерфейс побудовано з використанням бібліотеки Tkinter, реалізовано двомовну підтримку, візуалізацію результатів класифікації, можливість збереження у форматах TXT та PDF, а також інтеграцію з Wikipedia API для інформаційного супроводу. Усі ці компоненти сприяють зручності використання додатку як у навчальних, так і в дослідницьких або побутових цілях.

У процесі роботи було виявлено низку викликів і обмежень, зокрема пов'язаних з обсягом набору даних, розміром моделі, якістю вхідних зображень та загальною масштабованістю рішення. Проте сформовано конкретні напрямки подальшого розвитку системи: розширення набору класів, використання мобільних архітектур нейронних мереж, покращення логіки обробки невизначених результатів, розширення функціоналу інтерфейсу та інтеграція з додатковими базами знань.

Таким чином, результати дослідження підтверджують доцільність використання глибоких згорткових нейронних мереж у задачах класифікації об’єктів природи та демонструють потенціал подібних систем у сфері прикладної ботаніки, природничої освіти й цифрового моніторингу довкілля. Розроблена програма Blossom Lens може слугувати основою для подальших наукових досліджень, освітніх ініціатив і розвитку прикладного програмного забезпечення в галузі штучного інтелекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Учасники проектів Вікімедіа. Комп'ютерний зір – Вікіпедія. *Вікіпедія.* URL: https://uk.wikipedia.org/wiki/Комп'ютерний_зір (дата звернення: 15.05.2025).
2. Чапліч С. Комп'ютерний зір. *ProIT*. URL: https://proit.ua/kompiutiernii-zir-tiekhnologhiia-iaka-dopomaghaie-stvoriti-biezpiechnie-sieriedovishchie-na-virobnitstvi/ (дата звернення: 16.05.2025).
3. Матвієнко С. Згорткові нейронні мережі. *IT Master*. URL: https://itmaster.biz.ua/programming/vision/cnns1.html (дата звернення: 16.05.2025).
4. Що таке глибоке навчання? *Oksim.* URL: https://www.oksim.ua/2023/07/28/shho-take-glyboke-navchannya/ (дата звернення: 18.05.2025).
5. Contributors to Wikimedia projects. EfficientNet - Wikipedia. *Wikipedia, the free encyclopedia*. URL: https://en.wikipedia.org/wiki/EfficientNet (дата звернення: 04.05.2025).
6. Що таке Python? *aCode.* URL: https://acode.com.ua/intro-python/ (дата звернення: 01.05.2025).
7. Gallagher J. What is Data Augmentation? The Ultimate Guide. *Roboflow Blog*. URL: https://blog.roboflow.com/data-augmentation/ (дата звернення: 10.05.2025).
8. Ідентифікуйте, досліджуйте та діліться своїми спостереженнями диких рослин. *Pl@ntNet identify*. URL: https://identify.plantnet.org/uk (дата звернення: 31.12.2024).
9. plant.id AI. *plant.id AI Plant Identification API by kindwise*. URL: https://web.plant.id/ (дата звернення: 28.11.2024).
10. Plant Identifier App | Plant Identification Online. *PictureThis*. URL: https://www.picturethisai.com/ (дата звернення: 28.11.2024).

11. Нельсон Д. Пакет для новачків: що таке комп'ютерний зір?. *ФЕЙСЕР*. URL: https://www.facerua.com/pakiet-dlia-novachkiv-shcho-takie-kompiutiernii-zir/ (дата звернення: 13.11.2024).
12. Учасники проектів Вікімедіа. Сегментація зображення – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Сегментація_зображення (дата звернення: 23.11.2024).
13. Що таке згорткові нейронні мережі (CNN, ConvNet)?. *TheTransmitted*. URL: https://thetransmitted.com/adlucem/shho-take-zgortkovi-nejronni-merezhi-cnn-convnet/ (дата звернення: 18.11.2024).
14. Учасники проектів Вікімедіа. Глибоке навчання – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Глибоке_навчання (дата звернення: 24.11.2024).
15. Учасники проектів Вікімедіа. Python – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Python (дата звернення: 30.11.2024).
16. Учасники проектів Вікімедіа. TensorFlow – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/TensorFlow (дата звернення: 27.12.2024).
17. Учасники проектів Вікімедіа. matplotlib – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Matplotlib (дата звернення: 29.11.2024).
18. Visual Geometry Group - University of Oxford. *Information Engineering Main/Home Page*. URL: https://www.robots.ox.ac.uk/~vgg/data/flowers/102/categories.html (дата звернення: 22.11.2024).
19. Flowers Dataset. *Kaggle: Your Machine Learning and Data Science Community*. URL: https://www.kaggle.com/datasets/imsparsh/flowers-dataset?resource=download (дата звернення: 21.11.2024).
20. Учасники проектів Вікімедіа. Ознака (комп'ютерне бачення) – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/w/index.php?title=Ознака_(комп'ютерне_бачення) (дата звернення: 20.11.2024).

