КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра теорії та технології програмування

Кваліфікаційна робота

на здобуття ступеня бакалавра
за спеціальністю 122 Комп'ютерні науки

на тему:

Розробка та вдосконалення методів Data Science та Machine Learning

Виконав студент 4-го курсу
Юркевич Богдан Михайлович _____
(підпис)

Науковий керівник:
доцент, кандидат фіз-мат наук
Панченко Тарас Володимирович _____
(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«____» _____ 202_ р.,
протокол № ___
Завідувач кафедри
М. С. Нікітченко _____
(підпис)

Київ – 2021

РЕФЕРАТ

Обсяг роботи 44 сторінки, 11 ілюстрацій, 2 таблиці, 23 джерела посилань.

РОЗМИТТЯ ЗОБРАЖЕНЬ, ФІЛЬТР ГАУССА, МЕДІАННИЙ ФІЛЬТР, ДВОСТОРОННЄ РОЗМИТТЯ, РОЗМИТТЯ РУХОМ, УСУНЕННЯ РОЗМИТТЯ ЗОБРАЖЕНЬ, МЕТОД РІЧАРДСОНА-ЛЮСІ, ФУНКЦІЯ ЗГОРТКИ, НЕЙРОННА МЕРЕЖА.

Об’єктом роботи є методи розпізнавання розмиття та усунення розмиття зображень. Предметом роботи є аналіз та порівняння алгоритмів усунення розмиття зображень.

Метою кваліфікаційної роботи є розробка ефективних алгоритмів для розпізнавання та усунення розмиття зображень.

Методи розробки: для розробки програмної реалізації було використано мову програмування Python[1]. Це значно сприяє ефективності та розроблених додатків та мова потужні бібліотеки, такі як Numpy[2], SciPy[3] та інші.

Результат роботи: було детально розглянуто методи та моделі розмитих зображень, їх специфічні особливості, основні труднощі при розв’язанні задач розпізнавання та усунення розмивання зображень. Був виконаний загальний огляд методів розпізнавання розмиття зображень. Детально були розглянуті такі методи та алгоритми Data Science як метод опорних векторів, дерево прийняття рішень та логістична регресія. Розроблена програма для навчання моделі розпізнавання розмитих зображень та була протестована на ряді даних.

Зміст

ВСТУП

Оцінка сучасного стану об’єкта розробки. Зображення – це те з чим ми стикаємося кожного дня. Це зумовлено перш за тим що ми живемо у світі діджиталізації – переходу до зберігання інформації у цифровому вигляді, в основному це забезпечується активним розвитком ЕОМ, телефонії, тощо.

Основне призначення зображень це передавання інформації через наші когнітивні здібності, а зокрема зір. Водночас є багато проблем які пов’язані з розпізнаванням, інтерпретацією, розмиттям та усуненням розмиття зображень.

На сьогодні, одна з головних проблем пов’язаних з цією областю це розпізнавання розмиття та усунення його. Існують певні алгоритми для вирішення цих задач, зокрема методи Data Science, але дані задачі все ще не можуть бути ефективно і якісно вирішені за допомогою них.

Актуальність роботи та підстави для її виконання. Оскільки зображення у різних форматах зустрічаються нам кожного дня, вони є невід’ємною частиною нашого життя. А отже задачі розпізнавання та усунення розмиття є актуальними.

Аналізуючи і використовуючи наявні алгоритми та методи для вирішення даних задач можна прийти до висновку що їх результати не завжди є задовільними тому виникає потреба в удосконаленні та створенні нових алгоритмів розпізнавання та усунення розмиття зображень.

Мета й завдання роботи. Метою кваліфікаційної роботи є розробка ефективних алгоритмів для розпізнавання та усунення розмиття зображень.
Для досягнення цієї мети поставлено такі завдання:

- Теоретично дослідити та проаналізувати існуючі алгоритми розпізнавання розмитих зображень.
- Дослідити та проаналізувати існуючі алгоритми усунення розмитості зображень.
- Порівняти результати різних методів вирішення поставлених задач на прикладах.
- Реалізувати власний алгоритм усунення розмиття зображень на основі існуючих.

Об'єкт, методи й засоби розроблення. Об’єктом роботи є методи розпізнавання розмиття та усунення розмиття зображень. Предметом роботи є аналіз та порівняння алгоритмів усунення розмиття зображень.

Для розробки програмної реалізації було використано мову програмування Python[1]. Ця мова надає великі можливості, пришвидшує час розробки та є інтерпретованою, що значно сприяє ефективності та розроблених додатків та має потужні бібліотеки, такі як Numpy[2], SciPy[3], matplotlib[4], IPython[5] та інші.

Під час розробки кінцевого продукту використовувалися інтегровані середовище розробки PyCharm[6], IDLE[7] які дозволяють швидко та якісно створювати крос-платформні додатки мовою Python. Також, для відслідковування змін, під час розробки програми активно використовувалася система контролю версій git[8].

РОЗДІЛ 1. РОЗПІЗНАВАННЯ РОЗМИТИХ ЗОБРАЖЕНЬ

1.1 Означення розмитого зображення

Можна сказати що розмиття зображення робить зображення менш різким. Це можна зробити, згладивши перехід кольору між пікселями.

Для досягнення цієї мети нам потрібно застосувати до матриці зображення операцію згортки спеціалізованої матриці, яка називається ядром.

Математично кажучи, згортка двох матриць, A розміром m x n, і B розміром p x q, є матрицею (m + p -1) x (n + q-1) із записами:

$$C_{rs} = \sum_{i+k=r+1, j+l=s+1} A_{ij}B_{kl}, r = 1, \dots, m+p-1, s = 1, \dots., n+q-1$$

Простіше кажучи, згортка лише утворює нову матрицю, в якій записи - це суми добутку записів однієї матриці з відповідним записом іншої матриці. Усі ці добутки можна розрахувати по рядках і стовпцях.

Ядро - це матриця, яка має на меті перетворити зображення. Це не виключно розмиття зображення. Він також може використовуватися для виявлення країв, загострення країв та інших видів трансформації зображення. Ядро, яке використовується для розмиття зображення, є ядром фільтра низьких частот. Це дозволяє низькій частоті входити і зупиняти більш високу частоту.

Чому нам взагалі потрібно розмивати зображення? Зрештою, чи не робить це зображення менш помітним? Виявляється, розмиття зображення має кілька цілей.

По-перше, це зменшує шуми на зображенні. Випадкова пляма яскравості або неправильна кольорова пляма, залежно від типу шуму, може бути зменшена розмиттям зображення відповідним типом розмиття.

Розмиття зображення також зменшує розмір зображення. За допомогою відповідної функції розмиття ми можемо розмити розмите зображення на вихідному. Це може бути дуже корисним для передачі величезного розміру зображень.

Розмиття також використовується в засобах масової інформації. Наприклад, коли зображення новин невідповідне або явне. Інше використання полягає у приховуванні обличчя, імені та всіх приватних даних людей, які випадково потрапляють на зображення.

Нарешті, з метою розваги. Наприклад, у фільмах та цифрових творах мистецтва. Ефект розмиття може покращити відчуття чудової сцени сітілайтів. Або це може допомогти глядачам фільмів знати, що ця конкретна сцена відбулася в минулому.

Існує кілька типів фільтрів розмиття, які можна використовувати. Все має свої особливості.

1.1.1 Просторовий фільтр

Просторовий фільтр - це оптичний пристрій, який використовує принципи оптики Фур'є для зміни структури променя світла або іншого електромагнітного випромінювання, як правило, когерентного лазерного світла. Просторова фільтрація зазвичай використовується для "очищення" виходу лазерів, усунення аберацій в пучку через недосконалу, брудну або пошкоджену оптику або через зміни в самому середовищі лазерного посилення. Ця фільтрація може бути застосована для передачі чистого

поперечного режиму від багатомодового лазера при блокуванні інших режимів, випромінюваних оптичним резонатором. [1] [2] Термін "фільтрація" вказує на те, що бажані структурні ознаки вихідного джерела проходять через фільтр, тоді як небажані характеристики блокуються. Пристрій, що слідує за фільтром, ефективно бачить якісне, але менш потужне зображення джерела, замість фактичного джерела безпосередньо. Приклад використання просторового фільтра можна побачити в розширених налаштуваннях мікрораманівської спектроскопії.

