Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Факультет комп'ютерних наук
Спеціальність 125 «Кібербезпека»
Освітня програма «Безпека інформаційних та комунікаційних систем»

«Допущено до захисту»
В.о. зав. кафедрою БІСТ
Ольга МЕЛКОЗЬОРОВА
_____
« » 2023 р.

Пояснювальна записка
до кваліфікаційної роботи магістра
на тему: «Скелетизація дактилоскопічних зображень на основі адаптивного фільтра Габора»

оцінка « »
Голова ЕК
Олександр ЛЕМЕШКО _____

Керівник: к.т.н., доц. Нарєжній О.П.
Рецензент: PhD, старш. викл. Лисицький К. Є.
Виконавець: студент групи КБ-61
Демидов Єгор Михайлович

Харків - 2023

РЕФЕРАТ

Пояснювальна записка містить 64 сторінок, 43 рисунків, 2 таблиць, 1 додаток, 19 джерел.

Метою дипломної роботи є дослідження й розробка методу скелетизації зображень на основі адаптивного фільтра Габора, що дозволить виділяти структурні особливості та зменшити об'єм даних, що потрібно для їх збереження.

Об’єктом дослідження дипломної роботи є дактилоскопічні зображення, тобто зображення папіллярних ліній на пальцях людини.

Предметом розробки є метод обробки зображень, та скелетизація дактилоскопічних зображень. Розробка спрямована на використання адаптивного фільтра Габора для вирішення проблеми скелетизації дактилоскопічних зображень, зокрема для визначення структурних особливостей на зображеннях та зменшення об'єму даних, що потрібно для збереження цих зображень.

Методи дослідження:

- Фільтр Габора.
- Математична морфологія.
- Алгоритми скелетизації.
- Методи машинного навчання.
- Алгоритми обробки сигналів.

Результатом проведеної роботи є метод скелетизаццїї дактилоскопічних зображень, який може бути використаний в різних областях, пов'язаних з обробкою зображень та біометричними технологіями. Основні напрямки використання результатів дослідження можуть бути такі:

- Системи автоматичної ідентифікації осіб: Результати роботи можуть бути використані в системах автоматичної ідентифікації осіб на основі

дактилоскопічних зображень. Отриманий алгоритм скелетизації може бути використаний для ефективного визначення основних рис дактилоскопічних зображень та підвищення точності ідентифікації.

- Медичні дослідження: Результати роботи можуть бути використані для медичних досліджень, зокрема, для визначення аномалій у будові дактилоскопічних зображень та їх аналізу.

- Розпізнавання рукописного тексту: Отриманий алгоритм може бути використаний для розпізнавання рукописного тексту та покращення точності розпізнавання символів.

- Розпізнавання облич: Дослідження можуть бути використані для розпізнавання облич та підвищення точності розпізнавання особи на зображеннях.

Ключові слова: СКЕЛЕТИЗАЦІЯ, ДАКТИЛОСКОПІЧНІ ЗОБРАЖЕННЯ, АДАПТИВНИЙ ФІЛЬТР ГАБОРА, ОБРОБКА ЗОБРАЖЕНЬ, БІОМЕТРИЧНА ІДЕНТИФІКАЦІЯ, ІДЕНТИФІКАЦІЯ ОСІБ, МАТЕМАТИЧНІ МЕТОДИ ОБРОБКИ ЗОБРАЖЕНЬ, ВІДНОВЛЕННЯ СТРУКТУРИ ДАКТИЛОСКОПІЧНИХ СЛІДІВ.

ABSTRACT

The explanatory note contains 64 pages, 43 figures, 2 tables, 1 annex, 19 sources.

The aim of the thesis is to research and develop a method of image skeletonization based on an adaptive Gabor filter, which will allow highlighting structural features and reducing the volume of data required for their preservation.

The subject matter research of the thesis is dactyloscopy images, i.e., images of papillary lines on human fingers.

The scope of the study is a method of image processing and skeletonization of dactyloscopy images. The development is aimed at using an adaptive Gabor filter to solve the problem of skeletonization of dactyloscopy images to determine structural features in the images and reduce the amount of data required to save these images.

Research methods:

• Gabor filter.

• Mathematical morphology.

• Skeletonization algorithms.

• Methods of machine learning.

• Signal processing algorithms.

The work result is a method of skeletonization of dactyloscopy images, which can be used in various areas related to image processing and biometric technologies. The main areas of use of research results can be as follows:

- Systems of automatic identification of persons: The work result can be used in systems of automatic identification of persons based on dactyloscopy images. The resulting skeletonization algorithm can be used to effectively determine the main features of dactyloscopy images and increase the accuracy of identification.

- Medical research: The results of the work can be used for medical research, in particular, to determine anomalies in the structure of dactyloscopy images and their analysis.

- Handwriting recognition: The resulting algorithm can be used to recognize handwritten text and improve character recognition accuracy.

- Face recognition: Research can be used to recognize faces and improve the accuracy of face recognition in images.

Keywords: SKELETONIZATION, FINGERPRINT IMAGES, ADAPTIVE GABOR FILTER, IMAGE PROCESSING, BIOMETRIC IDENTIFICATION, IDENTIFICATION OF PERSONS, MATHEMATICAL METHODS OF IMAGE PROCESSING, RECOVERY OF THE FINGERPRINT STRUCTURE.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА СИМВОЛІВ

Classification	–	визначення належності об'єкта до вже заданих класів
Clustering	–	створення та присвоєння класів об'єктам на нерозмічених даних
CNN	–	convolutional neural network
Cross Validation	–	методи оцінки ефективності роботи алгоритмів за допомогою незалежних даних
Datasets	–	убудовані набори даних для тестування алгоритмів
Dimensionality Reduction	–	виділення найбільш значущих ознак і даних
Feature Extraction	–	визначення атрибутів у зображеннях та текстових даних
Feature Selection	–	виявлення найбільш значущих ознак для побудови моделей
FLOPS	–	Floating Point Operations Per Second
FSRCNN	–	fast super-resolution convolutional neural network.
GAN	–	generative adversarial network
Manifold Learning	–	множинне навчання для нелінійного скорочення розмірності даних
MNIST	–	Mixed National Institute of Standards and Technology
OpenCV	–	Open Source Computer Vision Library
SRCNN	–	super-resolution convolutional neural network
ДКП	–	дискретне косинусне перетворення
ДПФ	–	дискретне перетворення Фур'є
ІЗ	–	інформаційне забезпечення
ПЗ	–	програмне забезпечення
ТЗ	–	технічне забезпечення
ШПФ	–	швидке перетворення Фур'є

ВСТУП

Дактилоскопія - це наука та практика ідентифікації особистостей за допомогою унікальних характеристик пальців. Однією з найпоширеніших методів дактилоскопії є аналіз папілярних відбитків пальців, який передбачає застосування різних методів обробки та аналізу зображень для точного розпізнавання пальцевих відбитків. Один із таких методів - використання фільтра Габора в поєднанні з нейронними мережами.

Фільтр Габора - це математичний інструмент, який використовується для аналізу текстур і структури зображень. Він базується на гаусівському ядру, зміщеному на різні кути та частоти. Цей фільтр допомагає виділити текстурні особливості на зображенні та може бути ефективним інструментом для вилучення унікальних рис папілярних відбитків. Після застосування фільтра Габора до вхідного зображення отримуємо кілька нових зображень, кожне з яких має відповідати певному куту і частоті. Ці зображення можуть бути подані на вхід нейронній мережі для подальшого аналізу та класифікації.

Нейронні мережі, зокрема згорткові нейронні мережі (CNN), виявилися дуже ефективними у багатьох завданнях комп'ютерного зору, включаючи аналіз та розпізнавання зображень. Вони здатні автоматично вивчати характеристики зображень та виконувати класифікацію на основі цих характеристик. У випадку розпізнавання дактилоскопічних зображень, CNN може бути використана для аналізу зображень, отриманих після застосування фільтра Габора.

Процес використання фільтра Габора та нейронних мереж для розпізнавання дактилоскопічних зображень може бути поділений на декілька етапів:

1) Збір та підготовка даних: Спочатку потрібно зібрати базу дактилоскопічних зображень і підготувати їх для подальшого аналізу. Це може включати в себе обрізку та нормалізацію зображень.

2) Застосування фільтра Габора: Для кожного зображення застосовується фільтр Габора з різними параметрами (кути та частоти), щоб отримати кілька нових зображень, які відображають текстурні особливості.

3) Вивчення характеристик: За допомогою згорткової нейронної мережі або іншої архітектури можна вивчити характеристики цих зображень та створити модель, яка буде визначати унікальні риси папілярних відбитків.

4) Класифікація: На заключному етапі нейронна мережа може бути використана для класифікації папілярних відбитків на основі навчених характеристик.

Використання фільтра Габора в поєднанні з нейронними мережами дозволяє створити ефективну систему розпізнавання дактилоскопічних зображень. Цей підхід допомагає підвищити точність ідентифікації та забезпечити надійний рівень безпеки в сферах, де дактилоскопія є важливою технологією, таких як правоохоронні органи, банківська сфера та інші. Такий підхід може бути особливо корисним у великих системах ідентифікації, де потрібно обробляти великий обсяг дактилоскопічних даних.

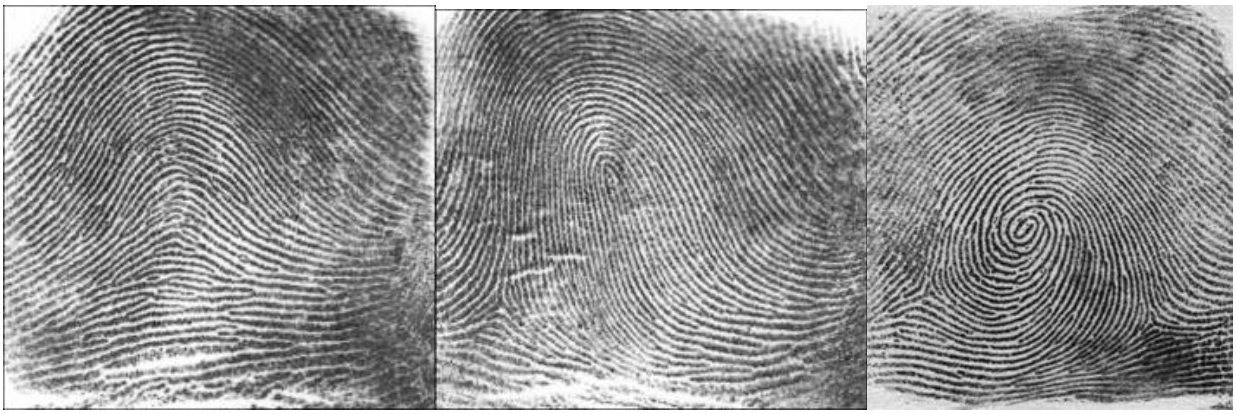
У підсумку, використання фільтра Габора та нейронних мереж у розпізнаванні дактилоскопічних зображень є потужним інструментом, який дозволяє досягти високого рівня точності та безпеки в ідентифікації особистостей на основі їх пальцевих відбитків. Цей підхід поєднує в собі математичні методи та технології штучного інтелекту для досягнення цієї важливої мети.

Основна задача роботи – програмна реалізація додатку для виконання скелетизації дактилоскопічних зображень на основі адаптивного фільтра Габора.

1 АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ, ДОСЛІДЖЕННЯ ІСНУЮЧИХ РІШЕНЬ, ЗБІР ТА АНАЛІЗ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1.1 Постановка завдань дослідження

Основна тема цієї роботи пов'язані з дактилоскопією – методом пізнання людини з відбиткам пальців, заснованому неповторності малюнка шкіри рук. В 1902 році вперше було застосовано з метою ідентифікації особистості. Після цієї дати метод дактилоскопічного пізнання особистості набув широкого поширення у різних країнах світу. Спочатку застосовувався у криміналістиці з метою ідентифікації особистості. Відсутність комп'ютерів у той час значно вплинула на розвиток дактилоскопії. Зі збільшенням кількості відбитків пальців виникла проблема їхньої класифікації, оскільки операцію порівняння доводилося виконувати вручну без автоматизації. Одним із методів класифікації було розділення відбитків за аналогією з папілярним візерунком, існує три основних типи відбитків: арки, петлі та завитки. Це вирішення проблеми дозволяло швидше проводити порівняння відбитків. (рис. 1.1).



(а) (б) (В)

Рисунок 1.1 - Різні типи візерунків відбитка пальця: арка (а), петля (б) та завиток (в).

Іншою характеристикою класифікації є так званий "лічильник ліній", який визначає кількість ліній у папілярному візерунку між різними точками зображення, наприклад, між центром та дельтою для відбитка типу "петля".

З появою комп'ютерів та їх використанням, комерційні корпорації також проявили інтерес до відбитків пальців. На сьогоднішній день дактилоскопічний метод є найбільш поширеним методом аутентифікації та ідентифікації. Він використовується для фізичного розмежування доступу. Проте, надійність дактилоскопічного методу залишається не доведеною - припущення про унікальність відбитків пальця не має належного наукового обґрунтування, і немає оцінки достовірності для впізнання відбитків. Насправді ж достовірність приймається рівною 100 %.

Існує два основних методи обробки образів відбитків.

Перший метод використовує кореляційне порівняння. При використанні цього методу відбиток пальця порівнюється з кожним еталоном із бази даних відбитків, обчислюючи різницю пікселів між вхідним та еталонним відбитками. Основною перевагою цього методу є те, що він працює з відбитками низької якості. Проте недоліками є великі вимоги до обсягу пам'яті для зберігання бази даних, бо кожен відбиток вимагає значної кількості пам'яті, і низька швидкість алгоритму. Це через те, що під час сканування палець може бути прикладений під різними кутами та трохи зміщений, що потребує багато ітерацій для порівняння, особливо при ідентифікації.

Другий метод використовує ключові точки, такі як мінуці. Процедура цього методу включає створення шаблону на основі відбитка пальця, виділення особливих точок, таких як кінцеві точки та точки розгалуження, і їх подальше порівняння з точками на вхідному зображенні відбитка. Рішення про ідентичність образів приймається на основі кількості збігаючихся точок. Перевагою цього методу є його швидкість. Алгоритми цього типу є найбільш поширеними. У цьому дослідженні розглядаються саме алгоритми порівняння за допомогою особливих точок.

Алгоритм порівняння відбитків за особливими точками вимагає відображень високої якості з мінімальними пошкодженнями. Це свідчить про

те, що для покращення якості відбитків пальця застосовуються спеціальні алгоритми обробки зображень.

Для вирішення математичної задачі, визначеної в роботі, необхідно встановити діапазон значень, в якому буде проводитися підбір параметрів та визначити кількість кроків підбору, позначену як N.

Нехай δmax_{min}- Вибраний діапазон значень, тоді $N = \frac{\delta_{max} - \delta_{min}}{N}$ - Величина кроку.

Значенняδна кроці і визначається формулою:

$$\delta_i = \delta \frac{i}{N} (\delta min_{max})_{min}, i = \overline{1, N}. \quad (1.1)$$

Необхідно знайти таке і, за якого $f(\delta) = a_1 P_1(\delta) + a_2 P_2(\delta) \underset{\delta}{\longrightarrow} min.$

Візьмем δmax_{min}, N=25, довжина кроку $N = \frac{\delta_{max} - \delta_{min}}{N} = 0.05.$

Результати підрахунків свідчать про те, що значення функції $f(\delta)$ мінімально при $\delta = 1.06.$

На рис. 1.2 проілюстровано роботу алгоритму за різних значень параметраδ.

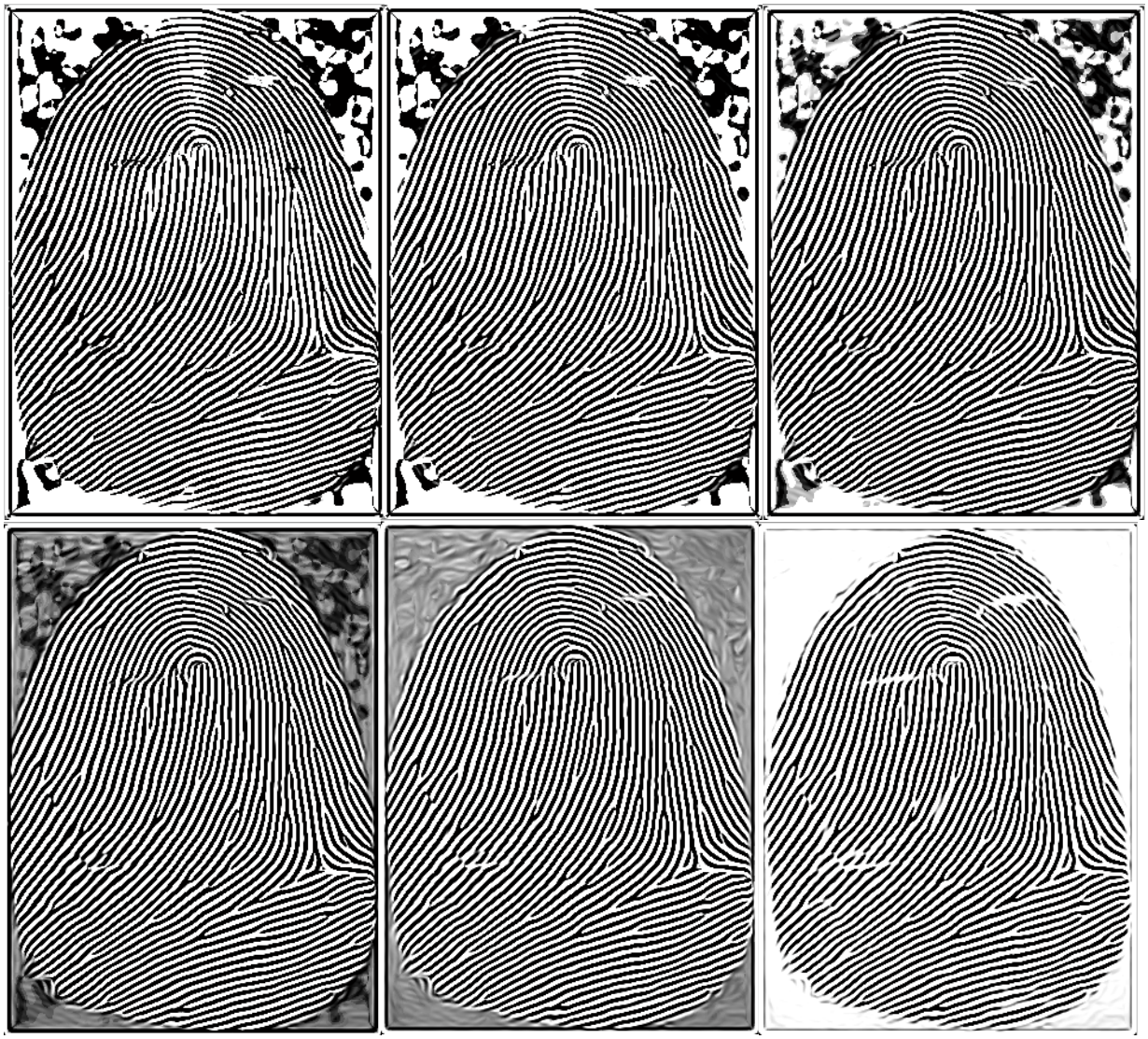


Рисунок 1.2 - Застосування фільтра Габора для обробки зображень з різними значеннями δ.

