Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут деревообробних та
комп'ютерних технологій і дизайну
(повне найменування інституту)

Кафедра інформаційних технологій
(повна назва кафедри)

Пояснювальна записка

до дипломної роботи

другий (магістерський)
(рівень вищої освіти)

на тему: «_Інтелектуальна система класифікації зображень клітин крові _»

Виконав: студент _VI_ курсу групи _КН-61м
спеціальності
122 "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

_____Волянський І.М._____
(прізвище та ініціали)

Керівник _Шиманський В.М._____
(прізвище та ініціали)

Рецензент_____
(прізвище та ініціали)

Львів – 2021 року

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ Деревообробних та комп'ютерних технологій і дизайну
Кафедра Інформаційних технологій
Рівень вищої освіти другий (магістерський)
Спеціальність 122 «Комп'ютерні науки»
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Крошний І.М.
"____" _____2021 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ
Волянський І.М.
(прізвище, ім'я, по батькові)

1. Тема роботи *Інтелектуальна система класифікації зображень клітин крові*
керівник роботи *Шиманський В.М., к.т.н.*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "31" 12 2020 року № С-593
2. Термін подання студентом проекту (роботи) 10 грудня 2021 року
3. Вихідні дані до проекту (роботи) 51 сторінка
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
4.1. Стан проблемної області.
4.2. Інформаційне забезпечення
4.3. Математичне забезпечення
4.4. Програмне забезпечення
4.5. Розроблення стартап проекту
4.6. Висновки
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
6. Дата видачі завдання 18.12.2021 року

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів дипломної роботи	Термін виконання	Відмітка про виконання
1	З'ясування загальної постановки завдання; розроблення першого розділу пояснювальної записки	01.03.21	виконано
2	Проектування інтелектуальної та математичної моделі, розроблення конструкторської документації.	17.05.21	виконано
3	Реалізація програмного забезпечення та налаштування інтелектуальної моделі.	08.11.21	виконано
4	Висновки.	10.12.21	виконано

Студент: Волянський І.М. _____
(підпис)

Керівник роботи: Шиманський В.М. _____
(підпис)

ТЕХНІЧНЕ ЗАВДАННЯ

Проаналізувати існуючі види нейронних меж, що використовуються для вирішення задач класифікації зображень. Вибрати та аргументувати структуру та тип нейронної межі, засобами якої буде проводитись дослідження. Підготувати навчальну вибірку для реалізації нейронної мережі. Розробити інтелектуальну систему класифікації зображень клітин крові. Дослідити швидкодію реалізації алгоритмів навчання. Проаналізувати отримані результати.

Розроблена інтелектуальна система повинна задовольняти наступним вимогам:

- мати зручний та інтуїтивно зрозумілий інтерфейс;
- надавати можливість навчати та адаптувати інтелектуальну систему;
- на основі вхідних даних класифікувати зображення клітини крові.

РЕФЕРАТ

Дипломна робота містить 51 сторінка пояснювальної записки, 13 рисунків, 1 таблицю, 1 додаток, 14 джерел.

В даній роботі розроблено інтелектуальну систему класифікації зображень клітин крові з використанням штучних нейронних мереж. Веб-орієнтована інтелектуальна система створена на базі фреймворку Dash. Вона дозволяє за заданим зображенням клітини крові класифікувати його. Вона має інтуїтивно зрозумілий інтерфейс.

Ключові слова: Python, машинне навчання, нейронна мережа, алгоритм навчання, клітина крові.

ABSTRACT

Thesis contains 51 pages of explanatory note, 13 figures, 1 table, 1 appendix, 14 sources.

In this paper, an intelligent system for classifying blood cell images using artificial neural networks has been developed. The web-based intelligent system is based on the Dash framework. It allows you to classify it according to a given image of a blood cell. It has an intuitive interface.

Keywords: Python, machine learning, neural network, learning algorithm, blood cell.

ЗМІСТ

ВСТУП

Актуальність роботи. Нейронна мережа для розпізнавання зображень – це, мабуть, найпопулярніший спосіб застосування нейронної мережі. При цьому незалежно від особливостей розв'язуваних завдань, вона працює за етапами, найважливіші серед яких розглянемо нижче.

Як образи, що розпізнаються, можуть виступати різні об'єкти, включаючи зображення, рукописний або друкований текст, звуки і багато іншого. Під час навчання мережі їй пропонуються різні зразки з позначкою того, до якого саме типу їх можна віднести. Як зразок застосовується вектор значень ознак, а сукупність ознак у умовах повинна дозволити однозначно визначити, з яким класом образів має справу нейронна мережа.

Важливо під час навчання навчити мережу визначати як достатню кількість і значення ознак, щоб видавати хорошу точність на нових зображеннях, а й не перенавчитися, тобто, зайве не «підлаштувати» під навчальну вибірку з зображень. Після завершення правильного навчання нейронна мережа повинна вміти визначати образи (тих класів), із якими вона мала справи у процесі навчання.

Важливо враховувати, що вихідні дані для нейромережі повинні бути однозначними і несуперечливими, щоб не виникали ситуації, коли нейронна мережа видаватиме високі ймовірності приналежності одного об'єкта до кількох класів.

Об’єкт дослідження. Зображення клітини крові.

Предмет дослідження. Штучна нейронна мережа для класифікації зображень клітин крові.

Мета роботи. Розробити інтелектуальну систему класифікації зображень клітин крові.

Наукова новизна одержаних результатів полягає у такому. Створено та реалізовано нейронну мережу для класифікації зображень клітин

крові. Досліджено швидкодію алгоритмів навчання на заданій навчальній вибірці.

Практична значимість. Дозволяє здійснювати класифікацію зображень клітин крові.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1 Типологія задач класифікації

У машинному навчанні завдання класифікації можна розділити на такі типи:

- Двокласова (бінарна) класифікація (binary classification);
- Багатокласова класифікація на класи, що не перетинаються. (Multiclass classification);
- Багатокласова класифікація на класи, що перетинаються. (Multilabel classification).

Поставлене завдання можна вирішити декількома способами. Оскільки одне зображення може бути віднесено відразу до кількох класам, найбільш очевидний шлях – вирішити її як завдання типу multilabel classification. Лише невелика кількість моделей можуть працювати із завданнями у цій постановці у чистому вигляді, серед них – штучні нейронні мережі. Такий підхід ускладнюється тим, що як вектора передбачень моделі видають вектор речових чисел від 0 до 1 і для кінцевого визначення набору міток необхідно підбирати граничне значення для кожного класу.

Завдання багатокласової класифікації на класи, що перетинаються можливо вирішити шляхом побудови набору бінарних класифікаторів за типом "один проти одного" і "один проти всіх". Підхід «один проти всіх» переважно через те, що в цьому випадку потрібно навчити меншу кількість моделей. Нестача таких рішень полягає в те, що при введенні в експлуатацію для об'ємної бази зображень фотобанку буде необхідно отримати результати відразу кількох класифікаторів, що займе більше часу Гідність підходу у відносній простоті вирішення задачі бінарної класифікації, можливості використовувати менш складні та вимогливі моделі.

Третій спосіб вирішення – звести завдання до багатокласового типу класами, що не перетинаються, ввівши такі класи, як «clip-art – photo», "clip-art - illustration", "background - photo" і т.п. Цей спосіб позбавлений недоліків двох попередніх і з цього погляду є найоптимальнішим.

1.2 Огляд існуючих методів класифікації

Сьогодні для класифікації зображень використовується безліч моделей машинного навчання. Один із найпоширеніших класифікаторів – штучна нейронна мережа, зокрема, згорткова нейронна мережа. Як класифікатор може виступати метод random forest та метод k-Nearest Neighbors, застосування якого до класифікації зображень розглянуто у роботі.

Алгоритм kNN (k-Nearest Neighbours, метод k найближчих сусідів) ґрунтується на гіпотезі компактності: припущенні, що при введенні деякої вдалої міри подібності об'єктів подібним об'єктам будуть відповідати подібні відповіді. Об'єкт відноситься до того класу, елементів якого виявиться більше серед його найближчих сусідів. Існує варіація алгоритму kNN – алгоритм k виважених найближчих сусідів, який вирішує проблему досягнення максимуму відразу на кількох класах шляхом строго спадної послідовності речових ваг wi, що задають внесок i-го сусіда в класифікацію.

Навчання класифікатора зводиться до запам'ятовування навчальної вибірки. Алгоритм має одну головну перевагу – простоту реалізації - і помітні недоліки, а саме, неефективна витрата пам'яті того, що доводиться зберігати всю навчальну вибірку, та порівняння об'єкта, що класифікується, з усіма об'єктами вибірки за O(n) операцій (для завдань з великими вибірками або високою частотою запитів це може бути накладно).

Виходячи з цього, можна зробити висновок, що алгоритм kNN малозастосовний для класифікації зображень. Тим не менш, у роботі було запропоновано поліпшення алгоритму kNN застосовно до зображень. Дослідження покращеного алгоритму не входить до рамки даної роботи,

тому аналіз його застосування до поточного завдання залишається предметом подальших досліджень.

Алгоритм random forest (випадковий ліс) може бути віднесений до мета-алгоритмам: алгоритмам, що здійснюють навчання множини класифікаторів і усредняющим їх значення. Він реалізує продовження ідеї алгоритму bagging: побудови безлічі класифікаторів на випадкових підвиборках навчальної вибірки та здійснення кінцевого передбачення шляхом «голосування» моделей за той чи інший клас.

Random forest реалізує навчання багатьох вирішальних дерев (decision trees) на різних випадкових підвиборках навчальної вибірки, при цьому, на відміну від bagging, процес побудови вирішального дерева ведеться на випадковій підвиборці ознак об'єкта, тобто. деякі випадкові ознаки у процесі побудови не враховуються. Цим збільшується стохастичність алгоритму, що дозволяє більш ефективно уникати проблеми перенавчання моделі, яка є слабкою місцем вирішальних дерев.

1.3 Машинне навчання

Машинне навчання як сферу досліджень дещо важко описати коротко, через широту області, кількість поточних досліджень, міждисциплінарні аспекти галузі та безліч інших факторів. Для цілей цього звітунаступне визначення в якості відправної точки буде використано, оскільки воно є досить точним:

«Кажуть, що комп'ютерна програма навчається на досвіді E щодо деякого класу завдань T та показника ефективності P, якщо її продуктивність при виконанні завдань у T, що вимірюється P, покращується з досвідом E."

З математичної точки зору проблему можна сформулювати як застосуваннямашинного навчання алгоритмудля пошуку невідомої математичної функції з відомою областю входу та відомою спільною

областю, що є вихідною. У більш неформальних термінах програма має знайти математичний зв’язок між даними та цільовим введенням.

Підключення залежить відзадачі. Відповідним прикладом вищезгаданих класів завдань є пошук зв'язку між векторизованим представленням вхідних даних і вихідних даних, що складається з набору дискретних категорій або розподілу ймовірностей. Конкретні приклади такого завдання можуть полягати в розробці програми для класифікації зображень і виявлення об'єктів, наприклад, розпізнавання облич на зображеннях та пошуку для них обмежувальних рамок. Іншими прикладами типів завдань є виявлення аномалій, наприклад виявлення кредитного шахрайства та машинний переклад, наприклад програмне забезпечення для перекладу.

Ефективність реалізації алгоритму машинного навчання зазвичай або виражається в точності реалізації, наприклад, який відсоток прикладів було класифіковано правильно, або в частоті помилок, наприклад, яке відношення неправильно класифіковано до правильного. приклади є. Складна частина тут полягає не стільки у виборі міри, скільки у тому, який (і) аспект(-и) результату, який слід виміряти в першу чергу. Чи слід, наприклад, вибрати відсоток правильно транскрибованих цілих послідовностей як єдину міру чи брати до уваги правильно транскрибовані частини послідовностей під час вимірювання продуктивності?

Іншим ускладнюючим фактором є те, що може бути важко виміряти обрану величину через практичні труднощі, і в таких випадках необхідно вибрати іншу міру.

В основному є два види навчання або досвіду, які алгоритм машинного навчання може мати під час процесу навчання або навчання; навчання без нагляду та контролю. Процес навчання базується на наборі даних, що складається з заданої кількості прикладів або точок даних.