21. Що таке комп'ютерний зір (Computer Vision, CV)? | TheTransmitted. *TheTransmitted*. URL: https://thetransmitted.com/adlucem/shho-take-kompyuternyj-zir-computer-vision-cv/ (дата звернення: 15.11.2024).
22. TensorFlow. *TensorFlow*. URL: https://www.tensorflow.org (дата звернення: 20.11.2024).
23. Modern approaches to computer vision / R. M. Tymchyshyn та ін. *Upravlâûŝie sistemy i mašiny*. 2018. № 6 (278). С. 46–73. URL: https://doi.org/10.15407/usim.2018.06.046 (дата звернення: 16.11.2024).
24. Machine Learning, ML. *IT-Enterprise*. URL: https://www.it.ua/knowledge-base/technology-innovation/machine-learning (дата звернення: 12.11.2024).
25. yaneya. Машинне навчання простими словами. Частина 1. *Механіко-математичний факультет Львівського національного університету імені Івана Франка*. URL: http://www.mmf.lnu.edu.ua/ar/1739 (дата звернення: 12.11.2024).
26. Учасники проектів Вікімедіа. Гауссове згладжування – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Гауссове_згладжування (дата звернення: 15.11.2024).
27. Учасники проектів Вікімедіа. Медіанна фільтрація – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Медіанна_фільтрація (дата звернення: 15.11.2024).
28. Учасники проектів Вікімедіа. Білатеральний фільтр – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Білатеральний_фільтр (дата звернення: 16.11.2024).
29. Contributors to Wikimedia projects. Adaptive histogram equalization - Wikipedia. *Wikipedia, the free encyclopedia*. URL: https://en.wikipedia.org/wiki/Adaptive_histogram_equalization (дата звернення: 15.11.2024).

ДОДАТКИ

Додаток А

Технічне завдання

1. Вступ

У сучасному світі технології комп’ютерного зору відіграють важливу роль у різних сферах, зокрема у флористиці, ботаніці, садівництві та екологічному моніторингу. Автоматичне розпізнавання квітів за зображеннями дозволяє значно спростити процес ідентифікації рослин, зробивши його доступним як для науковців, так і для аматорів.

Для вирішення цієї задачі була розроблена програма Blossom Lens, яка поєднує алгоритми комп’ютерного зору, методи машинного навчання та інтеграцію з Wikipedia API для автоматичного отримання додаткової інформації про розпізнані квіти. Програма дозволяє користувачам завантажувати фотографії квітів, отримувати прогнозовану назву виду разом із ймовірністю класифікації, а також переглядати короткий опис квітки з Вікіпедії безпосередньо у вікні застосунку.

Окрім цього, Blossom Lens має інтуїтивно зрозумілий інтерфейс, реалізований на базі Tkinter, що забезпечує зручну взаємодію з програмою. Новий функціонал дозволяє не лише отримувати класифікацію квітки, а й переглядати історію розпізнань та експортувати результати у текстовий або PDF-формат.

Розробка Blossom Lens спрямована на спрощення процесу ідентифікації квітів, автоматизацію аналізу рослин, підвищення точності класифікації, а також розширення можливостей користувачів через інтеграцію з Wikipedia API. Завдяки цим удосконаленням програма забезпечує швидкий доступ до актуальної інформації та підвищує зручність роботи з розпізнаними зображеннями.

