A COMBINED MACHINE LEARNING APPROACH FOR IMAGE BASED BANK CHEQUE SIGNATURE VERIFICATION

A Thesis Presented

By

Enkubahir Girmay Gebru

To

The Faculty of Informatics

Of

St. Mary's University

In Partial Fulfillment of the Requirements
For the Degree of Master of Science

In

Computer Science

January, 2024

ACCEPTANCE

A COMBINED MACHINE LEARNING APPROACH FOR IMAGE BASED BANK CHEQUE SIGNATURE VERIFICATION

By

Enkubahir Girmay Gebru

Accepted by the Faculty of Informatics, St. Mary's University, in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Thesis Examination Committee:

_____

Internal Examiner
{Full Name, Signature and Date}

_____

External Examiner
{Full Name, Signature and Date}

_____

Dean, Faculty of Informatics
{Full Name, Signature and Date}

January 2024

DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been duly acknowledged.

_____

Full Name of Student

_____

Signature

Addis Ababa

Ethiopia

This thesis has been submitted for examination with my approval as advisor.

_____Million Meshesha_____

Full Name of Advisor

_____million___Jan 24, 2024_____

Signature

Addis Ababa

Ethiopia

January, 2024

Addis Ababa, Ethiopia

Table of Contents

Acknowledgments

First and foremost, I would like to thank God for giving me the strength and ability to undertake this research study. Without his countless blessing, everything would not have been possible.

I would like to express my sincere gratitude to my advisor Dr. Million Meshesha, for his invaluable guidance and support throughout my master's program. His expertise and encouragement helped me to complete this research and write this thesis.

I must express my very profound gratitude to my parents, brothers, sisters, and friends for providing me with unfailing support and continuous encouragement through-out my years of study, through the process of researching and writing this thesis.

Finally, I would like to thank all of the participants in my study for their time and willingness to share their experiences. This work would not have been possible without their contribution.

List of Acronyms

The following are some abbreviations/Acronyms used in this Thesis document.

AER	----------	Average Error Rate
BCSD	----------	Bank Cheques Segmentation Database
DIP	----------	Digital Image Proccessing
FAR	----------	False Acceptance Rate
FCC	----------	Freeman Chain Code
FRR	----------	False Rejection Rate
GNNs	----------	Graph Neural Networks
JPEG	----------	Joint Photographic Experts Group
KNN	----------	K-Nearest Neighbors
RGB	----------	Red, Green and Blue
SVS	----------	Signature Verification System

List of Figures

List of Tables

Abstract

Signature recognition and verification is a crucial task in the field of biometrics, which aims to identify individuals based on their unique physiological or behavioral characteristics. In recent years, machine learning algorithms have been widely used for signature recognition and verification due to their high accuracy and efficiency. One such algorithm is the K-Nearest Neighbors (KNN) algorithm, which is a non-parametric method used for classification and regression tasks. However, KNN has limitations in handling complex data structures such as graphs. To overcome this limitation, Graph Neural Networks (GNNs) have been proposed as an effective solution for graph-based data. In this research, we propose a signature verification model by combining KNN with GNN algorithms. The proposed model first extracts features from the signature image using Freeman Chain Code (FCC).

We used a signature Database called CEDAR Signature dataset which consists of 792 signatures. These features are then used to train the KNN classifier, which is responsible for identifying the nearest neighbors of a given signature. However, instead of using the traditional Euclidean distance metric, we use a graph-based distance metric that takes into account the structural information of the signature. To further improve the performance of the system, we incorporate GNNs into the KNN classifier. The GNNs are used to learn the underlying graph structure of the signatures and capture their local and global dependencies. This allows the model to handle complex signatures with varying shapes and sizes. We use 82 percent of the data set for training and the remaining for testing. The experimental results show that our proposed model achieves 91 % accuracy and outperforms state-of-the-art methods.

One major constraint/weakness of the study is the potential sensitivity to variations in signature styles and dynamic aspects that are not fully captured by the proposed model. To address this limitation, future research could explore incorporating additional dynamic features such as the velocity and acceleration of the signature strokes. Additionally, considering temporal information and capturing the sequence of strokes in a signature might enhance the model's ability to handle variations in writing styles.

Keywords: Signature Recognition; Signature Verification; Digital Image Processing; Freeman Chain Code; Machine Learning

CHAPTER ONE

INTRODUCTION

1.1.Background of the study

Biometrics, often known as biometric authentication, is the identification of people based on their qualities or characteristics [1]. Biometrics is used for identification and access control in computer science. It's also used to find individuals in groups that are under observation. Biometric identifiers are unique characteristics that may be measured and utilized to classify and identify individuals. [2]. Physiological characteristics are often separated from behavioral characteristics in the classification of biometric identifiers. The physical characteristics of the body are linked to its shape. A few instances are iris recognition, fingerprint, facial recognition, DNA, palm print, hand geometry, retina, and smell/odor. [3]. A person's pattern of conduct can be inferred from a variety of behavioral indicators, including voice, movement, and typing rhythm. Some researchers have named this latter subset of biometrics "behaviometrics". More traditional approaches to access control include knowledge-based identity systems, such as passwords or personal identification numbers, and token-based identity systems, such as passports and driver's licenses. Biometric IDs are unique to each individual, making them a more dependable method of identification confirmation than token- and knowledge-based methods. But gathering biometric identifiers creates privacy concerns about what may happen to this data later on. [1].

A signature is a human biometrics that can alter owing to a variety of circumstances such as age, mood, and environment, which implies that two signatures from the same individual will not completely match [3]. For such a case, a Signature Verification System (SVS) is a viable option. The signature is a behavioral biometric attribute that includes the signer's neuromotor qualities as well as socio-cultural influences, and it is widely acknowledged as an identity identification method [3]. Online (dynamic) and offline (static) techniques can both be used to verify handwritten signatures [3][2]. Electronic gadgets such as a writing pad or a stylus connected to a computer can be used to capture an online signature. Only the digital signature

will be available in an offline signature, from which essential elements can be derived. A forged signature is a signature that appears to be fake.

Signature Verification is done by comparing one-to-one processes, which comprise data collecting, and preprocessing, feature extraction, and verification. It is critical in forensic, security, and resource access control applications such as banking, money laundering, and marriage approval. [6] Signature of a banker or a customer can be broadly divided into two categories: Initials and Full signature [7]. An initial signature is a very simplified form of a signature where the signatory has sealed the document. These initials are typically not accepted by bankers for a range of transactions. These are used on instrument copies and in internal business operations. Frequently, these signatures are not retained on file by institutions as samples. The Full signature, on the other hand, is one that is kept as a sample by the banker and is commonly used in financial transactions. Bank employees match these sample signatures to the signatures on the instruments. When a customer is a corporation or an institution, they also sign with their company seal or stamp.

The term 'bank cheque' describes a cheque that is issued by a bank. A cheque is a written, dated, and signed document that instructs a bank to pay the bearer a certain amount of money [7]. The person or entity writing the check is called the payor, sometimes known as the drawer, and the person or entity to whom it is written is known as the payee. The drawee is the bank that the cheque is drawn on. Several investigations have been conducted on this. Bank checks are normally treated by the law in the same way as regular checks. A bank check may not always be paid, despite the fact that some people consider them to be the same as cash. Even though these research have advanced the subject of signature verification significantly, it is still challenging to discern between skillful forgeries and genuine signatures. [1][7].

Identification is often required to enter many of the places we visit on a daily basis. Examples of verification include ATMs, credit cards, and internet accounts [9]. He or she must first present identification before attempting to access them. Typically, two types of identification are used for login: an alphanumeric password (occasionally with extra characters) and a Personal Identification Number (PIN). The primary shortcoming of the method is that it only offers "verification," not "identifying." These two terms are not interchangeable, despite their similarities. By determining if a password is correct or wrong, a system "verifies" it. It

makes no attempt to "identify" who has access. As a result, anyone can easily access and change the data of another individual if they know their password. There are two different types of Signature verification system (SVS): [4][5][1] static (offline) and dynamic (online). In an offline system the user signs a sheet of paper , which is subsequently scanned or captured on camera. To identify a signature, the SVS examines its static features or form. In contrast, an online approach needs the user to sign using a digitizing tablet, which instantly records the signature. Acquisition via PDAs using a pen, tablet, or smartphone is an additional choice. Dynamic signature features that are more difficult to forge can be captured by an online system. A web-based system is appropriate for real-time applications including office automation, document authentication, and finance. Only offline verification is the subject of this investigation.

Signature forgery is a crime committed when a person signs another party's name or alters a document in order to commit fraud or deceive others [12]. One common example of signature fraud is writing checks. When someone signs a check in the name of another person without that person's permission and the other person does not object, it is known as signature forging. Similarly, signature forging is the act of signing another person's name without that person's permission on a range of various written documents. The intent or motive behind an individual's signature on a document is a critical factor in determining whether or not the signature is being forgeried. Most countries require the signer of an act to have some form of ulterior intent or planned scheme in order for it to be considered a forgery. If you have the other person's permission, signing their name is not illegal.

Figure 1: sample bank cheque

A complete document created by someone else and signed with that person's name is considered a forgery and is illegal. This holds valid even if the signed document was created fraudulently in its entirety. However, if someone adds, removes, or changes any part of a properly signed document without the signer's knowledge or consent, many legal systems also consider that to be forging. Moreover, a document does not have to be a contract or other official legal documentation in order for it to be forged. A letter may also be considered fraudulent if it has been altered or falsely signed.

Signature forgeries fall into three categories [7][1][8]. First, there is **simple forgery**, when the forger simply does not know that a signature even exists; in reality, the forger does not try to mimic or trace a genuine signature. On a check without a reference signature, the fake signature is incredibly different from the original. Financial institutions are usually impacted when a firm accepts a fraudulent check and deposits it into their account at your bank. Forensicing one's own signature is known as **random forgery**, and it is the second kind. It only signifies that the counterfeit signature provided by the thief was not authentic.For instance, this kind of forgery happens when someone steals a box of checks from an unknown person but lacks any signed checks to base their counterfeit on. The final type is **skillful forgery**, in which the forger mimics or duplicates the real signature as closely as conceivable. Simulated freehand work is used in skillful forgery. A writing sample or even the target's actual signature is studied by the forger. When they recreate the victim's signature, they imitate the shapes and strokes that were employed in the original writing. However, the forger cannot typically conceal their own patterns. They frequently incorporate components from their own signatures, which removes the victim's signs of validity from the fake.