При просторовій фільтрації для фокусування променя використовується лінза. Через дифракцію промінь, який не є ідеальною плоскою хвилею, не буде фокусуватись на одній плямі, а навпаки, буде створювати малюнок світлих і темних областей у фокальній площині. Наприклад, недосконалий промінь може утворити яскраву пляму, оточену низкою концентричних кілець, як показано на малюнку праворуч. Можна показати, що ця двовимірна картина є двовимірним перетворенням Фур'є вихідного поперечного розподілу інтенсивності пучка. У цьому контексті фокальну площину часто називають площиною перетворення. Світло в самому центрі схеми перетворення відповідає ідеальній, широкій площині хвилі. Інше світло відповідає "структурі" в промені, а світло далі від центральної плями відповідає структурі з більш високою просторовою частотою. Візерунок з дуже дрібними деталями буде створювати світло дуже далеко від центральної точки площини трансформації. У наведеному вище прикладі велике центральне пляма та кільця світла, що його оточують, обумовлені структурою, яка виникає при

проходженні пучка через круглий отвір. Пляма збільшено, оскільки промінь обмежений отвором до кінцевого розміру, а кільця відносяться до гострих країв променя, створених краями отвору. Цей візерунок називається Ері, за іменем його першовідкривача Джорджа Ейрі.

Змінюючи розподіл світла в площині перетворення та використовуючи іншу лінзу для реформування колімітованого променя, можна змінити структуру променя. Найпоширеніший спосіб зробити це - розмістити в промені отвір, що пропускає бажане світло, одночасно блокуючи світло, яке відповідає небажаній структурі в промені. Зокрема, невеликий круглий отвір або "отвір", який проходить лише через центральну яскраву пляму, може видалити з балки майже всю дрібну структуру, створюючи плавний поперечний профіль інтенсивності, який може бути майже ідеальним гауссовим пучком. При хорошій оптиці та дуже маленькій отворі можна навіть наблизити плоску хвилю.

На практиці діаметр апертури вибирається на основі фокусної відстані лінзи, діаметра та якості вхідного променя та довжини його хвилі (довші хвилі вимагають більших апертур). Якщо отвір занадто малий, якість променя значно покращується, але потужність значно зменшується. Якщо отвір занадто великий, якість променя може не покращитися настільки, наскільки це потрібно.

Розмір отвору, який можна використовувати, також залежить від розміру та якості оптики. Для використання дуже маленької отвору необхідно використовувати фокусуючу лінзу з низьким числом f, і в ідеалі лінза не повинна додавати значних аберацій до променя. Дизайн такої лінзи стає все складнішим із зменшенням числа f.

На практиці найбільш часто використовувана конфігурація полягає у використанні об'єктива мікроскопа для фокусування променя та діафрагми, зробленої пробиванням невеликого, точного отвору у шматку товстої металевої фольги.

1.1.2 Фільтр Гаусса

В електроніці та обробці сигналів гауссівський фільтр - це фільтр, імпульсна характеристика якого є функцією Гауса (або наближенням до неї, оскільки справжня реакція Гауса фізично нездійсненна, оскільки має нескінченну підтримку). Гауссові фільтри мають властивості не перевищувати ступінчасту функцію функції, мінімізуючи час зростання і падіння. Така поведінка тісно пов'язана з тим, що гауссівський фільтр має мінімально можливу групову затримку. Він вважається ідеальним фільтром часової області, так само, як sinc фільтр є ідеальним фільтром частотної області. Ці властивості важливі в таких областях, як осцилографи та цифрові телекомунікаційні системи.

Математично, гауссівський фільтр модифікує вхідний сигнал за допомогою згортки з функцією Гауса; це перетворення також відоме як перетворення Вейєрштрасса.

Гауссова функція визначена на:

$$x \in (-\infty, \infty)$$

і теоретично вимагає нескінченної довжини вікна. Однак, оскільки воно швидко розпадається, часто доцільно скоротити вікно фільтра та застосувати фільтр безпосередньо для вузьких вікон, фактично використовуючи просту функцію прямокутного вікна. В інших випадках усічення може спричинити

значні помилки. Кращих результатів можна досягти, замість використання іншої функції вікна; докладніше див. реалізацію масштабного простору.

Фільтрація передбачає згортку. Функція фільтра називається ядром інтегрального перетворення. Ядро Гаусса є безперервним. Найчастіше дискретним еквівалентом є відібране ядро Гауса, яке отримується за допомогою точок відбору з неперервного гаусова. Альтернативним методом є використання дискретного ядра Гауса, яке має чудові характеристики для деяких цілей. На відміну від дискретизованого ядра Гауса, дискретне ядро Гауса є рішенням рівняння дискретної дифузії.

Оскільки перетворення Фур'є функції Гаусса дає функцію Гаусса, сигнал (бажано після поділу на перекриваються віконні блоки) може бути перетворений за допомогою швидкого перетворення Фур'є, помноженого на функцію Гауса і перетвореного назад. Це стандартна процедура застосування довільного фільтру кінцевих імпульсних характеристик, з тією лише різницею, що перетворення Фур'є вікна фільтра явно відомо.

Завдяки центральній граничній теоремі, Гауса можна апроксимувати кількома прогонами дуже простого фільтра, такого як ковзне середнє. Проста ковзаюча середня відповідає згортці з постійним B-сплайном (прямокутний імпульс), і, наприклад, чотири ітерації ковзного середнього дають кубічний B-сплайн як вікно фільтра, що досить добре наближає Гауса. Ковзне середнє є досить дешевим для обчислення, тому рівні можна каскадувати досить легко.

У дискретному випадку стандартні відхилення пов'язані між собою

$$\sigma \cdot \sigma_f = \frac{N}{2\pi}$$

де стандартні відхилення виражаються в кількості зразків, а N - загальна кількість зразків. Позичаючи терміни зі статистики, стандартне відхилення

фільтра можна трактувати як міру його розміру. Частота відсікання гауссового фільтра може бути визначена стандартним відхиленням в частотній області, що дає

$$f_c = \sigma_f = \frac{1}{2\pi\sigma}$$

де всі величини виражаються у своїх фізичних одиницях. Якщо

$$f_c = \frac{F_s}{2\pi\sigma}$$

вимірюється у зразках, частоту відсікання (у фізичних одиницях) можна обчислити за допомогою

$$f_c = \frac{F_s}{2\pi\sigma}$$

де F - частота дискретизації. Значення відгуку гауссового фільтра на цій граничній частоті дорівнює exp (-0,5) ≈0,607.

Однак частіше визначають частоту відсічення як половину потужності: де відгук фільтра зменшується до 0,5 (-3 дБ) в спектрі потужності або до 1 / √2 ≈ 0,707 в амплітудному спектрі (наприклад, фільтр Баттерворта). Для довільного значення відсікання 1 / c для відгуку фільтра частота відсікання визначається як

$$f_c = \sqrt{2\ln(c)} \cdot \sigma_f$$

Для c = 2 константа до стандартного відхилення в частотній області в останньому рівнянні дорівнює приблизно 1,1774, що становить половину повної ширини з половиною максимуму (FWHM) (див. Функцію Гауса). Для c = √2 ця константа дорівнює приблизно 0,8326. Ці значення досить близькі до 1.

Проста ковзна середня відповідає рівномірному розподілу ймовірностей, і, отже, її ширина фільтра за розміром n має стандартне відхилення

$$\sqrt{(n^2-1)/12}$$

.

Таким чином, застосування послідовних m ковзних середніх із розмірами $n_1 \ldots n_m$ дають стандартне відхилення

$$\sigma = \sqrt{\frac{n_1^2 + \cdots + n_m^2 - m}{12}}$$

(Зверніть увагу, що середньоквадратичні відхилення не підсумовують, а дисперсії роблять це).