2. Підстави для розробки

Розробка Blossom Lens зумовлена поєднанням кількох чинників: зростанням популярності ботанічних досліджень серед широких верств населення, розвитком глибинного навчання в обробці зображень, а також потребою у простих, локальних інструментах для швидкої ідентифікації рослин.

Серед доступних застосунків для розпізнавання квітів більшість працюють як мобільні додатки (наприклад, Pl@ntNet, PlantSnap), які потребують стабільного Інтернет-з'єднання та не завжди забезпечують гнучкість у збереженні результатів або модифікації функціоналу.

Blossom Lens вирішує ці проблеми шляхом створення настільного локального застосунку, який не потребує підключення до зовнішніх серверів для проведення класифікації. Користувач має повний контроль над даними, може зберігати результати локально, шифрувати їх і за потреби розширювати функціонал шляхом оновлення моделі чи навчального набору.

Додатковою перевагою є інтеграція з Wikipedia API, що дозволяє отримувати короткий опис про кожну розпізнану квітку без потреби в ручному пошуку. Така інтеграція дає змогу використовувати програму в навчальному процесі, а також як демонстраційний інструмент для ознайомлення з принципами машинного навчання у візуальних завданнях.

Отже, підставами для розробки Blossom Lens є:

– потреба в доступному інструменті для класифікації квітів у наукових, освітніх та прикладних цілях;

– бажання забезпечити повну автономність застосунку без зовнішньої залежності від онлайн-сервісів;

– інтеграція навчальних і довідкових функцій в одному середовищі;

– актуальність використання комп'ютерного зору та нейронних мереж у практичних задачах.

3. Призначення розробки

Програма Blossom Lens розроблена з метою створення доступного та ефективного інструменту для автоматичного розпізнавання квітів на зображеннях. Вона використовує можливості комп’ютерного зору та глибинного навчання, що дає змогу визначати вид квітки за її фотографією без потреби у спеціальних знаннях або ручному порівнянні з каталогами.

Blossom Lens має інтуїтивно зрозумілий інтерфейс і працює автономно на персональному комп’ютері, не потребуючи підключення до серверів. Це робить її зручною як для професіоналів, так і для аматорів — ботаніків, викладачів, студентів, школярів, флористів, садівників та всіх, хто цікавиться рослинним світом.

Завдяки поєднанню нейромережевої класифікації, інтеграції з енциклопедичним API Wikipedia та підтримки збереження результатів у різних форматах, програма може використовуватись як у наукових дослідженнях і навчальному процесі, так і в прикладних задачах — наприклад, для екологічного моніторингу чи оформлення флористичних композицій.

4. Вимоги до програми чи програмного продукту

Програмне забезпечення повинно забезпечувати завантаження зображень у форматах PNG, JPG або JPEG та їхню автоматичну обробку, включаючи масштабування, нормалізацію й аналіз за допомогою попередньо навченої нейронної мережі EfficientNetB0. Результатом класифікації має бути визначення виду квітки з відображенням назви, рівня впевненості моделі та короткого опису, отриманого з Wikipedia.

Програма повинна працювати у середовищі Windows 10 або новішому з попередньо встановленим Python 3.11 і необхідними бібліотеками. Основний функціонал має бути доступний офлайн, за винятком звернень до Wikipedia, які потребують з’єднання з Інтернетом.

Інтерфейс має бути графічним, реалізованим на базі бібліотеки Tkinter, із підтримкою української та англійської мов. Результати класифікації повинні

відображатися у вікні програми з можливістю їхнього збереження у форматах TXT і PDF. Усі результати повинні автоматично шифруватися та зберігатися у локальному CSV-файлі, а дії користувача — реєструватися у log-файлі.

Програма повинна бути стабільною, швидкою у роботі з одним зображенням, і забезпечувати базовий захист даних. Важливою вимогою є адаптивність інтерфейсу та можливість майбутнього оновлення класифікаційної моделі без повного переписування коду.