However, a signature needs to be seen as an image since writers sometimes enclose their names in groupings of characters, symbols, or letters, making it appear as though the writer is not using their name. An image is a depiction of something in visual form; it's usually a photograph or two-dimensional picture. It can be made with a variety of media, including computer tools, paint, sketching, and photography. Images are employed to tell stories, arouse feelings, and freeze moments in time. Multiple image formats, including JPEG, PNG, GIF, and BMP, are available for storage. [13]. These formats determine the quality and size of the image. JPEG is the most commonly used format for photographs due to its high compression rate and ability to

maintain image quality. PNG is used for images that require transparency while GIF is used for animated images.

Image processing is a method of performing operations on an image to extract useful information or enhance its visual quality. It involves various techniques such as image filtering, segmentation, feature extraction, and object recognition. Image processing has numerous applications in various fields such as medical imaging, remote sensing, surveillance, and robotics. [19][20] One of the most common techniques used in image processing is image filtering [26]. Image filtering is a process of modifying the pixels of an image to achieve a desired effect. There are various types of filters such as low-pass filters, high-pass filters, median filters, and Gaussian filters [26]. Low-pass filters are used to remove high-frequency noise from an image, while high-pass filters are used to enhance edges and details in an image. While Gaussian filters are used to blur an image, median filters are used to remove noise from images such as salt and pepper. Segmentation is a crucial technique in image processing. The process of segmenting an image into different parts or segments according to their attributes, including color, texture, or intensity, is known as segmentation. Applications for segmentation include computer vision, medical imaging, and object recognition. Another crucial method in image processing is feature extraction. The process of feature extraction is removing pertinent elements from an image so that they can be utilized for additional research or classification. It is possible to extract features according to their color, texture, or shape. The process of recognizing things in an image by their features is called object recognition. [22].

In image processing, machine learning algorithms play a pivotal role, particularly in the domain of image recognition. Image recognition involves the identification and classification of objects or patterns within an image [18]. Machine learning techniques bring a data-driven approach to this task, allowing systems to automatically learn and adapt to the inherent complexity and variability of visual data. These algorithms can be broadly categorized into supervised and unsupervised learning methods [16]. In supervised learning, the algorithm is trained on a labeled dataset, where each image is associated with a specific class or category whereas in Unsupervised learning the main goal is to discover patterns or groupings in the data without predefined labels [16]. Machine learning models can adapt to variations in lighting, orientation,

and other factors that may affect image quality. Also they can automatically learn intricate patterns and representations that may be challenging to hand-craft.

1.2.Statement of the Problem

Nowadays when a customer opens an account at any bank, the bank asks for a prototype of the customer's signature, which has been entered into the bank's system. When a customer writes a check, the bank teller manually compares the customer's signature on the check to the signature prototype. If the signature on the cheque does not match the system's prototype signature, the teller refuses to process the check and it may be returned. If the signature is matched, the check can be forwarded for processing. However, this is a manual procedure which is prone to error, inconsistent and inefficient during service delivery. When a customer deposits a cheque or gives over a cheque with an unauthorized signature to a bank teller, the teller can only examine data fields and, to some extent, the signature style from the signature prototype. However, he/she may be unable to recognize small signature flaws, resulting in the cheque being cleared. It is possible that the fraudulent transaction will take place here. It's impossible to verify signatures on all checks in a timely manner. Some solutions for this problem have been defined because signatures are a sort of biometric that can alter with mood, environment, and age. A decent signature database must be updated at a few particular intervals in order for it to remain current and useful. Furthermore, a person must sign in a consistent manner in order to create a succession of signatures that are almost identical. This age-old method for signature verification is manual one, in which a person manually checks the signature. If both the signatures are sufficiently similar, then the access is granted. An automated signature verification process will help improve the scenario. Signature recognition and verification is a critical aspect of identity management, particularly in fields such as finance, healthcare, and legal sectors [6]. This technology aims to identify and authenticate signatures for various purposes, including fraud prevention, secure access, and document authentication.

Several research studies have been conducted to improve the accuracy and efficiency of signature recognition and verification systems. Wu, Xing, and Bir Bhanu [35] have investigated the dynamics and variability of signatures to develop more robust recognition and verification algorithms. They also have explored various feature extraction and classification techniques to enhance the accuracy of signature recognition and verification systems. These methods involve

extracting specific characteristics from signatures and classifying them to identify or authenticate the signatures. Recent advancements in deep learning and convolutional neural networks (CNNs) have led to significant improvements in signature recognition and verification accuracy. Malik, M. I., Liwicki, M., Alewijnse, L., Ohyama, W., Blumenstein, M., & Found, B. [36] have explored the use of CNNs for feature extraction and classification, as well as developing novel deep learning architectures for signature recognition. Researchers have also investigated ensemble and multi-modal approaches to improve the accuracy and reliability of signature recognition and verification systems. These approaches involve combining different recognition and verification algorithms or using additional biometric information, such as facial or voice recognition, to enhance the overall performance.

In Ethiopia, research on signature recognition and verification is an emerging area of interest due to the increasing need for secure authentication systems in various sectors such as banking, government, and legal processes. While there may not be an extensive list of specific research studies solely focused on signature recognition and verification in Ethiopia, there are broader studies related to biometric authentication and security measures that encompass this field. These studies often provide valuable insights into the development and implementation of signature recognition and verification technologies in the Ethiopian context. While specific research solely focused on signature recognition and verification in Ethiopia may be limited, exploring related areas such as biometric authentication, technological advancements, legal considerations, and cultural influences provides a comprehensive understanding of the landscape surrounding this topic within the Ethiopian context.

It is therefore the aim of this study to investigate and develop image based bank cheque verification system using machine learning algorithm. To this end, this study answers the following research questions.

- What are the suitable image processing tasks and methods to apply for preparing data set?
- How K-nearest neighbor and Graph neural network are combined for signature recognition and verification?
- To what extent the proposed model performs in signature recognition and verification?

1.3.Objectives of the study

1.3.1. General objective

The general objective of this study is to construct a combined model of k-nearest neighbor along and graph neural network for an image-based signature recognition and verification using freeman chain code.

1.3.2. Specific objectives

- To review related works so as to identify suitable methods and techniques for the current work
- To collect and prepare data set for experimentation
- To select suitable algorithms for image processing and machine learning
- To propose an optimal model by combining k-nearest neighbor along with graph neural networks
- To evaluate the performance of the proposed model for signature recognition and verification

1.4.Significance of the study

- It will save time and energy of customers and employees
- It will help to prevent human error during the signature verification as well as the check payment process
- It will lower chances of fraud in the process of authentication.
- It will maintain human privacy and security in banking industry.
- It will increase accuracy and efficiency of signature verification
- It will verify the accuracy of signatures in a bank checks using a set of previously collected signatures (specimens).
- It will reduce the time required for Signature verification and recognition.
- It will enable bank tellers highly resist different kinds of impostors.
- It will accurately characterize each user's signature
- It will offer good verification and recognition performance.
- For researchers it will play a crucial role in advancing knowledge and improving understanding of various phenomena while using a hybrid model. Its significance lies in its ability to address the current gap, develop new models and algorithms.

1.5.Scope and limitation of the study

This study focuses on offline (static) signature verification which is based on static characteristics of the signature which are invariant. In offline signature system no need to use special hardware, ubiquitous etc. this study will investigate on creating new algorithms to find the best way to get good results in verifying the bank cheque signature using **freeman chain code** (a lossless compression algorithm used to represent a boundary by a connected sequence of straight line segments) and **k-nearest neighbor** (supervised machine learning algorithm used to solve both classification and regression problems)along with **artificial neural network**(a computational network based on biological neural networks that construct the structure of the human brain). The scope of using KNN with GNN for bank cheque signature verification is vast. It can be used by banks and financial institutions to prevent fraud and ensure the security of their transactions. The use of machine learning algorithms such as KNN and GNN can significantly reduce the time and effort required for manual signature verification, thereby increasing efficiency and accuracy.

However, there are also limitations to this approach. One limitation is that the accuracy of the model depends on the quality and quantity of data used for training. If the dataset used for training is not representative of the population, the model may not be able to accurately identify forged signatures. Additionally, there may be cases where genuine signatures are misclassified as forged ones or vice versa, leading to errors in the verification process. Another limitation is that this approach may not be effective against sophisticated forgery techniques that can mimic genuine signatures with high accuracy. In such cases, additional security measures such as biometric authentication may be necessary.

We have used a signature dataset with real and forged ones called CEDAR Signature dataset which is a database of off-line signatures for signature verification. Each of 22 individuals contributed 24 signatures thereby creating 528 genuine signatures. Some were asked to forge three other writers' signatures, four times per subject, thus creating 264 forgeries. Altogether there are 792 signature samples. Each signature was scanned at 300 dpi. The database has 24 genuine and 12 forgeries available for each writer.

1.6.Organization of the Thesis

This thesis document is organized as follows. Chapter one discuss the background of the study, Statement of the Problem, Objective, Significance and Scope and limitations of the study. Chapter Two discuss previously conducted research works on finding better results using different algorithms and image processing tools and state of the art approaches for signature verification. My proposed approach (methodology) for signature verification is elaborated in Chapter Three. The experimental setups, procedures evaluation metrics, results, and discussion are discussed in Chapter Four. Finally, Chapter five discusses the conclusion and future research direction on signature verification system.

CHAPTER TWO

LITERATURE REVIEW

2.1.Overview

The existing method of capturing images in remote sensing of the Earth using aircraft or satellite-based sensors is currently in digital format. The pixels represent localized spatial information, and the quantization levels in each spectral band correspond to the quantized radiometric measurements. It is most reasonable to view each image as a vector array, where pixels are arranged on a rectangular grid. The value of the image at each pixel is a vector whose elements correspond to radiometric levels, also known as intensity values or digital numbers, of the different bands. For each band, the resulting image is monochrome, and hence, the quantized values of such an image are referred to as gray levels. These images can be acquired in a few bands, forming a multispectral image, or in hundreds of bands, forming a hyperspectral image. Image processing operations for both types of images can be either scalar image-oriented, where each band is processed independently as a separate image, or vector image-oriented, where the operations consider the vector nature of each pixel. Image processing can occur at various levels. At its most fundamental level, the processing enhances an image or brings attention to specific objects for the analyst to observe. At higher levels, processing can involve automatically detecting objects in the image and classifying them.

2.2.Digital Image processing

Digital image processing, a subset of digital signal processing, has experienced significant advancements across various domains due to its extensive applications [17]. This field involves employing computer algorithms to enhance different aspects of an image, making it a crucial tool for image improvement. The primary focus in image processing lies in enhancing the quality, often referred to as clarity, through various techniques.