Застосовуючись у двох вимірах, ця формула утворює гауссову поверхню, яка має максимум у початку координат, контури якого - це концентричні кола з центром початку координат. Двовимірна матриця згортки попередньо обчислюється з формули та складається із двовимірних даних. Для кожного елемента в результуючій матриці нове значення встановлюється середньозваженим для сусідства цих елементів. Фокальний елемент отримує

найбільшу вагу (має найвище значення Гауса), а сусідні елементи отримують менші ваги, оскільки їх відстань до фокального елемента збільшується. При обробці зображень кожен елемент у матриці представляє атрибут пікселя, такий як яскравість або інтенсивність кольору, а загальний ефект називається розмиттям Гауса.

Гауссовий фільтр є непричинним, що означає, що вікно фільтра є симетричним щодо початку координат у часовій області. Це робить фільтр Гаусса фізично нездійсненним. Це зазвичай не має наслідку для програм, де пропускна здатність фільтра набагато більша за сигнал. У системах реального часу виникає затримка, оскільки вхідні вибірки повинні заповнити вікно фільтра, перш ніж фільтр може бути застосований до сигналу. Хоча жодна кількість затримок не може зробити теоретичну причинну причину Гаусова фільтра (оскільки функція Гауса скрізь ненульова), функція Гауса збігається до нуля настільки швидко, що причинне наближення може досягти будь-якого необхідного допуску із невеликою затримкою, навіть до точності подання з плаваючою точкою.

1.2 Означення задачі розпізнавання розмитих зображень

Видалення розмитості зображення - одна з найбільш класичних і складних проблем при обробці зображень на даний момент. Існують різні причини появи розмитості. Перша причина - відносний рух (рис. 1.1).

Рисунок 1.1 – розмите зображення через відносний рух

Якщо об'єкт в полі огляду камери переміщується під час експозиції, зображення, зняті камерою, є зображеннями з розмитими рухами. Область об'єкта на зображенні розмита при русі. Друга причина - в фокусі (рис. 1.2).

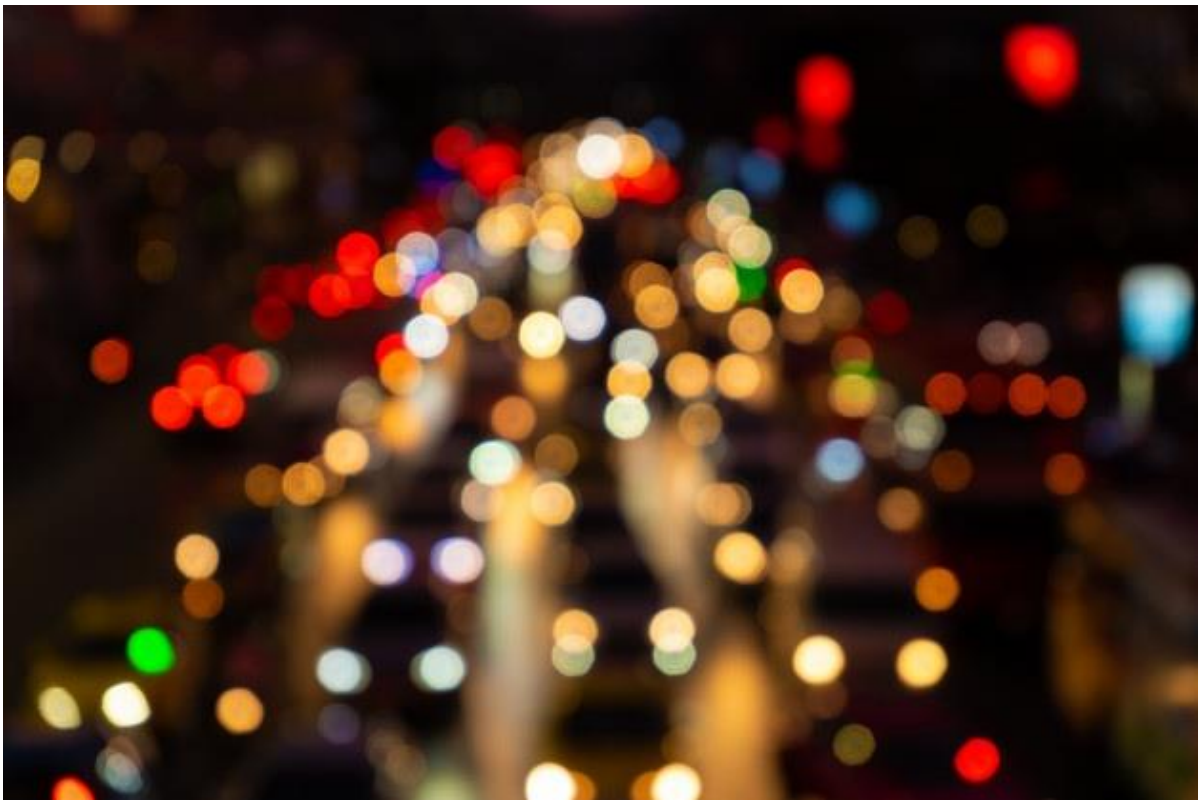

Рисунок 1.2 – неправильний фокус на фото широкого шоссе

Якщо об'єкт який знаходиться в полі огляду камери не перебуває у фокусі камери під час експозиції, зображення, зняті камерою, є розмитими зображеннями в розфокусуванні. Область об'єкта на зображенні розмита при розфокусуванні. Третя причина - змішання двох вищевказаних причин (рис. 1.3).

Рисунок 1.3 – розмите зображення через неправильне фокусування та відносний рух

Якщо об'єкт в поле огляду камери переміщується під час експозиції, а об'єкт не знаходиться у фокусі камери, зображення, одержувані камерою, є змішаними розмитими зображеннями. Область об'єкта на зображенні розмита.

Грунтуючись на наведеному вище аналізі можна зробити висновок, що розмиття відповідає об'єкту; це призводить до часткового непевного зображення. Області об'єкта на зображенні, які задовольняють одній з трьох вищезгаданих причин, розмиті. Решта областей зображення не розмиваються. Відновити частково розмите зображення - дуже складне завдання, тому що розмиті і світлі області співіснують в одному зображенні. Одним з рішень є відновлення всього часткового розмитого зображення з використанням того ж ядра розмиття; це викличе неприємні артефакти в чистих областях. Іншим можливим рішенням цієї проблеми є виявлення розмитих областей в частково

розмитих областях і подальше відновлення розмитої області з використанням алгоритму усунення розмитості зображення. За допомогою цього рішення можна уникнути появи артефактів в чистих областях.

Виявлення розмитих областей на частковому розмитому зображенні називається виявленням розмиття. Це дуже складна проблема, і вона дуже важлива для часткового відновлення розмитих знімків. Після того, як розмита область видалена з часткового розмитого зображення, наступна проблема полягає в оцінці ядер розмиття. Для розмитих областей дефокусування, руху і змішування їх ядра розмиття можуть бути виражені в математичній моделі.

1.3 Математичні моделі розпізнавання розмитості

Якщо клас розмиття може бути підтверджений і структура математичної моделі відома, нам просто потрібно оцінити параметри моделі. Але в багатьох ситуаціях клас розмиття, до якого належить розмита область, невідомий. Наприклад, ми не знаємо, чи є розмита область розмитою областю розфокусування, областю розмиття руху або областю змішаного розмиття.

Алгоритм класифікації розмиття може вирішити цю проблему він може класифікувати розмиту область в клас розмиття. Пропонований алгоритм відновлює часткове розмиття зображення за допомогою алгоритму класифікації розмиття і алгоритму виявлення розмиття. Його можна представити таким чином:

$$y = x \otimes k + n, \qquad (1)$$

де y позначає розмите зображення, x це приховане чітке зображення, $\otimes$ позначає оператор згортки, k розмитість ядра і n позначає шум. Ядра розмиття

різних видів розмиття неоднакові. Наприклад, математична модель розмиття руху - це лінія. Це можна показати наступним чином:

$$k(x,y) \tag{2}$$
$$= \begin{cases} \frac{1}{L}\ if\ \sqrt{x^2+y^2} \le L, tan\theta = \frac{y}{x} \\ 0\ else, \end{cases}$$

де k - ядро розмиття, сума всіх елементів дорівнює 1. x позначає горизонтальну координату, y означає вертикальну координату, L є шкалою розмиття та θ кутом розмиття. Математичною моделлю розмиття розфокусування є диск; це можна показати наступним чином:

$$k(x,y) = \begin{cases} \frac{1}{\pi R^2}\ if\ x^2 + y^2 \le R^2 \\ 0\ else, \end{cases} \tag{3}$$

де R шкала розмиття. Математичною моделлю ядра розмитої суміші є згортка вищезазначених двох ядер розмиття.