5. Вимоги до програмної документації

Програмна документація до Blossom Lens повинна містити повний набір матеріалів, що забезпечують розуміння структури програми, її функціональності, способу використання та підтримки. До обов'язкових компонентів документації належать: технічне завдання, опис архітектури програмного засобу, інструкція користувача, коментарі до програмного коду та результати тестування.

Документація має бути структурованою, зрозумілою та актуальною. Усі матеріали повинні супроводжуватись поясненнями щодо налаштування, запуску, оновлення моделі та можливих дій користувача у разі виникнення помилок. Особливу увагу слід приділити підтримці відповідності документації до змін, які можуть вноситися в програму після її впровадження.

Інструкція користувача повинна пояснювати встановлення програми, вибір зображення, отримання результатів, зміну мови та збереження звітів. Усі документи мають відповідати загальноприйнятим вимогам до оформлення технічної документації в галузі програмного забезпечення.

6. Техніко-економічні показники

Економічна доцільність створення Blossom Lens полягає в автоматизації рутинного процесу класифікації квітів, що дозволяє зменшити витрати часу та людських ресурсів. Завдяки локальному виконанню без потреби у підключенні до серверів знижується вартість обробки даних, а також зростає стабільність роботи програмного засобу.

Програма не потребує великої обчислювальної потужності, тому може використовуватись на звичайних комп’ютерах у навчальних аудиторіях, лабораторіях, офісах та вдома.

Blossom Lens має потенціал для подальшого комерційного розвитку — зокрема, адаптації до мобільних платформ, додавання нових функцій (наприклад, визначення пори року цвітіння, геолокації виду), а також розширення навчального набору. Це відкриває можливості як для індивідуального використання, так і для застосування у професійній діяльності.

7. Стадії і етапи розробки

Розробка Blossom Lens проходила кілька послідовних етапів. На першому етапі було здійснено аналіз потреб цільової аудиторії, вивчено наявні аналоги та визначено базові функціональні вимоги. Далі було спроєктовано архітектуру програмного забезпечення, розроблено модель класифікації на основі EfficientNetB0 та підготовлено датасет квітів для навчання.

Наступним кроком стала реалізація графічного інтерфейсу з підтримкою двох мов, адаптацією до форматів зображень та інтеграцією з Wikipedia API. Було додано функції збереження у форматах PDF і TXT, а також механізм шифрування історії результатів. Після цього проводились етапи тестування, оптимізації продуктивності, виправлення помилок, оформлення документації та підготовка до фінального впровадження.

8. Порядок контролю і приймання

Контроль якості програмного засобу здійснювався шляхом внутрішнього тестування всіх модулів: класифікаційної моделі, обробки зображень, генерації PDF і TXT-звітів, коректності багатомовного інтерфейсу та збереження результатів.

Під час перевірки тестувалися різні типи вхідних зображень (розмір, якість, формат), перевірялася точність класифікації, стабільність у випадку помилок

користувача (наприклад, відсутність моделі або неправильний формат файлу), а також робота програми без Інтернету.

Програму вважалося прийнятою після успішного виконання усіх тестів, перевірки працездатності функціоналу відповідно до технічного завдання, підтвердження зручності інтерфейсу й оформлення супровідної документації.

Додаток Б

Інструкція користувачу

1. Загальні відомості:

Blossom Lens — це локальний застосунок для автоматичного розпізнавання квітів на зображеннях, створений на основі згорткової нейронної мережі EfficientNetB0. Програма має графічний інтерфейс, реалізований засобами Tkinter, і підтримує дві мови: українську та англійську. Застосунок орієнтований на студентів, викладачів, ботаніків, а також усіх користувачів, які цікавляться природничими науками та технологіями штучного інтелекту. Програма працює повністю локально, без необхідності постійного інтернет-з’єднання (за винятком запитів до Wikipedia).