One such technique is image interpolation, where interpolation methods determine function values between sampled positions. Numerous interpolation techniques have been documented in the past. Image processing techniques have gained substantial importance in enhancing low-resolution images from sources such as CT scans, MRI, geographical images, mobile phones,

and satellites. These techniques allow for image resampling to either decrease or increase resolution, and the quality of the processed image is influenced by the chosen interpolation method. Over the last decade, various image processing techniques have been developed, including image restoration, filtering, compression, segmentation, and more. [13].

2.3.Steps in digital image processing

Digital image processing is the use of computer algorithms to perform operations on digital images. The process involves several steps that are essential in transforming raw image data into a form that can be analyzed and interpreted by humans or machines. [16][17] The following are the steps involved in digital image processing, as shown in figure 2 below.

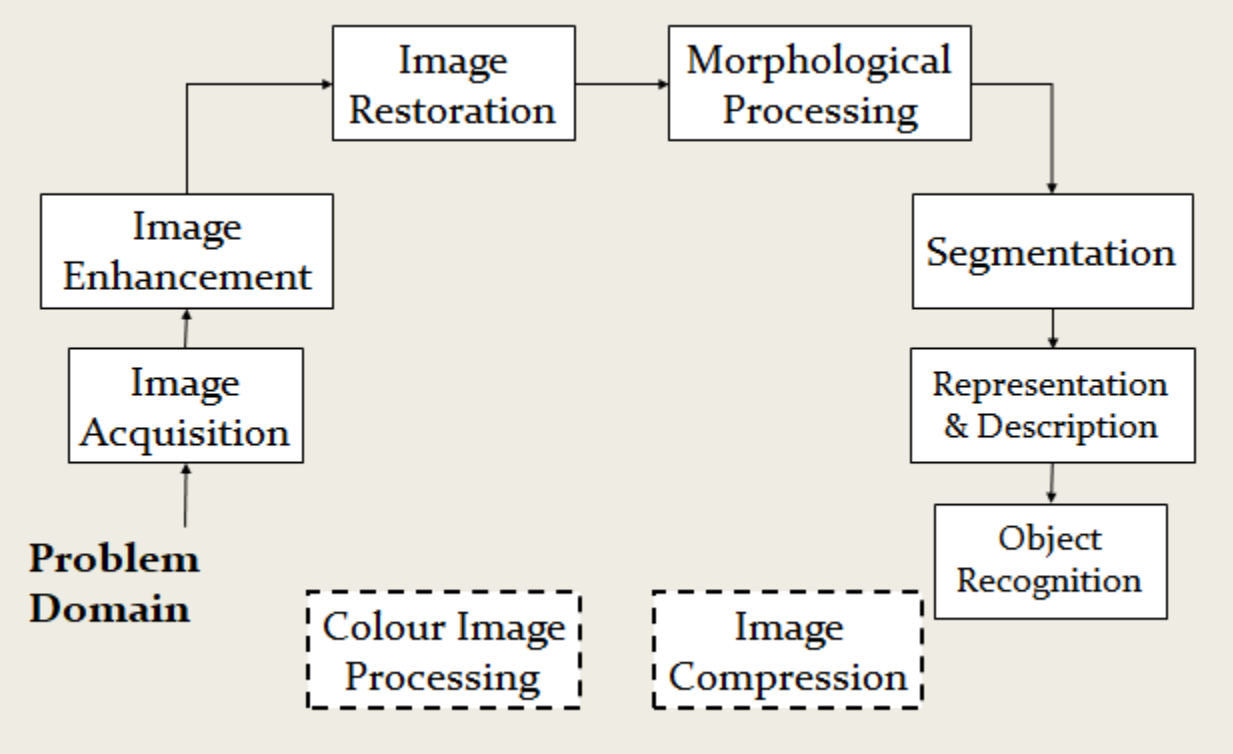

Figure 2: digital image processing stages

1. **Image Acquisition**: This is the first step in digital image processing, which involves capturing an image using a digital camera or scanner. The quality of the image captured at this stage determines the quality of the final output.

2. **Image Enhancement**: This step involves improving the quality of the image by removing noise, adjusting brightness and contrast, and sharpening edges. Image enhancement techniques include filtering, histogram equalization, and edge detection.

Filtering, histogram equalization, and edge detection are distinct techniques used in image processing with different objectives [23][24]:

Filtering - is a process used to enhance or suppress certain features or patterns within an image. It involves applying a mathematical operation to each pixel in the image to modify its intensity based on the values of neighboring pixels. Filtering can be used for various purposes such as noise reduction, edge enhancement, and feature extraction.

Histogram equalization - is a technique used to enhance the contrast of an image by adjusting the intensity distribution of its pixels. The method works by redistributing the intensity values of the pixels so that the histogram of the output image becomes more uniform. This results in an image with improved contrast and visual appearance.

Edge detection - is a fundamental technique in image processing used to identify boundaries within an image. The goal of edge detection is to locate significant changes in intensity, which often corresponds to object boundaries or transitions between different regions in the image.

While filtering and histogram equalization focus on modifying pixel intensities globally across the entire image, edge detection specifically targets identifying local changes in intensity that represent edges or boundaries [23][24].

3. **Image Restoration**: This step involves removing distortions caused by factors such as motion blur, camera shake, or atmospheric turbulence. Image restoration techniques include deblurring, denoising, and inpainting.

Deblurring - is a technique used to remove blurriness or distortion from images. This process is particularly helpful when dealing with images that have been captured through a camera lens or other imaging devices that may introduce blur or motion artifacts. Deblurring algorithms work by estimating the original sharp image from the blurry version, considering factors such as the type of blur, the camera motion, and the image content [25].

Denoising - is the process of removing noise from an image or signal, making it clearer and more visually appealing. Noise can be introduced during image acquisition, transmission, or storage, and it can significantly degrade the quality of the image. Denoising algorithms work by identifying and eliminating noise components while preserving the important image features, such as edges, textures, and contours [25].

Inpainting - is a technique used to fill in missing or damaged regions of an image, typically due to scratches, tears, or other defects. The goal of inpainting is to restore the image to its original state, making it appear as natural and seamless as possible. Inpainting algorithms work by estimating the content of the missing regions based on the surrounding image context, such as texture, color, and structure [25].

Generally, deblurring, denoising, and inpainting are three essential image processing techniques that address different types of image degradation. Deblurring aims to remove blurriness, denoising focuses on eliminating noise, and inpainting deals with filling in missing or damaged regions of an image. Each technique has its specific goals and approaches, and they can be used individually or combined to enhance the overall quality of an image [25].

4. **Color Image Processing**: This step involves manipulating color images to enhance their visual appearance or extract useful information from them. Techniques used in color image processing include color correction, color segmentation, and color feature extraction.
5. **Image Compression**: This step involves reducing the size of an image while preserving its quality. Image compression techniques include lossless compression and lossy compression.
6. **Image Segmentation**: This step involves dividing the image into regions or segments based on their characteristics (e.g., color, texture). Segmentation techniques include thresholding, clustering, and region growing.

Thresholding - is a technique used in image processing to convert an input image into a binary image. This is done by setting a threshold value, which separates the pixels of the input image into two categories: foreground (pixels with values greater than the threshold) and background (pixels with values less than the threshold). Thresholding is useful in various applications, such as OCR (Optical Character Recognition), document analysis, and medical image analysis.

In the process of thresholding, an image is first converted to a gray scale representation, and then a threshold value is chosen. This value can be manually set or determined using various techniques such as Otsu's method, which adaptively selects the optimal threshold value to maximize the between-class variance. Once the threshold value is chosen, all pixels with values greater than or equal to the threshold are considered as the foreground, and the remaining pixels are considered as the background [26].

Clustering - is an unsupervised learning technique used to group similar data points together. It is commonly used in image processing for segmentation and pattern recognition. Clustering algorithms analyze the input data and identify natural groupings or clusters based on similarity measures, such as distance or correlation.

In the context of image processing, clustering can be used to identify and group pixels with similar characteristics, such as color, intensity, or texture. This can be useful for segmenting an image into different regions, detecting objects, or identifying patterns within the image. There are several clustering algorithms available, such as K-means, hierarchical clustering, and DBSCAN. Each algorithm has its own advantages and limitations, depending on the specific application and the characteristics of the data [26].

Region growing - is a seeded region-based image segmentation technique that iteratively expands an initial seed point to form a homogeneous region. It is used to segment images by grouping pixels based on their similarity in terms of color, intensity, or texture. Region growing algorithms work by selecting an initial seed point and then adding neighboring pixels to the growing region if they meet a predefined similarity criterion. This process continues until no more pixels can be added to the region or a stopping criterion is met.

In comparison to thresholding and clustering, region growing is more adaptive and can handle images with varying intensity or color distributions. However, it can be computationally expensive, especially for large images or high-dimensional data [26].

7. **Representation and Description**: representation and description are two important steps in DIP that are used to extract meaningful information from digital images. Representation involves converting an image into a format that can be easily processed by a computer, while

description involves extracting useful information from the image using techniques such as feature extraction.

Here are some common feature extraction methods for signature recognition and verification [27]:

Histogram of Oriented Gradients (HOG):

HOG is commonly used in object detection and recognition tasks, including signature recognition. It analyzes the distribution of gradient orientations in local regions of an image. Signatures can be represented by the gradients in different parts of the signature, capturing important texture and shape information.

Scale-Invariant Feature Transform (SIFT):

SIFT is a feature detection algorithm that identifies key points and extracts descriptors. It is invariant to image scale and rotation, making it suitable for signature verification. SIFT captures distinctive local features such as loops and endpoints in a signature.

Local Binary Patterns (LBP):

LBP is a texture descriptor that captures patterns in local neighborhoods. It is effective for describing the texture of signatures and can be used in conjunction with other features. LBP is computationally efficient and robust to changes in illumination.

Zernike Moments:

Zernike moments are used for shape-based recognition and are particularly useful for capturing the global shape characteristics of signatures. They provide a set of complex moments that describe the shape of the signature in a rotation-invariant manner.

Discrete Wavelet Transform (DWT):

DWT decomposes an image into its frequency components at different scales. It is used for capturing both fine and coarse details in signatures. The wavelet coefficients can be used as features for signature recognition.

Gabor Wavelet Transform:

Gabor wavelets are effective in capturing texture and frequency information in an image. Gabor filters at different scales and orientations can be applied to signature images to extract relevant features.