1.4 Алгоритми розв'язання моделей розпізнавання розмитих зображень

На основі вищезгаданої моделі розмиття зображення теорії і алгоритми усунення розмитості були розроблені доволі швидко. Наприклад алгоритми усунення розмитості зображення на основі фільтрів, такі як фільтр Вінера [1], успішно використовуються для усунення розмитості зображення. Такий фільтр можна представити у вигляді наступного рівняння:

$$\hat{F}(u,v) = \tag{4}$$
$$\left[\frac{H^*(u,v)}{|H(u,v)|^2 + \frac{S_{\tilde{\eta}}(u,v)}{S_f(u,v)}}\right] G(u,v),$$

де $\hat{F}(u,v)$ позначає перетворення Фур'є відновленого зображення; ми можемо використовувати зворотне перетворення Фур'є, щоб отримати відновлене зображення; $H^*(u,v)$ позначає спряженість перетворення Фур'є ядра розмиття, $S_{\tilde{\eta}}(u,v)$ є спектром потужності шуму та $S_f(u,v)$ спектром потужності розмитого зображення. $G(u,v)$ позначає перетворення Фур'є розмитого зображення. Потім алгоритми розмивання зображень на основі регуляризації розроблялися дуже швидко; цей тип методів дуже добре працює при розмитті зображень; це все ще основний потік алгоритмів розмивання зображень. Основною ідеєю алгоритму, заснованого на регуляризації, є побудова терміну регуляризації, який може забезпечити, щоб відновлене зображення відповідало особливій умові. Побудова терміну регуляризації є ключовим моментом методів, заснованих на регуляризації; найвідоміший термін регуляризації називається TV (Total Variation) [9].

Розмивання зображень на основі регуляризації може бути змодельовано як таке рівняння:

$$x^* = \arg\min\left[\|x \otimes k - y\|_2{}^2 + \alpha(x)\right], \qquad (5)$$

де x^2 позначає відновлене зображення. Перший термін - термін вірності даних; другий член - термін регуляризації; це функція прихованого чіткого зображення x. Враховуючи математичну модель, ми можемо отримати відновлене зображення, розв'язуючи (5). Деякі умови регуляризації дуже складні; аналітичні рішення цих методів дуже важко отримати, тому для оптимального рішення використовується техніка оптимізації. Побудова терміну регуляризації та методи вирішення (5) - два найважливіші ключові

моменти. Більшість алгоритмів розмивання зображень на основі регуляризації зосереджені на вирішенні цих двох проблем. У міру розвитку техніки розмивання зображень пропонуються більш складні проблеми розмивання зображення. Відновлення частково розмитого зображення - одне з них. У цій роботі пропонується рішення цієї проблеми на основі виявлення та класифікації розмиття.

РОЗДІЛ 2. МЕТОДИ УСУНЕННЯ РОЗМИТТЯ ЗОБРАЖЕНЬ

2.1 Основні принципи

Розмиття - це процес видалення розмитих артефактів із зображень [вхідне зображення, нехай це буде B, що є розмитим зображенням, яке зазвичай відбувається внаслідок тремтіння камери або якогось іншого явища]. Тепер ми хочемо відновити вхідне зображення S із розмитого зображення, яке є B. Математично ми представляємо B = S * K, де B є розмитим вхідним зображенням, нам потрібно знайти як різке зображення S, так і K, яке є ядром розмиття і * називається згорткою або функцією згортки . Ми говоримо, що S взаємодіє з K, щоб генерувати розмите зображення B, де K - розмиття, спричинене розфокусованою аберацією, розмиттям руху, розмиттям за Гаусом або будь-яким розмиттям. Отже, наша мета зараз відновити S, яке є гострим нерозмитим зображенням, а також K, і процес відомий як Deblurring, і деякі люди називали це також Unblur, але Deblur - це правильне технічне слово.

Розмиття K, як правило, моделюється як функція згортки і складається з гіпотетичним різким зображенням S, щоб отримати розмите зображення B, де як різке зображення S (яке слід відновити), так і функція K розподілу точок невідомі. Це приклад оберненої задачі. Майже у всіх випадках на розмитому зображенні недостатньо інформації, щоб однозначно визначити правдоподібне вихідне зображення, що робить його неправильно поставленою проблемою. Крім того, розмите зображення містить додатковий шум, що ускладнює завдання визначення вихідного зображення. Як правило, це вирішується використанням терміну регуляризації для спроби усунення

неправдоподібних рішень. Ця проблема є аналогом видалення відлуння в області обробки сигналів. Проте, коли для формування зображень використовується когерентний промінь, функцію розподілу точок можна моделювати математично. Завдяки належній деконволюційній функції розподілу точок K та розмитого зображення B, розмите зображення B може бути дерозмито (Deblurred) і відновлено різке зображення S.

2.2 Функція згортки

У математиці деконволюція - це операція, зворотна конволюції. Обидві операції використовуються при обробці сигналів та обробці зображень. Наприклад, згортка може бути використана для застосування фільтра, і може бути можливо відновити вихідний сигнал за допомогою деконволюції.

Основи деконволюції та аналізу часових рядів значною мірою заклав Норберт Вінер з Массачусетського технологічного інституту у своїй книзі "Екстраполяція, інтерполяція та згладжування стаціонарних часових рядів" (1949) [2]. Книга була заснована на роботі, яку Вінер зробив під час Другої світової війни, але яка була класифікована на той час. Деякі з перших спроб застосувати ці теорії були в галузі прогнозування погоди та економіки.

Загалом, метою деконволюції є пошук рішення f рівняння згортки виду:

$$f * g = h \qquad (6)$$

Зазвичай h - це якийсь записаний сигнал, а f - це якийсь сигнал, який ми хочемо відновити, але до того, як ми його записали, був сформований із фільтром або функцією спотворення g. Функція g може представляти передавальну функцію приладу або рушійну силу, застосовану до фізичної системи. Якщо ми знаємо g або, принаймні, знаємо форму g, тоді ми можемо виконати детерміновану функцію згортки (деконволюцію). Однак, якщо ми не

знаємо g заздалегідь, тоді нам потрібно це оцінити. Найчастіше це робиться із використанням методів статистичної оцінки.

При фізичних вимірах ситуація зазвичай ближча до

$$(f * g) + \varepsilon = \mathrm{h} \qquad (7)$$

У цьому випадку ε - це шум, який потрапив у наш записаний сигнал. Якщо шумний сигнал або зображення вважається безшумним, статистична оцінка g буде неправильною. У свою чергу, оцінка f також буде неправильною. Чим нижчим буде відношення сигнал / шум, тим гіршою буде оцінка деконверсованого сигналу. Саме тому зворотна фільтрація сигналу, як правило, не є гарним рішенням. Однак, якщо існують хоча б деякі відомості про тип шуму в даних (наприклад, білий шум), оцінка f може бути покращена за допомогою таких методів, як деконволюція Вінера[10].

Деконволюція зазвичай виконується шляхом обчислення перетворення Фур'є записаного сигналу h та функції спотворення (загалом, вона відома як передавальна функція) g. Потім деконволюція виконується в частотній області (за відсутності шуму), використовуючи:

$$F = H/G \qquad (8)$$

де F, G та H - перетворення Фур'є f, g та h відповідно. Нарешті, обернене перетворення Фур'є функції F береться для знаходження оціненого деконвертованого сигналу f.

2.3 Основні алгоритми усунення розмиття

Цифрові зображення - це електронні знімки сцени, які зазвичай складаються з елементів зображення в сітці утворення, відоме як пікселі. Кожен піксель містить зліченну величину значення, яке представляє тон у певній точці. Зображення отримуються в районах, починаючи від повсякденної фотографії до астрономії, дистанційного зондування, медичної візуалізації та мікроскопії.