1.2 Аналіз основ дактилоскопії та процесу збору дактилоскопічних зразків

Розглянемо важливі, ключові характеристики під час обробки дактилоскопічних зображень.

1) Масштаб і орієнтація:

Під час подання дактилоскопічних зображень важливо враховувати їхній масштаб. Зображення може містити об'єкти різних розмірів, і, отже, обробка повинна дозволяти аналізувати їх на різних масштабах. Для двовимірного сигналу, зображення розкладається на різні спектральні області відповідно до їхнього масштабу і орієнтації. Орієнтація базових функцій визначає їхню здатність до коректного аналізу орієнтованих структур, які часто зустрічаються у дактилоскопії, такі як контури та лінії. Таким чином, для успішного аналізу важливо мати перетворення, яке розкладає вхідний сигнал на локальні частотні області.

2) Просторова локалізація:

Окрім частотної локалізації, базові функції також повинні бути локальними у просторі. Це особливо важливо, коли інформація про місцезнаходження деталей на зображенні має велике значення. Просторова локалізація повинна бути абсолютною, подібно до дискретного косинусного перетворення (ДКП), оскільки це допомагає зберегти властивість локальності у частотній області.

Найбільш часто застосовуваний підхід при аналізі полягає в наступному: сигнал дискретизується, потім виконується ДПФ. Що ж виходить у результаті? Спочатку сигнал розкладається по базису одиничного імпульсу, який не має частотної локальності, а потім по базису синусоїд з парними та непарними фазами, що не мають просторової локальності. Звичайно, уявлення сигналу в частотній ділянці виключно важливе для його аналізу. Однак це не означає, що вибір функцій імпульсу та синусоїди для вирішення цього завдання є найкращим. Ще 1946 року Д.Габор запропонував клас лінійних перетворень, які забезпечують локальність і частотної, й у

часовій області. Базис одиничного імпульсу та базис синусоїди можуть розглядатися як два екстремальні випадки цих перетворень.

3) Ортогональність.

Взагалі, перетворення не обов'язково має бути ортогональним. Так, ортогональність зазвичай не розглядається в контексті смугового кодування, хоча вейвлет зазвичай є ортогональним. Ортогональність функцій спрощує багато обчислень.

Крім того, як буде показано, "сильне" неортогональне перетворення може бути неприйнятним для кодування.

4) Швидкі алгоритми обчислення.

Це, мабуть, найважливіша властивість. Тому що неможливість практичної реалізації перетворення в реальному масштабі часу зводить нанівець усі його позитивні властивості.

Лінійні перетворення кінцевих сигналів

Система фільтрів аналізу-синтезу

Розглянемо лінійні перетворення сигналів кінцевого розміру, які можуть бути виражені в термінах згортки з КІХ - фільтрами. На рис.1.3 показано систему, яка здійснює таке перетворення з допомогою банків фільтрів аналізу – синтезу. Позначення малюнку є стандартними для цифрової обробки сигналів.

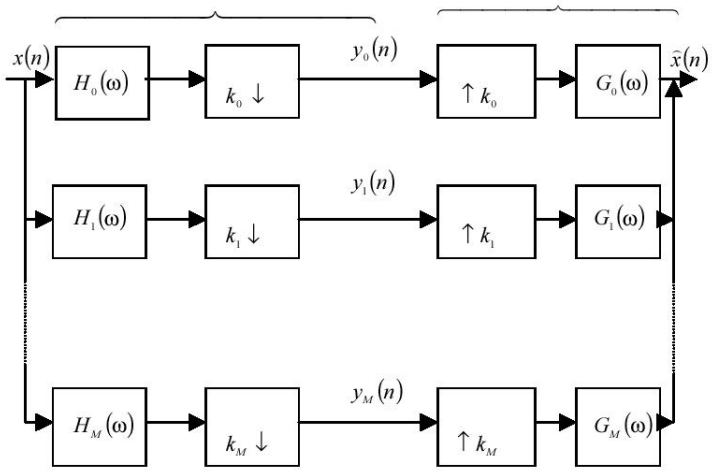


Рисунок 1.3 - Банк фільтрів аналізу-синтезу

|*Hi(ω)*| означає операцію кругової згортки вхідного сигналу довжиною N з імпульсною характеристикою КІХ – фільтра $h_i(n)$ а Фур'є – перетворення результату:

$$H_i(\omega) = \sum_n h_i(n) \cdot e^{-j\omega n} \tag{1.2}$$

Природно передбачається, що довжина фільтра менше розміру зображення. Блоки $k_i \downarrow$ означають децимацію в k_i разів, блоки $\uparrow k_i$ - інтерполяцію у k_i разів. Нагадаємо, що децимація означає залишення лише кожного k_i відліку, інтерполяція означає вставку k_i - 1 нулів між цими відліками. Передбачається, що k_i - цілі числа і ділять N.

Будемо називати таку систему системою А – С. Секція аналізу системи А – С виконує лінійне перетворення над вхідним сигналом $x(n)$ довжиною n. В результаті виходить М послідовностей $y_i(i)$ довжиною $\frac{N}{k_i}$. Операції, що виконуються секцією синтезу, є зворотними операціями секції аналізу. В результаті виходить сигнал $\widehat{x}(n)$. Так само будується система А – С і багатовимірного сигналу.

Таким чином, коефіцієнти перетворення обчислюються через пакунок. Інтуїтивно зрозуміло, що це добре, оскільки різні ділянки сигналу оброблятимуться однаковим чином. Далі формулювання проблеми в частотній області дозволяє легко розділити помилку реконструкції. $\varepsilon(n) = \widehat{x}(n) - x(n)$ на дві частини: елайзингову складову та складову, інваріантну до зсуву. Для цього запишемо вихідний сигнал схеми аналізу в частотній ділянці:

$$Y_i(\omega) = \frac{1}{k}\sum_{j=0}^{k-1} H_i\left(\frac{\omega}{k} + \frac{2\pi j}{k}\right) X\left(\frac{\omega}{k} + \frac{2\pi j}{k}\right) \tag{1.3}$$

Тоді вихідний сигнал схеми A - C:

$$\widehat{X}(\omega) = \sum_{i=0}^{M-1} Y_i(k\omega)\, G_i(\omega) \tag{1.4}$$

з урахуванням ефекту інтерполяції та децимації у частотній області. Об'єднуючи вирази (1.2) та (1.3), отримуємо:

$$\widehat{X}(\omega) = \sum_{i=0}^{M-1} \left[X\left(\omega + \frac{2\pi j}{k}\right)\right] G_i(\omega) =$$

$$= \frac{1}{k}\sum_{i=0}^{M-1} H_i(k\omega)\, G_i(\omega)X(\omega) + \frac{1}{k}\sum_{j=0}^{k-1} X_i\left(\omega + \frac{2\pi j}{k}\right)\sum_{i=0}^{M-1} X_i\left(\omega + \frac{2\pi j}{k}\right) G_i(\omega) \tag{1.5}$$

Тут перший доданок відповідає відгуку лінійної незалежної від часу системи, а другий відповідає елайзингу системи.

Каскадне з'єднання систем A – C

Перевагою запровадження поняття систем A – C і те, що дозволяє наочно уявити і проаналізувати ієрархічно побудовані перетворення. Припустимо, що система A – C задовольняє вимогу повного відновлення. Проміжний сигнал цієї системи може подаватися на вхід іншої системи A - C, в результаті чого виходить ієрархічна каскадно-з'єднана система. Приклад такої системи показаний на рис. 1.4, де система A - C застосовується повторно до свого проміжного сигналу.

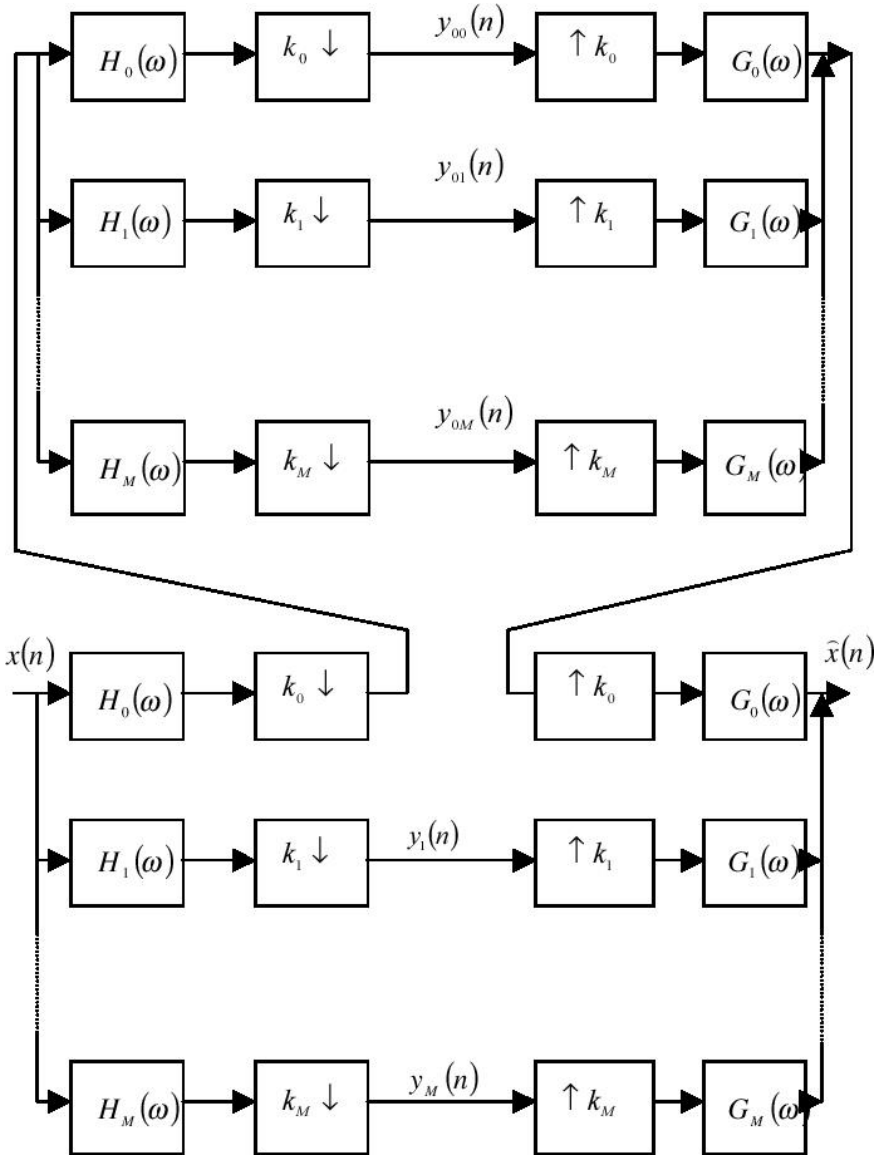


Рисунок 1.4 - Нерівномірний каскадний банк фільтрів аналізу-синтезу

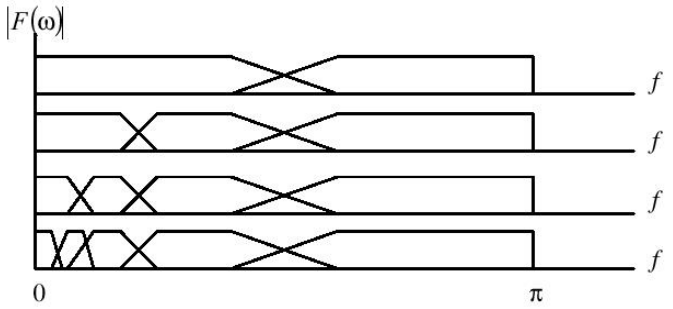


Рисунок 1.5 - Октавосмугове розбиття частотного плану

Якщо початкова система А - С мала властивість повного відновлення, то і двокаскадна система, що вийшла, також буде мати цю властивість. Якщо у подальшому розкладанню піддається кожен проміжний сигнал, така система називається рівномірною системою. В іншому випадку ми маємо справу з нерівномірною або пірамідальною системою, як показано на рис. 1.6. Таке каскадування призводить до октавосмугового розбиття частотного плану, як показано на ідеалізованій частотній діаграмі (рис. 1.5). Тут на верхній діаграмі показано розбиття частотного плану двоканальною системою А - С. Наступні діаграми демонструють послідовне повторення застосування тієї ж системи до низькочастотної частини сигналу. Таким чином, нижня діаграма відповідає чотирирівневому розбиттю частотної області.

Базисні функції перетворення повинні бути локалізовані як у просторовій (тимчасовій), так і частотній областях.

Одне з вирішення цієї проблеми було запропоновано Д.Габором. Габор представив перетворення, в якому базовими функціями є синусоїди, модульовані вікном Гауса. Перетворення Габора можна розглядати як виконання локалізованої частотної декомпозиції в ряд вікон, що перекриваються. Базисні функції Габора локалізовані за частотою та часом. Габор показав, що ці функції є оптимальними з погляду локалізації щодо обраного ним заходу. (Пізніше було доведено, що вибір іншого заходу веде до інших оптимальних функцій). Перші п'ять базисних функцій перетворення

Габора разом із спектрами показані на рис. 1.6. Як самі базисні функції, та їх спектр є гладкими і компактними. Функції Габора можна перенести у двовимірний випадок. Вони можуть бути використані для стискання зображення.

Головний недолік перетворення Габора полягає у неортогональності базисних функцій (тобто функції аналізу докорінно відрізняються від функцій синтезу). Функції аналізу перетворення Габора є погано обумовленими як у просторовій, так і у частотної областях. Це призводить до поширення помилок квантування коефіцієнтів по всій частотній та просторовій областях, незважаючи на те, що значення коефіцієнтів обчислювалися для локальної області. Цікаво відзначити, що локалізація базисних функцій Габора може бути значно покращена, якщо використовувати надмірну виставу. Воно виконується шляхом частішого, ніж потрібно, накладання вікна Гауса або шляхом розподілу на частини кожного частотного вікна. Однак це призводить до збільшення числа коефіцієнтів, що не застосовується для кодування.

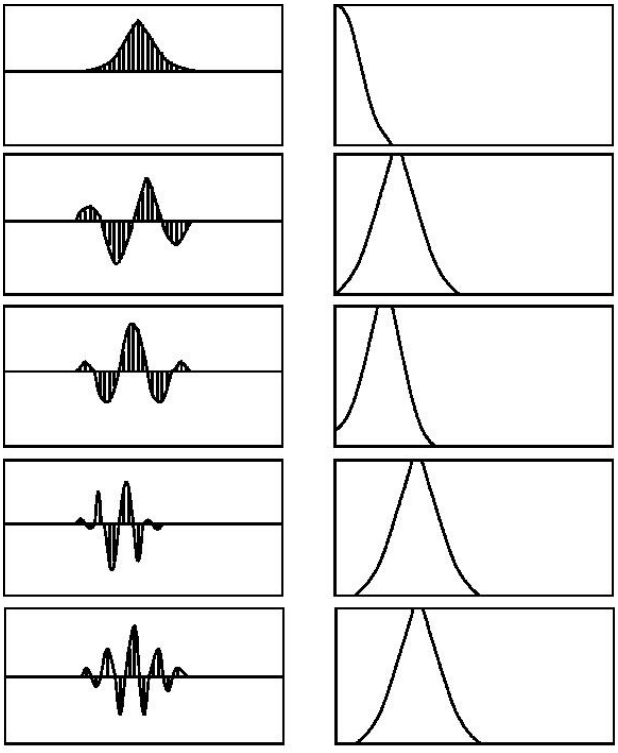


Рисунок 1.6 - Базисні функції Габора з відповідними спектрами Фур’є

1.3 Діаграми варіантів використання програмної системи

Додаток являє собою програмний засіб для обробки запитів одного користувача, тому основним елементів діаграми прецедентів є тільки він. Інші можливі режими роботи представлені на рис.1.7.