Principal Component Analysis (PCA):

PCA is a dimensionality reduction technique that can be applied to signature features. It helps in capturing the most significant variations in the data, reducing the computational load and improving recognition performance.

Radon Transform:

The Radon transform is used for extracting features related to the spatial distribution of ink in a signature. It is particularly useful for detecting changes in line thickness and intensity distribution.

Freeman Chain Coding (FCC):

Freeman chain coding, also known as Freeman code or Freeman chain, is a technique used for representing the contour of an object or a shape. It is particularly useful for representing the spatial relationship between consecutive points along the boundary of a shape. While Freeman chain coding is not as commonly used as some other feature extraction methods in modern signature recognition systems, it can still be applied to capture the shape characteristics of signatures.

Freeman chain coding represents the contour of an object by encoding the directions of transitions between consecutive points along the boundary. It uses a set of directions corresponding to the eight possible movement directions in a 2D grid (up, down, left, right, and diagonals). Each transition between two adjacent points in the contour is encoded with a numerical value corresponding to the direction of the movement. The resulting sequence of codes forms a compact representation of the object's shape.

Advantages of Freeman chain coding

Freeman chain coding offers several advantages, making it a useful technique in certain applications, particularly for representing the contours of objects. Here are some advantages of Freeman chain coding [27]:

- Compact Representation:

Freeman chain coding provides a compact representation of the shape or contour of an object. The encoding consists of a sequence of codes that describe the directions of transitions between consecutive points along the boundary, resulting in a concise representation.

- Memory Efficiency:

Due to its compact nature, Freeman chain coding is memory-efficient. This makes it suitable for applications where storage space is a concern, especially in scenarios with large datasets.

- Rotation and Translation Invariance:

Freeman chain codes are invariant to translation and rotation transformations. This property makes them useful in scenarios where the orientation or position of the object may vary.

- Simplicity and Computationally Efficient:

The algorithm is relatively simple and computationally efficient. It involves a straightforward process of encoding the directions of transitions between points, making it suitable for real-time applications or scenarios with limited computational resources.

- Robust to Scaling:

Freeman chain coding can be less sensitive to scaling compared to certain other shape representation methods. While not completely scale-invariant, it can handle some degree of scaling variations.

- Suitable for Contour Representation:

Freeman chain coding is well-suited for representing the contours of shapes and objects. It captures the sequence of movements along the boundary, providing a distinctive description of the shape.

- Applicability to Certain Recognition Tasks:

Freeman chain coding is commonly used in applications where the overall shape or contour of an object is critical, such as character recognition, signature recognition, and certain types of shape matching [27].

8. **Object Recognition**: This step involves identifying objects in the image and classifying them based on their features (e.g., shape, texture). Object recognition techniques include template matching, feature extraction, and machine learning.

In this study an attempt is made to apply machine learning algorithms for signature recognition and verification since it can be trained to make classifications or predictions, and to uncover key insights in the given signature data.

2.4.Machine learning

Machine learning (ML) is the study of algorithms and statistical models utilized by computer systems to perform specific tasks without explicit programming. Learning algorithms find application in various daily tasks, such as web search engines like Google, which employ learning algorithms to rank web pages effectively. These algorithms are utilized in data mining, image processing, predictive analytics, and other applications. The key advantage of machine learning lies in the ability of algorithms to automatically perform tasks once they have learned how to handle data [28].

Machine learning tackles the challenge of constructing computers that can enhance their performance through experience. Positioned at the intersection of computer science and statistics, it forms the core of artificial intelligence and data science, experiencing rapid growth. Progress in machine learning results from the development of new algorithms and theories, as well as the abundant availability of online data and cost-effective computation. The widespread adoption of data-intensive machine learning methods is evident in various fields such as

healthcare, manufacturing, education, financial modeling, policing, and marketing, leading to more informed decision-making [28].

Machine learning encompasses both supervised and unsupervised learning [1]. Supervised learning involves training a function to map input to output based on labeled examples. The algorithms require external assistance, with the input dataset divided into training and test datasets, where the training dataset includes output variables to be predicted or classified. In contrast, unsupervised learning lacks correct answers and a teacher. Algorithms autonomously discover and reveal interesting structures in the data. Unsupervised learning algorithms learn features from the data and, when introduced to new data, use previously acquired features to identify data classes. This type of learning is primarily employed for clustering and feature reduction.

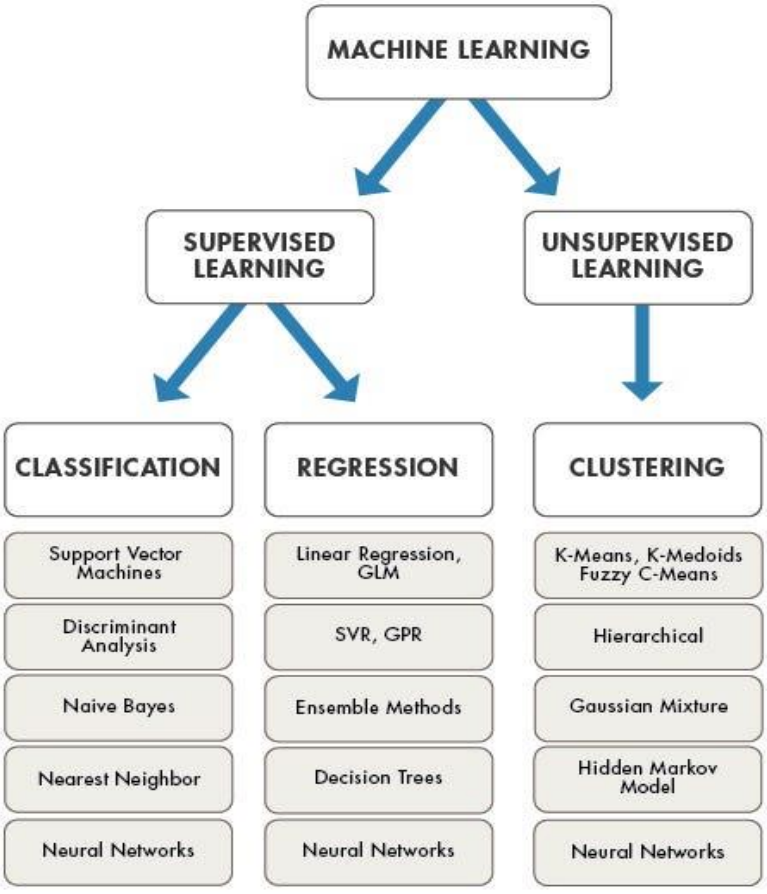

Figure 3: Major machine learning algorithms [23]

2.5.Supervised learning algorithms

Supervised learning is a category of machine learning algorithms that employ labeled datasets for training models or algorithms. The primary objective of these algorithms is to acquire a mapping from input data to output labels, enabling them to make predictions or classifications on novel, unseen data. These techniques utilize a set of training data with known inputs and desired outputs to learn a model capable of making accurate predictions on new data. Supervised learning algorithms find extensive application in diverse fields such as natural language processing, computer vision, and finance. This thesis delves into several prominent supervised learning algorithms, including linear regression, logistic regression, support vector machines, decision trees, k-nearest neighbors, and neural networks.

Linear Regression:

Linear regression, a widely utilized supervised learning algorithm, models the relationship between a continuous dependent variable (e.g., house prices) and one or more independent variables (e.g., square footage of a house). It assumes a linear correlation between these variables and aims to fit a straight line through the data points for making predictions. Linear regression finds application in economics, finance, and statistics, facilitating the understanding and prediction of relationships between variables.

Logistic Regression:

Logistic regression is a classification algorithm employed when the dependent variable is categorical, such as the presence or absence of a disease or the success or failure of a specific task. This algorithm estimates the probability of the event occurring based on the independent variables. Unlike linear regression, logistic regression uses a logistic function to model the relationship between variables. It is widely utilized in fields like medicine, marketing, and fraud detection.

Support Vector Machines (SVM):

Support Vector Machines are a supervised learning algorithm suitable for both classification and regression tasks. They seek the optimal hyperplane that separates data points of different classes, aiming to maximize the margin between the hyperplane and the closest data points (support

vectors). SVMs have demonstrated success in diverse domains, including text classification, face recognition, and bioinformatics.

Decision Trees:

Decision trees, a popular supervised learning algorithm, utilize a tree-like structure to represent decision rules learned from training data. At each node, a test on an independent variable is conducted, and based on the outcome, the data is partitioned into subsets. This process continues until all data points in a leaf node belong to the same class. Decision trees find applications in credit scoring, medical diagnosis, and fraud detection.

k-Nearest Neighbors (k-NN):

k-Nearest Neighbors is a supervised learning algorithm based on instance-based learning. The algorithm stores all available input examples and classifies new examples by comparing them to the stored examples. k-NN classifies a new data point based on the majority class of its k-nearest neighbors. Users typically choose the value of k, which commonly ranges from 1 to 20.

Neural Networks:

Neural networks, inspired by the structure and function of the human brain, represent a type of machine learning algorithm. Comprising interconnected nodes called neurons organized in layers, neural networks pass information through these layers. Each neuron computes a weighted sum of its inputs, applying a nonlinear activation function to generate its output. Neural networks are versatile, applicable to various tasks such as classification, regression, and even unsupervised learning. They excel in handling problems with high-dimensional input data, capturing complex relationships between inputs and outputs.

Advantages of K-Nearest Neighbors (KNN)

Flexibility: KNN is highly flexible and can be used for both classification and regression tasks. It can adapt to various types of data, including numerical, categorical, and even textual data. This makes it a suitable choice for a wide range of applications.

Easy to implement: KNN is relatively simple to implement compared to more complex algorithms like neural networks. This simplicity makes it an attractive option for beginners in machine learning and data science.

No training phase: Unlike many other machine learning algorithms, KNN does not require a training phase. This means that it can be applied to new data immediately without the need for additional preparation or retraining.

Robust to outliers: KNN is robust to outliers, as it takes into account the proximity of data points in the feature space. This means that it can handle noisy or incomplete data better than some other algorithms.

Non-parametric: KNN is a non-parametric algorithm, which means that it does not make any assumptions about the underlying distribution of the data. This can be an advantage when working with data that does not conform to standard statistical assumptions.

Advantages of Neural Networks

Ability to learn: Neural networks can learn from data and improve their performance over time. This is because they can adapt their weights and biases based on the input data, which allows them to generalize well to new, unseen data.

Scalability: Neural networks can be scaled to handle large datasets and complex tasks. They can be designed with multiple layers and a large number of nodes, which can help improve their performance on more complex problems.