Неконтрольоване навчання засноване на наборах даних, які містять функції, мета яких полягає в тому, щоб дізнатися корисні властивості структури набору даних або, точніше, знайти основний розподіл

ймовірностей набору даних. Навпаки, контрольоване навчання базується на наборах даних, що складаються з багатьох функцій, але з мітками до кожного прикладу. Мета навчання зполягає в тому, щоб навчитися передбачати мітку, пов’язане значення, з довільного прикладу в наборі даних після завершення сесії навчання. Терміни «контрольований» і «без нагляду» походять від того факту, що в алгоритмі навчання є вчитель, який показує алгоритму, що робити в першому випадку, але передбачається вчитися на даних без керівництва в останньому випадку.

Найскладнішою частиною машинного навчання, незалежно від алгоритму(ів), що використовується в процесі навчання, є створення програм, які добре узагальнюють; тобто програми, які працюють так само або майже так само добре на неспостережуваних входах, як і ті, які спостерігаються під час навчання. Головне, що відрізняє машинне навчання від оптимізації, полягає в тому, що мета полягає в мінімізації як помилки навчання, так і помилки тесту. Вищезгадані набори даних розділені на два заздалегідь визначені набори для використання в процесі навчання, один для навчання, а другий для тестування.

Мета машинного навчання тепер може бути визначена більш точно; щоб мінімізуватинавчання помилкута мінімізувати розрив між помилкою навчання та помилкою тесту. У спробі досягти вищезгаданої мети виникають ще дві проблеми; недообладнання та переобладнання на тренувальному наборі.

Недооснащення відбувається, коли модель не може досягти достатньо низької похибки навчання, а перенавчання відбувається, коли розрив між помилкою навчання та помилкою тесту занадто великий. Перша проблема спричинена моделлю з низькою репрезентативною ємністю, яка не може належним чином підходити до навчального набору, друга може бути викликана моделлю з високою ємністю, яка запам'ятовує властивості навчального набору занадто добре, щоб працювати достатньо на тестовому наборі. Переобладнання також може бути викликано навчальним набором, який занадто малий для належного узагальнення.

Недостатність можна досить легко виправити за допомогою моделі з достатньою ємністю, але переобладнання може бути важче виправити через те, що отримати більшу кількість даних не завжди можливо. Яскравим прикладом техніки, яку можна запровадити під час навчання, є регуляризація, яка використовується для обмеження простору потенційних функцій (оскільки мета – знайти найкращу). Тут не так багато місця, щоб вдаватися в деталі, і проблеми, які можуть виникнути, і відповідні рішення будуть обговорені в наступних розділах.

Параметри алгоритму машинного навчання, які не адаптуються або не змінюються самим алгоритмом під час навчання, називаються гіперпараметрами, параметрами, які можна налаштувати, щоб змінити поведінку алгоритму навчання. Тут доречно ввести додатковий розділ набору даних, вихідних навчальних даних у навчальний набір і набір для перевірки.

Навчальний набір використовується виключно для налаштування внутрішніх параметрів, наприклад ваг і зміщень, під час навчання, а набір перевірки використовується для вимірювання поточної помилки узагальнення та відповідного налаштування гіперпараметрів. Набір перевірки не використовується для налаштування внутрішніх параметрів. Важлива відмінність між набором перевірки та тестовим набором полягає в тому, що останній взагалі не використовується під час навчального процесу і не дає жодного внеску в навчальний процес – він просто використовується для вимірювання продуктивності.

Як останнє спостереження в цій главі; Існує багато алгоритмів машинного навчання на вибір, багато чого можна детально описати тут або навіть зробити побіжний огляд. Оскільки фреймворки в дослідженні є реалізацією компонентів, функціональності та алгоритмів, необхідних для навчання нейронних мереж і впровадження глибокого навчання, немає потреби.

1.4. Висновки

Сьогодні одними з найважливіших чадам комп'ютерної обробки інформації є завдання автоматичного розпізнавання образів. Для вирішення даного класу завдань вже понад півстоліття ведуться дослідження в таких галузях науки як математична статистика, штучний інтелект, прийняття рішень, цифрове оброблення сигналів тощо.

Під час вирішення завдань розпізнавання зображень однією з найважливіших є проблема вибору векторів ознак. Оскільки аналізована під час вирішення таких завдань предметна область є погано формализуемой, універсальних критеріїв вибору векторів ознак завдання розпізнавання. Тому дослідникам доводиться вибирати відповідний набір ознак експериментально, керуючись специфікою розв'язуваного завдання. Таким чином, отримання критеріїв вибору ознак зображень у завдання розпізнавання є актуальною проблемою.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Згорткові нейронні мережі

Однією із слабких сторін звичайної нейронної мережі з прямим зв'язком із повністю зв'язаними шарами є те, що вона не має попереднього вбудованого припущення щодо даних, з яких вона має вчитися. Він агностичний щодо структури даних і однаково обробляє всі дані. Це легко стають проблемою, частково через надлишкові параметри, що виникають, частково через розмір отриманої надлишковості, оскільки нейронна мережа може, як згадувалося кілька разів раніше, дуже сильно розростатися. Також нерозумно використовувати цей підхід у випадку, наприклад, даних, що складаються із зображень, оскільки це означатиме, що просторова структура зображення буде ігнорована, а навчання почнеться з припущення, що всі пікселі однаково пов’язані.

Зрозуміло, що потрібен інший підхід, тому в цьому підрозділі йдеться про особливий типіз нейронних мережпрямим зв’язком – згорткові нейронні мережі. Згорткові нейронні мережі розроблені з певними припущеннями щодо даних, припущеннями, які відповідають даним зображення, а також іншими даними з подібною внутрішньою структурою.

Найфундаментальнішими операціями згорткових нейронних мереж є, мабуть, не дивно, згортки. Концепція згорток у контексті нейронних мереж починається з уявлення про шари, що складаються з нейронів з локальним рецептивним полем, тобто нейронів, які підключені до обмеженої області вхідних даних, а не до всього. У випадку даних зображення це означає, що кожен нейрон підключений лише до обмеженої області пікселів, наприклад, до квадрата 3 x 3 пікселі у верхньому лівому куті зображення. Рецептивні поля нейронів такожперекриваються певною мірою, оскільки (продовжуючи приклад із попереднього речення) два сусідніх нейрони мають рецептивні поля, які зміщені на один або кілька пікселів по горизонталі або вертикалі

відносно один одного. Можна візуалізувати ковзне вікно заданого розміру, яке ковзає зліва направо і зверху вниз (можна вважати, що дані мають двовимірну структуру відтепер), з'єднуючи вихід вікна на кожному проході з нейроном.

Результатом буде власна двовимірна структура, карта або зображення активацій від кожного нейрона. «Ковзне вікно» в цьому контексті правильніше назвати ядром, набором ваг і зміщенням. Це означає, що кожен нейрон у такому шарі або карті має однакові параметри, і, таким чином, можна сказати, що він вивчає ту саму функцію, що призводить до власної назви таких структур, карт ознак. Операція, описана вище, застосування ядра до вхідної карти та створення карти об'єктів, є згорткою. Використання згорток, інтуїтивно, дуже підходить для дуже просторово корельованих даних, оскільки можна стверджувати, що, наприклад, два пікселі в оці та навколо нього на зображенні обличчя мають більш значущий зв'язок, ніж два пікселі. пікселів, вибраних випадковим чином із зображення.

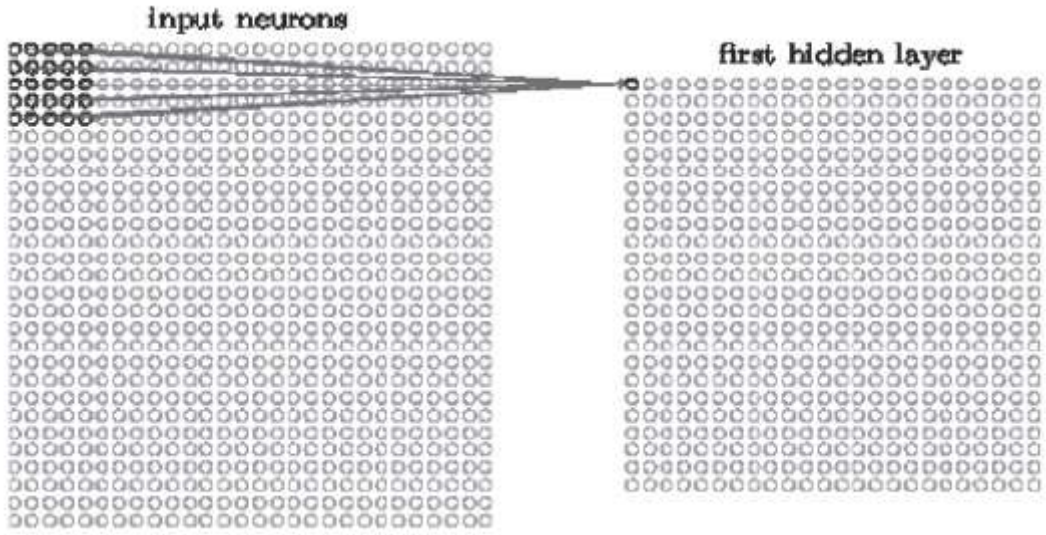


Рис 2.1. Прихований шар нейронів з локально сприйнятливими полями.

Згорткові шари в згорткових нейронних мережах складаються з множиннихознак, що складаються картразом із пов'язаними нейронами. Ось чому ядра, пов'язані з картами ознак, також називаються фільтрами або каналами. Згорткові шари також можна поєднувати з іншими згортковими

шарами, які в більш неформальних термінах можна описати як створення карт об’єктів на картах об’єктів або вивчення об’єктів вищого рівня. Ще один важливий аспект згорткових нейронних мереж полягає в тому, що спільне використання параметрів зменшує абсолютну кількість необхідних параметрів, на додаток до значного відносного зменшення параметрів у порівнянні з мережею, яка використовує лише повністю підключені шари для виконання того ж завдання.

Загалом, формула для кількості параметрів у згортковому шарі, що виконує двовимірні згортки : (M _X _ Y +1) _N, де M — кількість карт ознак у вхідних даних (з урахуванням червоно-зелено-синіх каналів у кольорові зображення), X та Y — висота й ширина згортки (3 x 3, 5 x 5 тощо) і N — кількість карт об’єктів у вихідних даних, доданих для врахування зміщення. Два останні аспекти згорток, які будуть обговорюватися в цьому розділі, — це крок і відступ. У попередньому абзаці неявно було зазначено, що розмір кроку, або кроку, під час згортки був один, але можна використовувати і вищі кроки. Заповнення вхідних даних використовується для збереження просторової роздільної здатності під час згортки, оскільки без нього воно стискається на постійний коефіцієнт в обох вимірах залежно від розміру згортки.

Іншим видом важливого рівня в згорткових нейронних мережах є шари об'єднання, які застосовують операцію об'єднання. Загалом об’єднання складається з перетворення карт об’єктів уоб’єктів менші, більш агреговані карти. Найпоширенішим видом об'єднання є максимальний пул, який бере неперекриваються області заданого розміру та виводить найбільшу активацію в цьому регіоні, наприклад, розбиваючи вхідну карту на секції розміром 2 x 2, вибираючи максимальне значення і створення вихідної карти, що складається з цих значень, але лише на четверту частину початкового розміру. Таким чином, шар об’єднання об’єднує вхідні дані з попереднього згорткового шару, зберігаючи кількість карт об’єктів.

Обгрунтування об’єднання полягає в тому, щоб зменшити просторову розмірність у мережі при збереженні просторової інформації,

використовуючи сильну кореляцію між точками даних, які знаходяться близько один до одного. Як і у випадку звивин, шари об’єднання можуть використовувати крок для досягнення ще більшої агрегації.

Завдяки введеним шарам об’єднання структуру згорткової нейронної мережі можна зробити більш зрозумілою. Головною метою згорткової нейронної мережі є перетворення просторового представлення вхідних даних у представлення, багате ознаками, великий простір ознак, з якого можна класифікувати. Таким чином, базова структура згорткової нейронної мережі — це згорткові шари, які чергуються як підгонка з шарами об’єднання, у поєднанні з використанням методів, обговорених у попередньому розділі, і зазвичай пара повністю пов’язаних шарів з класифікатором на кінці.

Ця структура існує протягом багатьох років, принаймні з архітектури Yann LeCuns LeNet5 в 1998 році, але ті самі ідеї є корисними й сьогодні. Варто зазначити, що основний підхід все ще працює, тим більше, що дослідження згорткових нейронних мереж є інтенсивними і що було зроблено і робиться багато великих проривів, особливо в області класифікації зображень і виявлення об’єктів.