На жаль, усі зображення в кінцевому підсумку стають більш-менш розмитими. Це пов'язано з тим, що в системі навколишнього середовища, а також у камері розмиття або деградація зображення може бути спричинена багатьма факторами такі як рух під час захоплення, використовуючи довгий час експозиції, використовуючи ширококутний об'єктив тощо. Зображення розмивання використовується, щоб зробити зображення чіткими та корисними з використанням математичної моделі.

2.3.1 Модель деградації

Процес деградації можна візуалізувати за допомогою наступної системи:

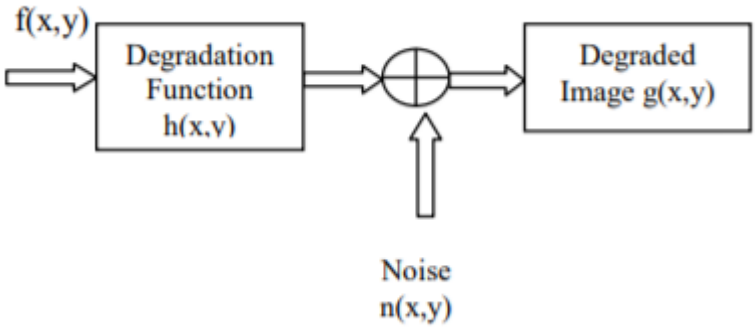

Рисунок 2.1 – модель деградації

Оригінальне введення - це двовимірне (2D) зображення f (x, y). Цим зображенням керує система h (x, y) і після додавання шуму n (x, y). Можна отримати деградоване зображення g (x, y). Цифрове відновлення зображення можна розглядати як процес, в якому ми намагаємось отримати наближення до f (x, y). Розмите зображення можна описати за допомогою наступного рівняння:

$$g\,(x,y) \;=\; h\,(x,y) \;*\; f\,(x,y) \;+\; n\,(x,y) \qquad (9)$$

2.3.2 Алгоритм Річардсона-Люсі

Алгоритм Річардсона – Люсі, також відомий як деконволюція Річардсона – Люсі - це ітеративна процедура для відновлення прихованого зображення, яке було розмитим відомою функцією згортки.

$$\mathrm{C}_i = \Sigma_j * p_{ij} * u_j \qquad (10)$$

Де: p_{ij} - функція розподілу точки (частка світла походить з справжнього розташування j, яке спостерігається в положенні i), u_j - значення пікселя в місці j на прихованому зображенні, і c_i - спостережуване значення в місці розташування пікселів i. Це рівняння виконується при припущенні, що u_j – розподіл Пуассона, що підходить для фотонного шуму в даних. Основна ідея полягає в обчисленні найбільш ймовірного заданого u_j при спостережуваному c_i і відомому p_{ij}. Це призводить до рівняння для u_j, яке можна вирішити ітеративно згідно:

$$u_j = u_j{}^t \Sigma_j * p_{ij}, \qquad (11)$$

Емпірично показано, що якщо ця ітерація зближується, вона сходиться до максимальної ймовірності рішення для u_j.

2.3.3 Нейронна мережа

Нейронна мережа (рис. 2.2) - це низка алгоритмів, які намагаються розпізнати основні взаємозв'язки в наборі даних за допомогою процесу, що імітує роботу людського мозку(нейронів). У цьому сенсі нейронні мережі відносяться до систем нейронів, органічних або штучних за своєю природою. Нейронні мережі можуть адаптуватися до змінних входів; таким чином мережа дає найкращий можливий результат без необхідності переробляти критерії виводу. Концепція нейронних мереж, корінням якої є штучний інтелект, на сьогоднішній день є одним з найбільш популярних методів Data Science.

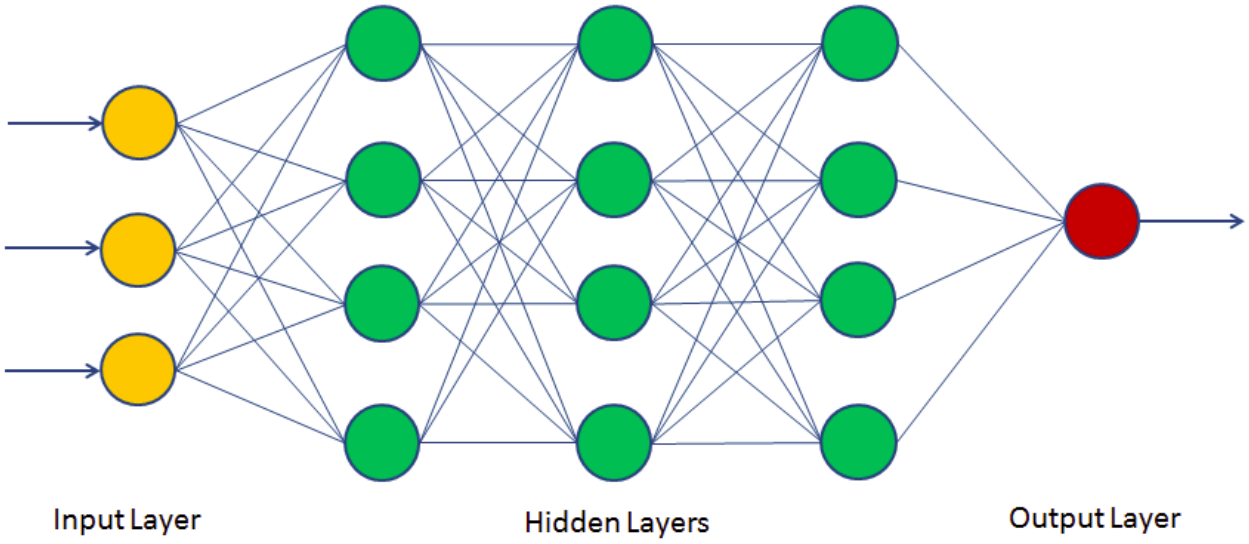

Рисунок 2.2 – схема нейронної мережі

Нейронна мережа працює подібно до нейронної мережі людського мозку. Нейрон у нейронній мережі - це математична функція, яка збирає та класифікує інформацію відповідно до конкретної архітектури. Мережа сильно нагадує статистичні методи, такі як підгонка кривих та регресійний аналіз.

Нейронна мережа (CNN) містить шари взаємопов'язаних вузлів. Кожен вузол є персептроном і схожий на багаторазову лінійну регресію. Персептрон подає сигнал, що виробляється багаторазовою лінійною регресією, у функцію активації, яка може бути нелінійною.

У багатошаровому персептроні (MLP) персептрони розташовані взаємопов’язаними шарами. Вхідний рівень збирає вхідні шаблони. Вихідний рівень має класифікації або вихідні сигнали, до яких можуть вказуватися шаблони введення. Наприклад, моделі можуть містити перелік величин для технічних показників про цінний папір; потенційні результати можуть бути "купувати", "утримувати" або "продавати".

Приховані шари тонко налаштовують вхідні зважування, поки похибка нейронної мережі не стане мінімальною. Існує гіпотеза, що приховані шари екстраполюють суттєві риси у вхідних даних, які мають прогнозуючу силу щодо виходів. Це описує вилучення особливостей, яке виконує корисність, подібну до статистичних методів, таких як аналіз основних компонентів.

Оскільки CNN є адаптивним алгоритмом(supervised), то ми повинні навчити її, щоб вона стала адаптивною до будь-якого з поданих входів.

З метою навчання збирається набір розмитих зображень та відповідні їм основні зображення без дефектів. І для того, щоб подати зображення як вхід для цієї нейронної мережі, ми повинні зробити це зображення в певному форматі зображення, тобто потрібно стиснути його як матовий файл. Отже,

спочатку розмиті та основні істинні зображення поділяються на блоки. Деяка частина цих блоків використовується для навчання, а інші використовуються для тестування мережі. Блоки, що використовуються для тренувань отримують ярлик 1, а тих, що використовуються для тестування, ярлик 2. Блоки оригінального зображення та його розмитість версії обидва перетворюються в 4D єдиний формат. Це як структура подається як вхід для мережі.