Fault tolerance: Neural networks are fault-tolerant, meaning that they can still function and provide reasonable results even if some of their nodes or connections are damaged or not functioning correctly.

Parallel processing: Neural networks can be efficiently implemented using parallel processing techniques, which can significantly reduce computation time and improve performance.

Versatility: Neural networks can be used for a wide range of tasks, including classification, regression, pattern recognition, and even natural language processing.

Generally, K-nearest neighbors and neural networks are two powerful machine learning algorithms that offer unique advantages for different types of problems. KNN is a simple and flexible algorithm that is suitable for a wide range of applications, while neural networks excel at handling complex, non-linear problems and can be efficiently parallelized. By understanding the strengths and weaknesses of each algorithm, the researcher makes an informed decision about which method to use for a given problem.

2.6.Related works

In this area many researches have been and are being conducted by different scholars using different algorithms and image processing tools to get better results on signature verification and identification of the real signer. For instance the three Malaysians Aini Najwa Azmi, Dewi Nasien & Fakhrul Syakirin Omar have conducted a research on signature verification on 2016. They stated that SVS can be decomposed into three major stages: data acquisition and preprocessing, feature extraction and verification. Data acquisition is the process of sampling signals that measure real physical conditions and converting the resulting samples into digital values that can be manipulated by a computer, for example in varying color, gray level, or binary format. MCYT Bimodal Offline Signature database was used in their entire stages. There were 75 users in this database, each having 15 genuine signatures and 15 forgery signatures.

In preprocessing, the image was converted to binary values. Noise removal was applied to the signature images before cropping. Finally, thinning algorithm is used to remove all redundancy by eliminating excess foreground pixels. In converting raw binary image to thinned binary image (TBI), thinning function in Image Processing Toolbox in MATLAB software is used. As the first important stage, image and data preprocessing performs the purpose of extracting regions of interest, enhancing and cleaning up the images, so that they can be directly and efficiently processed by the feature extraction component in the next stage. In image processing, feature extraction is a special form of dimensionality reduction. When the input data to an algorithm is too large to be processed and it is suspected to be redundant, it was transformed into a simplified representation set of feature vector by carefully choosing relevant information from the input data. In order to perform this important task, Freeman chain code (FCC) was used, constructed using boundary based style.

Lastly, k-NN was used as a classifier, chosen because this classifier is performing excellently in pattern recognition system. The first one, k-NN, has been proven to perform well in previous researches and the generalization is in the given distance measure. Almost all verifiers depend, in one way or the other, on a given distance measure. K-NN is advantageous in that it reflects the human decision making because it is only based on the distance measure designed by the researcher. K-NN also does not involve a lot of parameters like other verifiers. The performance quality is measured by error rates such as FAR (False Acceptance Rate) and FRR (False Rejection Rate), and AER (Average Error Rate).

Based on the experimental results it was recorded that, the lowest error rate for FRR and FAR were 6.67% and 12.44% with AER 9.85% which is better in term of performance compared to other works using that same database.

Another well-formed research was Biometric Signature Processing & Recognition Using Radial Basis Function Network which was conducted by Satam (2010), Chadha (2010) and Wali (1997). First, new image, also known as user image is made to undergo Rotation-Scaling-Translation removal, so that it resembles original signature stored in database. Database is created by collecting 700 signature samples. These signatures are stored in database after feature extraction process for which DCT is employed. It is also applied on corrected user image. Now this image is provided to Radial Basis Function Network (RBFN), which is earlier trained using database. If RBFN identifies user image as the one in database then access for that particular individual is granted.

The advantage of this system is that it uses neural network effectively, i.e., it requires only few samples to train the network and then it performs for signatures newly added in the database. The algorithm used in this paper uses correlation to detect the rotation and implements simple cropping method to eliminate the effect. RBFN is used for verification purpose. Back-propagation algorithm is used for signature recognition. In this algorithm, neural network is created with R, G and B pixel values as input and gray value as output. To calculate the gradient, unsigned char to unsigned int and float to unsigned char conversions are needed to be performed. This increases computation. Also, the network is trained multiple times so that proper weight matrix is obtained. Use of RBFN reduces this need of multiple training and achieves the goal in few epochs.

For the experimental purpose, a database of 700 samples was created by collecting 10 samples each from 70 people. The network for identification was created by using RBFN fewer neurons model. All the images were used for training of the network. One of the images was processed by using rotation-translation-scaling algorithm. The signature recognition system presented in this paper was developed, trained, and tested using MATLAB™ 7. The computer was a Windows 8 machine with a 2.5 GHz Intel Core I5 processor and 4 GB of RAM. The preprocessed grayscale images of size 8 × 8 pixels are reformed in MATLAB to form a 64 × 1 array with 64 rows and 1 column for each image. This technique is performed on all test images to form the input data for testing the recognition system. Similarly, the image database for training uses 50 images and forms a matrix of 64 × 50 with 64 rows and 50 columns. The input vectors defined for the RBFN are distributed over a 2D-input space varying over [0 255], which represents intensity levels of the grayscale pixels. As many as 5 test images are used with the image database for performing the experiments. Training and testing sets were used without any overlapping. Finally the paper presents a novel method to signature recognition using RBFN. The system was evaluated in MATLAB using an image database of 700 signatures, containing 70 people and each person with 10 signatures. After training for approximately 200 samples the system achieved a recognition rate of 80%. A reduced feature space substantially reduces the computational requirements of the method as compared with standard DCT feature extraction methods. Tambade, Varma and Sonawale [7] also presented that Signatures are imperative biometric attributes of humans that have long been used for authorization purposes. Most organizations primarily focus on the visual appearance of the signature for verification purposes. Many documents, such as forms, contracts, bank cheques, and credit card transactions require the signing of a signature. Therefore, it is of upmost importance to be able to recognize signatures accurately, effortlessly, and in a timely manner. An artificial neural network based on the well-known Back-propagation algorithm is used for recognition and verification. To test the performance of the system, the False Reject Rate, the False Accept Rate, and the Equal Error Rate (EER) are calculated. They approach the problem in two steps. Initially a set of signatures are obtained from the subject and fed to the system. These signatures and pre-processed Then the pre-processed images are used to extract relevant geometric parameters that can distinguish signatures of different persons. These are used to train the system. The mean value of these features is obtained. In the next step the scanned signature image to be verified is fed to the

system. It is pre-processed to be suitable for extracting features. It is fed to the system and various features are extracted from them. These values are then compared with the mean features that were used to train the system. The Euclidian distance is calculated and a suitable threshold per user is chosen. Depending on whether the input signature satisfies the threshold condition the system either accepts or rejects the signature. In their investigation Different methods have been tried with varying results, about 99% at the best.

On the other hand in online signature verification Salinca, Rusu and Diaconescu [37] presented several signatures databases acquisitions (PHILIPS, SVC'2004 Development Set, MCYST Signature Subcorpus, BIOMET Signature Subcorpus, BioSecure Signature Subcorpus DS2 and DS3) and includes a survey on acquisition devices, procedures of acquiring genuine signatures and several types of forgeries, and the main results obtained in their evaluation and in international evaluations like SVC'2004 - First International Signature Verification Competition (Bernadette, Chollet, and Petrovska-Delacrétaz, 2009). The signatures database collected for SVC'2004 contains samples from only 60 people, and for privacy reasons, the signatures are not "real". There were two sets of signatures: one containing only coordinate information and the other containing also pen pressure and orientation. The team from Sabanci University of Turkey obtained the best result in the evaluation of a Dynamic Time Warping based system for both sets (the first set had an ERR of 2.84% and the second set had an ERR of 2.89%) (Yeung et al., 2004). A recent article presents evaluation results of online signature obtained in BioSecures Signature Evaluation Campaign (BSEC'2009), depending on the signatures quality captured on fixed (DS2) or mobile platforms (DS3) from a total of 382 people. The main task was to evaluate the algorithms' results in different acquisition conditions of signatures as well as the complexity of information contained in signatures. The results revealed that system robustness depends on the quality of signatures (on DS2 an ERR of 2.2% for skilled forgeries and an ERR of 0.51% for random forgeries, and also, on DS3 an ERR of 4.97% for skilled forgeries and an ERR of 0.55% for random forgeries).

As to the researcher knowledge there is no study conducted in signature recognition and verification system in Ethiopia". One of the primary research gaps this study attempts to investigate in signature recognition and verification is the need for more accurate and efficient algorithms. Current algorithms may struggle with variations in handwriting styles, ink thickness, and pen pressure, leading to errors in recognition. Another research gap lies in the security and privacy aspects of signature-based authentication systems. Current systems may be vulnerable to forgery, tampering, or unauthorized access. So this research focuses on developing algorithms that can better adapt to these variations and improve overall accuracy and efficiency.

CHAPTER THREE

METHODOLOGY

3.1.Overview

Signature recognition and verification is a rapidly growing field in biometrics that aims to accurately identify and authenticate individuals based on their handwritten signatures. The methodology used in this research involves a combination of various techniques, including machine learning, pattern recognition, and digital image processing. In this section, we discuss the different approaches and methodologies used in signature recognition and verification. The methodology used in this research involves data collection and preprocessing, feature extraction and selection, classification techniques, performance evaluation, and security and privacy considerations. As the technology continues to evolve, it is expected that signature recognition and verification systems will become more accurate, efficient, and secure, ultimately leading to increased adoption in various applications.

3.2.Proposed Architecture

The proposed architecture for signature verification leverages a synergistic combination of Freeman Chain Code, k-Nearest Neighbors (KNN), and Graph Neural Networks (GNN) to enhance the accuracy and robustness of signature authentication systems. Freeman Chain Code is employed to represent the spatial configuration of signature patterns, capturing the directional information of the strokes. The use of KNN introduces a powerful mechanism for comparing the input signature against a database of reference signatures, aiding in the identification of similar patterns. Furthermore, the integration of GNN allows for the modeling of complex relationships within the signature data, considering the inherent graph-like structure of signatures. The GNN is adept at learning intricate dependencies and contextual information, contributing to the improvement of overall verification performance. This proposed architecture not only exploits the strengths of individual components but also fosters a holistic approach to signature verification, addressing challenges related to variability in writing styles and ensuring a more reliable and efficient authentication process. Hereunder figure 3 presents the proposed architecture that enables to achieve the objective of this study.