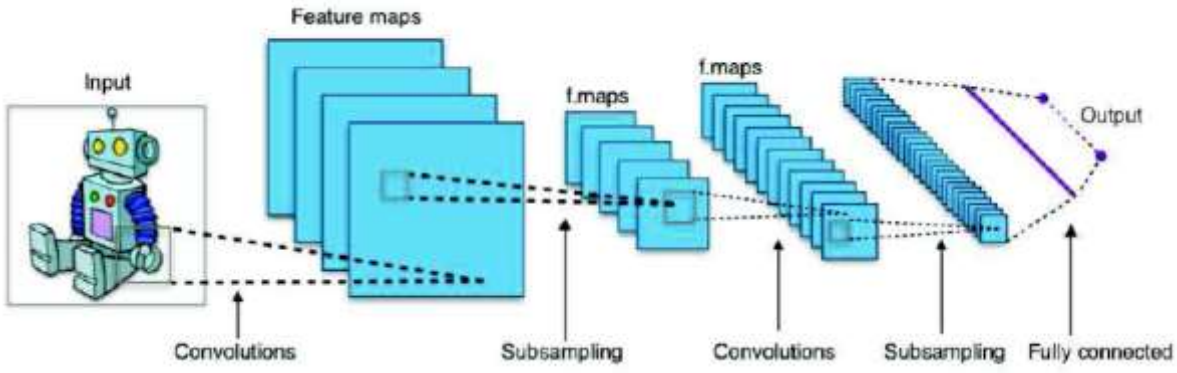


Рис 2.2. Згорткова нейронна мережа.

2.2.Конструкції та архітектури нейронних мереж

У цьому розділідеякі з найбільш значущих проектів і архітектур будуть докладно представленінейронних мереж, зокрема згорткових нейронних мереж.

Загальним завданням, на яке спрямовані всі наведені нижче архітектури, є класифікація зображень, щоб мати можливість якомога правильніше класифікувати зображення в ряд заздалегідь визначених категорій.

Першим серйозним проривом після моделі LeNet5, згаданої в попередньому розділі, став AlexNet Алекса Крижевського у 2012 році, внесок у конкурс ImageNet. ImageNet— це база даних, що складається з мільйона зображень, відсортованих за тисячами категорій, які використовуються для дослідження розпізнавання зображень та виявлення об’єктів. AlexNet побудований на основі підходу LeNet5, використовував випрямлені лінійні блоки та працював на двох GPU. Цей підхід був додатково вдосконалений за допомогою мереж VGG (Visual Geometry Group), мереж із набагато більшою глибиною та точністю.

Основним нововведенням було використання менших згорток розміром 3x3 і 5x5 замість набагато більших згорток, які використовуються в AlexNet, і накладання шарів із використанням цих менших згорток один на одного.

Однак мережі стали досить великими, і навчання доводилося проводити по частинах через величезну складність. Продуктивність мереж VGG під час виведення або класифікації після навчання також була досить витратною.

Як уже згадувалося, згаданий вище підхід має деякі очевидні недоліки, особливо щодо обчислювального навантаження на навчання та обслуговування моделі. Дослідницька група в Google скористалася іншим підходом, враховуючи ці міркування, підхід, який призвів доу переможця конкурсу ImageNet2014 році, GoogleNet, дизайну мережі на основіInception модуліві першої мережі, яка використовує архітектуру Inception. Основна

ідея модуля Inception полягає в тому, щоб шари працювали на вхідних даних паралельно, як згорткові, так і об’єднані шари, шари з різними розмірами ядер, а потім об’єднували вихідні дані в один шар. GoogleNet розпочався зі звичайного набору шарів, як у вищезгаданих моделях AlexNet і VGG, великої середньої частини початкових модулів, накладених один на одного, і, нарешті, глобального рівня усереднення з класифікатором softmax, остання частина, натхненна документом Network In Network.

Іншим важливим аспектом архітектури Inception є використання згорток 1x1. Згортки 1x1 служать двом корисним цілям у згорткових нейронних мережах; їх можна використовувати, щоб дешево, з точки зору обчислень, додати нелінійність за допомогою додаткових активацій і дешево зменшити кількість функцій на вихідному рівні. Ці згортки використовуються для побудови так званих шарів вузьких місць, конструкцій, що складаються з шару, що використовує 1x1 згорткидля зменшення вибірки вихідних об'єктів зазвичай у чотири рази, другий шар із використанням більшого ядра з меншою кількістю функцій і остаточний шар згорток 1x1 для повторної вибірки функцій. Ці вузькі місця можуть зменшити кількістьобчислень необхіднихмайже в десять разів.

Поєднання вузьких місць і модулів Inception призвело до успіху архітектури та наступних мережевих проектів Inception V2 і V3. Мережі Inception, порівняно з мережами VGG та підходом за ними, підвищили точність у ImageNet Challenge, одночасно зменшивши кількість параметрів та кількість операцій.

Остаточний підхід, який буде представлено тут, — це підхід команди Microsoft, яка стоїть за ResNet, переможця конкурсу ImageNet 2015 та архітектурного підходу, який досяг

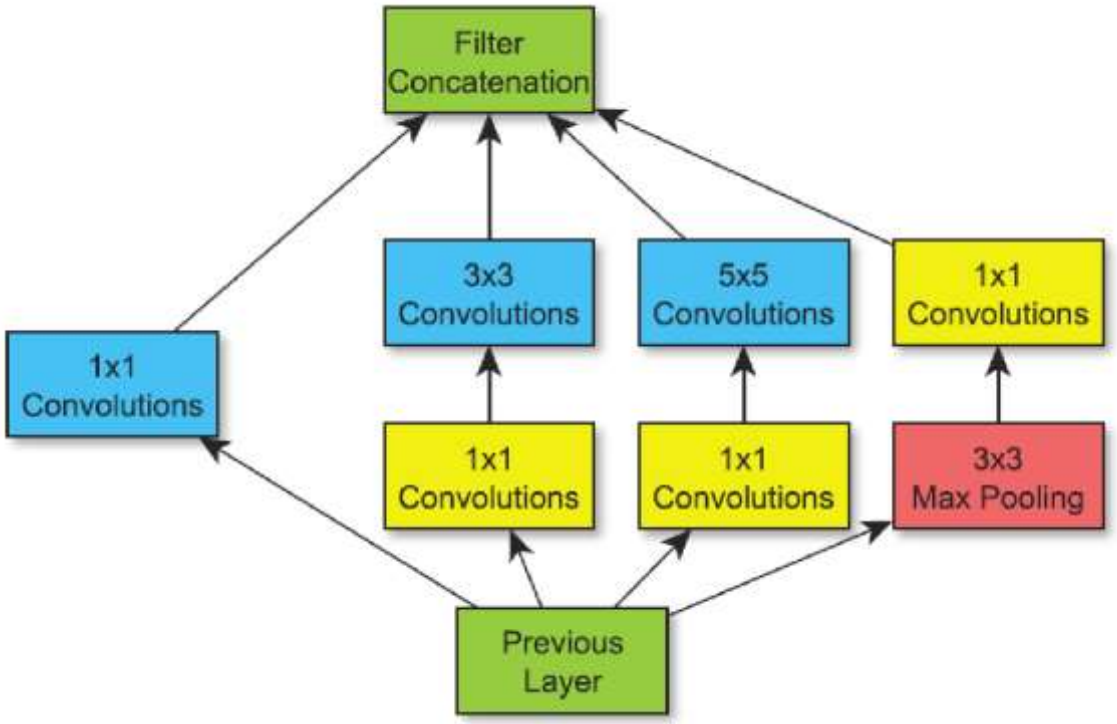


Рис 2.3: Початковий модуль V1.

безпрецедентну глибину мережі за допомогою нової техніки: залишкові блоки, що складаються з кількох шарів, і мережі, що складаються з цих блоків, залишкові нейронні мережі. Основна ідея залишкових блоків полягає в тому, щоб запустити вхід через два або більше шарів, а потім, що важливо, додати початковий вхід до виходу другого шару, нейрон за нейроном, і застосувати функцію активації для отримання результату блок. Можна сказати, що блок складається із залишкового з'єднання, шарів, через які проходить вхід, і з'єднання пропуску, що обходить початковий вхід.

Залишкове з'єднання зазвичай є вузьким місцем, аз пропуском з'єднаннязазвичай є лише вхідним, але існують і інші варіанти, насамперед для зменшення просторової розмірності. Команді ResNet вдалося використати залишкові блоки для розробки татисячою навчання згорткової нейронної мережі з понадшарів, що є рекордом у глибокому навчанні.

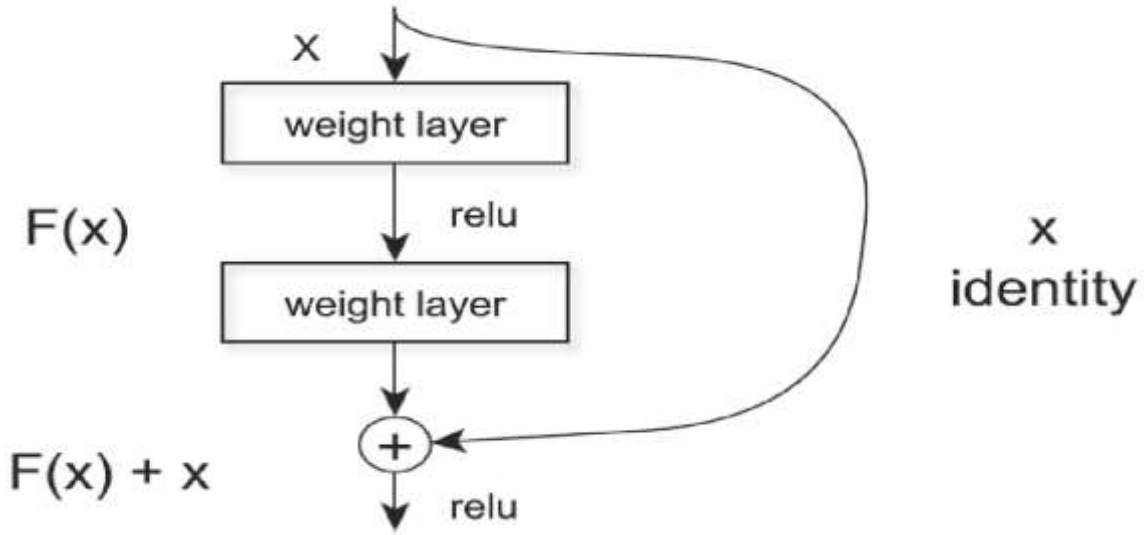


Рис 2.4. Залишковий блок із звичайним з'єднанням із пропуском.

Дослідження, що стосуються нових проектів і нових архітектурних підходів, публікуються постійно і швидко, і ми не визначили пріоритетність представлення найактуальніших проектів тут. Те, що було представлено тут, є одними з найбільш значущих підходів наступних років.

Є звичайно безліч інших цікавих конструкційякі можуть бути згадані, наприклад, конструкції з такою ж продуктивністюяк AlexNetале з 50 раз менше параметрів, конструкціями зі змінним акумулюванням шарів замінений сверточное шарами і дизайн, який використовує варіант ResNet для досягнення найсучаснішої продуктивності для певних проблем.

2.3. Висновки

У цій главі представлений загальний огляд штучних нейронних мереж (ANN) та їх значення у технологічному розвитку, натхнення системи людського мозку, їх архітектура та компоненти, навчання та тестування процес. Їх алгоритми та закони навчання, а також пояснення алгоритму навчання зворотного розповсюдження, який використовується для навчання системи стоматологічної ідентифікації в цій дипломній роботі. Стоматологічна система ідентифікації буде детально описана в наступному розділі.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Глибоке навчання

Хоча перший крок до нейронних мереж був зроблений у 1943 році з роботою Уоррена МакКалока та Уолтера Піттса, перше практичне застосування штучних нейронів з'явилося з винаходом Френка Розенблата, персептрона. Персептрон — це найпростіший варіант штучного нейрона, і він має такі основні суттєві атрибути:

• Один або кілька числових входів із відповідними вагами, позитивними чи негативними, для кожного входу.

• Упередження, яке може бути як позитивним, так і негативним.

Неформально можна описати як стійкість нейронів до «вистрілу».

• Функція активації (у випадку персептрона, функція одиничного кроку).

• Єдине вихідне значення, функція активації, застосована до суми зважених вхідних даних і зміщення.