Чим більша кількість зображень, що використовуються для тренувань, тим буде більшою ефективність CNN. Після цього CNN готова до перевірки розмивання. Тепер розмите зображення можна подати як вхід для CNN, і ми отримаємо зображення яке є чітким зображенням як вихідний результат.

РОЗДІЛ 3. РОЗРОБКА ПРОГРАМИ

3.1 Попередня обробка файлів

Перед тим як використовувати будь який алгоритм машинного навчання нам потрібно обробити дані. І в цьому випадку у нас може бути безліч зображень з різними розмірами.

Тому якщо на вході програма має зображення з різними розмірами потрібно їх або видалити з набору або змінити їх розмір до підходящого як зображено на рис. 3.1.

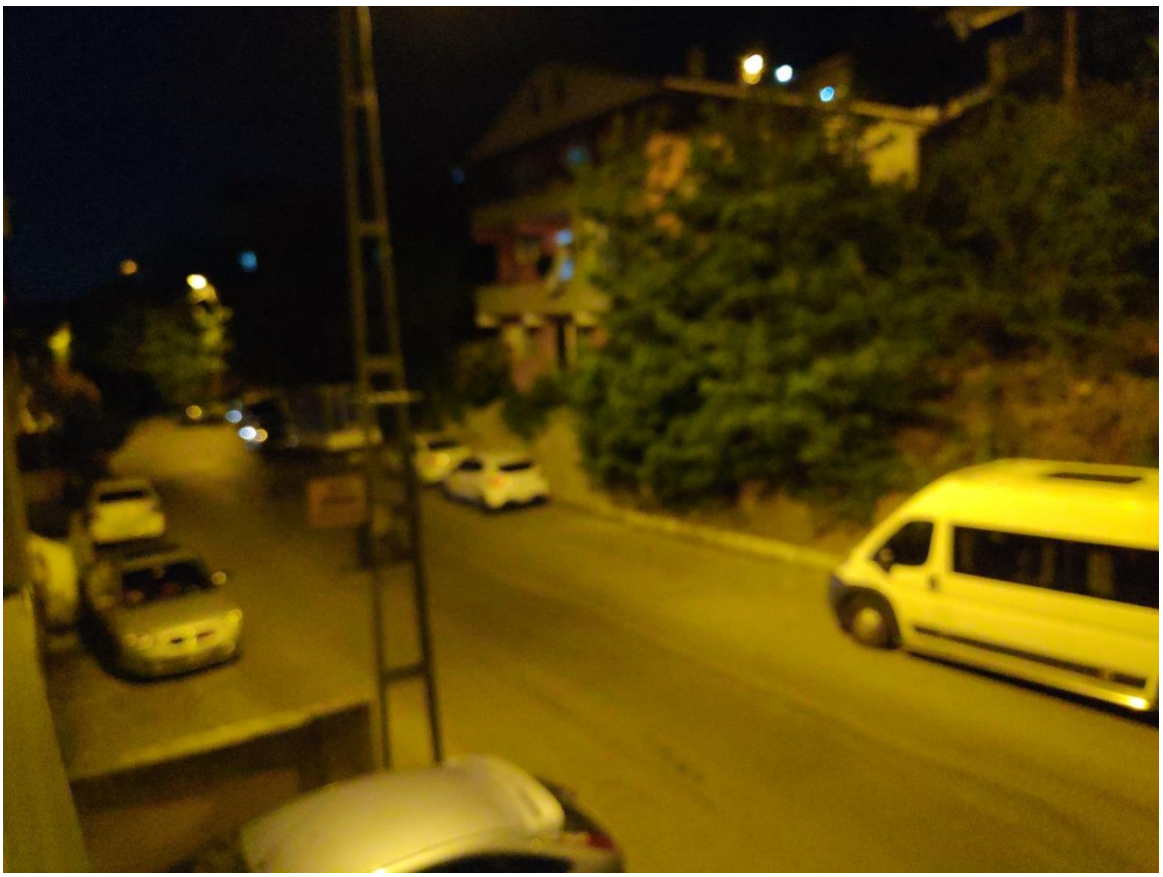

Рисунок 3.1 – розмите зображення розміром 2048x1800

3.2 Загальний алгоритм роботи програми

Основна ідея алгоритму програми полягає в операторі Лапласса[11] та використанні алгоритму машинного навчання - SVM[12].

3.2.1 Оператор Лапласса

Згідно з Вікіпедією[11], лапласіан функції f у точці p є (до коефіцієнта) швидкістю, з якою середнє значення f над сферами з центром у p відхиляється від $f\ (p)$, оскільки радіус кулі зменшується до 0 Зазвичай його позначають символами $\nabla\ \cdot\ \nabla, \nabla\ \hat{}\ 2$ (де ∇ - оператор набла) або Δ.

Простіше кажучи, лапласіанський оператор визначається як розбіжність градієнта функції f.

$$\Delta\ f\ (x, y)\ =\ div\ (grad\ (f)) \qquad (12)$$

А градієнт – найкрутіший схил. По суті, він дає інформацію про напрямок, точку (на кривій) по якій слід рухатись, щоб досягти найвищого акценту, тобто локальних максимумів. Так само негативний градієнт дає напрямок локальних мінімумів.

Наприклад, якщо врахувати графік, породжений функцією $z\ =\ x\ *\ exp\ (-x\ \hat{}\ 2\ +\ y\ \hat{}\ 2))$ та градієнт у кожній точці кривої (представлений лінією зі стрілкою). Ми можемо отримати таку фігуру, як нижче (рис. 3.2):