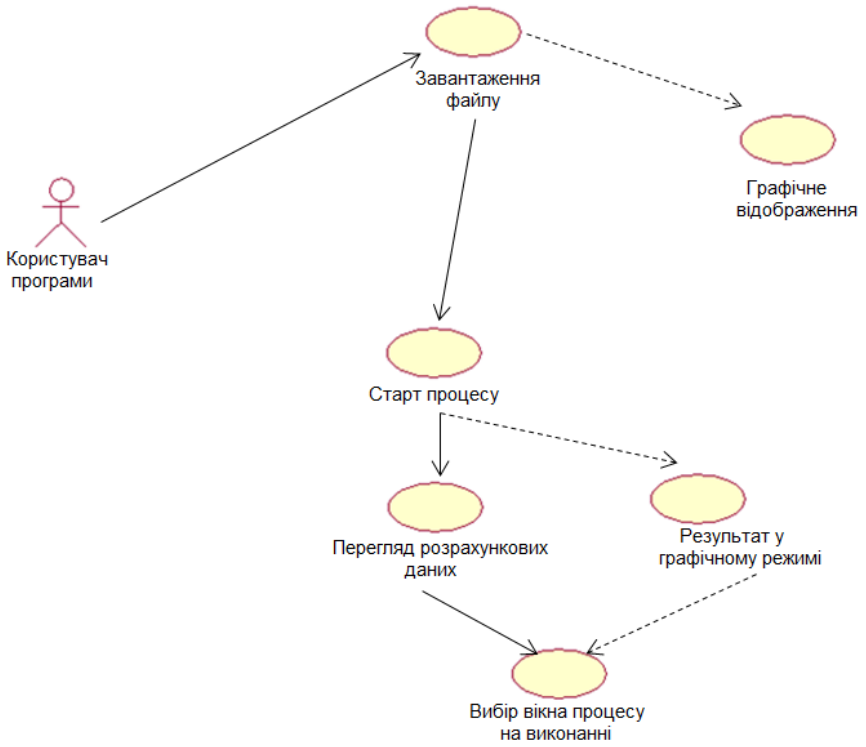


Рисунок 1.7 - UML-схема системи розділення дактилоскопічної візуальної інформації

1.5 Діаграми діяльності програмного засобу

Концептуальна схема являє собою узагальнені функціональні та інформаційні компоненти проєктованого програмного продукту, принципи їхньої взаємодії між собою, з користувачем та зовнішнім середовищем. За результатами вищеописаної роботи було спроектовано концептуальну схему системи постановки процесів на виконання та структуризації оперативної пам'яті, яку показано на рис. 1.8, де показано дві концептуальні частини - компонент оброблення даних, що надійшли в систему, та компонент аналізу ідентифікованих характеристик.

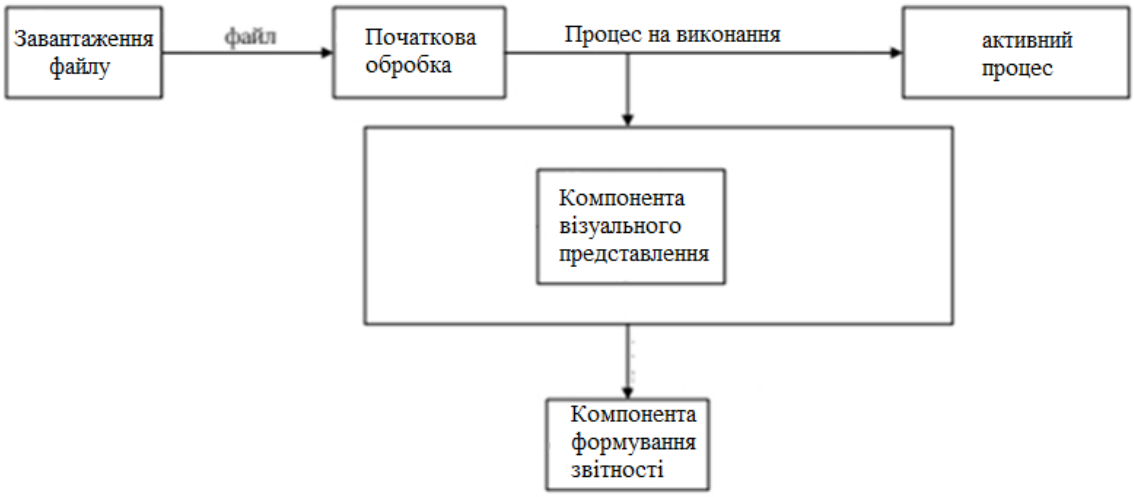


Рисунок 1.8 - Концептуальна схема програмної системи розпізнавання

На моделі конфігурації перцептрону можна спостерігати, що кожен нейрон у першому прошарку розділяє площину x-y на дві частини: одна частина забезпечує вихідну одиницю для вхідних даних нижче верхньої лінії, а інша - для вхідних даних вище нижньої лінії. Результатом цього подвійного розділення є вихідний сигнал нейрона другого прошарку, який дорівнює одиниці лише всередині V-подібної області. У другому прошарку також можуть бути використані три нейрони для подальшого розділення площини і створення області трикутної форми. Шляхом додавання достатньої кількості нейронів у вхідний прошарок, можливо створити опуклий многокутник будь-якої заданої форми.

Припустимо, що вектор X визначає розпізнаваний образ на демонстраційній карті. Кожен компонент (квадрат) вектора X помножується на частину вектора V, і сума цих множень визначає вихід Y нейрона. Коли сума перетинає певний кордон, то вихід Y встановлюється на одиницю, в протилежному випадку - залишається нульовим. Це може бути ефективно записано у векторному вигляді, після чого використовується порогова функція.

Під час навчання мережі, образ X подається на вхід, і вихід Y обчислюється. Коли Y коректний - нічого не змінюється. Проте, коли навпаки, що приєднані до входів, які призводять до помилкового результату, змінюються, щоб зменшити похибку.

Для легшого розуміння цього процесу, давайте розглянемо ситуацію, коли на вхід подається демонстраційна карта з зображенням цифри 3, і вихід Y дорівнює 1 (вказуючи на непарність). Так, як це є правильною відповіддю, ваги залишаються без змін. Проте, якщо карта з номером 4 і вихід Y дорівнює одиниці (неправильний результат), то ваги, пов'язані з входами, які призвели до помилки, змінюються з метою зменшення цієї похибки. Також, коли третя видає нульовий вихід, ваги, що пов'язані з входами, які спричинили цю помилку, змінюються для корекції.

1.6 Методи та засоби розв'язання задачі скелетизації дактилоскопічних зображень

Якщо, вихідні дані є дискретним набором точок з неоднозначним X , то маємо такі розкладання на гармоніки.

Функція:

$$f(t) = A\cos\left(\frac{2\pi tm}{T+\varphi}\right) 2f''(t)$$

$$= A\cos\left(\frac{2\pi t(Nm)}{T-\varphi}\right) i\, f'(t) + f''(t) = \left(\frac{A}{2}\right)\cos(\frac{2\pi tm}{T+\varphi})$$

$$+\left(\frac{A}{2}\right)\cos\left(\frac{2\pi tm}{T-\varphi}\right)$$

Підтвердження дзеркального ефекту.

У результаті ДПФ гармонійної функції практично виходять дві гармоніки. Однак цей емпіричний факт не доводився. Доведемо тепер суворо, які гармоніки дає довільна гармонійна функція:

$f(t) = \mathrm{A}\cos(\frac{2\pi Tm}{T+\varphi})$ за цілого $m \in [0, N]$.

Нагадаємо формулу прямого ДПФ:

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{-j2\pi kn}{N}}$$

В даному випадку

$$x_n = f(t_n) = f\left(\frac{T_n}{N}\right) = A\cos\left(\frac{2\pi T_n m}{NT + \varphi}\right) = A\cos(\frac{2\pi nm}{N + \varphi})$$

Введемо позначення:

$$w_n = \frac{2\pi n}{N}$$

$$Z_{k,}n = \left(\frac{f(t^n)}{A}\right)\frac{e^{-j2\pi kn}}{N} = \cos(w^n m + \varphi)\, e^{-j2\pi kn}$$

В результаті формула прямого ДПФ спрощується до:

$$X_k = A\sum_{n=0}^{N-1} Zk, n$$

Тепер перетворимо *Zk, n*:

$$Z_{k,}\mathrm{n} = \cos(w^n m + \varphi)\, e^{-j2\pi kn}$$

Застосовуємо формулу Ейлера:

$$\cos\,(w^n m + \varphi)(\cos(-w^n k) + j\sin(w^n k) = \cos(w^n m + \varphi)(\cos(w^n k) - j\sin(w^n k))$$

Розкриваємо дужки:

$$\cos(w^n m + \varphi)\cos(w^n k) - j\cos(w^n m + \varphi)\sin(w^n k) =$$

$$\left(\frac{1}{2}\right)[\cos\big((w^n m + \varphi) - w^n k\big) + \cos\,((w^n m + \varphi) + w^n k)] = (\tfrac{1}{2})[\sin\,(w^n k - (w^n m + \varphi)) + \sin(w^n k + (w^n m + \varphi))] = (\tfrac{1}{2})[\cos\big((w^n m + \varphi) - w^n k\big) + \cos((w^n m + \varphi + w^n k) - j\sin\big(w^n k - (w^n m + \varphi)\big) - j\sin(w^n k + (w^n m + \omega))] = (\tfrac{1}{2})[\cos\big((w^n m + \varphi) - w^n k\big) + \cos\big((w^n m + \varphi) + w^n k\big) + j\sin\big((w^n m + \varphi) - w^n k\big) - j\sin((w^n + \varphi) + w^n k)] = (\tfrac{1}{2})[-w^n k) + j\sin\,(w^n m + \varphi - w^n k) + \cos\;\,(w^n m + \varphi + w^n k) - j\sin\,(w^n m + \varphi + w^n k)]$$

Застосовуємо формулу Ейлера (зворотній напрямок):

$$(\frac{1}{2})\,[e^{j}\,(w^{n}m\,+\,\varphi\,-\,w^{n}k)\,+\,\,e^{-j}\,(w^{n}m+\,\varphi\,+\,w^{n}k)]$$

Суми S1 та S2 є геометричними прогресіями, а формула суми геометричної прогресії нам відома:

$$SN\,=\,\frac{a^{0}(q^{N}\,-\,1)}{(q\,-\,1)},q\,\neq\,1$$

Перший елемент прогресії обох випадках дорівнює a0 = 1.

Знаменники прогресій рівні

$q1\,=\,e^{j2\pi\,\frac{(m-k)}{N}}$ *для S1*

$q2\,=\,e^{-j2\pi\,\frac{(m+k)}{N}}$ *для S2.*

Умова q ≠ 1 призводить до розв'язання рівняння: $e^{j2\pi(m - k) / N} = 1$, і $e^{-j2\pi(m + k) / N} = 1$,

Враховуючи Теорему 0 отримаємо, що умова q ≠ 1 не виконується при k = m для S1 і при k = (N - m) для S2.

У разі коли виконуються обидві умови: $k = m$ і $k = (N - m)$, тобто $k = m = N /2$ обидві суми не можна вважати за формулою геометричної прогресії.

2 ВИБІР ТЕХНІЧНИХ РІШЕНЬ І ХАРАКТЕРИСТИК, АЛГОРИТМІВ ДЛЯ СТВОРЕННЯ ДОДАТКУ СКЕЛЕТИЗАЦІЇ ДАКТИЛОСКОПІЧНИХ ЗОБРАЖЕНЬ

2.1 Розробка адаптивного фільтра Габора для відокремлення основних орієнтацій ліній у дактилоскопічному зображенні

Зазвичай, зображення, отримане за допомогою сканера, характеризується низькою якістю. Багато факторів впливають на якість отриманого зображення, такі як сила натиску пальця під час сканування, вологість повітря, чистота пальця та самого сканера. З цією метою використовують різні методи фільтрації для поліпшення якості вихідного зображення.

На етапі скелетизації бінарне зображення відбитка пальця стає більш тонким і витонченим до формування скелета. Скелетом лінії вважається простий ланцюг (u, v) з вершинами u та v, суміжними, що проходить близько до геометричного центру лінії. Кожна вершина $p_1 \in (u, v)$ має рівно дві суміжні вершини p_2 і p_3, при цьому вершини p_2 і p_3 не є суміжними.

Після скелетизації виконується пошук мінуцій, тобто ключових точок. Більшість алгоритмів для розпізнавання використовує лише два типи мінуцій: закінчення та розгалуження. У скелеті зображення вони визначаються так:

Закінчення - це вершина p_1 скелета така, що для неї існує одна суміжна вершина p_2.

Розгалуження - це вершина скелета p_1, для якої існує рівно три суміжні вершини p_2, p_3 і p_4, які попарно не суміжні.

Наступним кроком є пошук мінуцій на зображенні, де для кожної ключової точки визначається її тип (закінчення або розгалуження), координати x і y, орієнтація.

Результати етапів фільтрації, скелетизації та пошуку мінуцій для образу відбитка пальця показані на рисунку 2.1.

а б в

Рисунок 2.1 –Зразки зображень на етапі після фільтрації скелет

Останнім кроком в процесі аутентифікації за відбитком пальця є порівняння отриманого зображення алгоритму з відповідним зразком із бази даних. Рішення про надання або відмову в доступі приймається на підставі аналізу всіх наявних даних з використанням вирішального правила.

Розглянемо декілька найпоширеніших алгоритмів обробки відбитків пальця, які використовуються в сучасних системах:

Фільтр згладжування:

Фільтр використовується для видалення шумів зі зображення загалом, а також видалення перешкод, пов'язаних із відбитком пальця. Суть його полягає в тому, що вікно розміром n × n проскановує всю область зображення, перелічуючи значення інтенсивності кожного пікселя.

$$I(i,j) = \frac{\sum_{(x,y)\in N} I(x,y)}{n^2}, \quad (2.1)$$

де$I(i,j)$ - новий показник інтенсивності пікселя, (i,j),$I(x,y)$ - вихідний показник інтенсивності.

(а) (б)

Рисунок 2.2 - Вхідне зображення та зображення після обробки згладжувальним фільтром.

Медіанний фільтр є ще одним широко використовуваним методом для видалення шумів у зображеннях. Зображення проскановується вікном розміром n × n, і значення інтенсивності пікселів всередині кожного вікна сортуються за порядком зростання (або зменшення). Значенням виходу є інтенсивність пікселя, який знаходиться в середині вікна.

Метод просторової фільтрації образу використовує фізичний процес поглинання та відображення світла. Робота алгоритму розпочинається з напівтонового зображення R з відтінками сірого. На початковому етапі алгоритму застосовується порогова обробка відбитка пальця, щоб отримати бінарне зображення.

$$R(i,j) = \begin{cases} 1, & якщо \;\; G(i,j) > R_0 \\ 0, & інакше \end{cases}, \qquad (2.2)$$

де$R(i,j)$ - Значення пікселя з координатами(i,j) у бінарному зображенні.

2) Напівтонове зображення, що має форму n × n, аналізується для центрального пікселя вікна, який знаходиться на координатах (i, j). Обчислюється коефіцієнт відображення, який представляє собою відношення кількості білих пікселів, що потрапили в дане вікно N(i, j), до розмірності цього вікна:

$$k(i,j) = \frac{N(i,j)}{n^2}, \tag{2.3}$$

де$k(i,j)$ – коефіцієнт відображення пікселя з координатами(i,j).

3) На третьому кроці визначається інтенсивність для кожного пікселя. Ця інтенсивність визначається як добуток коефіцієнта відбиття та максимальної інтенсивності світла:

$$I(i,j) = k(i,j) \cdot I_{max}, \tag{2.4}$$

де I_{max}- максимальне значення інтенсивності

Обробка зображення із застосуванням фільтра Габора

Обробка образу відбитка пальця даним алгоритмом здійснюється у кілька етапів:

1) Нормалізація зображення необхідна для визначення попередніх середніх значень та стандартних відхилень. Зображення після нормалізації, позначене як G, є квадратним зображенням розміром N×N, G(i, j) представляє значення нормалізованої яскравості пікселя. Для розрахунку нормалізованого зображення використовуються середнє та середньоквадратичне відхилення вихідного зображення.:

$$G(i,j) = \begin{cases} M_0 + \sqrt{\dfrac{VAR_0 \cdot (I(i,j) - M)^2}{VAR}}, & якщо \quad I(i,j) > M \\ M_0 - \sqrt{\dfrac{VAR_0 \cdot (I(i,j) - M)^2}{VAR}}, & в\ іншому\ випадку \end{cases}, \tag{2.5}$$

деM_0іVAR_0- задані значення середнього та середньоквадратичного відхилення відповідно,MіVAR - вихідні значення середнього та середньоквадратичного відхилення, обчислюються за формулами:

$$M = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} I(i,j), \tag{2.6}$$

$$VAR = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(I(i,j) - M)^2, \quad (2.7)$$

2) Використовуючи нормалізоване зображення визначається орієнтаційне зображення, яке є квадратним розміром N×N, де O(i, j) вказує на локальну орієнтацію (кут нахилу) виступу в пікселі з координатами (i, j):

$$O(i,j) = \frac{1}{2}arctg(\frac{d_x^2(i,j)d_y^2(i,j)}{2d_x(i,j)d_y(i,j)}), \quad (2.8)$$

де$d_x(i,j)$ і$d_y(i,j)$ - градієнти пікселя з координатами(i,j)

3) З використанням нормалізовано зображення обчислюється частотне представлення. Частотне зображення F представляє собою N×N зображення, де F(i, j) вказує на локальну частоту виступу, що визначається як частота папілярних ліній образу відбитка пальця. У випадку, коли через особливості візерунка неможливо визначити частоту, його частота обчислюється як середнє значення частоти сусідніх блоків.

Нехай λ – кількість пікселів між двома сусідніми вершинами гребенів у блоку розмірністю W×W, центром якого є піксель з координатами (i, j).

$$F(i,j) = \frac{1}{\lambda}. \quad (2.9)$$

4) Застосування бінаризації до зображення визначає зображення R розмірності N×N, де R(i, j) вказує категорію пікселя (i, j) як піксель западини або гребеня.

$$R(i,j) = \begin{cases} 1, & якщо \;\; G(i,j) > R_0 \\ 0, & інакше \end{cases}, \quad (2.10)$$

5) Використання фільтрів Габора, налаштованих локальну орієнтацію виступів; застосовується до нормалізованого вхідного зображення:

$$H(i,j{:}\,\phi,f) = exp\left(-\frac{1}{2}\left(\frac{x_\phi^2}{\delta_x^2} + \frac{y_\phi^2}{\delta_y^2}\right)\right) \cdot cos(\,2\pi f x_\phi), \quad (2.11)$$

Де $x_\phi^2 = i \cdot cos\,\phi + j \cdot sin\,\phi; y_\phi^2 = -i \cdot sin\,\phi + j \cdot cos\,\phi;$

ϕ - орієнтація фільтра Габора,

f – частота синусоїдальної плоскої хвилі,

δ_xтаδ_y– Константи огинальної Гауса визначаються в просторі вздовж вісей x і y відповідно. Ці значення встановлюються і коригуються на підставі емпіричних даних про функціонування алгоритму.