Більш неформально зазначено, що персептрон виводить 1, якщо сума зважених вхідних даних і зміщення більше 0, і 0, якщо ні. Незважаючи на те, що персептрони не використовуються на практиці, вони привели до наступного логічного кроку, багатошарового персептрона (MLP) або нейронної мережі з прямим зв’язком.

Нейронна мережа з прямим зв’язком – це просто штучні нейрони в шарах, при цьому всі вихідні сигнали від кожного нейрона попереднього шару подаються вперед, а не назад, до кожного нейрона наступного шару, винятком є вхідний шар (що складається з пасивних нейронів, які не трансформувати вхідний) і вихідний шар. Шари між першим і останнім називаються прихованими шарами, що надає глибину мережі і, отже, веде до

першої частини назви цієї глави, глибоке навчання, яка також є загальною назвою для використання глибокої нейронної системи. мережі в цілому та пов'язані з ними методи.

Оскільки кожен прихований шар і вихідний шар складаються з нейронів, які окремо пов’язані з виходом кожного нейрона попереднього шару, ці шари в мережі з прямим зв’язком називаються повністю підключеними. шари. Усі нейрони в мережі мають унікальний набір ваг, а функції активації є нелінійними функціями. Ця остання частина буде викладена пізніше в цьому розділі.

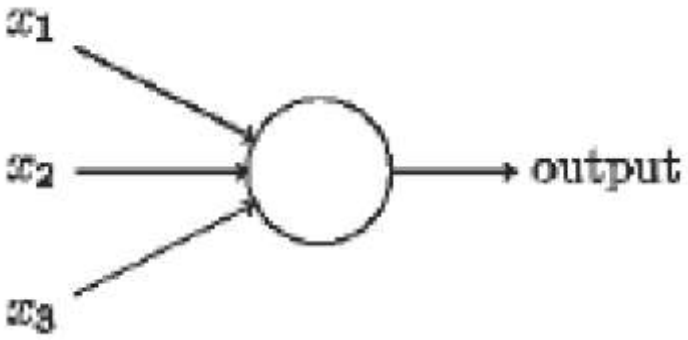


Рис 3.1. Штучний нейрон.

Перш ніж розглянути більш технічні деталі нейронних мережбільш фундаментальну із прямим зв'язком, спочатку необхідно представитивластивість нейронних мереж із прямим зв'язком; щопрямим зв’язком нейронні мережі зпрацюють як апроксиматори універсальних функцій. Один персептрон або будь-який інший штучний нейрон не дуже корисний, але мережа з принаймні одним прихованим шаром може апроксимувати будь-яку безперервну функцію, що на практиці означає, що будь-яку розривну функцію також можна апроксимувати.

Опис формального доказу виходить за рамки цієї доповіді, але концептуально штучний нейрон можна порівняти зелементами NAND або NOR, логічнимиоскільки він працює як універсальний будівельний блок.

Основна відмінність полягає в тому, що штучний нейрон має параметри, які можна налаштувати і, отже, тренувати.

3.2. Основи нейронних мереж

Говорячи про параметри, доцільно ввести функції активації, які використовуються на практиці. Функція одиничного кроку, згадана вище, не використовується на практиці через той факт, що невелика зміна вхідних даних може призвести до великої зміни на виході (від 0 до 1), що є небажаною властивістю, оскільки постійна зміна виходу є бажаною. Сигмовидна функція і гіперболічна тангенсова функція, більш гладкі версії функції одиничного кроку, використовувалися на практиці, але сьогодні функцією активації вибору є випрямлена лінійна одинична функція (ReLU), яка визначена. повертаючи лише позитивні значення та варіанти цієї функції. Вихідний рівень, або шар класифікації, використовує softmax.

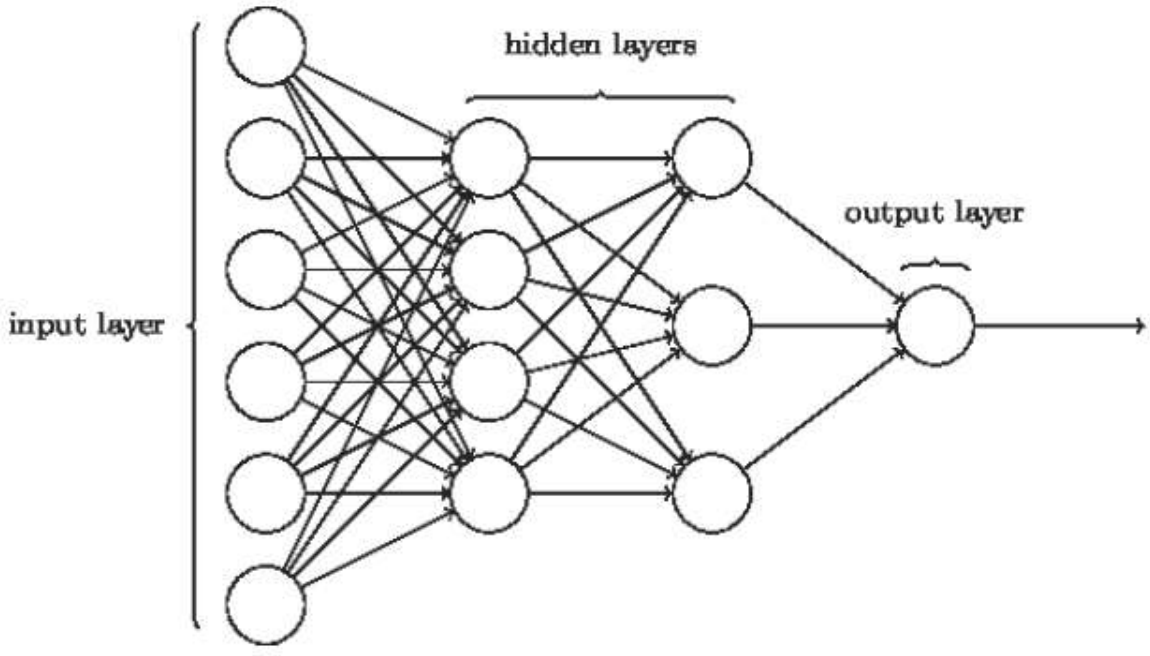


Рис 3.2. Штучна нейронна мережа

Функція, яка виводить розподіл ймовірностей за всіма категоріями. Більш неформально функція softmax виводить найбільш вірогідну категорію, класифікацію, яку мережа вважає найбільш вірогідною.

Щоб навчити нейронну мережу, нам потрібен якийсь інший тип міри того, наскільки велика поточна помилка за межами помилки навчання, якась міра того, наскільки вагові та зміщення мережі в цілому відрізняються. Для вирішення цієї проблеми вводиться функція вартості або цільова функція, яка вимірює загальну поточну помилку. Двома важливими властивостями, якітака цільова функція повинна мати, є те, що вона невід’ємна для всіх вхідних даних і що вона дорівнює нулю або близька до нуля, якщо помилка невелика. Таким чином, безпосередньою метою навчання є мінімізація цільової функції.

Простий підхід тут повиненби вибрати середню квадратичну помилкуяк вартістьщоб звестимінімуму, але на практиці функція крос ентропії замість використовується зза своєю природою більш високої продуктивності.

Нейронні мережі можуть містити мільйони, десятки мільйонів і навіть мільярди параметрів, і немає можливого способу знайти мінімум за допомогою методів зі звичайного обчислення. Натомість

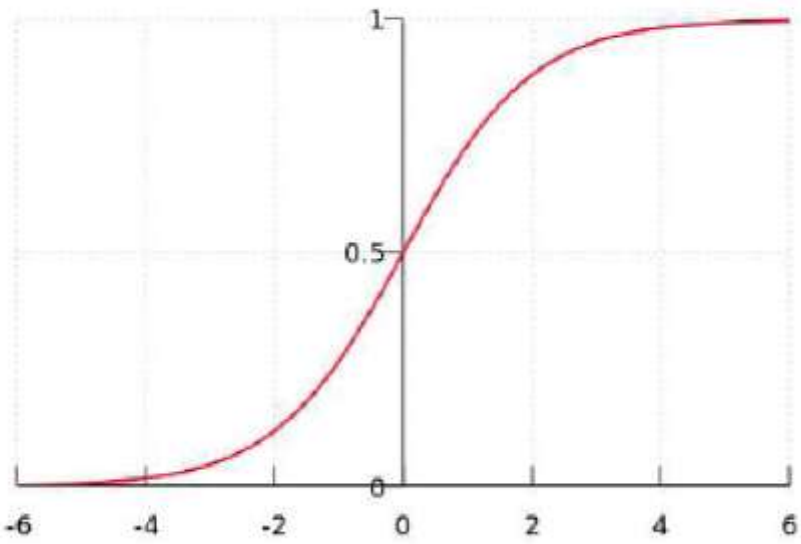


Рис. 3.3. Графік сигмовидної функції.

Незважаючи на те, що стохастичний градієнтний спуск, як описано вище, є алгоритмічним рішенням математично нерозв’язної проблеми, він все ще потребує подальшої роботи, щоб бути корисним на практиці. Знову ж таки, нейронні мережі мають величезну кількість параметрів, і обчислення кожної часткової похідної цільової функції, як у наївній реалізації вище, чисельно нездійсненно.

Рішенням проблеми, що робить стохастичний градієнтний спуск корисним на практиці, є зворотне поширення, алгоритм, який, простіше кажучи, обчислює помилку одного шару на основі помилки попереднього шару і відповідно оновлює його параметри. Назва алгоритму походить від тієї властивості, що помилка і виправлення помилки поширюються назад по всій мережі від вихідного рівня і назад. Елегантність алгоритму полягає в тому, що він відображає шлях активації в мережі і несе приблизно однакові обчислювальні витрати. Як останнє зауваження щодо градієнтного спуску, сьогодні використовуються більш просунуті варіанти, які ґрунтуються на стандартній версії із зворотним поширенням, варіанти, які додають додаткові елементи, такі як динамічна швидкість навчання та інші налаштування.

Детальне пояснення цих алгоритмів виходить за рамки цього звіту, але варто зазначити, що вони існують.

Як згадувалося в попередньому розділі, є вбудовані проблеми, пов’язані змашинного процесомнавчання, і нейронні мережі не є винятком. Дві проблеми, які будуть досліджені тут, це переобладнання у випадку нейронних мереж і проблема нестабільного градієнта, проблема, специфічна для алгоритму стохастичного градієнтного спуску, застосованого до навчання нейронних мереж. Переобладнання в цьому контексті усувається такими методами, як збір більшої кількості та кращих даних, про що згадувалося раніше в цій главі, і регуляризація. Це було згадано в попередньому розділі, але варто повторити, що регуляризацію можна описати як обмеження кількості функцій, які може генерувати алгоритм машинного навчання.

У випадку нейронних мереж існує багато видів регуляризації, але обговорення тут обмежиться такими методами: регуляризація L1, регуляризація L2, вилучення та збільшення даних.

Регуляризація L1 і L2 є двома варіантами однієї теми і побудована на додаванні додаткового терміна до функції вартості, терміну, що складається із середньозваженої суми всіх ваг у мережі. Різниця між ними полягає в тому, що сума абсолютних значень ваг у першому випадку і квадратів ваг у другому випадку. Причина, чому додається цей термін, полягає в тому, щоб покарати мережі із занадто великими вагами, покарання, ефект якого, неформально зазначений, полягає в тому, щоб зробити мережу кращою узагальненням, змушуючи її вибирати менш складні функції, що пов’язують вхід і вихід.

Dropout – це техніка, яка заснована на викиданні випадкової кількості нейронів у прихованих шарах під час кожного раунду міні-пакетів з остаточним коригуванням ваги в кінці тренувального запуску.

Передбачуваний ефект, якз L1 і L2, для забезпечення кращого узагальнення. Останній метод, даних збільшення, заснований на штучному розширенні доступних даних шляхом внесення випадкових змін у приклади, наприклад, перегортання зображення по горизонталі, зміщення його в певному напрямку або злегка повороту. Оскільки найкращим засобом проти переобладнання є більше даних, ця методика працює в цьому напрямку, щоб покращити узагальнення мережі.