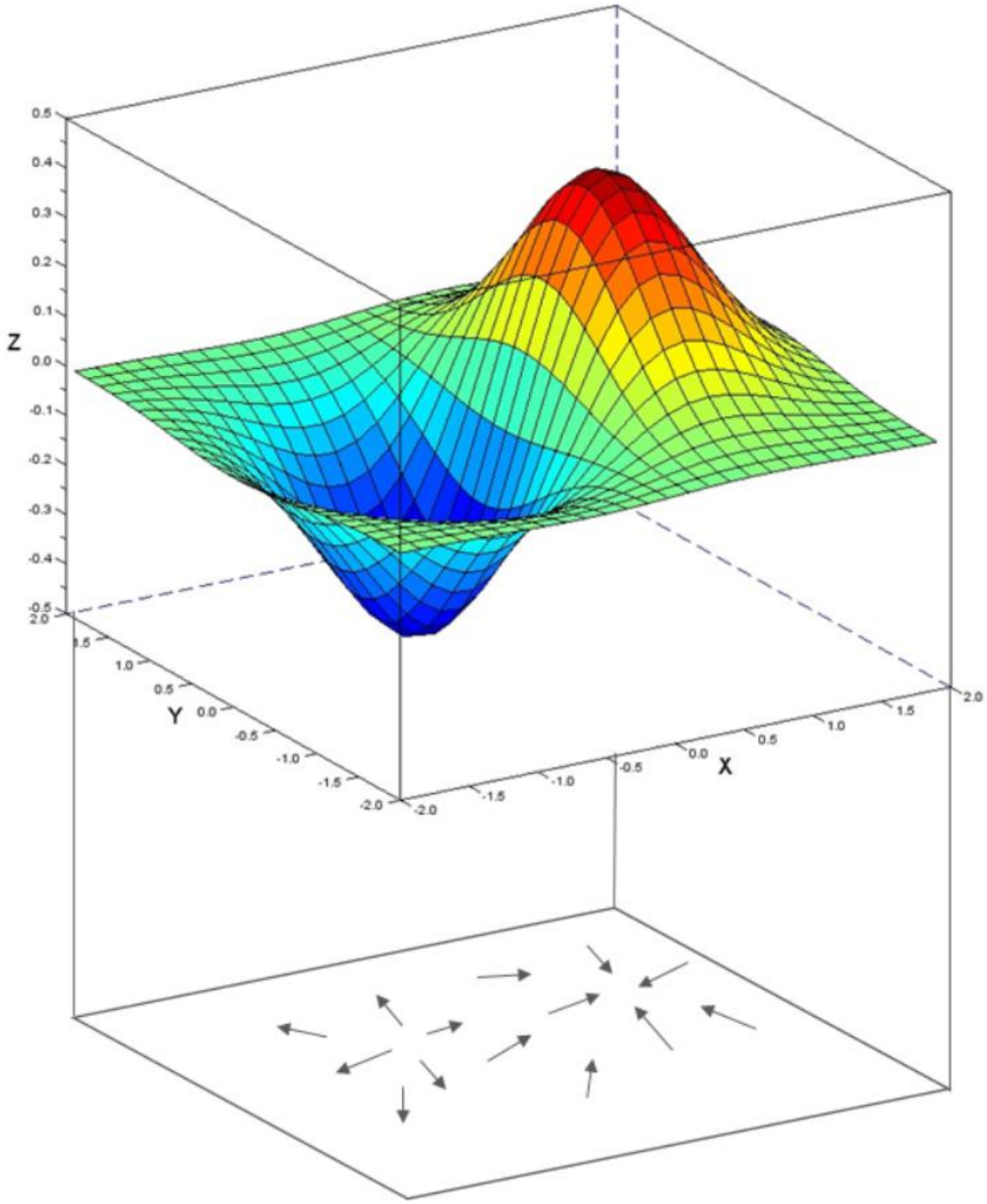

Рисунок 3.2 – зображення функції $z = x * exp(-x^2 + y^2))$ та її градієнтів в різних точках

Ми можемо розуміти розбіжність як векторне поле, що відповідає певному виду руху, породженого потоком рідини. На пагорбі графіка, коли

кожне векторне поле вказує на область, що спричиняє розбіжність, є негативною, а в долині, де кожен вектор вказує, розбіжність є додатною.

Отже, певним чином, лапласіанський оператор є своєрідним показником того, наскільки мінімальна точка дорівнює x, y. Таким чином, діючи як друга похідна для багатовимірної функції зі скалярним значенням.

Друга похідна функції може бути використана для визначення локальних екстремумів для функції. Якщо функція має критичну точку f ’(x) = 0, на якій друга похідна точки f’’(x) <0, тобто від’ємна, то f має локальний максимум у точці. Це можна додатково проілюструвати на малюнку нижче (рис. 3.3):

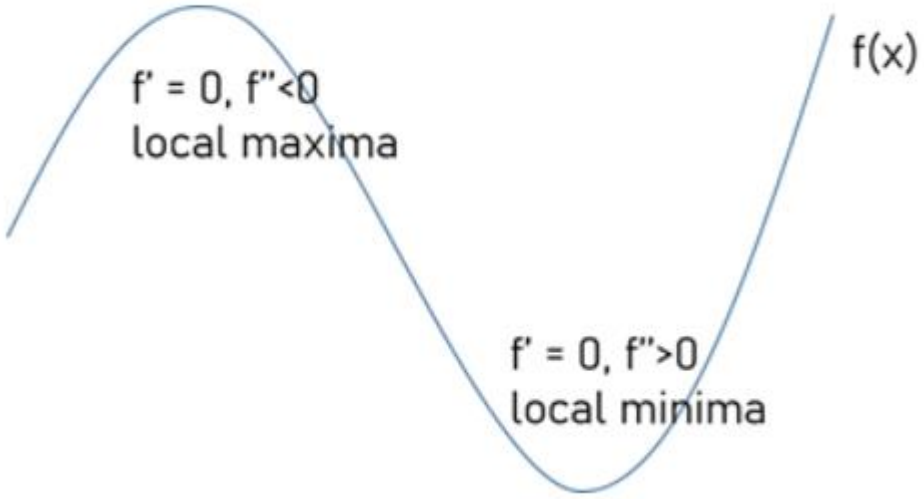

Рисунок 3.3 – локальний мінімум та максимум функції $f(x)$

Лапласіан може бути представлений як:

$$\nabla^2 f = \frac{\partial^2 \mathrm{f}}{\partial^2 \mathrm{x}} + \frac{\partial^2 f}{\partial^2 y} \qquad (13)$$

І поєднуючи рівняння для часткових похідних, ми можемо отримати:

$$\nabla^2 f = [f(x+1, y] \qquad (13)$$

Тепер розглянемо, як виглядає фільтр. 3 * 3 фільтр для зображення може бути представлений для координат (x, y) і може бути представлений у вигляді (рис. 3.4):

x-1,y-1	x,y-1	x+1,y-1
x-1,y	x,y	x,y+1
x-1,y+1	x,y+1	x+1,y+1

Рисунок 3.4 – фільтр Лапласса

Ця матриця разом із зображенням, дає трансформоване зображення на основі лапласіана. Вона використовується для пошуку області швидких змін на зображеннях, якщо немає шуму. По суті, оператор приймає другу похідну зображення на деякій вищій розмірній площині. Якщо зображення в основному однорідне, результат буде нульовим. Де б не відбулася зміна, результуюча матриця матиме позитивні елементи на темній стороні і негативний елемент на яскравій стороні (рис. 3.5).

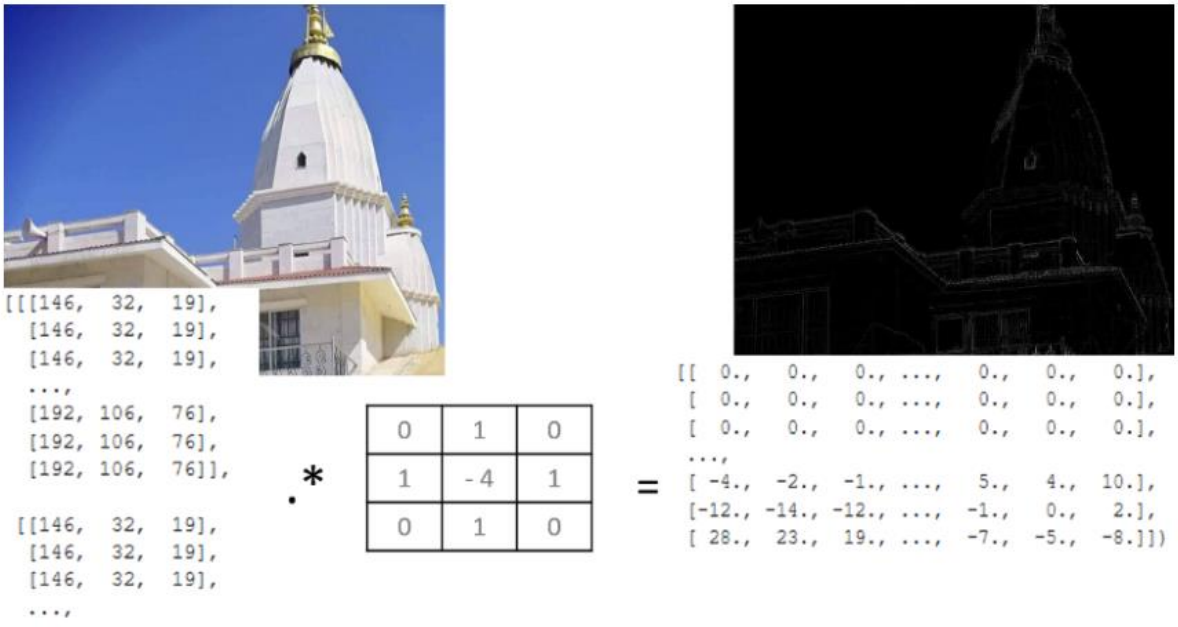

Рисунок 3.5 – операція згортки використовуючи оператор Лапласса

Таким чином, ми можемо за допомогою оператора Лапласса визначити, розмито зображення чи ні. Поріг можна встановити на основі продуктивності даних, які ми маємо. Це теоретично найкраще буде працювати на зображеннях, отриманих з одного джерела, наприклад, зображення з живого каналу, а не колекція зображень, отриманих з різних джерел камери різних телефонів в різний час доби, але має місце експериментування.

3.2.2 SVM

Метод опорних векторів (SVM) - це керований алгоритм машинного навчання, який може застосовуватися як для класифікації, так і для регресії. SVM частіше використовуються у проблемах класифікації, як і задана, тому я вирішив використовувати саме цей метод.

SVM базуються на ідеї пошуку гіперплощини, яка найкраще ділить набір даних на два класи, як показано на зображенні нижче.