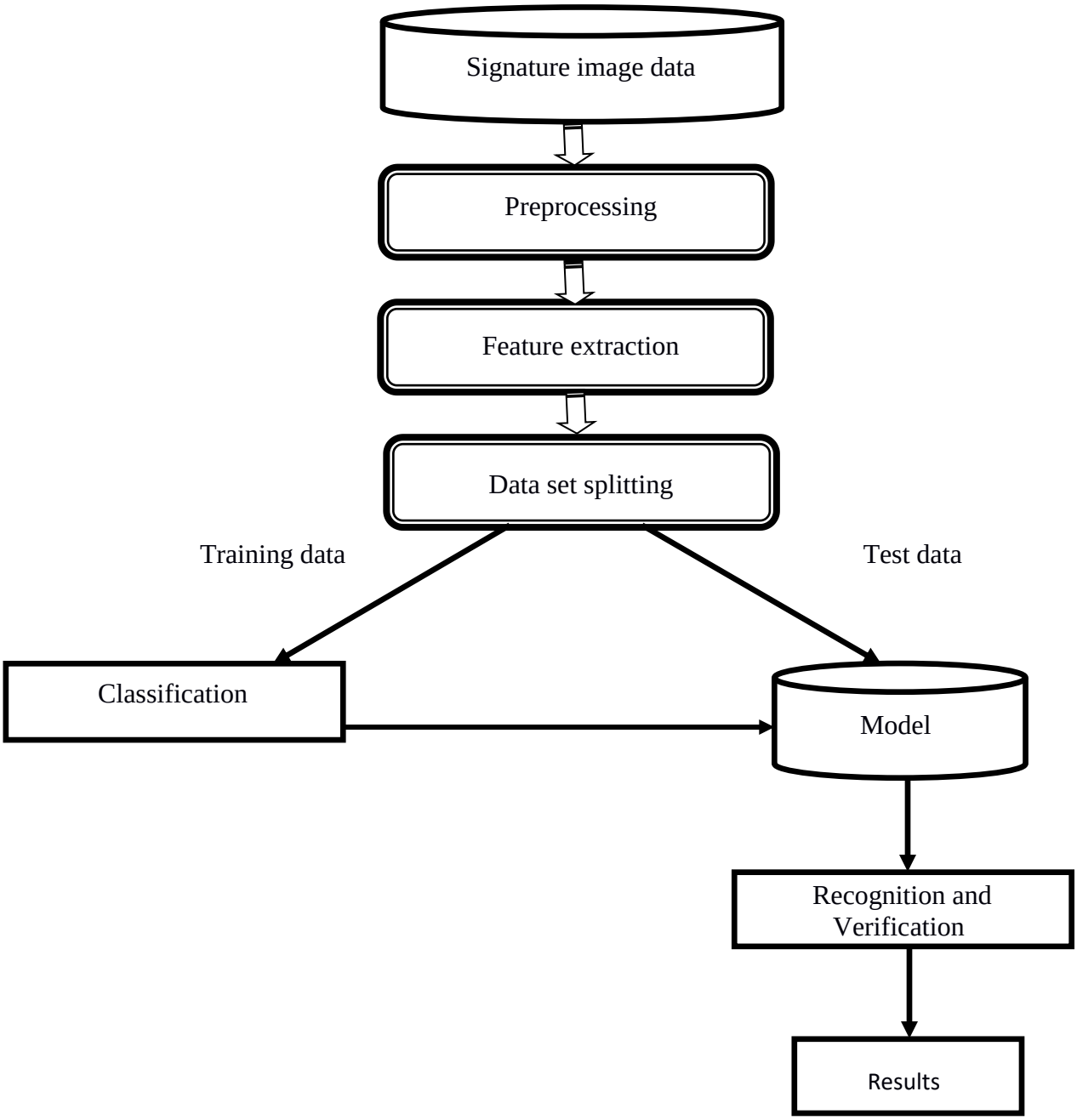

Figure 4: proposed model workflow

3.2.1. Data acquisition

In the realm of image processing, image acquisition refers to the action of retrieving an image from an external source for subsequent processing. This initial step in the workflow is crucial, as no processing can take place without the availability of an image. Data acquisition systems are utilized to measure various parameters, often within a single system. The process involves sampling signals that represent real physical conditions and converting these samples

into digital values, such as varying color, gray levels, or binary code, a procedure known as data acquisition [23].

The image acquisition phase is fundamental in the workflow sequence for image processing, providing an unprocessed baseline image generated by the employed apparatus. The objective is to maintain precise and controlled parameters, allowing the same image to be nearly replicated under similar conditions when necessary. This approach facilitates the identification and removal of anomalous variables.

Images in Python can be read as a 2-D matrix using the OpenCV library. OpenCV, standing for Open Source Computer Vision Library, is an open-source software library for computer vision and machine learning, available in languages such as C++, Python, MATLAB, and Java. It is widely used by data scientists for image processing, offering a range of algorithms and techniques. Therefore, the OpenCV library in Python and sklearn can be employed for image processing and machine learning, respectively.

3.2.2. Preprocessing

The purpose of pre-processing is to enhance the image's quality so that the feature extraction stage can use it effectively [23]. For offline signature verification systems, images may contain blurry pixels, complicated backgrounds, blurriness, and poor contrast. Consequently, a variety of pre-processing methods have been used to produce an optimized image that is suitable for other sophisticated tasks. The typical pre-processing steps involve **cropping** and **thinning**, **extracting the signature** from the entire image, **removing the background**, **aligning the signature**, **turning it into grayscale**, using specific filters to **remove noise or distortion**, **normalizing** the signature size, **binarizing**, **normalizing** the distance, **skeletonizing**, etc.

Cropping and Tinning

The word Crop can be defined as “to trim” or “cut back” [23]. The Crop tool in most image processing programs is used to trim off the outside edges of a digital image. Cropping can be used to make an image smaller (in pixels) and/or to change the aspect ratio (length to width) of the image.

Similar to erosion or opening, **thinning** is a morphological procedure used to eliminate certain foreground pixels from binary pictures. It may be applied to many other situations, but skeletonization is one where it shines. By decreasing all lines to a single pixel thickness in this mode, edge detector output is frequently cleaned up. Thinning typically only affects binary images, and the result is another binary image.

The thinning operation is related to the hit-and-miss transform and can be expressed quite simply in terms of it [23]. The calculation for the thinning procedure involves translating the origin of the structuring element to each potential pixel position in the image and comparing it with the underlying image pixels at each such position. The image pixel underneath the structuring element's origin is set to background (zero) if the foreground and background pixels in the structuring element exactly match the foreground and background pixels in the image. Otherwise it is left unchanged and the structuring element must always have a one or a blank at its origin if it is to have any effect.

Binarization

The images are not viewed by machines in the same way that people do. Images are nothing but pixels. An image is represented as a 2-dimensional matrix with the x-axis being the width and the y-axis being the height. The matrix is made up of pixel values, where each pixel's value can range from 0 to 255. In this case, a pixel value of 0 indicates a black color, whereas a value of 255 indicates a white color. Typically, the RGB representation of each matrix element is used to represent it [23].

Binarization of the image is converting a grayscale image into a black and white image (i.e. 0 and 255 pixels respectively). This can be achieved with the help of a process called thresholding. Thresholding can be defined as a process in which each pixel is converted to either 0 or 255 depending on whether its value is greater than or less than a threshold value. If the value of the pixel is greater than the threshold value, then it is converted to 255 otherwise it is converted to 1. This is how thresholding works.

Noise removal

Preprocessing of a scanned image is required to separate the signature portion and remove any unwanted noise before extracting the features. Noise is one of the challenge in Digital Image or in Image Processing. Noise happens in digital images during image acquisition, coding, transmission, and processing steps [23]. And there are different noise removal techniques in order to denoise or in reducing the noise from the image.

Image denoising is one of the fundamental challenges in image processing and computer vision. By removing noise from the image, denoising estimates the original image. Digital image processing always includes a component called noise as a built-in component. In the computer form, noise is a signal's random disturbance. The signal in this instance is an image. Image noise is a random variation in the brightness and color of an image. Image noise is an arbitrary variation in the brightness or hue of the images that are being taken. Noise is degradation in the picture signal brought on by outside factors. The brighter the area, the noisier the image is when there is multiplicative noise present [23].

As noted by R.C. Gonzalez and R.E. Woods [23], some of the sources of image noise are the following:

- While image being sent electronically from one place to another.
- **Sensor heat** while clicking an image.
- With varying **ISO Factor** which varies with the capacity of camera or digital scanner to absorb light.

While applying color conversion from RGB images in to gray scale image, there will be some noise like Impulse noise affecting the quality of images. Impulse noise is also called Salt-and-pepper noise as the name suggests salt is white and pepper black such that salt is white spots in the dark regions or pepper is black spots in the white regions. In other words, an image having salt-and-pepper noise will have a few dark pixels in bright regions and a few bright pixels in dark regions. An effective noise reduction method for this type of noise is a median filter a morphological filter and also BM3d Filtering algorithms [23].

Median filtering:

The median filter is a non-linear digital filtering technique, often used to remove noise from an image. Median filtering is very widely used in digital image processing. Median filtering is widely used as it is very effective at removing noise while preserving edges. It is particularly effective at removing 'salt and pepper' type noise. The median filter considers each pixel in the image in turn and looks at its nearby neighbors to decide whether or not it is representative of its surroundings. Instead of simply replacing the pixel value with the mean of neighboring pixel values, it replaces it with the median of those values [23].

The median filtering is calculated by first sorting all the pixel values from the window (with 3x3 or 5x5 or other window sizes) into numerical order, and then replacing the pixel being considered with the middle (median) pixel value.

Morphological filtering (Dilation and Erosion):

Morphological filter are shrink and let grow process. The word "shrink" means using median filter to round off the large structures and to remove the small structures and in grow process, remaining structures are grow back by the same amount.

Dilation is a morphological operator which works for the grow process. Whereas Erosion is a morphological operator which works for the shrink process [23].

Dilation: this is a process in which the image is expanded from its original shape. The way the image is expanded is determined by the structuring element. This structuring element is smaller in size compared to the image itself, and normally the size used for the structuring element is mostly with a size of 3 x 3. The dilation process shifted from left to right and from top to bottom, at each shift; the process will look for any overlapping similar pixels between the structuring element and that of the binary image. If there exists an overlapping then the pixels under the center position of the structuring element will be turned to 1 or black. If this is the case, the position where the structuring element is being centered on the image will be 'ON'. This process is illustrated in Fig. a. The black square represents I and the white square represents 0.