Проблема нестабільного градієнта виникає через те, що градієнт, або швидкість зміни параметрів, у шарі є продуктом зміни швидкості всіх шарів перед ним. Це може призвести до повного зникнення швидкості змін, проблеми зникаючого градієнта, або різкого зростання, проблеми градієнта, що вибухає. Небажані ефекти накопичуються тим швидше, чим раніше в мережі знаходиться відповідний шар. Ця проблема, принаймні історично, була серйозною перешкодою для навчальних мереж за межами певної глибини. На щастя, здається, що проблема частково вирішена, частково через

те, що вищезгадана випрямлена лінійна функція та її варіанти стали стандартними функціями активації.

Основною причиною проблеми було використання функцій активації насичення, таких як сигмоїдна та гіперболічна тангенс, що означає насичення в цьому контексті, що означає, що швидкість зміни або похідна функції зводиться до нуля, коли вхід стає занадто великим від додатного чи від'ємного числа. На відміну від цього функція ReLU має похідну, яка дорівнює 0 або 1, що означає, що вона не насичується в своїй додатній області і завжди поширює градієнт у зворотному напрямку. Були й інші прориви в цій галузі, прориви, про які піде мова пізніше.

Під час цього викладу терміни нейронні мережі та нейронні мережі прямого зв'язку вживалися як синоніми, що не дивно, оскільки саме останні обговорювалися та пояснювались протягом більшої частини цієї глави. Однак терміни не є синонімами, оскільки існують нейронні мережі з циклами та зв'язками зворотного зв'язку. Цей підклас нейронних мереж називається рекурентними нейронними мережами і використовує більш складні шари, ніж згадані досі, такі як блоки довготривалої пам'яті (LSTM).

Детально про те, як рекурентні нейронні мережі працюють, виходить за рамки цієї доповіді, але вони важливі та варті згадки через те, що вони стоять за деякими з останніх проривів у обробці тексту та мовлення. Існують інші види нейронних мереж, але в центрі уваги цієї роботи є нейронні мережі з прямим зв'язком та їх похідні.

3.3. Висновки

Мабуть, найпопулярнішим завданням нейромереж є розпізнавання візуальних образів. Сьогодні створюються мережі, в яких машини здатні успішно розпізнавати символи на папері та банківських картках, підписи на офіційних документах, детектувати об'єкти тощо.

Ці функції дозволяють суттєво полегшити працю людини, а також підвищити надійність та точність різних робочих процесів за рахунок відсутності можливості припущення помилки через людський фактор.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Інтерфейс розробленої інтелектуальної системи

В даному розділі описано інтерфейс розробленої інтелектуальної системи, що дозволяє здійснювати класифікацію зображень клітин крові по заданих характеристиках.

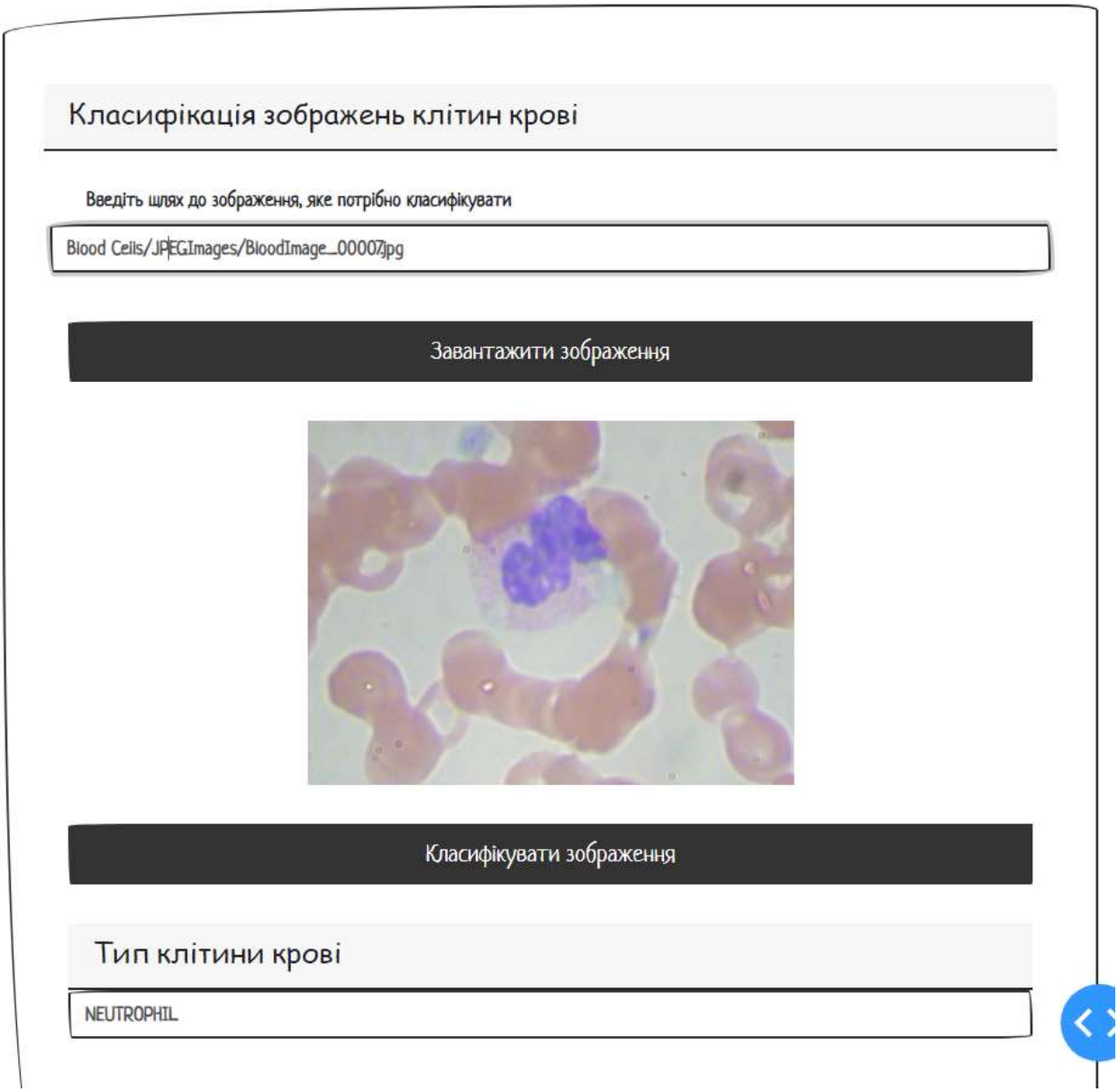


Рис. 4.1. Інтерфейс інтелектуальної системи.

На Рис. 4.1. зображено інтерфейс створеної інтелектуальної системи. За допомогою її функціоналу можна здійснити класифікацію зображення клітини крові попередньо задавши шлях до файлу, де знаходиться зображення.

Натиснувши на кнопку «Класифікувати зображення» запускається алгоритм реалізації нейронної мережі, а саме класифікації зображення клітини крові.

Результат прогнозування, виводиться у полі «Тип клітини крові»

4.2. Аналіз отриманих результатів

На рис. 4.2 зображено структуру розробленої згорткової нейронної мережі, що складається з 10 шарів:

1 шар – Conv2D шар з розміром входу 224 x 224 x 32

2 шар – MaxPooling2D шар з розміром 112 x 112 x 32

3 шар – Conv2D шар з розміром 112 x 112 x 32

4 шар – MaxPooling2D шар з розміром 56 x 56 x 32

5 шар – Conv2D шар з розміром 56 x 56 x 64

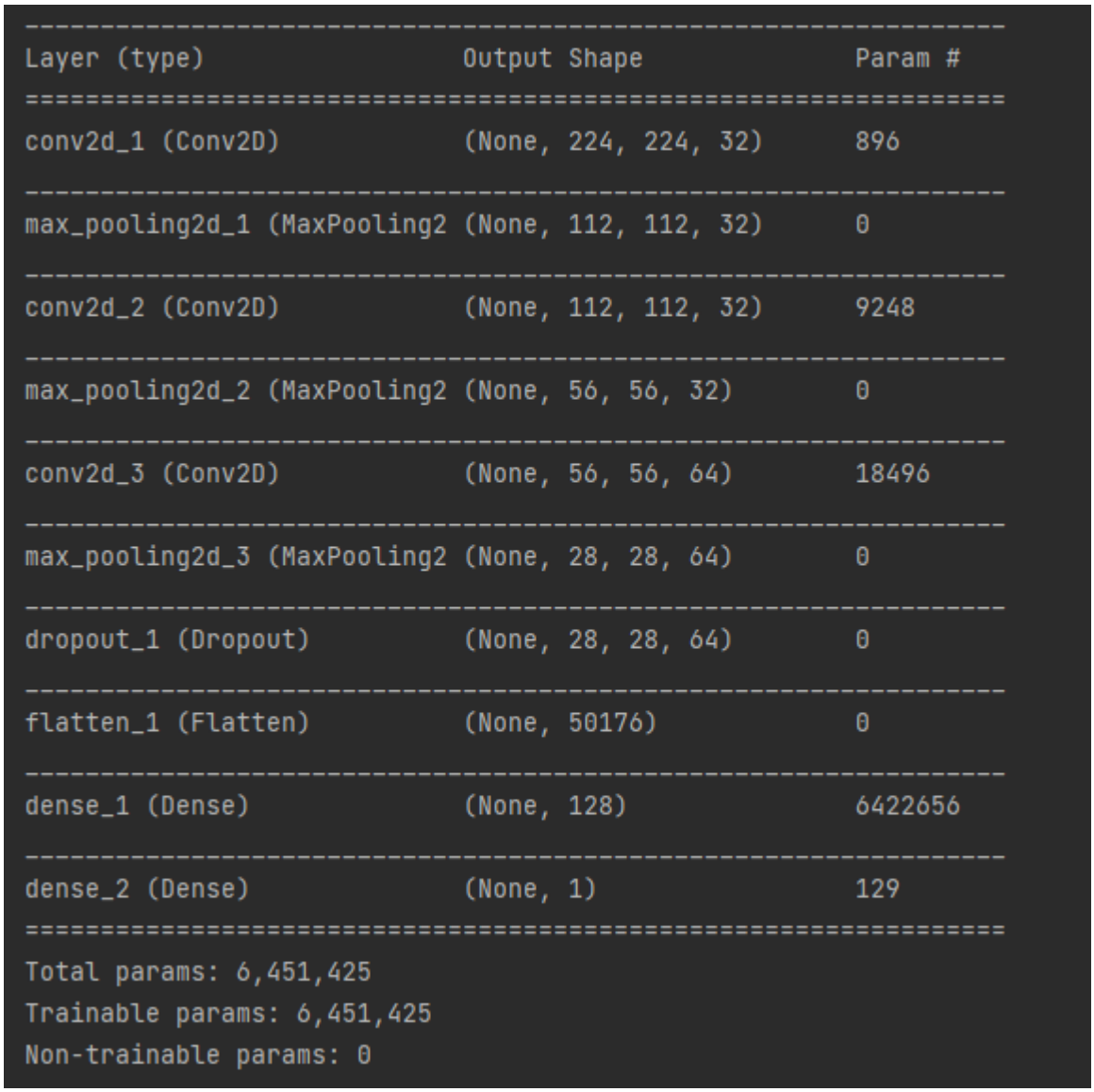
6 шар – MaxPooling2D шар з розміром 28 x 28 x 64