(а) (б)

Рисунок 2.3 – Вхідне зображення та результат обробки після фільтра

2.2 Розробка алгоритму скелетизації на основі адаптивного фільтра Габора

Нехай у нас є функціяf, тоді перетворення Фур'є для цієї функції виглядає так:

$$\hat{f}(\omega) = \int_R e^{-i\omega t} f(t) dt \tag{2.12}$$

Нехай задано функцію:

$$g_\alpha(t) = \frac{1}{2\sqrt{\pi\alpha}} e^{\frac{-t^2}{4\alpha}}, \tag{2.13}$$

$$(\Gamma^\alpha f)(\omega, b) = \int_R \left(e^{-i\omega t} f(t)\right) g_\alpha(t-b) dt \tag{2.14}$$

Для побудови одновимірного фільтра Габора застосовується формула:

$$G(x) = exp\left(-\frac{x^2}{2\sigma^2}\right) \cdot cos(2\pi\theta x), \tag{2.15}$$

деσ - стандартне відхилення гаусового ядра, що визначає амплітуду функції. Чим більше це значення, тим більше пологий вид набуде функція, і навпаки, що менше значення, то гострий пік вийде внаслідок побудови графіка функції.θ - Частота коливань, що визначається як:

$$\theta = \frac{1}{T}, \tag{2.16}$$

де T – період функції косинуса.

На рисунку 2.4 зображена функція фільтра Габора.

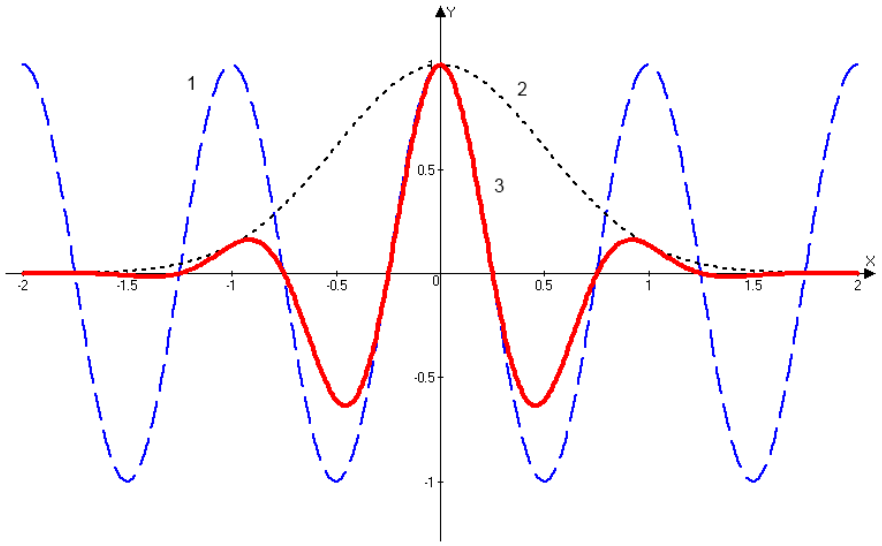


Рисунок 2.4 - Ілюстрація фільтра Габора.

2.3 Дослідження та вибір оптимальних параметрів адаптивного фільтра Габора для досягнення максимальної точності та ефективності скелетизації дактилоскопічних зображень

Перевага даного методу полягає у найвищій якості обробки зображень відбитків пальців порівняно зі сучасними фільтрами. Однак недоліком є високі вимоги до обчислювальної потужності всіх алгоритмів. З огляду на завдання вибору алгоритму, що забезпечує високу якість фільтрації та збереження ключової інформації, раціональним вибором є алгоритм, оснований на фільтрах Габора.

Це обумовлено тим, що він відповідає цим критеріям краще, ніж інші доступні методи.Формула функції Габора виглядає так:

$$g(x,y) = s(x,y) \cdot w_r(x,y), \tag{2.17}$$

Можна уявити синусоїду як дві дійсні функції, розташовані в дійсній та уявній частині комплексної функції (рис. 2.5).

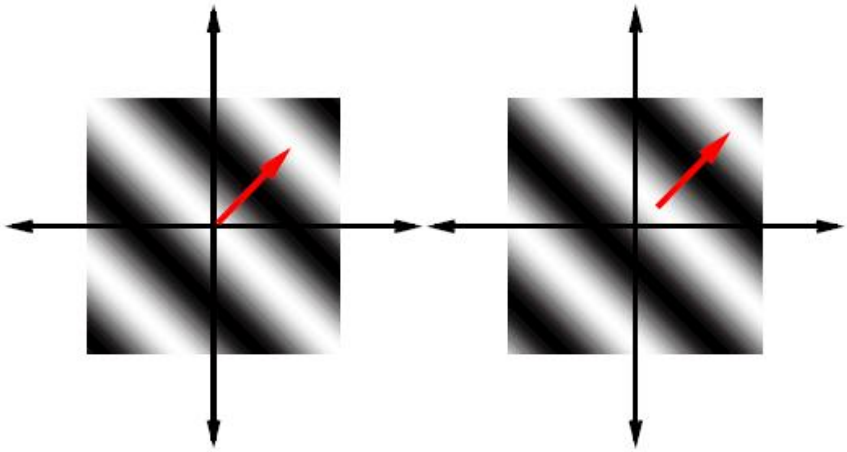


Рисунок 2.5 - Частина синусоїди

Форма синусоїди представлена її дійсною та уявною частинами:

$$Re\big(s(x,y)\big) = cos(2\pi(u_0 x + v_0 y) + P)$$
$$Im\big(s(x,y)\big) = sin(2\pi(u_0 x + v_0 y) + P) \tag{2.19}$$

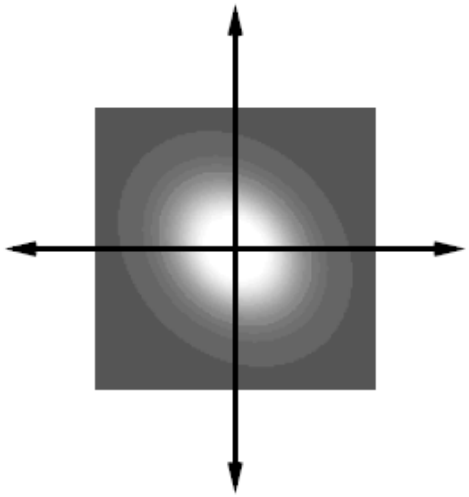


Рисунок 2.6 - Огинаюча Гауса

Кожна комплексна функція Габора складається з двох частин, розташованих у дійсній та уявній частині функції (рис. 2.7).

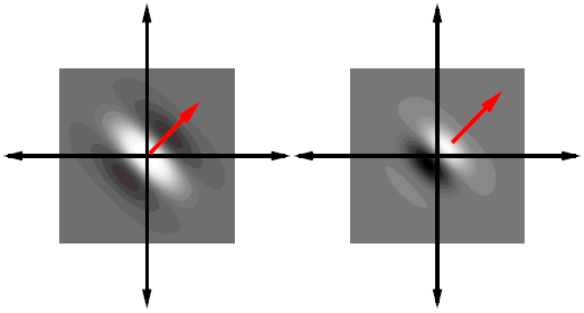


Рисунок 2.7 - Функція Габора.

2.4 Розробка методів попередньої обробки зображень для зменшення шуму та поліпшення якості зображень перед застосуванням адаптивного фільтра Габора

Опишемо практичну реалізацію всіх етапів цього методу.

а) Фільтрування зображення.

Фільтрування зображення здійснюють за допомогою цифрових фільтрів. На сучасному етапі немає необхідності займатися їх розробкою. Більшість із них реалізовано у вигляді доступних програмних бібліотек.

При виборі фільтра необхідно прагнути максимальної усунення шумів, і водночас, максимального збереження структури папілярних ліній відбитка. На рис. 2.8 представлено два відбитки пальця однієї людини.

а) б)

Рисунок 2.8 - Відбитки пальця однієї людини

На рис. 2.9 представлені самі відбитки після обробки фільтром Гаусса з параметрами (7x7) (функція GaussianBlur(img, img, Size(7, 7), 0), бібліотека OpenCV)

а) б)

Рисунок 2.9 - Відбитки пальця однієї людини після обробки фільтром Гауса

На рис. 2.10 представлені відбитки після обробки фільтром Гаусса з параметрами (7x7) і фільтрації адаптивним фільтром Гаусса (функція adaptiveThreshold(img, img, 255, ADAPTIVE_THRESH_GAUSSIAN_C, CV_THRESH_BINARY, 15, 45). також бінаризацію зображення.

а) б)

Рисунок 2.10 - Відбитки пальця однієї людини після обробки фільтром Гауса та адаптивним фільтром Гауса

Як видно з рисунку 2.10, шуми між папілярними лініями вдалося забрати, але самі папілярні лінії порвані на безліч дрібних частин. При подальшій скелетизації методом Габора це призведе до утворення безлічі помилкових точок закінчення.

На рис. 2.11 представлені відбитки після обробки фільтром Габора fingerprint_enhancer, функція fingerprint_enhancer.enhance_Fingerprint(img). Зображення інвертовано.

а) б)

Рисунок 2.11 - Відбитки пальця однієї людини після обробки фільтром Габора

З рис. 2.11 видно, що після обробки фільтром Габора вдалося не тільки прибрати шуми, а й зберегти цілими більшість папілярних ліній. Структура відбитка чітко видно.

Таким чином, для оптимального вибору алгоритму фільтрації, застосовувався фільтр Габора.

б) Бінаризація зображення глобальним способом.

Для реалізації бінаризації зображення глобальним способом необхідно задати поріг перетворення. Оскільки фон зображення може змінюватись у широких межах, то вибір порога стає непростим завданням. Неправильно вибраний поріг може призвести до суттєвих втрат зображень папілярних ліній або повної втрати зображення. Для того щоб не втратити зображення зовсім, потрібно знайти значення яскравості найсвітлішого пікселя і найтемнішого. Поріг слід вибирати в діапазоні між ними.

Бінаризація зображень реалізується за допомогою функції threshold(img, img, 127, 255, cv::THRESH_BINARY), де 127 поріг бінаризації, бібліотека OpenCV.

в) Скелетизація зображення.

Метод Габора не вимагає вхідних параметрів і є одним із найпростіших. Алгоритм є багатопроходовим (обробляє зображення поки

не виключить пікселі, що підлягають зміни). Внаслідок цього час обробки досить великий.

Опишемо алгоритм.

1) Створимо матрицю байтів такого ж розміру як зображення відбитка. Проініціалізуємо нулями. У ній фіксуватимемо пікселі, які змінювали, встановлюючи 1;

2) Виставимо прапор закінчення алгоритму finish = true;

3) Введемо наступну матрицю;

P9	P2	P3
P8	P1	P4
P7	P6	P5

4) Піксель змінює свій колір із білого на чорний, якщо виконується ряд умов:

а) 2 <= P2+P3+...+P8+P9 <=6

*б) P2 * P4 * P6 = 0*

*в) P4 * P6 * P8 = 0*

Для роботи з крайніми лівими пікселями в) та г) пункти необхідно замінити на:

*г) P2 * P4 * P8 = 0*

*д) P2 * P6 * P8 = 0*

Алгоритм скелетизації відображено на рис. 2.12.

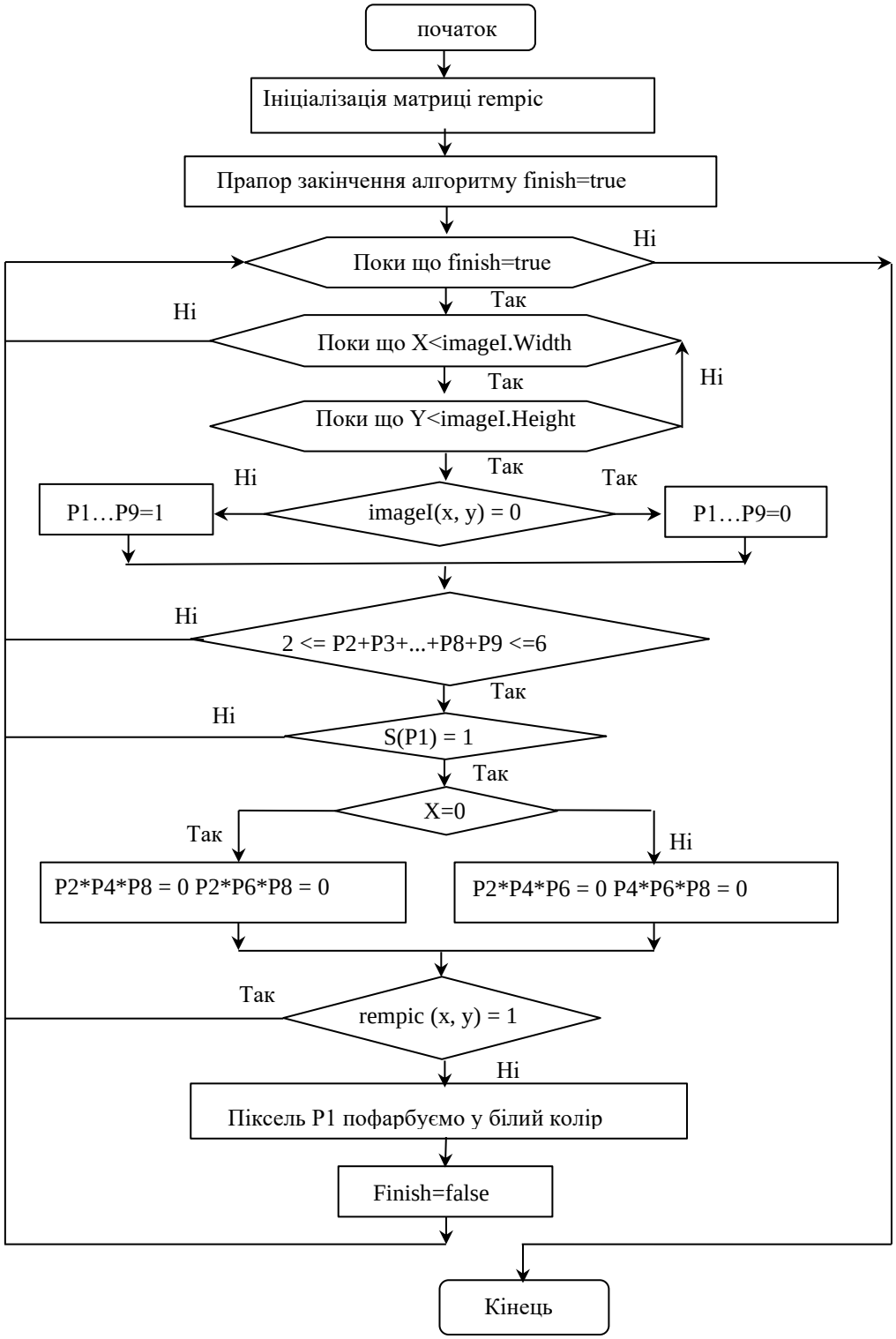


Рисунок 2.12 - Алгоритм скелетизації

На рис. 2.13 представлені результати скелетизації відбитків пальців після обробки фільтром Габора.

Рисунок 2.13 - Результати скелетизації відбитків

В результаті обробки зображення алгоритмом папілярні лінії стали завтовшки 1 піксель, при цьому не порвані і структура відбитків збережена.

2.5 Обґрунтування проєктних рішень щодо ТЗ

На вибір комплексу технічних засобів впливає велика кількість чинників, а саме:

– фактори, що впливають на вибір ПК: продуктивність процесора, потужність відеокарти, об'єм оперативної пам'яті, об'єм жорсткого диска, можливість підключення пристроїв введення - виведення, підтримувана операційна система, сумісність роботи обладнання різних типів;

– вартісні фактори: вартість і обслуговування устаткування, кількість обслуговуючого персоналу;

– фактори, що впливають на габарити пристроїв: площа, необхідна для розміщення устаткування;

– чинники, пов'язані обробкою інформації: обсяг вхідної та вихідної інформації, яким чином вхідна інформація буде надходити на комп'ютер;

– фактори, пов'язані з ергономічністю і комфортністю: якісний для роботи монітор, зручність користування клавіатурою і мишкою.

Наведемо рекомендовані вимоги до апаратного забезпечення оператора, користувача інформаційної системи реалізації фільтру Габора:

Таблиця 2.1 - Склад рекомендованого апаратного забезпечення

	Рекомендовані вимоги
Операційна система	Windows10 64bit
процесор	Pentium4 3.0GHz
Оперативна пам'ять	2 GB RAM
графічна карта	GeForce 6600 GT / Radeon X1600Pro
Direct X	DirectX 9.0c і вище.
Місце на жорсткому диску	0,5 GB

2.6 Обґрунтування проєктних рішень з ІЗ

Сучасний світ насичений великою кількістю зображень, що є важливою частиною інформаційного потоку. Обробка цих зображень стала актуальною задачею для багатьох галузей, включаючи медицину, комп'ютерне бачення, дизайн і багато інших. У цьому контексті технологія Visual Studio, розроблена компанією Microsoft, виявляється надзвичайно корисною та потужною платформою для створення програм для обробки зображень.

Перш за все, Visual Studio надає розробникам доступ до величезного набору інструментів і бібліотек, які спрощують процес обробки зображень. Вона підтримує мови програмування, такі як C++, C#, Python та інші, що дозволяє вибирати найбільш підходящий інструмент для конкретного завдання. Наприклад, для швидкого алгоритмічного аналізу можна використовувати C++, а для простої обробки зображень - Python.

Однією з ключових переваг Visual Studio є її інтегроване середовище розробки (IDE), яке забезпечує зручний редагування коду, налагодження і

відлагодження програм. Також, IDE має багатий набір інструментів для візуалізації зображень, що значно полегшує розробку програм для обробки графіки. Редагування коду відбувається відразу ж з можливістю перегляду змін на зображенні в реальному часі, що значно прискорює ітераційний процес розробки.

Крім того, Visual Studio надає доступ до розширень та бібліотек, які спеціалізуються на обробці зображень. Наприклад, бібліотека OpenCV, яка є однією з найпопулярніших у світі, може бути інтегрована в проекти Visual Studio для розв'язання різноманітних завдань обробки зображень, включаючи відслідковування об'єктів, розпізнавання облич, фільтрацію та аналіз.