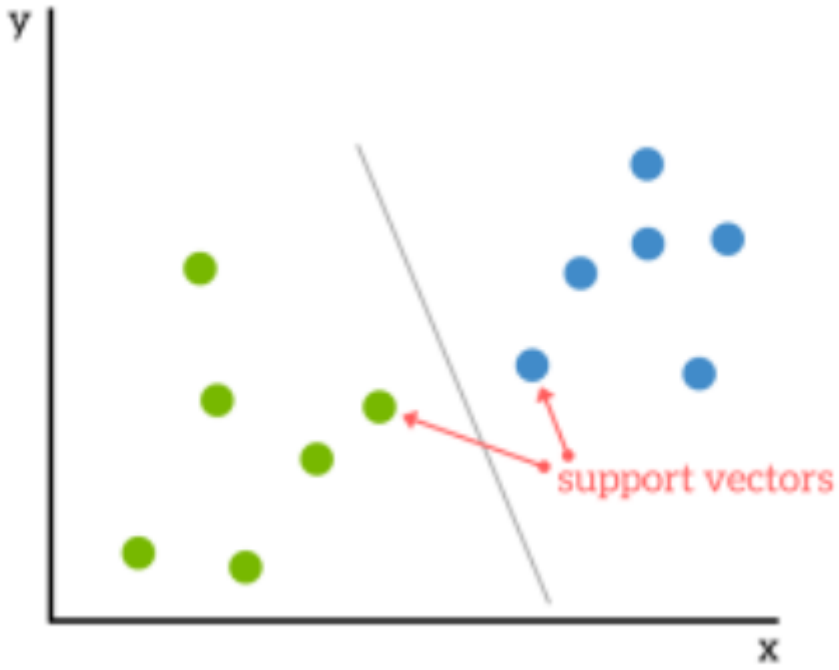

Рисунок 3.6 – опорні вектори

Вектори опори - це точки даних, найближчі до гіперплощини, точки набору даних, які, якщо їх видалити, змінять положення розділювальної гіперплощини. Через це їх можна вважати критично важливими елементами набору даних.

Як простий приклад, для завдання класифікації лише з двома ознаками (як на малюнку вище) можна уявити гіперплощину як лінію, яка лінійно відокремлює та класифікує набір даних.

Інтуїтивно зрозуміло, що чим далі від гіперплощину лежать наші точки даних, тим впевненіше ми впевнені в тому, що їх правильно класифікували. Тому ми хочемо, щоб наші точки даних були якомога далі від гіперплощини, при цьому залишаючись на правильній її стороні.

Отже, коли додаються нові дані тестування, будь-яка сторона гіперплощини, на яку вони потрапляють, вирішить клас, який ми йому присвоюємо.

Відстань між гіперплощиною та найближчою точкою даних з будь-якого набору відома як межа. Мета полягає в тому, щоб вибрати гіперплощину із максимально можливим запасом між гіперплощиною та будь-якою точкою в навчальному наборі, що дає більший шанс на правильну класифікацію нових даних.

Тому за такими ознаками саме цей алгоритм є найбільш підходящим для даної задачі.

3.3 Результати роботи

Використовуючи такі бібліотеки Python як Numpy та SciPy була реалізована програма, яка створює та навчає дану модель машинного навчання. Для її навчання було використано датасет “Blur detection dataset”[5].

Для порівняння результатів було використано ще 2 алгоритма машинного навчання для даного датасету: дерево прийняття рішень та логістична регресія. Результати можна побачити на наступній таблиці 3.1:

Табл. 3.2 – Результати роботи алгоритмів

Алгоритм	Ефективність на тренувальному наборі	Ефективність на валідаційному наборі
Метод опорних векторів	0.95	0.85

Дерево прийняття рішень	0.84	0.78
Логістична регресія	0.81	0.71

ВИСНОВКИ

В даній роботі було детально розглянуто методи та моделі розмитих зображень, їх специфіку, основні труднощі при розв’язанні задач розпізнавання та усунення розмивання зображень. Був виконаний загальний огляд методів розпізнавання розмиття зображень. Детально були розглянуті такі методи та алгоритми Data Science як метод опорних векторів, дерево прийняття рішень та логістична регресія.

Розроблена програма для навчання моделі розпізнавання розмитих зображень та була протестована на ряді даних. Результати тестування були порівняні за допомогою ще двох популярних алгоритмів машинного навчання. Метод опорних векторів, на користь якого було надано перевагу забезпечив кращу ефективність розпізнавання, що говорить про доцільність його використання. Шляхом реалізації більш складних та досконалих методів стиснення ефективність програми може бути підвищена.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Python [Електронний ресурс] – Режим доступу до ресурсу: https://www.python.org/.
2. NumPy [Електронний ресурс] – Режим доступу до ресурсу: https://numpy.org/.
3. SciPy [Електронний ресурс] – Режим доступу до ресурсу: https://www.scipy.org/.
4. matplotlib [Електронний ресурс] – Режим доступу до ресурсу: https://matplotlib.org/.
5. IPython [Електронний ресурс] – Режим доступу до ресурсу: https://ipython.org/.
6. PyCharm [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/pycharm/.
7. IDLE [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/3/library/idle.html.
8. git [Електронний ресурс] – Режим доступу до ресурсу: https://git-scm.com/.
9. Total Variation [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Total_variation_denoising.
10. Wiener Deconvolution [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Wiener_deconvolution.
11. Laplace Operator [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Laplace_operator.
12. Support-vector machine [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Support-vector_machine.

13. Zaharescu M. Image Deblurring: Challenges and Solutions [Електронний ресурс] / M. Zaharescu, C. Boiangiu. – 2013. – Режим доступу до ресурсу: https://www.researchgate.net/publication/260074006_Image_Deblurring_Challenges_and_Solutions.
14. Dejee S. A Survey on Various Image Deblurring Techniques [Електронний ресурс] / S. Dejee, R. K. Sahu – Режим доступу до ресурсу: https://ijarcce.com/wp-content/uploads/2012/03/IJARCCE6D-a-_Dejee_A_SURVEY_ON.pdf.
15. Sharma A. A NOVEL APPROACH FOR THE DETECTION OF BLUR USING SVM AND KNN CLASSIFICATION TECHNIQUES IN IMAGE PROCESSING [Електронний ресурс] / A. Sharma, D. Shukla // 2016 – Режим доступу до ресурсу: https://www.ijates.com/images/short_pdf/1469550785_1110ijates.pdf.
16. Sagar. Laplacian and its use in Blur Detection [Електронний ресурс] / Sagar. – 2020. – Режим доступу до ресурсу: https://medium.com/swlh/laplacian-and-its-use-in-blur-detection-fbac689f0f88.
17. Bambrick N. Support Vector Machines: A Simple Explanation [Електронний ресурс] / Noel Bambrick – Режим доступу до ресурсу: https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html.
18. Blur Detection Dataset [Електронний ресурс] – Режим доступу до ресурсу: http://www.cse.cuhk.edu.hk/~leojia/projects/dblurdetect/dataset.html.
19. Работа с изображениями на Python [Електронний ресурс] // 2018 – Режим доступу до ресурсу: https://habr.com/ru/company/oleg-bunin/blog/425471/.

20. Rosebrock A. OpenCV Fast Fourier Transform (FFT) for blur detection in images and video streams [Електронний ресурс] / Adrian Rosebrock // 2020 – Режим доступу до ресурсу: https://www.pyimagesearch.com/2020/06/15/opencv-fast-fourier-transform-fft-for-blur-detection-in-images-and-video-streams/.
21. Brief B. Mobile Image Blur Detection with Machine Learning [Електронний ресурс] / Benedikt Brief – Режим доступу до ресурсу: https://medium.com/snapaddy-tech-blog/mobile-image-blur-detection-with-machine-learning-c0b703eab7de.
22. Opencv blur detection and Python [Електронний ресурс] – Режим доступу до ресурсу: https://www.programmersought.com/article/46582774111/.
23. [Python image detection] image blur detection algorithm [Електронний ресурс] – Режим доступу до ресурсу: https://www.programmersought.com/article/13406764045/.

ДОДАТКИ