7 шар – DropOut шар з розміром 28 x 28 x 64

8 шар – Flatten шар з розміром 50176

9 шар – Dense шар з розміром 128

10 шар – Dense шар з розміром 1



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 224, 224, 32)      896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 112, 112, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 112, 112, 32)      9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 56, 56, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 56, 56, 64)        18496
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 28, 28, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 28, 28, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 50176)             0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               6422656
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 6,451,425
Trainable params: 6,451,425
Non-trainable params: 0
```

Рис. 4.2. Структура розробленої нейронної мережі.

На рис. 4.3. зображено інтерактивну ілюстрацію процесу навчання нейронної мережі. Для прикладу зображено тільки перші 2 епохи навчання. Як можна побачити з часу проходження епохи, то процес навчання розробленої мережі досить тривалий в часі.

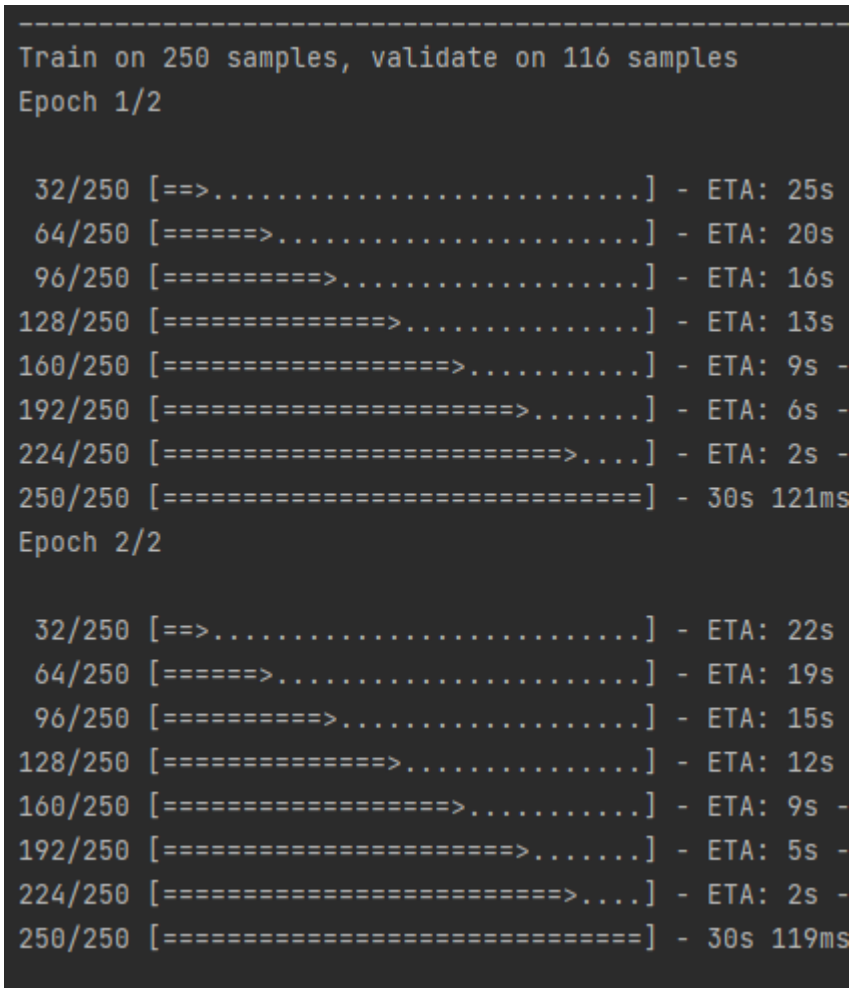


Рис. 4.3. Ілюстрація процесу навчання нейронної мережі.

На рис. 4.4- 4.6 зображе приклади типів клітин крові, згідно яких буде проводитись класифікація вхідного зображення за допомогою розробленої інтелектуальної системи.

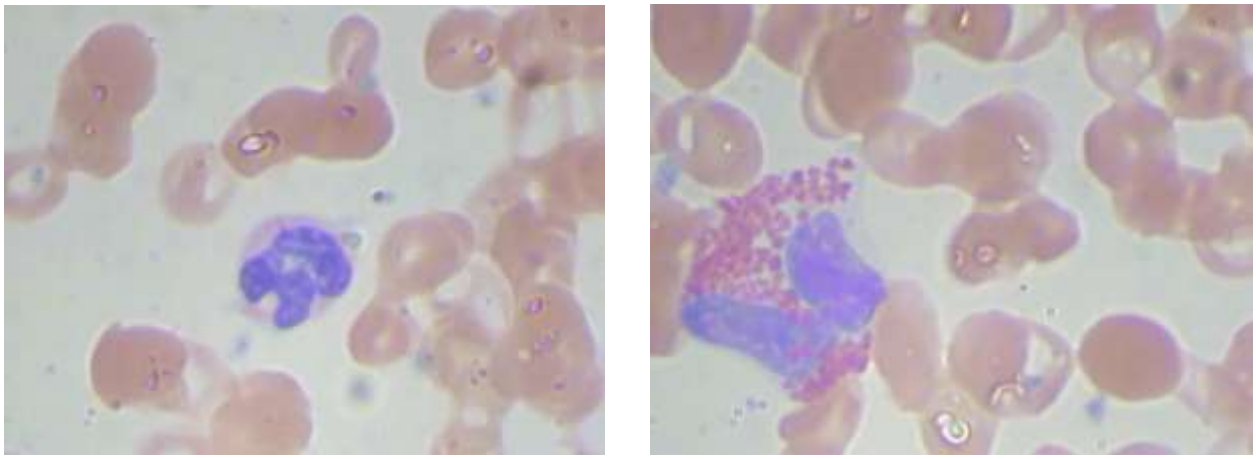


Рис. 4.4. Клітина крові типу NEUTROPHIL і EOSINOPHIL відповідно

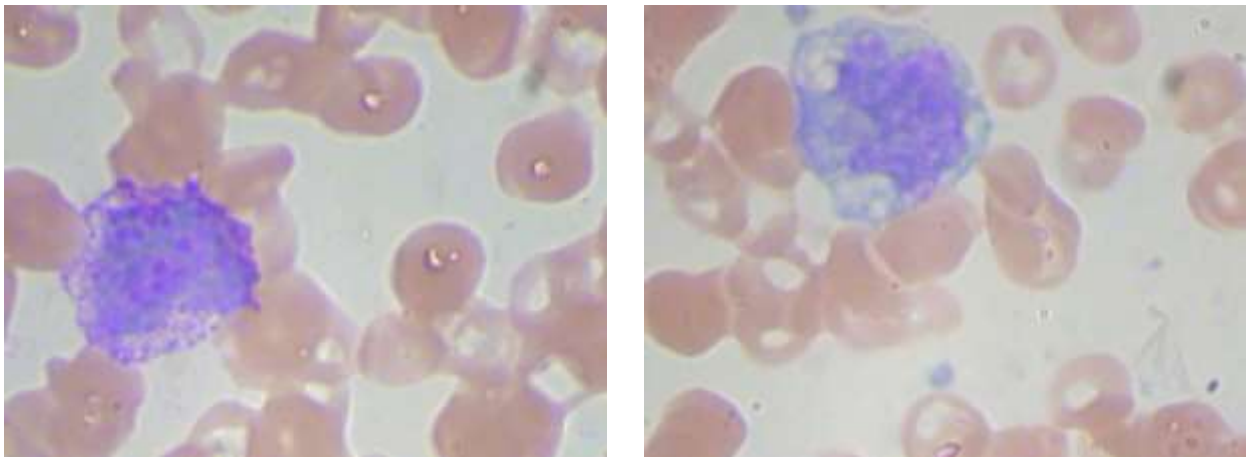


Рис. 4.5. Клітина крові типу BASOPHIL і MONOCYTE відповідно

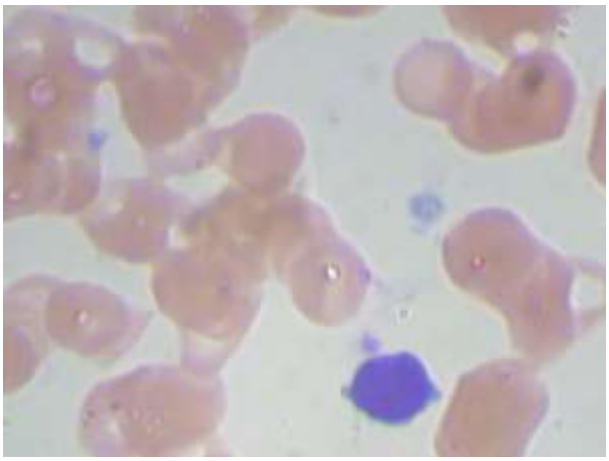


Рис. 4.6. Клітина крові типу LYMPHOCYTE

4.3. Аналіз процесу навчання нейронної мережі

Для побудови нейронної мережі було використано бібліотеку Keras, мови програмування Python. Було проаналізовано точність та втрати навчання залежно від епох та розміру навчальної вибірки.

Розглянемо отримані результати для навчальної вибірки з 366 зображень. Як видно точність при 10 епохах навчання досягається близько 0,882, а втрати складають 0,35.

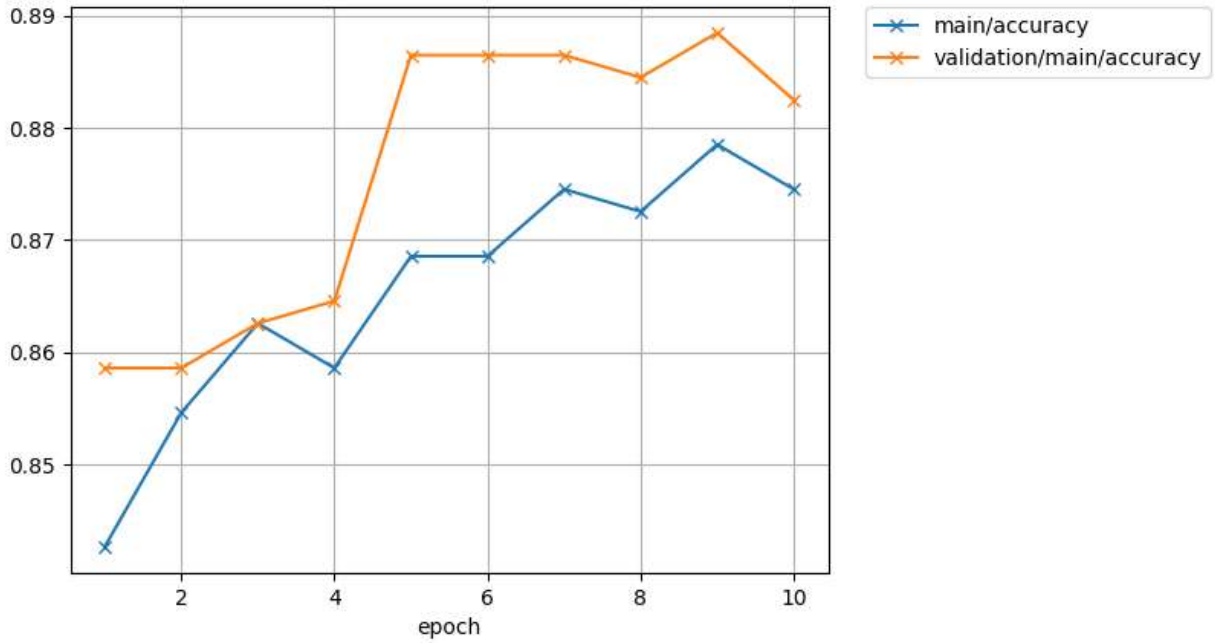


Рис. 4.5. Точність побудованої моделі нейронної мережі залежно від епох

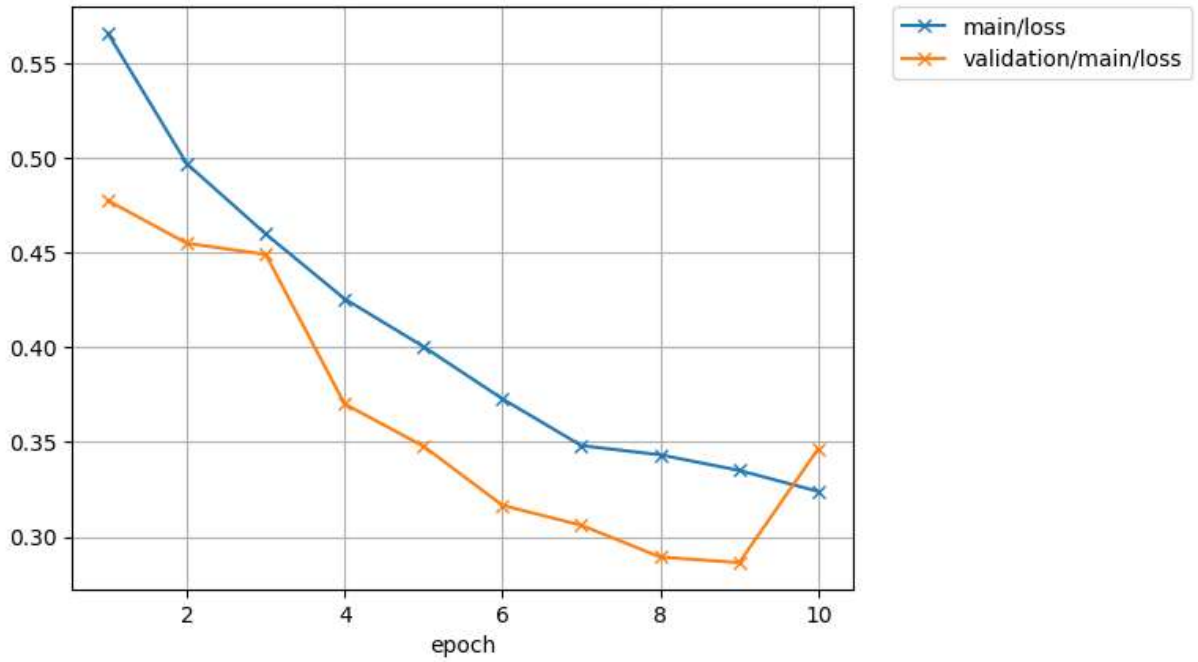


Рис. 4.6. Похибки нейронної мережі залежно від епох

Розглянемо отримані результати для навчальної вибірки з 3285 зображень. Як видно точність при 10 епохах навчання досягається близько 0,873, а втрати складають 0,32.

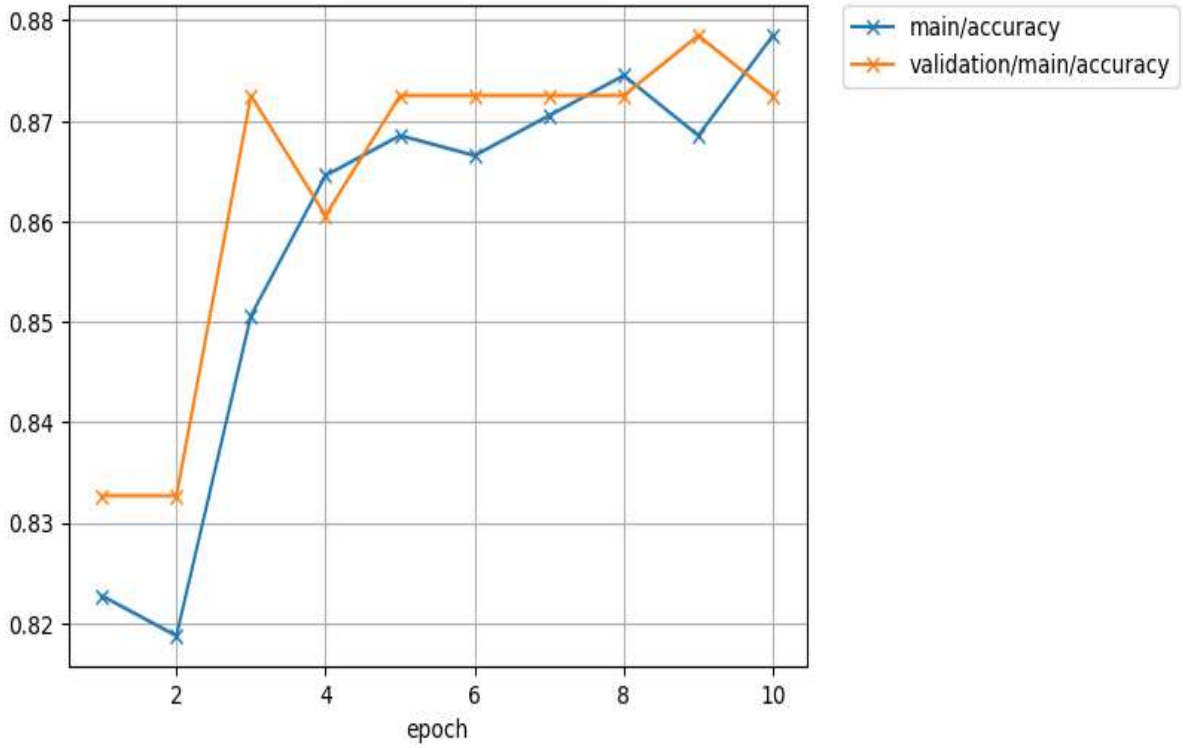


Рис. 4.5. Точність побудованої моделі нейронної мережі залежно від епох

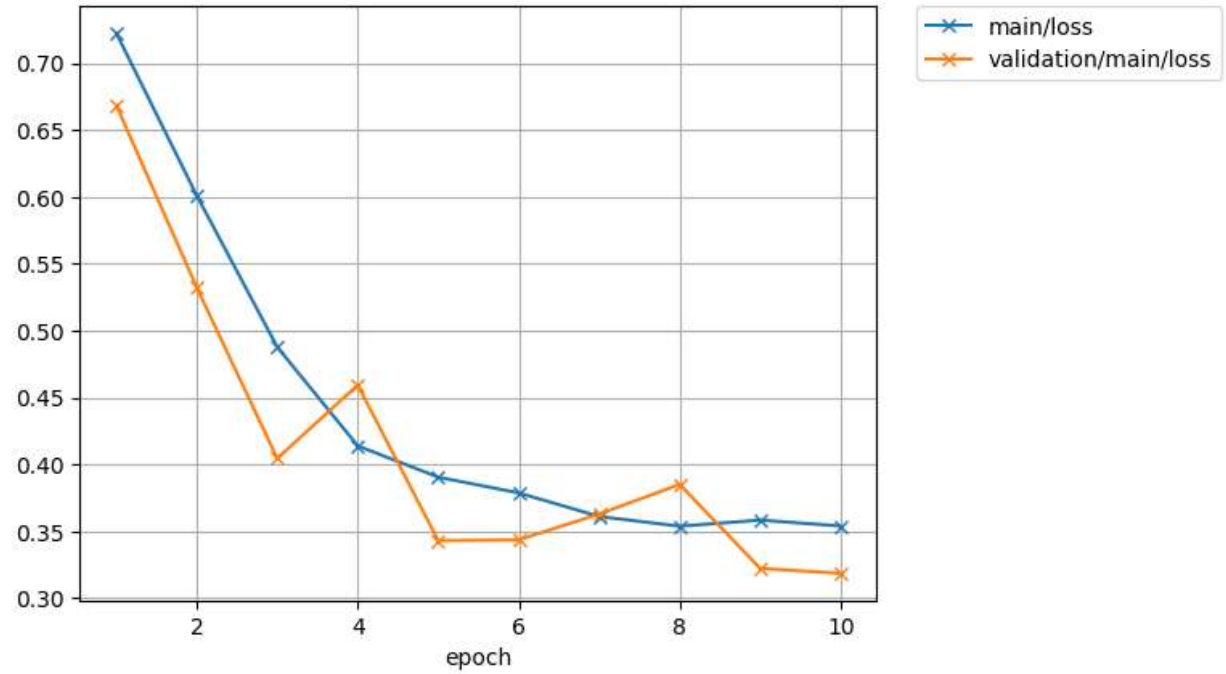


Рис. 4.6. Похибки нейронної мережі залежно від епох

Проаналізувавши отримані результати можна прийти до висновку, що із збільшенням кількості епох навчання точність нейронної мережі підвищується, як на тестовій так і на навчальній вибірці. Проте також видно,

що навчальна вибірка складена з 366 елементів забезпечує схожу точність, що і вибірка з 3285 елементів.

4.4. Висновки

Було розроблено інтелектуальну систему із зручним інтерфейсом для класифікації зображення клітини крові. Створено згорткову штучну нейронну мережу з оптимальною структурою для реалізації поставленої задачі. Дана система також підтримує можливість адаптації параметрів під нові навчальні дані.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

5.1. Опис ідеї проекту

Таблиця 5.1. Інформаційна карта проекту

Назва	Опис
Вид проекту	Інтелектуальна система
Назва проекту	«Інтелектуальна система класифікації зображень клітин крові»
Назва ВНЗ в якому розробляється проект	Національний лісотехнічний університет України, кафедра інформаційних технологій
Прізвище, ім’я, побатькові	Волянський І.М.
Цілі та задачі проекту	1. На основі бази знань, що складається із класифікованих зображень клітин крові, створити нейромережеву модель. 2. Розробити механізм емуляції нейронної мережі та класифікації зображень клітин крові. 3. Розроблена система повинна мати інтуїтивно зрозумілий інтерфейс.
Короткий зміст проекту	Проект покликаний створити інтелектуальну систему класифікації зображень клітин крові. Це дозволить автоматизувати процес ідентифікації типу клітини крові по його зображенню.
Терміни виконання проекту	6 місяців
Бюджет проекту	147 000 грн.

5.2. Передумови реалізації бізнес ідеї

Продуктові інновації є незамінними для підтримки прибутковості та конкурентоспроможності. Незважаючи на широке визнання цієї необхідності, методична і систематична процедура в підприємств майже не спостерігається. Кілька опитувань показали, що безперервне застосування методів на підприємствах має місце лише рідко. Особливо це стосується планування ранні фази інноваційного проекту Причини недостатнього застосування різноманітні.

Прикладами причин, які описані в літературі, є брак часу в продукті процес розробки або хибне розуміння методів, що призводить до невиконання очікувань користувачів.

Перед реалізацією нової бізнес-ідеї («Пуск») зазвичай надають бізнес-план. З іншого боку, в літературі існує багато фазових моделей, які описують процес інноваційний проект. Підходи бізнес-плану та фазових моделей можуть бути об’єднані на думку авторів. Далі ці два підходи коротко представлені та обговорюються їх переваги та недоліки. Згодом власний пропонується підхід.

5.3. Фазові моделі інноваційного процесу

Можуть бути різні думки щодо ідеально-типового процесу та масштабу інноваційного проекту знайти в літературі. У цьому контексті були розроблені численні фазові моделі. Ці моделі суттєво відрізняються один від одного за кількістю використовуваних фаз і рівень абстракції. Проте більшість фазових моделей мають спільне те, що вони розділяють інноваційний процес на послідовні підпроцеси. Більшість фазових моделей починаються з фази реалізації проблеми. На цьому етапі виводиться потреба в інновації - на основі загальний розвиток ринку і технологій або шанс на підприємстві. Фазові моделі переважно закінчуються впровадженням на ринок товарних інновацій.

Критична оцінка

Фазові моделі є чітким, але абстрактним уявленням про інноваційний процес. Вони пропонують а орієнтація, як можна організувати інноваційний процес. З іншого боку, фазові моделі інноваційного процесу можна критикувати через те, що практична доцільність в моделей недостатньо. Зокрема, моделі навряд чи можуть служити малого та середнього розміру підприємства як посібник з дій. Крім того, не є фактором успіху орієнтації на клієнта достатньо враховано в багатьох фазових моделях. Крім того, багато фазових моделей цього не роблять розглянути необхідність постійної оцінки та контролю інноваційного процесу.

У літературі дається багато посилань, як можна систематизувати, описати (проектні) методи. Але відсутнє опис того, як проходять фази фази моделі можуть підтримуватися (маркетинговими) методами. Ще одним моментом для критики є послідовна процедура. Інноваційний процес характеризується високим поділом праці і мультидисциплінарність. Тому багато етапів процесу проходять паралельно, а не послідовно.

Бізнес-план служить в першу чергу для документації нової бізнес-ідеї. Мета полягає в тому, щоб знайти фінансові джерела або покупців для підприємства. Перш за все, бізнес-план – це а інструмент комунікації та має переконати читача в прибутковості нової справи за допомогою а чітке і зрозуміле тлумачення. Необхідність складання бізнес-плану може відбуватися в різний час. Таким чином, потрібен план не тільки від засновників існування, а також підприємствами, які розвивають нову сферу бізнесу або які претендують на IPO.