У висновку, технологія Visual Studio є потужним інструментом для розробки програм для обробки зображень. Вона надає зручний інтерфейс розробки, доступ до багатьох інструментів та бібліотек, що робить її ідеальним вибором для розробників, які працюють у цій галузі. Завдяки Visual Studio, обробка зображень стає більш доступною та ефективною, що сприяє розвитку багатьох сфер діяльності.

2.7 Обґрунтування вибору програмних засобів

Однією з основних особливостей, що відрізняють Python від інших мов, є наявність великої кількості модулів і бібліотек, що вільно розповсюджуються, які значно розширюють функціональні можливості мови. Відповідно до завдання на ВКР було обрано такі модулі та бібліотеки:

- Numpy;
- SciPy;
- Scikit - learn;
- Matplotlib.

У багатьох галузях науково-дослідних і дослідно-конструкторських роботах потрібне просте у використанні програмне забезпечення, що має великі можливості. Для забезпечення цих потреб було створено SciPy –

екосистему відкритого програмного забезпечення для математики, наукових досліджень та інженерних робіт. SciPy як екосистема ПЗ складається з наступних основних пакетів:

- NumPy;
- SciPy (як бібліотека);
- Matplotlib;
- IPython;
- Sympy;
- Pandas.

У стек пакетів SciPy входить однойменна бібліотека, яка відповідає за обробку даних для наукових та інженерних розрахунків. У її можливості та функції входять:

- Константи (constants): фізичні константи та коефіцієнти для їх перерахунку;
- Кластеризація (cluster): алгоритми кластеризації (Векторне квантування, К-середні);
- Пакет fftpack: алгоритми дискретного перетворення Фур'є;
- Модуль «integrate»: інструменти для інтегрування;
- Інтерполяція (interpolate): інструменти для інтерполяції даних (рис. 2.13);
- Модуль «io»: введення та виведення різних даних;
- Модуль «lib»: інтерфейси для роботи із зовнішніми бібліотеками;
- Модуль лінійної алгебри "linalg": різні інструменти для роботи з лінійною алгеброю;
- Модуль «misc»: різні утиліти загального призначення;
- Модуль «ndimage»: різні функції для роботи з багатовимірними зображеннями;
- Алгоритми оптимізації («optimize»): різні алгоритми оптимізації, у тому числі й лінійне програмування;
- Обробка сигналу (signal): інструменти для обробки сигналів;

- Модуль «sparse»: Розряджені матриці та алгоритми для роботи з ними;

- Модуль «spatial»: просторові дані та алгоритми для роботи з ними, наприклад, вирішальні дерева найближчий сусід;

- Модуль «special»: спеціальні функції;

- Статистичні функції (stats);

- Модуль «weave»: інструменти для впровадження коду C/C++ у Python у вигляді багаторядкових коментарів

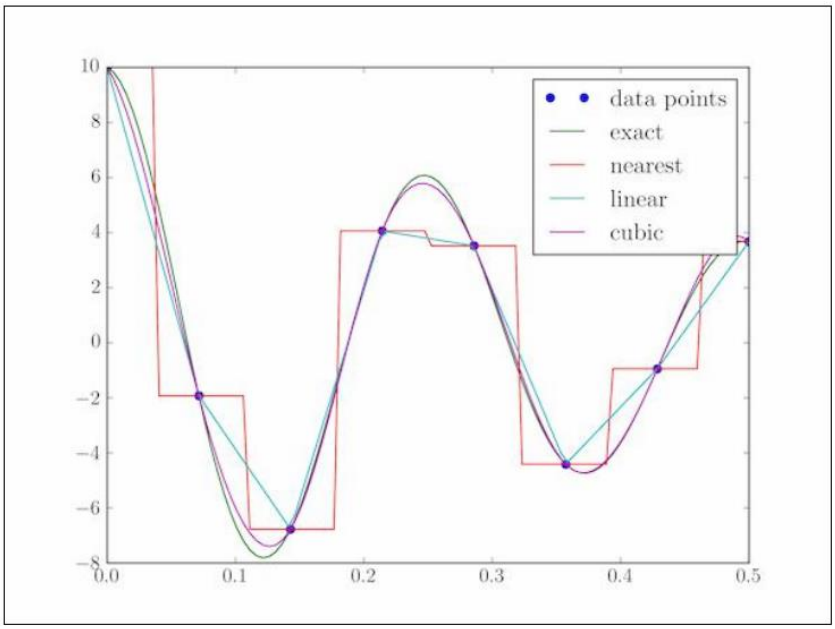


Рисунок 2.14 - Різні методи інтерполяції даних у scipy.interpolate

Для розширення можливостей екосистеми програмного забезпечення SciPy було створено набір додаткових пакетів під назвою SciKits (SciPy Toolkits). У набір входять різні модулі: інструменти для створення та роботи з нейронними мережами, машинне навчання, нечітка логіка, робота із зображеннями, розрахунок аеродинаміки, геодезія та багато інших. Для аналізу оброблених даних було вирішено використовувати модуль SciKit-learn, який включає засоби для роботи з великою кількістю даних та їх аналізу, машинне навчання та комп'ютерного бачення.

Нижче наведено список актуальних завдань у сфері обробки даних завдань, які SciKit-learn здатний вирішити:

- Кластеризація (Clustering): створення та присвоєння класів об'єктам на нерозмічених даних;

- Класифікація (Classification): визначення належності об'єкта до вже заданих класів (рисунок 2.14);

- Скорочення розмірності (Dimensionality Reduction): виділення найбільш значущих ознак і даних;

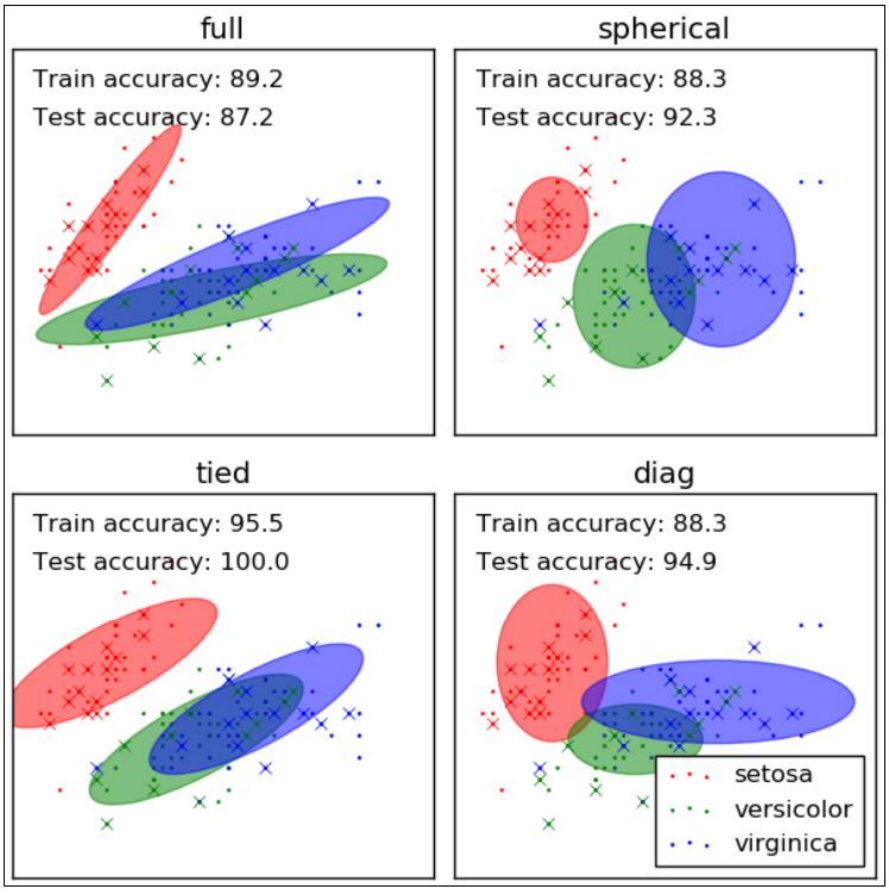


Рисунок 2.15 - Класифікація видів трьох видів ірису за допомогою чотирьох видів GMM

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ЗАСТОСУВАННЯ МЕТОДУ СКЕЛЕТИЗАЦІЇ ДАКТИЛОСКОПІЧНИХ ЗОБРАЖЕНЬ

3.1 Структура вхідної та вихідної інформації

Вхідними даними являється зображення. Позначимо зображення, яке ми отримуємо в результаті роботи нейронної мережі, як I SR, зображення низької роздільної здатності, яке необхідно збільшити в n раз, як I LR.

Зображення з високою роздільною здатністю (без обробки) позначимо як I HR. I LR виходить в результаті зменшення I HR в n раз.

В результаті вивчення необхідно отримати генеруючу функцію G. Генератор навчається як згорткова нейронна мережа прямого поширення з параметрами ΘG:

$$\theta_G = \{W_{1:L}; b_{1:L}\}, \quad (3.1)$$

де $W_{1:L}$ і $b_{1:L}$ – це, відповідно, ваги і зміщення нейронної мережі глибиною в L шарів. Цей параметр обчислюється шляхом оптимізації складовою функції втрат lSR. Для вивчення нейронної мережі знаходимо значення:

$$\theta_G = min\frac{1}{N} l_{SR}(G(I_n^{LR}), I_n^{HR}), \quad (3.2)$$

Функція втрат l_{SR} складається з наступних компонентів:

$$l_{SR} = l_{MSE} + 10^{-3} l_{GEN},$$

де l_{MSE} – середня квадратична помилка при порівнянні згенерованого зображення I SR з його оригіналом I HR;

l_{GEN} – змагальна функція втрат, яку можна представити наступною формулою:

$$l_{GEN} = \sum_{n=1}^{N} - log\left(D\left(G(I^{LR})\right)\right), \quad (3.3)$$

де D (G (I^{LR})) – це ймовірність розпізнавання дискримінатором того, що створене генератором зображення G (ILR) є зображенням високої роздільної здатності IHR.

Варто зазначити, що автори дослідження [1] пропонують використовувати при порівнянні зображення не MSE, а попередньо навчену нейронну мережу VGG-19, тому що результати її роботи більше відповідають тому, як людське око сприймає різницю між двома зображеннями.

На даний момент існує велика кількість методів, спрямованих на підвищення роздільної здатності зображень на основі єдиного вихідного. До першої групи можна віднести методи, що використовують інтерполяцію як спосіб досягнення найкращої апроксимації кольору та інтенсивності пікселів на основі значень їхніх сусідів. Основні алгоритми інтерполяції можуть бути віднесені до однієї з двох категорій: адаптивні та неадаптивні. Неадаптивні алгоритми включають метод найближчих сусідів, білінійну, бікубічну і сплайнову інтерполяції, фільтрацію Ланцоша [6]. Тоді як неадаптивні алгоритми обробляють всі пікселі однаковим чином, адаптивні методи змінюють свої коефіцієнти залежно від того, що вони інтерполують (різкі градієнти та межі, або гладкі текстури) [7].

Друга група методів заснована на пошуку схожих ділянок зображення та створення комплексного відображення між зображеннями з низьким (LR) та високим (HR) роздільною здатністю. Як навчальні дані використовуються пари LR-HR зображень і при відновленні використовуються частини LR зображень, для яких відомі HR аналоги.

У дослідженні Гласнера [8] використовується надмірність інформації, отриманої з кількох подібних елементів зображення. Пошук подібних ділянок проводиться у піраміді масштабів (набір версій вихідного зображення різних розмірів). Було запропоновано підхід з використанням розрідженого згорткового кодування, який обробляє зображення повністю, а

не тільки окремі частини, що перетинаються, що дозволяє поліпшити однорідність результатів.

До третьої групи можна віднести регресійні методи, для яких характерний більш загальний підхід до відображення пар ділянок зображень на основі ядерної регресії з регуляризацією L2. Це дозволяє вирішити проблему перенавчання, що виникає для підходів, що підвищують роздільну здатність за рахунок пошуку схожих ділянок у маломірній множині навчальної вибірки низької роздільної здатності та суміщення відповідних ділянок високої роздільної здатності для відновлення зображення. Регресійне формулювання цього завдання може бути вирішене за допомогою гауссівських процесів, вирішальних дерев та випадкового лісу.

В даний час алгоритми, засновані на згорткових нейронних мережах, дозволили досягти результатів, які перевершують описані вище методи. В одній із перших робіт [6] було реалізовано кодування вихідного представлення простору зображень у повнозв'язкову архітектуру нейронної мережі, заснованої на пороговій обробці та ітеративному алгоритмі стиснення.

Позначимо вихідне зображення низької роздільної здатності, яке збільшили до необхідного розміру методом бікубічної інтерполяції як Y. Метою підвищення роздільної здатності є отримання зображення F(Y), яке буде найбільш близьким до еталонного зображення з високою роздільною здатністю X. При використанні згорткових нейронних мереж архітектура мережі складається таким чином, щоб під час навчання було визначено відображення F з низької роздільної здатності у високу роздільну здатність. Ця процедура складається з трьох основних кроків:

1) Виділення ділянки зображення та побудова уявлення. Проводиться витягування ділянок зображення низької роздільної здатності Y, що перекриваються, і перетворення інформації про дану ділянку в багатовимірний вектор. Набір таких векторів утворює карту ознак зображення.

2) Нелінійне відображення. Кожному багатовимірному вектору нелінійним чином зіставляється багатовимірний вектор, що відповідає уявленню, вивченому нейронною мережею. Отримане зіставлення характеризує ділянку зображення високої роздільної здатності.

3) Відновлення. Виконується об'єднання окремих ділянок зображення високої роздільної здатності в єдине ціле.

Реалізація цих трьох кроків забезпечує формування нейронної мережі, як показано на рис. 3.1.

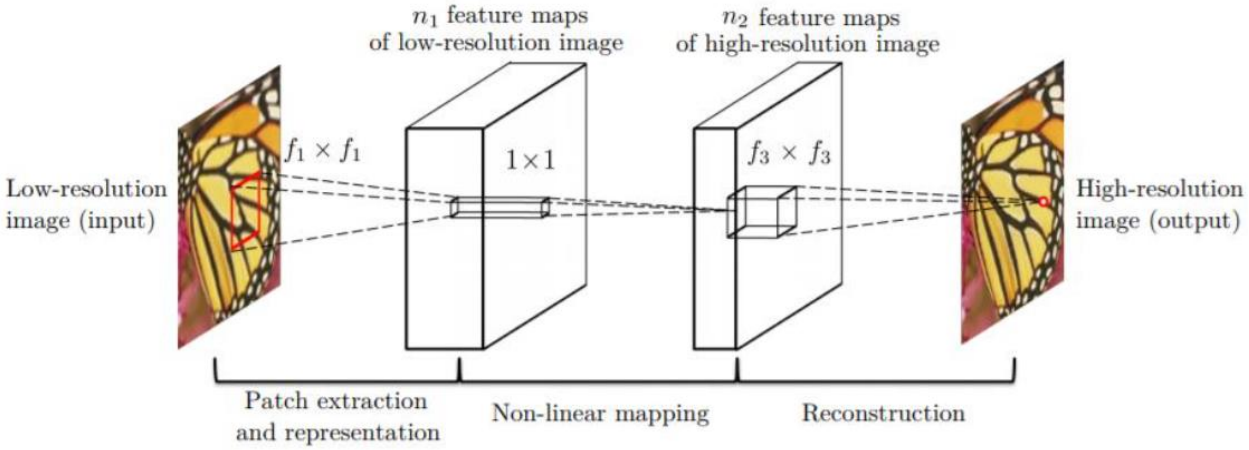


Рисунок 3.1 - Загальна схема згорткової нейронної мережі підвищення роздільної здатності зображення

На вхід подається зображення низької роздільної здатності Y, перший згортковий шар витягує карту ознак зображення. Другий шар нелінійно відображає ці ознаки на ознаки зображення високої роздільної здатності. Останній шар об'єднує передбачення нейронної мережі для окремих ділянок зображення для отримання результуючого зображення високої роздільної здатності F(Y) У роботі [6] представлена мережа згортання на основі глибокої рекурсії, яка дозволяє визначити залежності навіть для пікселів зображення, сильно розділених просторово, при цьому не збільшуючи значної кількості параметрів моделі (рис. 3.2). Слід зазначити дослідження

[8], в якому введено перцепційну функцію втрат, що дозволяє оптимізувати мережу для відновлення більш фотореалістичних зображень.

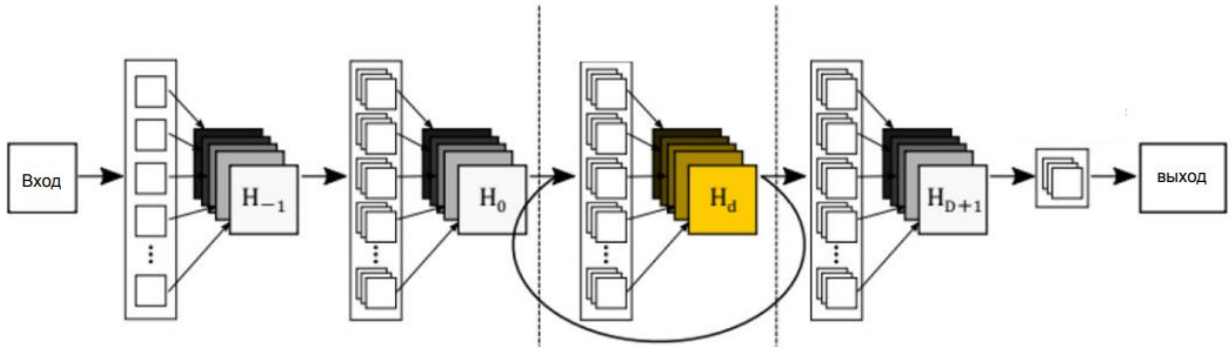


Рисунок 3.2 - Схема нейронної мережі з урахуванням глибокої рекурсії підвищення роздільної здатності зображень

Частина мережі, відповідальна за виведення відповідностей між ділянками низької та високої роздільної здатності, виділена жовтим кольором [7].

Підходи, засновані на використанні згорткових нейронних мережах глибокого навчання, найбільш якісно відновлюють деталі зображень, зберігаючи при цьому високу швидкість роботи.