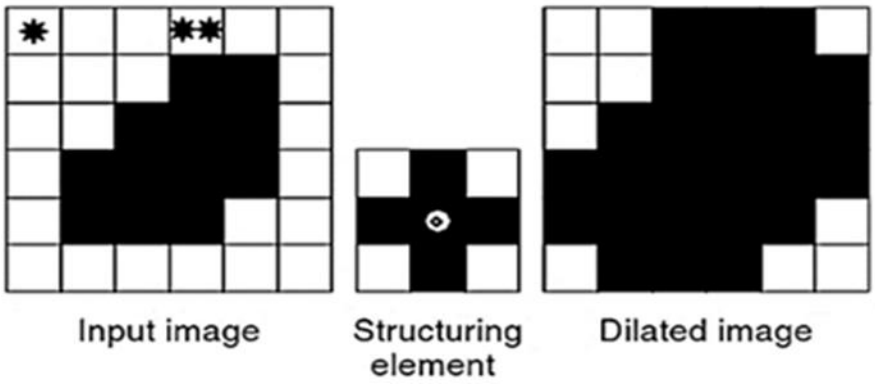

Figure 5: Dilation operation result

Erosion: this is the counter-process of dilation. If dilation enlarges an image, then erosion shrinks the image. The way the image is shrunk is determined by the structuring element. The structuring element is normally smaller than the image with a 3 x 3 size. This will ensure faster computation time when compared to larger structuring-element size. Almost similar to the dilation process, the erosion process will move the structuring element from left to right and top to bottom. At the center position, indicated by the center of the structuring element, the process will look for whether there is a complete overlap with the structuring element or not.

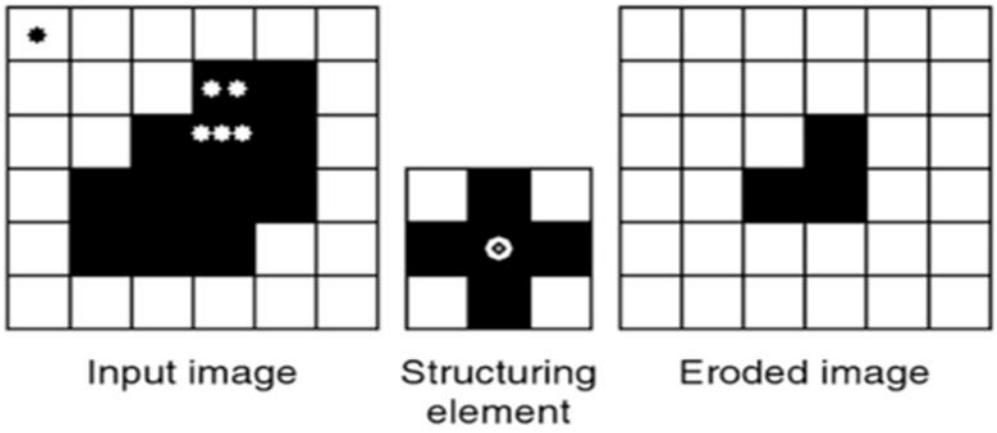

Figure 6: Erosion operation result

The result of dilation and erosion in gray-scale morphology is contributed from maximum and minimum operation.

BM3D filtering:

Block-matching and 3D filtering algorithm (BM3D) is the current state-of-the-art for image de-noising. This algorithm has a high capacity to achieve better noise removal results as compared with other existing algorithms. BM3D is a 3-D block-matching algorithm used primarily for noise reduction in images. A block matching algorithm involves dividing the current frame of a video into macro blocks and comparing each of the macro blocks with a corresponding block and its adjacent neighbors in a nearby frame of the images [23].

Lock-matching and 3D filtering makes use of this approach to solve various image restoration inverse problems such as noise reduction and de-blurring in both images and digital video.

3.2.3. Segmentation

Image segmentation is a crucial step in bank cheque signature verification. It involves dividing an image into multiple segments or regions to simplify image analysis and processing. In the context of bank cheque signature verification, image segmentation is used to isolate the signature from the rest of the cheque image, making it easier to compare with the reference signature [23].

There are several techniques for image segmentation, including thresholding, edge detection, region growing, and clustering [24]. Thresholding involves setting a threshold value and classifying pixels as foreground or background based on their intensity values. Edge detection involves identifying edges in an image using gradient-based or Laplacian-based methods. Region growing involves grouping pixels based on their similarity in color or texture. Clustering involves partitioning an image into clusters based on pixel similarity.

In bank cheque signature verification, thresholding is often used for image segmentation due to its simplicity and effectiveness. The cheque image is first converted to grayscale, and then a threshold value is selected to separate the signature from the background. The resulting binary image is then processed further to extract features that can be used for signature verification.

There are 3 different types of thresholding techniques: Simple Thresholding, Adaptive Thresholding and Otsu’s Thresholding [23].

Simple Thresholding

When using the simple thresholding technique one has to provide the function with the threshold value parameter. Thus, the threshold value is manually determined by the implementer after many hits and trials to determine whether the specific number is appropriate for the image processing. Every pixel has the same threshold value, which is translated to 0 if the value is below the threshold value and to 255 otherwise.

Adaptive Thresholding

When the image has varying lighting, adaptive thresholding method can be more helpful for binarization. As compared to simple thresholding where a single global threshold value can be applied to all pixels when performing simple thresholding. When adaptive thresholding algorithm divides the image into smaller sections and automatically chooses a threshold value for each section. Compared to simple thresholding, this method produces significantly superior results.

Otsu’s Thresholding

The method of Otsu's thresholding is comparable to simple thresholding . The only significant distinction is that, unlike simple thresholding, where the value of the threshold is explicitly required, there is no need to select the global threshold value for Otsu’s thresholding. In the case of Otsu’s thresholding, the threshold value is automatically determined by this algorithm and used to achieve the optimal outcome.

3.2.4. Feature extraction and selection

Feature extraction is a special form of dimensionality reduction. When an algorithm's input data is too large to analyze and appears to be redundant, it will be converted into a simplified representation set of feature vector by selectively omitting irrelevant data. This crucial work was carried out using Freeman chain code (FCC), which is built in a boundary-based manner [23].

Freeman Chain Code (FCC) is initially made by Herbert Freeman which is a concise way of describing an object's outlines. According to Freeman et al [29], line drawings are represented using a variety of techniques called Chain Coding, which aids in the effective transmission and

storage of such data. The 2D figure must be quantized and encoded for this reason. The line drawing is overlayed with a consistent mesh. Let us suppose that T is the distance between two adjacent mesh nodes. Curve points are used to quantify or approximate a line drawing. By identifying mesh nodes that are near the curve in terms of the various quantization schemes, offering different representations for each scheme, curve points are picked to represent points on the line drawing. These next three quantization techniques (schemes) are described [24].

Square quantization

As seen in the picture below, each mesh node is assumed to be the center of a square with side T. A mesh node is designated as a curve point when a portion of a line drawing or curve falls inside its square. This quantization approach creates the pattern with the curve points depicted in magenta in the example curve shown in the picture.

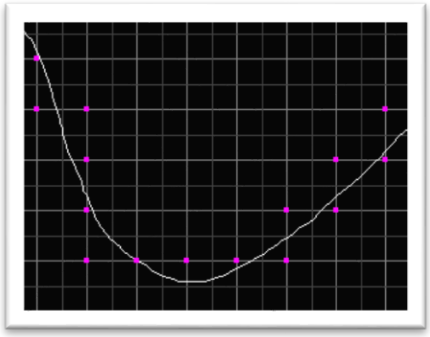

Figure 7: square quantization

Circular Quantization

This quantization system uses the same concept as square quantization, with the exception that each mesh node is regarded as the center of a circle with radius T/2. As illustrated in the image below. A mesh node is regarded as a curve point when a portion of the line drawing lies within the circle whose center is a mesh node.

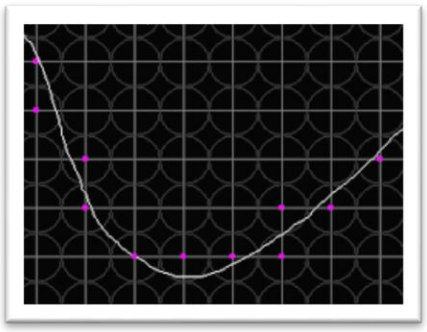

Figure 8: circular quantization

Grid-Intersect Quantization

In this technique the points on the curve that intersect the mesh lines between adjacent nodes are chosen for the curve. The mesh nodes connected to the mesh line are taken into account every time the line drawing crosses the mesh. As a curve point, the node closest to the intersection is chosen. There is only one curve point selected when there are multiple intersecting points near the mesh node.

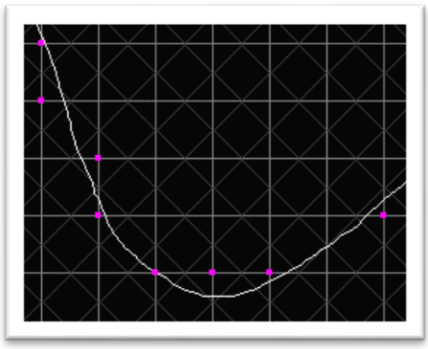

Figure 9: grid-intersect quantization

For the same line drawing, the various quantization techniques frequently produce substantially different chain codes. The quantity of diagonal line segments in the different designs is one way in which they differ from one another. It is clear that, in contrast to the circular and grid-intersect quantizations, the square quantization does not produce any diagonal segments (provided that the line drawing never crosses two mesh lines). For a decent or close approximation, diagonal elements, according to Freeman, should appear half the time for random configurations.

Freeman chain code is a technique used in digital image processing to represent the boundary of an object in an image. It is a sequence of directions that describe the path of the boundary pixels. There are two most commonly used types of Freeman chain code, which are as follows [23]:

- **4-Connected Freeman Chain Code:** In this type of chain code, there are four possible directions that can be taken by the boundary pixels. These directions are represented by the numbers 0, 1, 2, and 3. The direction 0 represents a move to the right, direction 1 represents a move to the upper right, direction 2 represents a move to the top, and direction 3 represents a move to the upper left.
- **8-Connected Freeman Chain Code:** In this type of chain code, there are eight possible directions that can be taken by the boundary pixels. These directions are represented by the numbers 0 to 7. The direction 0 represents a move to the right, direction 1 represents a move to the upper right, direction 2 represents a move to the top, direction 3 represents a move to the upper left, direction 4 represents a move to the left, direction 5 represents a move to the lower left, direction 6 represents a move to the bottom, and direction 7 represents a move to the lower right.
- **Mixed Freeman Chain Code:** This type of chain code is used when there are both straight and diagonal edges in an object's boundary. It combines both 4-connected and 8-connected chain codes.
- **R-Directed Freeman Chain Code:** In this type of chain code, each pixel is assigned an R-value based on its position relative to its neighbors. The R-value determines which direction should be taken next in order to follow the boundary.