Підготовка бізнес-плану переслідує певну мету і адресована певній людині група. Бізнес-проект - для якого написаний Бізнес-план, а мета – для що мається на увазі, тому мають вирішальний вплив на його структурування. Бізнес-план для а Початкова компанія виглядає інакше, ніж для підприємства, яке хоче розвивати нове сфера бізнесу. Оскільки він не читається в присутності автора, який міг би дати додаткові запити і пояснення, важливо чітке формулювання.

Елементи бізнес-планів

У літературі наводяться різні точки зору, які елементи повинен мати бізнес-план і які міра відповідна. Причина – мінливість підприємств і проектів. Немає загальноприйнятий план. Незважаючи на всі відмінності, бізнес-плани мають певні основні елементи, які повинні бути присутніми в кожному конкретному випадку.

5.4. Бізнес план нових продуктів

Бізнес план поділяється на стратегічний етап, етап ідентифікації продукту, реалізації продукту та супровід продукту. Це визначення є занадто розширеним для цілей планування. Натомість вихідною точкою концепції планування є вже існуюча ідея продукту. Ні стратегічні міркування, ні реалізація продукту не належать до змісту. Акцент робиться на плануванні та експертизі ідеї.

Кроками першого етапу є аналіз ринку, сегментування ринку та аналіз конкуренції. Початком аналізу ринку є систематичний збір і оцінка інформація про відповідні ринкові обставини, галузі та клієнтів.

Підприємство не повинно намагатися взяти участь у конкуренції в усіх сферах - і часто проти конкуренти з кращими умовами. Він має зосередитися на відповідному сегменті ринку. Потенційні клієнти повинні бути диференційовані на визначені категорії. Вимоги покупеця мають бути визначені для відповідних категорій.

Однак вирішальним є використання географічних, демографічних, психографічних і критерії сегментації, орієнтовані на поведінку. Аналіз конкуренції має завдання вказати на вплив конкуренції на продажі перспективи. Обсяг і глибина конкурентного аналізу визначаються доступністю потрібної інформації та витратами на надання інформації.

Відповідна інформація про конкуренцію: кількість продажів, частка ринку, цінова стратегія, зростання, структури витрат, обслуговування клієнтів, маркетинг, ланцюг збуту тощо.

Зміст другого етапу планування – конкретизація ідеї продукту та його планування реалізації. Кроками другого етапу є визначення комплексу маркетингу:

- прогноз збуту, попереднє планування виробництва, планування організації та персоналу, фінанс
- планування та визначення цільових показників для цільової калькуляції.

Відправною точкою є планування інструментів маркетингу: продукту, ціни, розміщення та просування (Marketing Mix).

Прогноз продажів може бути проведений кількісно і якісно. Кількісний прогноз не підходить для малих і середніх підприємств через високі витрати. А можливість якісного прогнозу - це оцінка (зовнішніми) експертами. Прогноз має охоплювати позитивний, оптимістичний і песимістичний випадок.

Наступні два кроки охоплюють приблизне планування виробництва та організаційне планування, яке тут більше не обговорюються. На основі результатів попередніх кроків планування тепер можна отримати дефолтні витрати для наступного дизайну продукту. Відповідний метод являє собою цільову калькуляцію витрат. Перший

Крок цільового калькулювання - це визначення загальних цільових витрат на продукт. Другий крок, цільові витрати розподіляються на компоненти продукту за рахунок їх внесок для виконання вигоди клієнта.

Після визначення та кількісної оцінки всіх відповідних блоків витрат можна надати фінансовий план. Фінансовий план має на меті оцінити потребу в капіталі, яка випливає з нового продукт на короткострокову та довгострокову перспективу. До фінансової належить також інвестиційна оцінка план. Метою оцінки інвестицій є оцінка очікуваного грошового потоку викликаний інноваційним проектом виправдовує необхідні інвестиції.

Після останнього кроку планування підприємство може прийняти рішення про реалізацію продукту розвиток.

5.5. Висновки

Одним з підходів до процесу планування є концепція бізнес-плану, оскільки вона використовується для підвищення капіталу для новостворених компаній або розширення існуючого бізнесу. Однак, різне призначення спільного бізнес-плану обмежує його застосування для підтримки малих та середніх компаній під час процесу планування ідеї нового продукту.

Концепція планування проводить компанію через два етапи планування. На першому етапі акцент плануючої діяльності робиться на визначення факторів, що впливають на успіх проекту, і порівняння їх фактори з можливостями та ресурсами компанії. Для цього відповідний сегмент ринку ідентифікується та визначається попит цього сегмента на продукт.

ВИСНОВКИ

У цій роботі було розглянуто деякі сучасні методи розпізнавання образів та аналізу зображень. Було проведено огляд основних груп ознак, що використовуються під час вирішення задачі розпізнавання зображень та описані деякі методики вибору ознак. Розглянуто методи класифікації ознак у процесі розв'язання задач розпізнавання.

Проведений аналіз існуючих підходів до розпізнавання та обробки зображень дозволив виявити сильні та слабкі сторони цих підходів. Проаналізовано існуючі види нейронних меж, що використовуються для вирішення задач класифікації зображень. Побудована згорткова нейронна мережа, що дозволяє реалізувати ситему класифікації зображень клітин крові. Досліджено швидкодію реалізації алгоритмів навчання. Проаналізовано отримані результати.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Волянський І. Інтелектуальна система класифікації зображень клітин крові / І. Волянський, Б. Пришляк, В. Шиманський // Матеріали третьої науково-практичної конференції студентів, аспірантів та молодих вчених (Львів, 14-16 жовтня 2021 р.). – Львів: кафедра інформаційних технологій НЛТУ України, 2021. – с. 24-26.
2. Воронцов К.В. Математические методы обучения по прецедентам (теория обучения машин). [Электронный ресурс] URL: http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf.
3. Гонсалес Р., Вудс Р. Цифровая обработка изображений: Пер. с англ. — Техносфера, 2005
4. Boiman O., Shechtman E., Irani M. In Defense of Nearest-Neighbor Based Image Classification. // IEEE, 2008. – No. 1.-P. 1–8.
5. Breiman L. Random forests. // Machine Learning, 2001.-No. 1 (45). - P. 5–32
6. Dalal N., Triggs B. Histograms of Oriented Gradients for Human Detection // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), 2005. - P. 886–893.
7. Deng, L. Deep Learning: Methods and Applications.// Foundations and Trends in Signal Processing, 2014.- Vol. 7 (3–4). - P. 197–387.
8. Hinton G.E., Srivastava N., Swersky K. Lecture 6a- overview of mini-batch gradient descent. // COURSERA: Neural Networks for Machine Learning, 2012. - P. 31.
9. Kostenetskiy P.S., Safonov A.Y. SUSU Supercomputer Resources. // Proceedings of the 10th Annual International Scientific Conference on Parallel Computing Technologies (PCT 2016). Arkhangelsk, Russia, March 29- 31, 2016. CEUR Workshop Proceedings, 2016. - V. 1576. - P. 561-573.

10. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet Classification with Deep Convolutional Neural Networks. // Advances In Neural Information Processing Systems, 2012. – P. 1–9.

11. McCulloch W.S., Pitts W. A logical calculus of the ideas immanent in nervous activity. // The Bulletin of Mathematical Biophysics, 1943. – No. 4 (5). – P. 115–133.

12. Mitchell T.M. Decision Tree Learning. // Machine Learning, 1997. – P 52–80.

13. Sokolova M., Lapalme G. A systematic analysis of performance measures for classification tasks. // Information Processing and Management, 2009. – No. 4 (45). – P. 427–437.

14. Srivastava N. et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. // Journal of Machine Learning Research, 2014. Vol. 15. – P. 1929–1958.

ДОДАТОК А