Всі згорткові нейронні мережі (CNN) мають три виміри: ширину, глибину та роздільну здатність. Глибина означає кількість шарів, ширина – кількість каналів (наприклад, три канали для RGB), а роздільна здатність – кількість пікселів у зображенні. Кожен з цих вимірів можна масштабувати, і кожен певною мірою підвищує точність CNN:

1) Масштабування ширини – підвищує кількість каналів у зображенні (або нейронів у шарі). Це дозволяє шарам вивчити більш деталізовані ознаки та використовувалося в таких архітектурах, як WideResNet та MobileNet. Однак у міру збільшення ширини зростає і складність вивчення комплексних ознак.

2) Масштабування глибини – збільшення кількості шарів CNN. Це дозволяє мережі вивчати складніші ознаки. Однак, як було сказано вище, проблема градієнтів, що зникають, ускладнює навчання глибоких нейронних

мереж. І хоча пакетна нормалізація та обхідні зв'язки допомогли послабити цю проблему, емпіричні дослідження продемонстрували швидке падіння приросту точності. Наприклад, ResNet-100 має таку саму точність, як ResNet-1000.

3) Масштабування дозволу – збільшує роздільну здатність зображення, і, отже, кількість пікселів. Це дозволяє мережі знаходити дрібніші структури за рахунок додаткових деталей зображення. Як і інші види масштабування, саме собою масштабування дозволу забезпечує обмежений приріст точності.

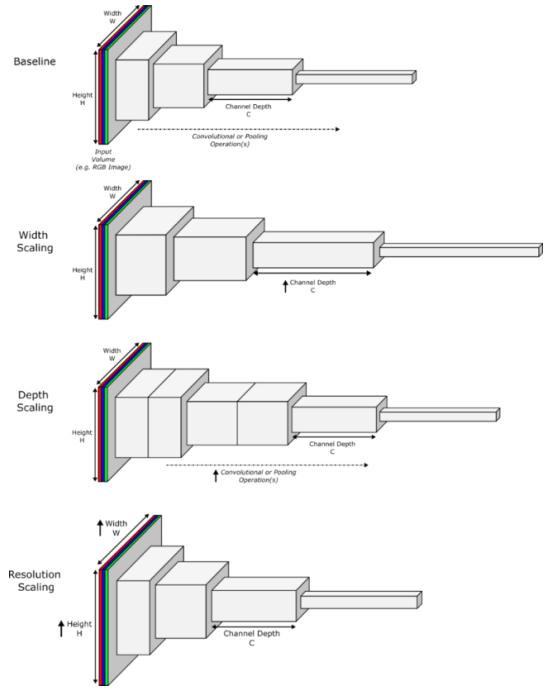


Рисунок 3.3 - Методи масштабування нейронних мереж

Дослідники виявили, що балансування масштабуванням по кожному з вимірювань (ширина, глибина та роздільна здатність мережі) було ключем до отримання максимального приросту точності при мінімальному зростанні обчислювальної складності.

Найдорожчі в обчислювальному сенсі операції CNN - це операції згортки. Більше того, кількість операцій з плаваючою точкою (FLOP'ів) на операцію згортки приблизно пропорційно d, w2 і r2 - тобто, подвоєння

глибини подвоїть і кількість FLOP'ів, а подвоєння ширини та роздільної здатності збільшить кількість FLOP'ів вчетверо. Грунтуючись на цих припущеннях, автори [3] запропонували просту техніку масштабування, засновану на комбінованому коефіцієнті phi (що описує кількість наявних ресурсів), щоб визначити, як ефективно масштабувати alpha, beta і gamma (що стосуються ширини, глибини та роздільної здатності нейронної мережі).

$$depth{:}d=\alpha^{\phi} width{:}w=\beta^{\phi} resolution{:}r=\gamma^{\phi} s.t. \alpha\cdot\beta^2\cdot\gamma^2\approx 2 \alpha\geq 1, \beta\geq 1, \gamma\geq 1$$

Як обговорювалося вище, це співвідношення встановлює, що масштабування мережі (за шириною, глибиною або роздільною здатністю) збільшить кількість FLOP'ів (aplha*beta2*gamma2)phi разів. Обмеження "(aplha*beta2*gamma2) приблизно 2" введено, щоб гарантувати, що кількість FLOP'ів не зросте більше, ніж у 2phi разів. Таким чином, ми можемо використовувати комбінований коефіцієнт phi для масштабування необхідної кількості FLOP'ів у відому кількість разів – 2phi.

Базова нейронна мережа EfficientNet-B0 використовувалася для пошукового мережі оптимальних параметрів масштабування з фіксованим комбінованим коефіцієнтом phi, рівним одиниці, і кращими параметрами виявилися aplha=1.2, beta=1.1 і gamma=1.15. Потім, фіксуючи оптимальні значення aplha, beta і gamma, дослідники масштабували кількість наявних phi ресурсів для створення моделей більшого розміру від EfficientNet-B1 до EfficientNet-B7.

3.2 Архітектура програмного комплексу

Операція з матрицями є важливою частиною для ММ навчання, таких як лінійна регресія. TensorFlow має всі матричні операції, такі як множення, інверсія, обчислення визначника, розв'язання лінійних рівнянь та багато інших задач.

Ілюстрація розповсюдження сигналу в С-шарі зображено на рис. 3.4:

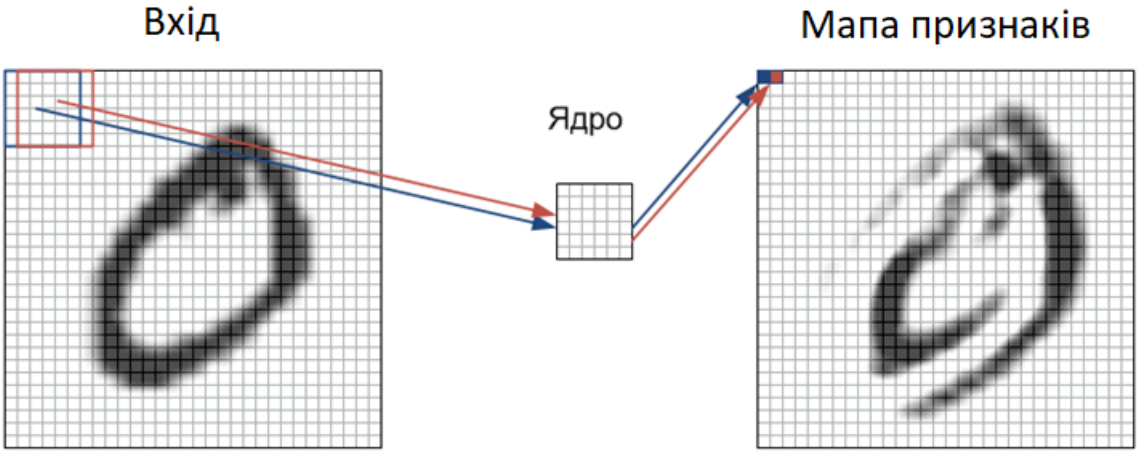


Рисунок 3.4 - Використання матричних операцій

Важливим є правильний вибір алгоритмів обробки та визначення ознак для досягнення оптимального результату на кожному з цих етапів.

Мультиспектральне зображення і двох компонент. Отримано два зображення в діапазоні каналу 1і діапазоні каналу 2, рисунок 20.

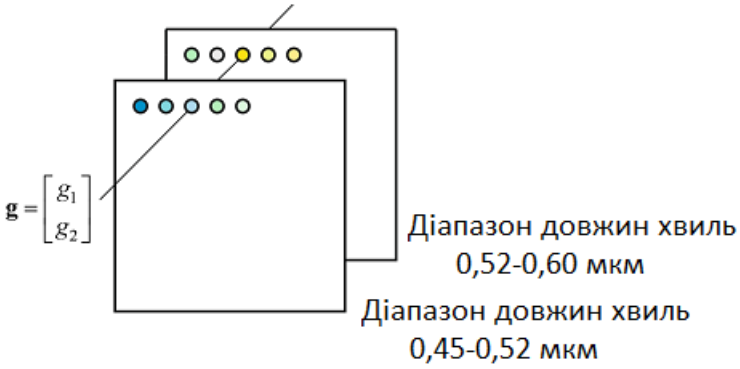


Рисунок 3.5 - Дві складові мультиспектрального зображення

Значення кожного пікселя мультиспектрального зображення можна у вектора, координати якого визначаються зображеннями каналу 1і каналу 2, тобто.

$$\boldsymbol{g} = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}.$$

Один вектор описує один піксель мультиспектрального зображення. Наприклад, якщо таке зображення має розмір$M \times N$, воно характеризується

двомірними векторами чи масивом даних величиною . Якщо реєструється компонент (зображень) вектор$K = MN2MNnn$

$$.\boldsymbol{g} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix} n - \text{стає мірним}$$

Вектор можна подавати як вибірку з випадкового процесу.$\boldsymbol{g}$Метод основних компонентів використовує поняття коваріації.

Нехай встановлено суміщене зображення розміром, отримане з двох зображень для двох каналів (двох діапазонів). Поєднане зображення включає наступні двовимірні вектори:$1 \times 5K = MN = 5$

$$\boldsymbol{g}_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, , , , . \boldsymbol{g}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \boldsymbol{g}_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \boldsymbol{g}_4 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} g_5 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Через $\boldsymbol{m}_g$ позначимо середній вектор $\boldsymbol{m}_g = E\{g\}Em_g$, де E – оператор математичного очікування. Оцінка вектора обчислюється за формулою усереднення

$$.\boldsymbol{m}_g = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{g}_i = \frac{1}{5}\begin{bmatrix} 5 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Типова архітектура згорткової мережі виглядає приблизно так

· згорткові шари (16 каналів, 1-3 шари)

· max-pooling(в 2 рази)

· згорткові шари (32 канали, 1-3 шари)

· max-pooling(в 2 рази)

· згорткові шари (64 канали, 1-3 шари)

· max-pooling(в 2 рази)

І так поки картинка не стане досить маленькою (у випадку з YOLO) або навіть не перетвориться на картинку 1×1 з великою кількістю каналів (VGG, AlexNet).

Можливі варіації із заміною max-pooling на згортку зі stride.

Перші шари отримують на вхід повнорозмірне зображення, його обробка виконується достатньо довго, тому початкові шари мають невелику кількість каналів.

Після зменшення картинки у 2 рази її площа зменшується у 4 рази, ми збільшуємо кількість каналів у 2 рази, через що складність обчислення для кожного пікселя зростає у 2×2 = 4 рази. Таким чином, виходить, що всі шари приблизно рівні за складністю обчислення. Це досить логічно, якби час роботи різних шарів становило б прогресію типу 1, 2, 4, то більшу частину часу обчислювався один шар і виникла б ідея або зробити його тоншим, або збільшити кількість каналів в інших шарах, так як вони обчислювалися все одно швидко.

Зменшення картинки збільшує швидкість роботи мережі або дозволяє використовувати більш "товсті" шари. З практичних міркувань було обрано зменшення вихідної картинки в 8 разів - достатньо знати про наявність області виділення з точністю до 8 пікселів, та й типова область виділення буде більше цього розміру.

Ще важлива кількість пам'яті, що займається. Для зберігання хоча б 8 каналів зображення 1440x900 потрібно як мінімум 1440x900x8x4 = 41472000 байт - більше 40 мб пам'яті. При зменшенні розмірів шару вдвічі і збільшення кількості каналів вдвічі (тобто, при постійної обчислювальної складності) ми отримуємо економію пам'яті вдвічі. Через це максимум споживання пам'яті посідає перші верстви. Це досить актуально для мобільних пристроїв – на них обчислення початкових шарів виходило дуже повільним.

Зазвичай згортка робиться з розміром ядра меншим або рівним 3×3. Але так вхідне зображення зменшується тільки у 8 разів, хотілося б, щоб область вихідного зображення, на підставі якої приймається рішення про те, чи належить піксель виділення, була щонайменше 100 × 100 пікселів у початковому зображенні. Чим більше ця область, тим більше навчальних прикладів потрібно, проте мережа буде брати до уваги якусь інформацію з досить далеких частин зображення.

Існують два підходи:

1) Зменшити картинку і потім збільшити назад (все всередині однієї мережі)

2) Використати згортку з параметром dilation, що дозволяє отримувати інформацію від віддалених областей зображення.

Звичайний згортковий шар робить відразу дві важливі речі: обробляє просторову складову зображення (аналізує відразу кілька пікселів), а також поканальну - аналізує відразу всі канали (з усіх пікселів) і на основі їх значень обчислює нові. Нижче описано архітектуру Xception шару, в якому ці дві функції розділені між шарами мережі.

Якщо додати проміжні поточкові згортки, мережа працює краще. Підсумкова архітектура:

· convolution(8 channels)

· max-pooling(2)

· convolution(16 channels)

· max-pooling(2)

· convolution(32 channels)

· max-pooling(2)

· convolution(64 channels)

· pointwise convolution(32 channels)

· convolution(64 channels, dilation = 2)

· pointwise convolution(32 channels)

· convoltuion(64 channels, dilation = 4)

· pointwise convolution(32 channels)

· convoltuion(64 channels, dilation = 8)

· pointwise convolution(32 channels)

· convoltuion(64 channels)

· convolution(1 channel)

Що важливо, мережа з проміжними шарами поточки краще розпізнає, містить менше коефіцієнтів і працює швидше.

3.3 Опис особливостей програмної реалізації

Для створення програми, яка покращує якість дактилоскопічних зображень за допомогою нейронної мережі, ми можемо використовувати бібліотеку Python, таку як TensorFlow або PyTorch. Послідовність виконання етапів навчання мережі:

1) Підготовка даних: Перш за все, потрібно підготувати набір даних, який містить пари вхідних та вихідних зображень. Вхідні зображення - це низькоякісні версії, а вихідні - високоякісні оригінали. Розмір навчального набору даних може бути досить великим.

```
overall_dataset_indexes = list(range(0, len(input_images_fnames)))
# Split indexes for train and test
train_indexes, test_indexes = train_test_split(overall_dataset_indexes, test_size=.25, random_state=42)

input_train_fnames = [str(input_images_fnames[i]) for i in train_indexes]
output_train_fnames = [str(output_images_fnames[i]) for i in train_indexes]

input_test_fnames = [str(input_images_fnames[i]) for i in test_indexes]
output_test_fnames = [str(output_images_fnames[i]) for i in test_indexes]
```

```
BATCH = 1

train_dataset = Dataset.from_tensor_slices((input_train_fnames, output_train_fnames))
test_dataset = Dataset.from_tensor_slices((input_test_fnames, output_test_fnames))

train_dataset = train_dataset.map(load_and_preprocess_images, num_parallel_calls=tf.data.experimental.AUTOTUNE)
test_dataset = test_dataset.map(load_and_preprocess_images, num_parallel_calls=tf.data.experimental.AUTOTUNE)

train_dataset = train_dataset.batch(BATCH)
test_dataset = test_dataset.batch(BATCH)
```

Рисунок 3.6 – Процес підготовки даних

2) Створення моделі: Вибираємо архітектуру нейронної мережі для завдання покращення зображень. Один з популярних підходів - це використання глибоких конволюційних мереж (CNN). Одним із прикладів може бути модель, що базується на архітектурі Generative Adversarial Network (GAN) або Super-Resolution Convolutional Neural Network (SRCNN).

3) Навчання моделі: Використовуємо навчальний набір даних для тренування моделі. Структура коду для навчання моделі за допомогою бібліотеки TensorFlow:

```
saving_callback = tf.keras.callbacks.ModelCheckpoint
('./{epoch:02d}_best_model.h5', monitor='val_mse', save_best_only=True)

model.compile(optimizer='adam', loss='mse', metrics=['mse'])
history = model.fit(
    train_dataset,
    epochs=20,
    callbacks=[saving_callback],
    shuffle=True,
    validation_data=test_dataset,
    verbose=1
)
```

Рисунок 3.7 – Структура навчання моделі

Створення моделі

```
model = Sequential([
    # encoder
    layers.Input(shape=(320, 320, 1)),
    layers.Conv2D(128, (3, 3), activation="relu", padding="same"),
    layers.MaxPooling2D((2, 2), padding="same"),
    layers.Conv2D(64, (3, 3), activation="relu", padding="same"),
    layers.MaxPooling2D((2, 2), padding="same"),
    layers.Conv2D(32, (3, 3), activation="relu", padding="same"),
    layers.MaxPooling2D((2, 2), padding="same"),
    # decoder
    layers.Conv2DTranspose(32, (3, 3), strides=2, activation="relu", padding="same"),
    layers.Conv2DTranspose(64, (3, 3), strides=2, activation="relu", padding="same"),
    layers.Conv2DTranspose(128, (3, 3), strides=2, activation="relu", padding="same"),
    layers.Conv2D(1, (3, 3), activation="sigmoid", padding="same"),
])
model.summary()
```

Рисунок 3.8 – Структура моделі

4) Застосування моделі до нових зображень: Після навчання моделі її можна використовувати для покращення якості зображень.