2. Функціональне призначення:

Blossom Lens дозволяє користувачеві обрати зображення квітки зі свого комп’ютера, автоматично обробити його, класифікувати вид за допомогою нейромережі та отримати результат у вигляді назви квітки й рівня впевненості моделі. Після класифікації програма також відображає короткий опис про відповідний вид із Wikipedia мовою, що відповідає мові інтерфейсу. За бажанням користувач може зберегти результати класифікації у форматі TXT або PDF, а також переглянути повну енциклопедичну статтю через браузер. Додатково реалізовано автоматичне шифрування та резервне копіювання результатів у зашифрованому CSV-файлі.

3. Умови застосування сервісу:

Програма працює на операційній системі Windows (версія 10 або новіша) та потребує встановленого Python 3.11. Необхідно також мати встановлені бібліотеки: TensorFlow, NumPy, OpenCV, Pillow, Wikipedia, pandas, matplotlib, seaborn, fpdf, cryptography та tkinter. Для коректної роботи потрібно, щоб у папці

з програмою містилися попередньо збережені файли моделі (.keras) та словник класів (class_names.json). У разі відсутності ключа шифрування (encryption_key.key), він буде згенерований автоматично. Зображення, що використовуються для класифікації, повинні бути у форматах JPG, JPEG або PNG.

4. Повідомлення користувачу:

Після запуску застосунку на екрані з’являється головне вікно з логотипом та кнопками. Користувач натискає «Оберіть зображення» та обирає файл із комп’ютера. У випадку успішної обробки зображення програма відобразить його, назву квітки, рівень довіри (ймовірність), коротку енциклопедичну довідку та запропонує зберегти результати або переглянути більше у Wikipedia. Якщо ймовірність класифікації занизька або квітка не розпізнана, на екрані з’явиться відповідне повідомлення про помилку. Також передбачені повідомлення про відсутність необхідних файлів або бібліотек, а дії користувача автоматично журналюються у лог-файл (blossom_lens.log).

5. Опис роботи сервісу:

Після запуску застосунок завантажує навчену модель нейронної мережі та список класів. Коли користувач завантажує зображення, воно піддається попередній обробці — масштабуванню, нормалізації та фільтрації шуму. Потім воно подається на вхід моделі, яка обчислює ймовірності належності до кожного з класів. Клас із найвищою ймовірністю визначається як результат, який відображається на екрані разом із довідковою інформацією. Користувач може натиснути кнопку «Зберегти як TXT» або «Зберегти як PDF» для створення звіту. Всі результати додатково зберігаються у резервний зашифрований файл для подальшого аналізу або безпечного зберігання. У будь-який момент користувач може очистити інтерфейс і повторно розпочати розпізнавання. У разі реалізації перемикача мов, мову інтерфейсу можна змінити без перезапуску програми.

АНОТАЦІЯ

Малярчук Б. М. — **Дослідження методів комп’ютерного зору для класифікації зображень на прикладі розпізнавання видів квітів. Рукопис**

Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр» за спеціальністю 122 Комп’ютерні науки. Волинський національний університет імені Лесі Українки, Луцьк, 2025 р.

Робота присвячена розробці локального програмного застосунку Blossom Lens, призначеного для автоматичного розпізнавання квітів за зображеннями із використанням методів комп’ютерного зору та глибокого навчання. У дослідженні розглянуто основні підходи до класифікації зображень із допомогою згорткових нейронних мереж, виконано вибір архітектури моделі, підготовку та аугментацію набору даних, проведено навчання та оцінювання якості моделі.

У роботі реалізовано повнофункціональний настільний застосунок з графічним інтерфейсом, підтримкою української та англійської мов, інтеграцією з Wikipedia API, а також можливістю експорту результатів у формати TXT і PDF. Передбачено автоматичне збереження класифікацій у захищеному форматі, побудову матриці плутанини та можливість донавчання моделі.

Розроблений програмний продукт протестовано на різних прикладах зображень, що підтвердило його стабільність, зручність у використанні та високу точність класифікації. Blossom Lens може бути використаний у навчальному процесі, наукових дослідженнях і в повсякденному житті для розпізнавання об’єктів флори.

Ключові слова: комп’ютерний зір, глибоке навчання, згорткова нейронна мережа, класифікація зображень, інтерфейс користувача, Python, Wikipedia API, Blossom Lens, Tkinter, PDF.