```
import dash
import dash_core_components as dcc
import dash_html_components as html
import numpy as np
from dash.dependencies import Input, Output, State
import matplotlib.pyplot as plt
from keras.models import Sequential
from keras.layers import Dense, Conv2D , MaxPool2D , Flatten , Dropout
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam
import tensorflow as tf
import cv2
import os
import base64


labels = ['NEUTROPHIL', 'EOSINOPHIL', 'LYMPHOCYTE', 'BASOPHIL', ' MONOCYTE ']
img_size = 224

image_filename = '/Blood Ceils/JPEGImages/BloodImage_00007.jpg' # replace with your own image
encoded_image = base64.b64encode(open(image_filename, 'rb').read())

def get_data(data_dir):
    data = []
    for img in os.listdir(data_dir):
            try:
                img_arr = plt.imread(os.path.join(data_dir, img), )[..., ::-1] # convert BGR to RGB format
                resized_arr = cv2.resize(img_arr, (img_size, img_size)) # Reshaping images to preferred size
                data.append([resized_arr, labels[index]])
            except Exception as e:
                print(e)
    return np.array(data)


def predict_result(x_to_pred):
    data = get_data("/Blood Ceils/JPEGImages/")

    train = data[0:250]
    val = data[250:]
    l = [i[1] for i in data]

    x_train = []
    y_train = []
    x_val = []
    y_val = []

    for feature, label in train:
```

```
    x_train.append(feature)
    y_train.append(label)

for feature, label in val:
    x_val.append(feature)
    y_val.append(label)

# Normalize the data
x_train = np.array(x_train) / 255
x_val = np.array(x_val) / 255

x_train.reshape(-1, img_size, img_size, 1)
y_train = np.array(y_train)

x_val.reshape(-1, img_size, img_size, 1)
y_val = np.array(y_val)

datagen = ImageDataGenerator(
    featurewise_center=False,  # set input mean to 0 over the dataset
    samplewise_center=False,  # set each sample mean to 0
    featurewise_std_normalization=False,  # divide inputs by std of the dataset
    samplewise_std_normalization=False,  # divide each input by its std
    zca_whitening=False,  # apply ZCA whitening
    rotation_range=30,  # randomly rotate images in the range (degrees, 0 to 180)
    zoom_range=0.2,  # Randomly zoom image
    width_shift_range=0.1,  # randomly shift images horizontally (fraction of total width)
    height_shift_range=0.1,  # randomly shift images vertically (fraction of total height)
    horizontal_flip=True,  # randomly flip images
    vertical_flip=False)  # randomly flip images

datagen.fit(x_train)

model = Sequential()
model.add(Conv2D(32, 3, padding="same", activation="relu", input_shape=(224, 224, 3)))
model.add(MaxPool2D())

model.add(Conv2D(32, 3, padding="same", activation="relu"))
model.add(MaxPool2D())

model.add(Conv2D(64, 3, padding="same", activation="relu"))
model.add(MaxPool2D())
model.add(Dropout(0.4))

model.add(Flatten())
model.add(Dense(128, activation="relu"))
model.add(Dense(1, activation="softmax"))

model.summary()
epoch_count = 2
opt = Adam(lr=0.000001)
model.compile(optimizer=opt, loss=tf.keras.losses.sparse_categorical_crossentropy,
              metrics=['accuracy'])
```

```
    history = model.fit(x_train, y_train, epochs=epoch_count, validation_data=(x_val, y_val))

    loss = history.history['loss']
    val_loss = history.history['val_loss']

    epochs_range = range(epoch_count)

    plt.figure(figsize=(15, 15))
    plt.plot(epochs_range, loss, label='Training Loss')
    plt.plot(epochs_range, val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.title('Training and Validation Loss')
    plt.show()

    return model.predict(x_to_pred)

app = dash.Dash()
app.title = 'Класифікація зображень клітин крові'

app.layout = html.Div([
    html.Div([
        html.Div([
            html.Div([
                html.H4('Класифікація зображень клітин крові',
                        className='my-0 font-weight-normal')
            ], className="card-header"),

            html.Div([
                html.Div([
                    html.Label('Введіть шлях до зображення, яке потрібно класифікувати',
classNameName='col-sm-6 col-form-label'),
                    html.Div([
                        dcc.Input(id='path', placeholder='Введіть сезон', type='text',
                                  className='form-control')
                    ], className='form-group row')
                ]),

                html.Div([
                    html.Label('', className='col-6 col-form-label'),
                ]),

                html.Button('Завантажити зображення', id='submit_image',
                            className='btn btn-primary btn-lg btn-block'),

                html.Div([
                    html.Label('', className='col-sm-6 col-form-label'),
                ], className='form-group row'),

                html.Div([
                    html.Img(src='{}'.format(encoded_image.decode()),
                        height=300)
                ], style={'textAlign': 'center'}),
```

```
            html.Div([
                html.Label('', className='col-6 col-form-label'),
            ], className='form-group row'),

            html.Button('Класифікувати зображення', id='submit_clasification',
                        className='btn btn-primary btn-lg btn-block'),

            html.Div([
                html.Label('', className='col-sm-6 col-form-label'),
            ], className='form-group row'),

            html.Div([
                html.H4('Тип клітини крові', className='my-0 font-weight-normal'),
            ], className="card-header"),
            html.Div(className='form-control', id='target'),

        ], className="card-body")

    ], className="jumbotron")
], className='jumbotron')

], className='jumbotron')

inputs = [Input('submit_image', 'n_clicks'), Input('submit_clasification', 'n_clicks')]

states = [State('path', 'value'),
          ]

@app.callback(Output('target', 'children'), inputs, states)
def home_graph(n_clicks, *args):
    if n_clicks:
        try:
            input_names = [item.component_id for item in states]
            kwargs_dict = dict(zip(input_names, args))
            path = kwargs_dict['path']

            predict = predict_result(path)

            return f'${format(predict, "8,d")}'
        except:
            return 'Введіть коректні значення'

    else:
        return ''

if __name__ == '__main__':
    app.run_server(debug=True, port=8060)
```