Завантаження навченої моделі

Завантаження низькоякісного зображення, яке треба покращити

Покращення зображення за допомогою моделі

Збереження покращеного зображення

Компілюємо модель та задаємо функцію втрат

Зберігаємо навчену модель для подальшого використання

Застосовуємо модель до тестових даних (зображень з дефектами) і отримуємо результати

```
# Loading best model
denoiser = tf.keras.models.load_model('./19_best_model_.h5')
# Shuffle test dataset
shuffled = test_dataset.shuffle(test_dataset.cardinality())
test_dataset_iterator = shuffled.as_numpy_iterator()

num_of_predicted_examples = 5
visualization_list = list()

for i in range(num_of_predicted_examples):
    in_image, gt_image = next(test_dataset_iterator)
    # Predictions
    prediction = denoiser.predict(in_image)
    # Backward transformation
    in_image = (in_image[0] * 255).astype(np.uint8)
    pred_image = (prediction[0] * 255).astype(np.uint8)

    visualization_list.append((in_image, pred_image))

visualiza_image_paires(visualization_list, ['Image with noise', 'Denoized image'])
```

Рисунок 3.9 – Застосування навченої моделі

Виконання зі зміною параметрів CNN

```
Epoch 1/10
1875/1875 [==============================] - 20s 10ms/step - loss: 0.3521 - accuracy: 0.9025
Epoch 2/10
1875/1875 [==============================] - 19s 10ms/step - loss: 0.2099 - accuracy: 0.9391
Epoch 3/10
1875/1875 [==============================] - 19s 10ms/step - loss: 0.1540 - accuracy: 0.9564
Epoch 4/10
1875/1875 [==============================] - 20s 11ms/step - loss: 0.1165 - accuracy: 0.9671
Epoch 5/10
1875/1875 [==============================] - 19s 10ms/step - loss: 0.0965 - accuracy: 0.9720
Epoch 6/10
1875/1875 [==============================] - 18s 10ms/step - loss: 0.0835 - accuracy: 0.9758
Epoch 7/10
1875/1875 [==============================] - 18s 10ms/step - loss: 0.0741 - accuracy: 0.9786
Epoch 8/10
1875/1875 [==============================] - 18s 10ms/step - loss: 0.0674 - accuracy: 0.9800
Epoch 9/10
1875/1875 [==============================] - 19s 10ms/step - loss: 0.0624 - accuracy: 0.9817
Epoch 10/10
1875/1875 [==============================] - 19s 10ms/step - loss: 0.0578 - accuracy: 0.9830
<keras.callbacks.History at 0x7f87bab60b80>
```

Рисунок 3.10 – Результат роботи зі зміною параметрів CNN

Будуємо графік

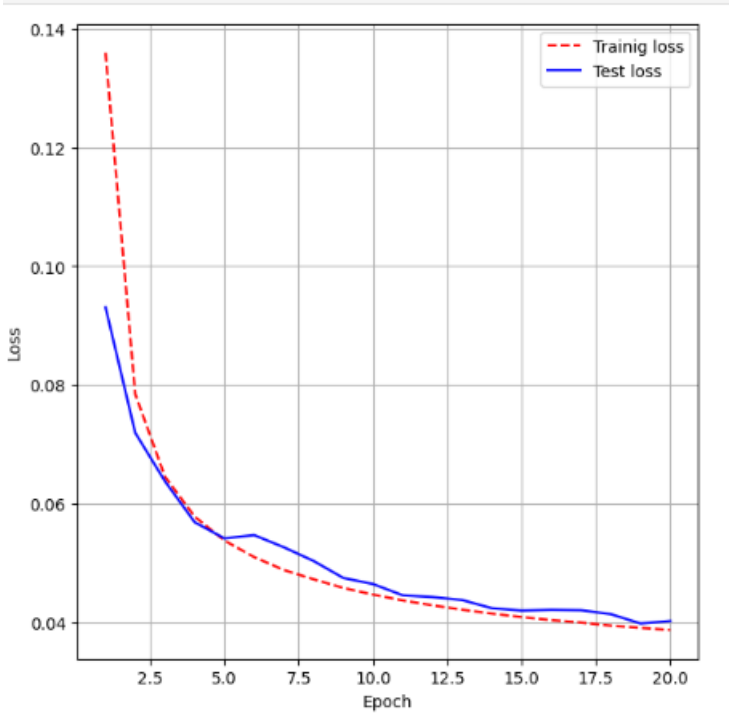


Рисунок 3.11 – Графік залежності похибки від кількості епох

Змінюємо параметри моделі, виконуючи збільшення кількості ітерацій навчання та розмірності партій даних:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 320, 320, 128)     1280

 max_pooling2d (MaxPooling2  (None, 160, 160, 128)     0
 D)

 conv2d_1 (Conv2D)           (None, 160, 160, 64)      73792

 max_pooling2d_1 (MaxPoolin  (None, 80, 80, 64)        0
 g2D)

 conv2d_2 (Conv2D)           (None, 80, 80, 32)        18464

 max_pooling2d_2 (MaxPoolin  (None, 40, 40, 32)        0
 g2D)

 conv2d_transpose (Conv2DTr  (None, 80, 80, 32)        9248
 anspose)

 conv2d_transpose_1 (Conv2D  (None, 160, 160, 64)      18496
 Transpose)

 conv2d_transpose_2 (Conv2D  (None, 320, 320, 128)     73856
 Transpose)

 conv2d_3 (Conv2D)           (None, 320, 320, 1)       1153

=================================================================
Total params: 196289 (766.75 KB)
Trainable params: 196289 (766.75 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Рисунок 3.12 – Процес виконання обчислень

Результати обчислення свідчать про більш точну апроксимацію даних завдяки кількості ітерацій:

```
Epoch 1/20
750/750 [==============================] - 173s 229ms/step - loss: 0.1360 - mse: 0.1360 - val_loss: 0.0931 - val_mse: 0.0931
Epoch 2/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0785 - mse: 0.0785 - val_loss: 0.0719 - val_mse: 0.0719
Epoch 3/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0646 - mse: 0.0646 - val_loss: 0.0638 - val_mse: 0.0638
Epoch 4/20
750/750 [==============================] - 170s 226ms/step - loss: 0.0578 - mse: 0.0578 - val_loss: 0.0569 - val_mse: 0.0569
Epoch 5/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0538 - mse: 0.0538 - val_loss: 0.0542 - val_mse: 0.0542
Epoch 6/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0510 - mse: 0.0510 - val_loss: 0.0547 - val_mse: 0.0547
Epoch 7/20
750/750 [==============================] - 169s 226ms/step - loss: 0.0488 - mse: 0.0488 - val_loss: 0.0527 - val_mse: 0.0527
Epoch 8/20
750/750 [==============================] - 170s 226ms/step - loss: 0.0473 - mse: 0.0473 - val_loss: 0.0504 - val_mse: 0.0504
Epoch 9/20
750/750 [==============================] - 169s 226ms/step - loss: 0.0458 - mse: 0.0458 - val_loss: 0.0475 - val_mse: 0.0475
Epoch 10/20
750/750 [==============================] - 171s 228ms/step - loss: 0.0447 - mse: 0.0447 - val_loss: 0.0465 - val_mse: 0.0465
Epoch 11/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0437 - mse: 0.0437 - val_loss: 0.0446 - val_mse: 0.0446
Epoch 12/20
750/750 [==============================] - 170s 226ms/step - loss: 0.0429 - mse: 0.0429 - val_loss: 0.0443 - val_mse: 0.0443
Epoch 13/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0421 - mse: 0.0421 - val_loss: 0.0438 - val_mse: 0.0438
Epoch 14/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0415 - mse: 0.0415 - val_loss: 0.0424 - val_mse: 0.0424
Epoch 15/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0409 - mse: 0.0409 - val_loss: 0.0420 - val_mse: 0.0420
Epoch 16/20
750/750 [==============================] - 170s 227ms/step - loss: 0.0404 - mse: 0.0404 - val_loss: 0.0421 - val_mse: 0.0421
Epoch 17/20
750/750 [==============================] - 171s 228ms/step - loss: 0.0400 - mse: 0.0400 - val_loss: 0.0421 - val_mse: 0.0421
Epoch 18/20
750/750 [==============================] - 170s 226ms/step - loss: 0.0395 - mse: 0.0395 - val_loss: 0.0414 - val_mse: 0.0414
Epoch 19/20
750/750 [==============================] - 171s 227ms/step - loss: 0.0391 - mse: 0.0391 - val_loss: 0.0398 - val_mse: 0.0398
Epoch 20/20
750/750 [==============================] - 174s 232ms/step - loss: 0.0387 - mse: 0.0387 - val_loss: 0.0402 - val_mse: 0.0402
```

Рисунок 3.13 – Результат обчислень після збільшення кількості ітерацій навчання

Нейромережа навчання представлена трьома згортковими шарами з ядром згортки 5x5, кроком 1 і доповненням 2 та трьома пов'язними шарами. Архітектура згорткової нейромережі зображена на рис. 3.14.

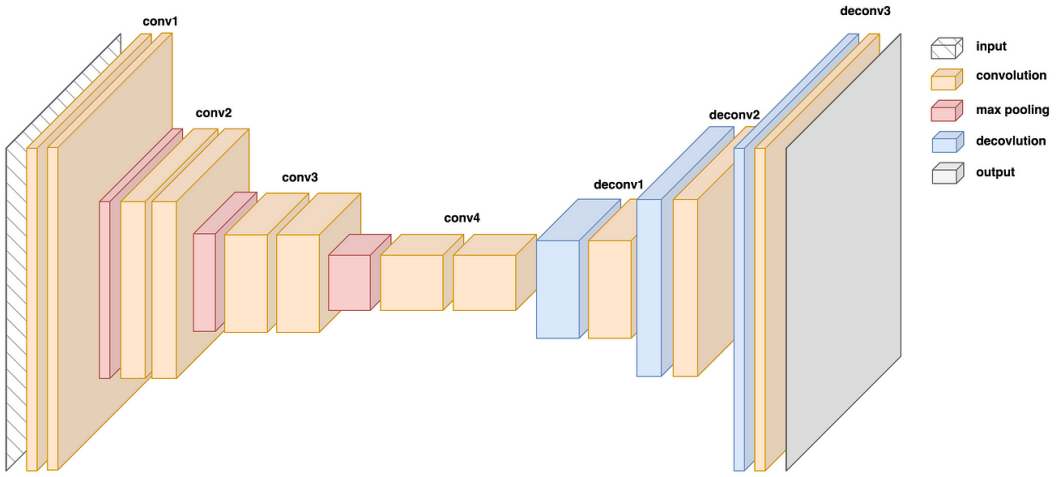


Рисунок 3.14 - Архітектура згорткової нейромережі для розпізнавання дактилоскопічних зображень

3.4 Опис інтерфейсів програмного продукту

Інтерфейс додатку передбачає наявність початкового зображення та кінцевого, після обробки:

```
    def enhance(self, img, resize=True):

        if(resize):
            rows, cols = np.shape(img)
            aspect_ratio = np.double(rows) / np.double(cols)

            new_rows = 350
            new_cols = new_rows / aspect_ratio

            img = cv2.resize(img, (int(new_cols), int(new_rows)))

        self.__ridge_segment(img)
        self.__ridge_orient()
        self.__ridge_freq()
        self.__ridge_filter()
        return(self._binim)

if __name__ == '__main__':

    image_enhancer = FingerprintImageEnhancer()
    if(len(sys.argv)<2):
        print('завантаження тестового зображення дактилоскопії');
        img_name = '2.jpg'
        img = cv2.imread('../images/' + img_name)
    elif(len(sys.argv) >= 2):
        img_name = sys.argv[1];
        img = cv2.imread('../images/' + img_name)

    if(len(img.shape)>2):
         img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

    out = image_enhancer.enhance(img)
    image_enhancer.save_enhanced_image('../enhanced/' + img_name)
```

Рисунок 3.15 – Інтерфейс додатку

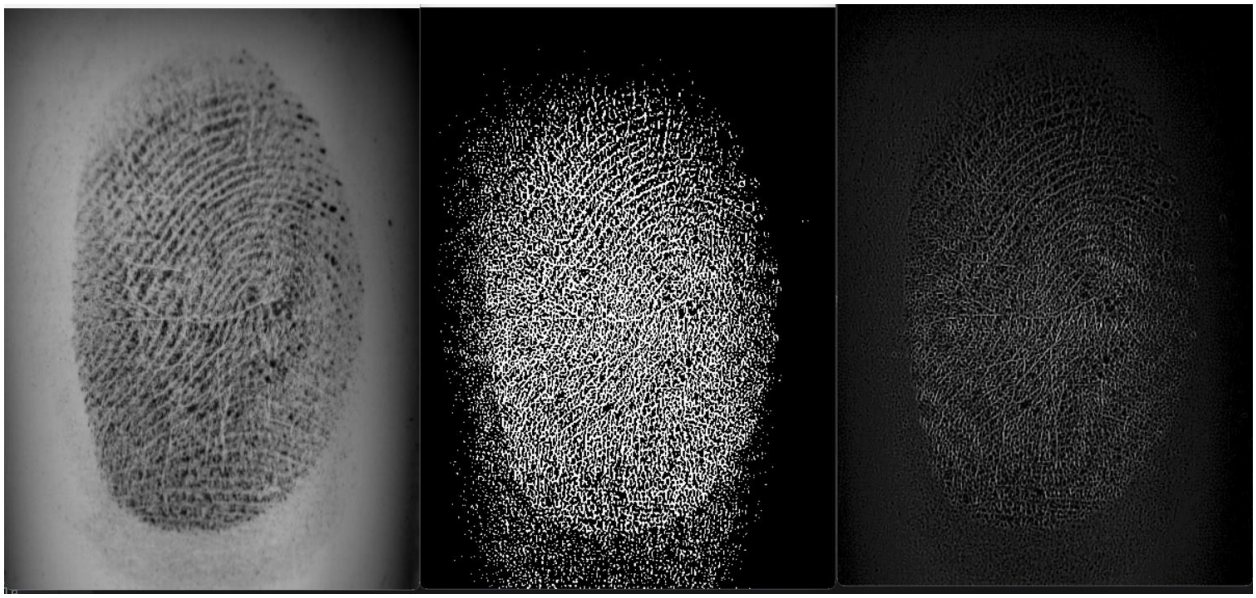


Рисунок 3.16 - Накладання фільтру Габора на тестове зображення

3.5 Оцінка ефективності методу скелетизації дактилоскопічних зображень

Розглянемо роботу фільтра №1 (виділеного на Рис. 3.14). Після того, як на вхід мережі подано зображення, фільтр переміщається по всьому об'єму, крок за кроком зсуваючись вниз або вправо, і для кожного положення обчислюється операція згортки. Тобто кожен елемент у ядрі множиться на значення відповідного пікселя зображення, потім отримані значення складаються та поділяються на кількість елементів. Таким чином, кожна відповідність елементів призводить до підсумкового значення 1. Аналогічно будь-яка невідповідність відображається результатом -1. У графічному вигляді це показано на рис.3.13.

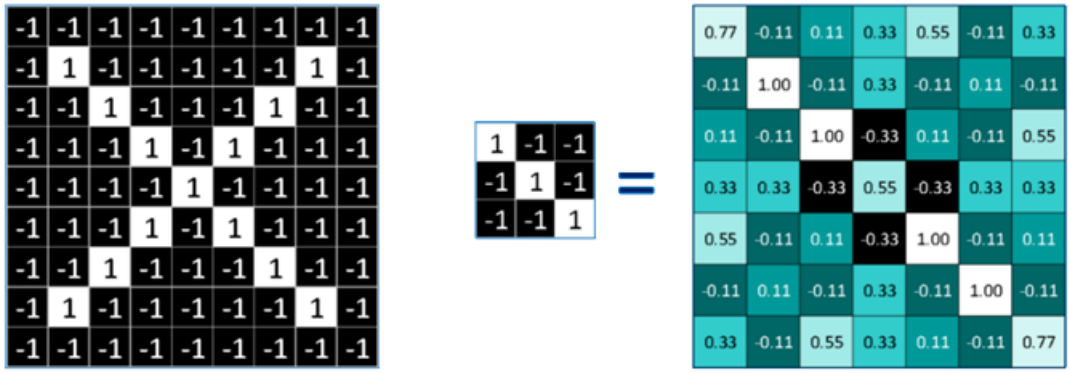


Рисунок 3.13 - Застосування фільтра до зображення

Застосувавши аналогічно ядра №2 і №3, отримаємо три відфільтрованих варіанти вихідного зображення, які називаються картами ознак.

Далі продемонструємо роботу ще одного інструменту згорткової мережі – шар субдискретизації. Він необхідний масштабування великих зображень (шарів) і, здійснюючи стиск, зберігає у своїй найзначнішу інформацію. Механізм його роботи простий: накласти невеликий фільтр на зображення та отримати максимальне значення із цієї області на кожному кроці. На практиці використовується ядро розміром 2 або 3 пікселя та кроком 2.4

Після вибору зображення має приблизно чверть пікселів, щодо вихідного. Результатом роботи цього шару є те, що мережі згортки можуть визначити, чи знаходиться потрібний фільтр у зображенні, незалежно від невеликих зрушень і спотворень.

Важливим допоміжним кроком на цьому етапі є «Лінійна ретифікація», тобто, кожне негативне значення замінюється на нуль. Це допомагає згорткової мережі від утримання вихідних значень, від застрягання близько 0 або збільшення до нескінченності.

Проводячи отримані карти ознак далі по згортковим шарам, ми маємо можливість обчислювати дедалі складніші функції, тобто розпізнавати високорівневі аспекти зображення, як-от форми і візерунки. На останньому шарі отриманий з вихідного зображення вектор подається на вхід повнозв'язного шару, нейрони якого визначають оцінку класу (Рис. 3.14).

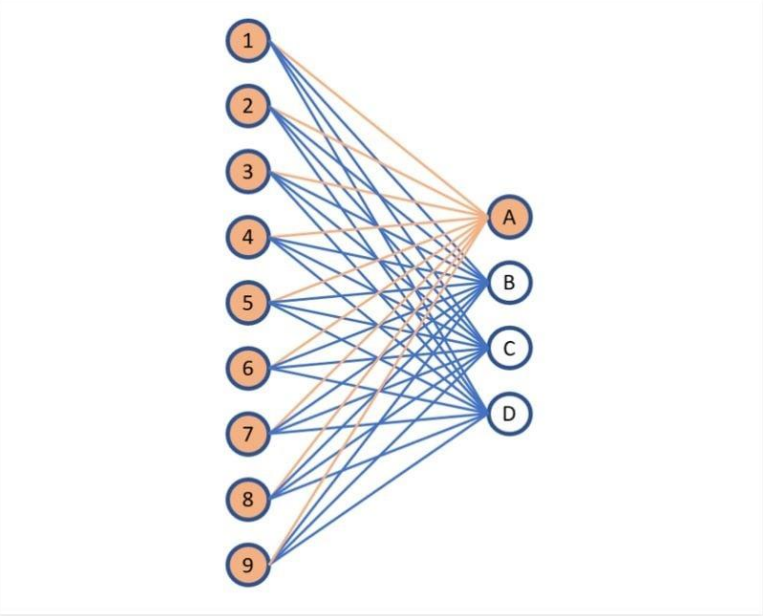


Рисунок 3.14 - Результат роботи повнозв'язного шару

Згорткова нейронна мережа (CHC) дозволяє виділити найважливіші як пікселі зображення, а й більш абстрактні структури, такі як лінії чи області і має складнішу структуру [2]. У прикладі буде використано мережу з наступною архітектурою: перший шар CHC – згортковий з фільтром 3 на 3

пікселя з додаванням порожнього рядка та стовпця, завдяки чому кінцевий розмір зображення співпадатиме з початковим.

Зображення розбивається на 32 карти ознак, далі відбувається операція MaxPooling, з матрицею 2 на 2 пікселя, яка дозволяє виявити найзначніші пікселі в кожній області розмірністю 2 на 2. Таким чином, зображення стискається в 2 рази. Наступним етапом наведені операції повторюються ще раз, але з використанням 64 фільтрів на згортковому шарі. У результаті отримане зображення і переведене в одномірний вектор, подається на вхід персептрону з 1 прихованим шаром з 128 нейронів і вихідний має також 10 нейронів - за кількістю класів зображень.

У всіх моделях використовувалася функція помилки "категоріальна кросентропія", оптимізатор навчання Адама. Підготовка набору даних Для коректної обробки масиву даних необхідно нормалізувати. Набір даних MNIST містить 50 000 зображень у градаціях сірого розмірів 28 на 28 пікселів. Один канал кольору означає характеристику пікселя як інтенсивність кольору, 0 (чорний) до 255 (білий). Тому доречно розділити значення пікселя на 255 діапазон значень буде в межах від 0 до 1.

Дані про правильні відповіді вибірки були нормалізовані за допомогою технології «One Hot Encoding» і перетворені на вектор виду [1,0,0,0,0,0,0,0,0,0], де порядковий номер одиниці означає клас зображення:

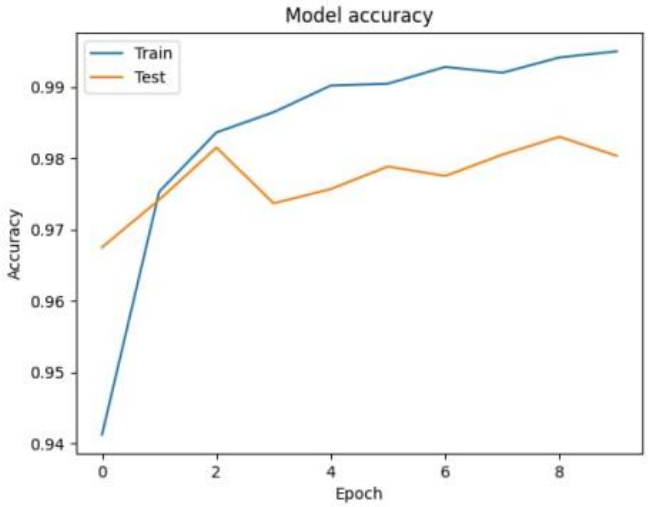


Рисунок 3.15 - Графік точності навчання персептрону

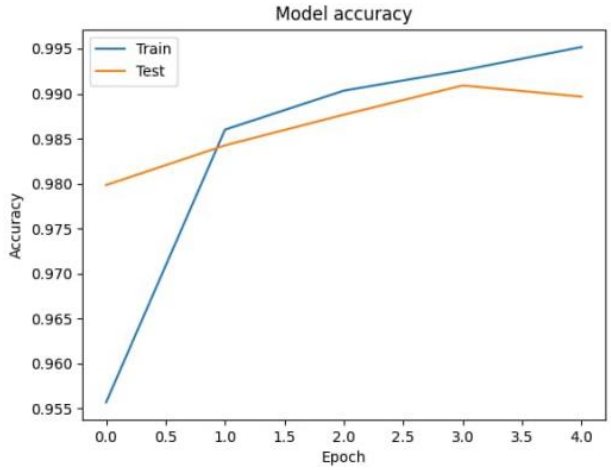


Рисунок 3.16 - Графік точності навчання СНР

Отримано порівняльну таблицю ефективності навчання нейронних мереж.

Таблиця 3.1 - Порівняння ефективності моделей СР

СР і завдання	Характеристика	Кількість епох	Точність навчальної/ тестової вибірки, %	Значення функції помилки в навчальній/ тестової вибірки	Час на навчання, сек
ПНР MNIST		5	99,5\98	0,016\0.068	113
СНР MNIST		5	99,5 \ 98,9	0.015\0.034	121

З таблиці, можна дійти невтішного висновку у тому, що найкращою моделлю у цій задачі виявилася згорткова нейронна мережу, зі значенням точності на тестової вибірці 98,9. Мінусом цієї моделі виявляється час навчання, на 5 ітерацій було витрачено 121 секунда (значення округлені до цілих).

Найгіршою у задачі класифікації та реконструкції зображень виявилася модель персептрона з більшим часом навчання (113 секунд), що пов'язано з великою кількістю нейронів у прихованих шарах, та найгіршим значенням на тестовому наборі даних. РНМ має найменший розкид у значеннях навчальної та тестової вибірки.

ВИСНОВКИ

Автоматична обробка дактилоскопічних зображень у загальному випадку включає в себе процес отримання символічних описів або відомостей зі зображень реального світу. Ці символічні описи можуть бути використані для подальшої інтерпретації сцени або виконання різних завдань аналізу та обробки зображень. Основні мети автоматичної обробки зображень включають:

1) Інтерпретація сцени: Автоматична обробка дозволяє визначати об'єкти, обличчя, області інтересу та їх взаємозв'язки на зображенні. Це корисно для розуміння того, що відображено на зображенні.

2) Відомості та аналіз: Обробка зображень може включати в себе видобуття різних відомостей, таких як кольори, текстури, контури, розміри об'єктів і т. д. Ці відомості можуть бути використані для подальшого аналізу.

3) Візуальне удосконалення: Автоматична обробка може включати фільтрацію шуму, підвищення роздільної здатності, підсилення контрасту та інші операції для покращення якості зображення.

4) Визначення аспектів зображення: Зображення може містити інформацію, яка не є безпосередньо очевидною у початковій формі. Автоматична обробка може допомогти визначити ці аспекти, такі як геометричні характеристики, особливості об'єктів і т. д.

Загалом, автоматична обробка зображень є важливою галуззю комп'ютерного зору та обробки сигналів, і вона знаходить застосування у багатьох сферах, включаючи комп'ютерне бачення, медичну діагностику, розпізнавання обличчя, виробничий контроль і багато інших.

Згорткові нейронні мережі знайшли своє застосування можна при обробці зображень. Якщо уявити, що зображення - двовимірні функції, то різні перетворення зображень - це операція згортка функції зображення з

локальною функцією, яка називається ядром згортки. Операція згортки чергується з операцією підвибірки.

Найбільш поширеними методами розпізнавання зображень є екстремально-кореляційні, статистичні, структурно-лінгвістичні, геометричні інваріанти [2]. Перш ніж показати зв'язок цих підходів з машинною графікою, дамо їх загальну коротку характеристику, визначимо ті причини, які затримують методи розпізнавання в рамках лабораторій та залишають проблему далекою від широкого практичного застосування.

Ідея екстремально-кореляційного рішення зводиться до обчислення кореляційної функції K(x,y) вихідного зображення I(x,y) та зображення еталона E(x,y). Якщо у вихідному зображенні знайдеться фрагмент, ідентичний E(x,y), то в I(x,y) виникне локальний екстремум. Все поле кореляційної функції піддають високочастотної фільтрації для придушення шумів, розмитих піків і граничним методом селектують положення еталонного об'єкта.

Екстремально-кореляційний метод, як і будь-який інший, має свої переваги та недоліки. До недоліків цього методу відносяться:

1) Висока чутливість до розбіжності масштабу, орієнтації та яскравості: Метод може бути менш ефективним, коли об'єкт на зображенні має відмінності у масштабі, орієнтації або яскравості порівняно з еталоном.

2) Значний обсяг обчислень: Робота з екстремально-кореляційним методом може вимагати великої кількості обчислень, що може бути ресурсомоємним завданням.

3) "Хибні тривоги": Метод може виводити "хибні тривоги" в тих випадках, коли пошуковий об'єкт відсутній на вихідному зображенні, але метод все одно виявляє подібність до еталону.

4) Суб'єктивність вибору параметрів: При використанні методу потрібно вибирати параметри, такі як поріг, вид високочастотного фільтра та інші. Це може впливати на результат і може вимагати суб'єктивного втручання.

Однак метод також має позитивні риси:

1) Працює безпосередньо із зображенням: Метод може опрацьовувати зображення без необхідності виділення окремих ознак або об'єктів.

2) Статистичні методи: Статистичні методи використовують статистичні характеристики зображення, такі як математичне очікування, дисперсія, моменти вищого порядку, гістограма, що може бути корисним для аналізу.

3) Структурні методи: Структурні методи дозволяють виділяти ознаки та зв'язки між ними, що може бути корисним для вирішення завдань ідентифікації.

4) Метод геометричних інваріантів: Метод геометричних інваріантів враховує геометричні властивості об'єктів і може бути корисним у випадках, коли масштаб, орієнтація та інші характеристики змінюються.

Кожен метод має свої сильні та слабкі сторони, і вибір методу залежить від конкретного завдання та обставин.

Усі перелічені методи поєднує загальна ідея порівняння еталонного та аналізованого зображень безпосередньо або через вторинні ознаки. Незалежно від методу якість порівняння сильно залежить від ідентичності умов освітлення та спостереження аналізованого та еталонного зображень.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Boser, B., Guyon I., Vapnik, V. A training algorithm for optimal margin classifiers. In D. Haussler, editor, proceedings of the 5th Annual ACM Workshop on COLT, Pittsburgh, 1992, P. 144-152.
2. Breiman, L. Bagging predictors. Machine Learning, Vol. 24 (2), 1996, P. 123- 140
3. Shoghian, Sh., Kouzehgar, M. A Comparison among Wolf Pack Search and Four other Optimization Algorithms. World Academy of Science, Engineering & Technology, Vol. 6, Issue 72, 2012, P. 418-423.
4. Zahadat, P., Schmickl, T. Wolfpack-inspired evolutionary algorithm and a reaction-diffusion-based controller are used for pattern formation. In proceedings of the 2014 Conference on Genetic and Evolutionary Computation (GECCO‘2014), 2014, P. 241-248.
5. Lia, C.-M., Duc, Y.-C., Wua, J.-X., Lind, C.-H., Hoe, Y.-R., Lina, Y., Chen, T. Synchronizing chaotification with support vector machine and wolf pack search algorithm for estimation of peripheral vascular occlusion in diabetes mellitus. Biomedical Signal Processing and Control, Vol. 9, 2014, P. 45-55.

6 Claude E. Duchon. Lanczos Filtering in One and Two Dimensions, 1979

7 Hao Jiang, C. Moloney. A new direction adaptive scheme for image interpolation, 2002.

8 D. Glasner, S. Bagon, M. Irani. Super-Resolution from a Single Image, 2009.

9. Антоненко В. М. Сучасні інформаційні системи і технології: управління знаннями : навч. Посібник / В. М. Антоненко, С. Д. Мамченко, Ю. В. Рогушина. – Ірпінь : Нац. університет ДПС України, 2016, 212 с.
10. Андреас М. Введення в машинне навчання за допомогою Python. Керівництво для фахівців по роботі з даними / Мюллер Андреас // М .: Альфакнига, 2017, 487 с.

11. Бенгфорт Б. Прикладний аналіз текстових даних на Python. Машинне навчання та створення додатків обробки природної мови / Б. Бенгфорт // СПб .: Питер, 2019, 368 с.
12. Годун В.М. Інформаційні системи і технології в статистиці: навч. посіб. / В.М. Годун, Н.С. Орленко, М. А. Сендзюк; за ред. В.Ф. Ситника. – К.: КНЕУ, 2003, 267 с.
13. Каллан, Р. Нейронні мережі: Короткий довідник / Р. Каллан. - К .: Вільямс І.Д., 2017, 288 с.
14. Ніколенко С. Глибоке навчання / С. Ніколенко, А. Кадурін, Е. Архангельська // СПб .: Питер, 2018, 480 с.
15. Плас Д. Python для складних завдань. Наука про дані і машинне навчання. Керівництво / Плас Джейк Вандер // К .: ВМС, 2018, 759 с.
16. Редько В.Г. Еволюція, нейронні мережі, інтелект: Моделі і концепції еволюційної кібернетики / В.Г. Редько // М .: Ленанд, 2019, 224 с.
17. Хайкін С. Нейронні мережі: повний курс / С. Хайкін // М .: Діалектика, 2019, 1104 с.
18. Alvertos Benroumpi. Aff-wild Database and Aff-wild Net. 2018
19. Демидов Є.М. «Застосування біометричного та дактилоскопічного фільтра Габора при оцінюванні якості навчання здобувачів освіти.»/ Демидов Є.М., Нарєжній О.П.// Науково-дослідна робота в системі підготовки фахівців-педагогів у природничій, технологічній і комп’ютерній галузях: матеріали IX Всеукраїнської науково-практичної інтернет-конференції, м. Запоріжжя, 21-22 вересня 2023 року, с. 59–60.

ДОДАТОК А

```
    def __ridge_filter(self):

            final_temp = temp1 & temp2 & temp3 & temp4

            finalind = np.where(final_temp)



            maxorientindex = np.round(180 / self.angleInc)
            orientindex = np.round(self._orientim / np.pi * 180
/ self.angleInc)


            for i in range(0, rows):
                for j in range(0, cols):
                    if (orientindex[i][j] < 1):
                        orientindex[i][j] = orientindex[i][j] +
maxorientindex

            self._binim = < self.ridge_filter_thresh

        def save_enhanced_image(self, path):
            cv2.imwrite(path, (255 * self._binim))

        def enhance(self, img, resize=True):

            if(resize):
                rows, cols = np.shape(img)
                aspect_ratio = np.double(rows) / np.double(cols)

                new_rows = 350
                new_cols = new_rows / aspect_ratio

                img     =     cv2.resize(img,     (int(new_cols),
int(new_rows)))

            self.__ridge_segment(img)
            self.__ridge_orient()
            self.__ridge_freq()
            self.__ridge_filter()
            return(self._binim)

    if __name__ == '__main__':

        image_enhancer = FingerprintImageEnhancer()
        if(len(sys.argv)<2):
            print('завантаження       тестового       зображення
дактилоскопії');
            img_name = '2.jpg'
```

```
            img = cv2.imread('../images/' + img_name)
        elif(len(sys.argv) >= 2):
            img_name = sys.argv[1];
            img = cv2.imread('../images/' + img_name)

        if(len(img.shape)>2):
             img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

        out = image_enhancer.enhance(img)
        image_enhancer.save_enhanced_image('../enhanced/'      +
img_name)
```

ДОДАТОК Б

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНА АКАДЕМІЯ ПЕДАГОГІЧНИХ НАУК УКРАЇНИ
БЕРДЯНСЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІМЕНІ М.П. ДРАГОМАНОВА

Сертифікат

учасника IX Всеукраїнської
науково-практичної конференції

***«Науково-дослідна робота
в системі підготовки фахівців-педагогів
у природничій, технологічній
і комп'ютерній галузях»***

Демидов
Єгор Михайлович

21-22 вересня 2023 року
м. Бердянськ, Україна

Ректор БДПУ
професор, доктор
педагогічних наук

І.Т. Богданов

Демидов Є.М.,
здобувач другого (магістерського)
рівня вищої освіти
(Харківський національний
університет імені В.Н. Каразіна)
Нарєжній О.П.
к.т.н, доцент кафедри безпеки
інформаційних систем і технологій
(ХНУ імені В.Н. Каразіна)

ЗАСТОСУВАННЯ БІОМЕТРИЧНОГО ТА ДАКТИЛОСКОПІЧНОГО ФІЛЬТРА ГАБОРА ПРИ ОЦІНЮВАННІ ЯКОСТІ НАВЧАННЯ ЗДОБУВАЧІВ ОСВІТИ

Сьогодні розпізнавання осіб використовуються в різних системах: біометрична ідентифікація, людино-машинний інтерфейс, зір роботів, комп'ютерна анімація, відеоконференції. Завдання розпізнавання особи включає процес пошуку та обробки особи. Одним із способів виявлення обличчя на зображенні є використання фільтрів Габора.

Цифрова обробка зображень є самостійною областю знань, яка швидко розвивається і охоплює великий спектр методів, які мають дуже широке застосування. Біометрична ідентифікація є додатковим рівнем захисту, оскільки біометричні дані людини складно підробити. Так само біометричні дані незмінні та унікальні для кожної людини, що є їхньою універсальністю. Основна перевага аутентифікації за біометричними параметрами очевидна: дані неможливо забути, втратити, передати іншій людині чи вкрасти, відтворити у повному обсязі.

Використання адаптивного фільтра Габора, як метод підтвердження особи, може сприяти покращенню оцінки якості навчання здобувачів освіти на всіх її рівнях. Цифровізація та впровадження сучасних технологій в систему освіти приводить до випадків, коли об'єктивно неможливо визначити справжність знань здобувача освіти. Створення єдиної бази здобувачів освіти разом з використанням метода автентифікації людини на основі адаптивного фільтра Габора призведе до значного росту якості навчання. Автентифікації людини на етапі оцінки якості освіти за допомогою дактилоскопічних та біометричних зображень призведе до неможливості підробки та фальсифікації відповідей, й може гарантувати прозорість оцінки знань.

Тестування є одним з найефективніших способів оцінки знань і умінь студентів. Застосування комп'ютерного тестування підвищує ефективність учбового процесу, активізує пізнавальну діяльність студентів, дає можливість швидкого зворотнього зв'язку викладача з студентом. Однією з важливих переваг застосування фільтра Габора під час комп'ютерного тестування є достовірність та об'єктивність результатів здобувачів освіти, яка підтверджується дактилоскопічними та біометричними даними здобувачів. Застосування методів автентифікації людини підвищує рівень мотивації навчання, формує готовність та здатність здобувачів освіти до виконання завдань різного характеру, та спонукає більший інтерес до самого процесу навчання. Завдання викладача, організувати учбову діяльність так, щоб отримані знання на заняттях студентів були результатом їх власних пошуків. Але ці пошуки необхідно організувати, при цьому управляти студентами, розвивати їх пізнавальну активність.