The 8 connectivity Freeman chain code is a technique used in digital image processing to represent the boundary of an object in an image. It is a sequence of numbers that describes the direction of each pixel in the boundary relative to its previous pixel. The chain code is computed by traversing the boundary of the object in a clockwise or counterclockwise direction and assigning a number to each pixel based on its position relative to the previous pixel.

The 8 connectivity Freeman chain code uses eight directions to represent the movement from one pixel to another. These directions are represented by numbers from 0 to 7 (see figure 10), where 0 represents a movement to the right, and the other numbers represent movements in a clockwise direction. The Freeman chain code has several applications in image processing, including object

recognition, shape analysis, and pattern recognition. It is particularly useful for recognizing objects with irregular shapes or contours.

Freeman chain code is a technique used to extract features from handwritten signatures. Developed by Freeman, this method converts a signature image into a sequence of line segments that capture the essential characteristics of the signature. By analyzing these line segments, it becomes possible to classify and compare signatures for various applications, such as signature verification and forgery detection.

The Freeman chain code algorithm works as follows:

Pixel traversal: The algorithm starts at a random point on the signature image and moves in a particular direction, usually counterclockwise. As it traverses each pixel, it checks if the pixel is a corner or a ridge point.

Corner and ridge point detection: A corner point is a pixel that has an 8-neighborhood with at least three pixels of different intensity. A ridge point is a pixel that has at least two neighboring pixels of the same intensity. The algorithm keeps track of the direction and distance traveled from the starting point to record the chain code.

Line segment extraction: When the algorithm encounters a corner point, it records the current direction and distance as the endpoint of a line segment. If it encounters a ridge point, it continues moving in the current direction and updates the distance.

Feature extraction: The extracted chain code is a sequence of line segments, representing the signature's essential features. These features can be used for comparison, or other analyses [23].

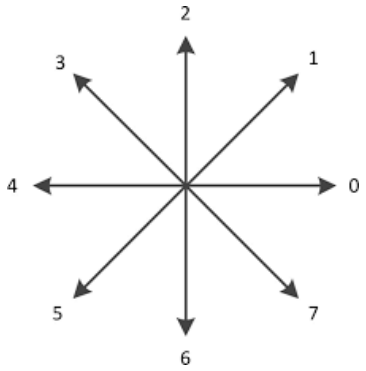

Figure 10: 8-Connectivity Freeman Chain Code

To implement the Freeman Chain Code in Python, the following steps are followed [18]:

- Import the required libraries, such as NumPy and OpenCV.
- Read the image or load the contour data.
- Apply a thresholding technique to binarize the image or extract the contour.
- Use the cv2.findContours () function to find the contours in the binary image or the contour data.
- For each contour, apply the Freeman Chain Code algorithm to obtain the sequence of values.
- Store the sequence of values for each contour.

3.2.5. Classification

Classification in signature recognition and verification refers to the process of categorizing signatures into predefined groups or classes based on their similarities and differences [31]. This is typically done by analyzing various features and characteristics of the signatures, such as pen pressure, speed, and the shape of the strokes. The primary goal of classification is to improve the performance of the recognition system by reducing the number of false positives and false negatives.

The process of classification is essential for several reasons [31]:

- Improved Accuracy: By categorizing signatures into groups based on their similarities, classification helps the recognition system to better differentiate between genuine and forged signatures. This results in a higher overall accuracy rate.
- Reduced Computational Complexity: Classification can simplify the recognition process by focusing on specific classes or groups of signatures. This reduces the computational complexity and enables faster processing times.
- Customization: Classification allows the system to be tailored to the needs of specific applications, such as financial transactions or legal documents. Different classification methods can be employed depending on the desired level of security and accuracy.

- Adaptability: As new signatures are acquired and added to the system, classification enables the system to adapt and learn from these new examples, improving its overall performance.

KNN classification

K nearest neighbor (KNN) is a popular algorithm in machine learning for classification and regression problems. It works by finding the K nearest data points to a query point in a given dataset, and using their labels or values to predict the label or value of the query point. The k-nearest neighbor (kNN) algorithm is one of the most widely used learning algorithms in machine learning research. [5] The main concept of kNN is to predict the label of a query instance based on the labels of k closest instances in the stored data, assuming that the label of an instance is similar to that of its kNN instances. kNN is simple and easy to implement, but is very effective in terms of prediction performance. kNN makes no specific assumptions about the distribution of the data. Because it is an instance-based learning algorithm that requires no training before making predictions, incremental Florin Leon learning can be easily adopted. For these reasons, kNN has been actively applied to a variety of supervised learning tasks including both classification and regression tasks. [5][8]

The KNN algorithm works by finding the K nearest neighbors to a given data point in the feature space. In the case of bank cheque signature verification, the feature space would consist of various features extracted from the signature image, such as stroke width, curvature, and direction. The distance between two data points in the feature space is typically calculated using Euclidean distance. Once the K nearest neighbors has been identified, the algorithm assigns a class label to the data point based on the majority class of its neighbors. For example, if 5 out of the 7 nearest neighbors are classified as genuine signatures, then the algorithm would classify the data point as a genuine signature. To train the KNN algorithm for bank cheque signature verification, a dataset of labelled signatures is required. This dataset would consist of genuine and forged signatures, along with their corresponding feature vectors. The algorithm would then use this dataset to identify the optimal value of K and to calculate the distances between data points in the feature space.

The main idea behind the KNN algorithm is to calculate the similarity or distance between the new data point (query point) and all the points in the training dataset. The similarity is often measured using the Euclidean distance, Manhattan distance, or Minkowski distance. Once the similarity or distance is calculated, the K-Nearest Neighbor algorithm identifies the K closest neighbors to the query point in the feature space. The prediction or classification of the query point is then based on the majority class or average value of its K nearest neighbors.

The value of K is a crucial parameter in the KNN algorithm. It can be chosen by cross-validation, grid search, or other optimization techniques. The choice of K can significantly impact the algorithm's performance, with smaller values of K leading to more localized predictions and larger values of K leading to smoother and more global predictions.

KNN has several advantages, including its simplicity, versatility, and ability to handle both categorical and numerical attributes. However, it also has some drawbacks, such as its high computational cost, sensitivity to noise, and the need for careful preprocessing of data.

In the context of signature verification, KNN involves the following steps:

Training Set: A set of genuine and forged signatures is required to train the KNN model. The training set is used to learn the similarities and differences between genuine signatures and forged ones, based on the extracted features.

K-Nearest Neighbor Classification: During the classification phase, a test signature is compared to the reference signatures in the training set. The Euclidean distance between the features of the test signature and those of the reference signatures is calculated. The K-nearest neighbors are then determined by selecting the K reference signatures with the smallest distances to the test signature.

Decision Making: Finally, the test signature is classified as genuine or forged based on the majority class of its K-nearest neighbors. If the majority of the K-nearest neighbors are genuine signatures, the test signature is considered genuine; otherwise, it is classified as forged.

In general, K-Nearest Neighbor in signature verification is a versatile and powerful technique that relies on the similarity of features between input signatures and reference signatures to

classify signatures as genuine or forged. The algorithm's effectiveness is dependent on the quality and quantity of the training set, as well as the chosen features and value of K.

Advantages of K-Nearest Neighbor

- Flexibility: KNN is a versatile algorithm that can be applied to both classification and regression problems. It can also be easily adapted to handle various types of data, including continuous, discrete, and categorical variables.
- Non-parametric: KNN is a non-parametric algorithm, which means that it does not make any assumptions about the underlying distribution of the data. This makes it suitable for a wide range of applications and data sets.
- Robust to outliers: KNN is less sensitive to outliers compared to other machine learning algorithms, as it takes into account the proximity of all data points in the neighborhood rather than relying on a single decision boundary.
- Easy to implement: KNN is a relatively simple algorithm to implement, and it does not require any complex mathematical calculations. This makes it accessible to a wide range of users, including those with limited programming skills.
- No hyperparameter tuning: Unlike some other machine learning algorithms, KNN does not require any hyperparameter tuning. This simplifies the model selection process and makes it easier to deploy in real-world applications.

Disadvantages of K-Nearest Neighbor

- Memory consumption: KNN requires storing all the training data in memory, which can be computationally expensive for large data sets.
- Computational complexity: The computational complexity of KNN is relatively high, as it requires calculating the distance between each data point and all other data points in the training set. This can be time-consuming for large data sets.
- Sensitivity to noise: Although KNN is robust to outliers, it can be sensitive to noise in the data. Noisy data points can have a significant impact on the classification or regression results.

- Curse of dimensionality: KNN is susceptible to the curse of dimensionality, which means that the performance of the algorithm degrades as the number of dimensions increases. This can be a problem for high-dimensional data sets.
- Lack of interpretability: KNN does not provide a clear interpretation of the model, making it difficult to understand how the algorithm is making its predictions.

To implement the kNN algorithm for signature verification in Python, the following steps can be applied:

Preprocess the data: Before applying the kNN algorithm, the signatures need to be preprocessed to ensure they are in a standard format. This can involve normalizing the signatures, converting them to a vector format, and removing noise.

Create a dataset: The dataset should consist of two columns: one for the reference signatures and the other for the given signatures. Each row in the dataset represents a unique signature pair.

Import the necessary libraries: To implement the kNN algorithm in Python, we will need to import the required libraries, such as Sklearn numpy, pandas, and matplotlib.

Implement the kNN model: The kNN model can be implemented using the following steps:

- Load the dataset: Read the dataset containing the reference and given signatures.
- Split the dataset: Divide the dataset into training and testing sets, ensuring that the training set contains the reference signatures, and the testing set contains the given signatures.
- Preprocess the data: Normalize the signatures and convert them to a vector format.
- Choose the value of k: Select the value of k (number of nearest neighbors) that will be used to classify the given signatures.
- Perform classification: For each given signature in the testing set, find the k-nearest neighbors in the training set and calculate the similarity between the given signature and each of its neighbors.
- Determine the class: Based on the similarity scores, determine whether the given signature is authentic or not.

To perform signature verification using k-nearest neighbor (KNN) in Python, you can use the scikit-learn library, which provides a simple and efficient tool for data mining and data analysis.

Graph neural network (GNN)

A graph neural network (GNNs) is a class of neural networks designed to work with graph data structures [28]. They have been used in various applications, including social network analysis, drug discovery, and recommendation systems.

The process of bank cheque signature verification involves comparing the signature on a cheque with the signature on file for the account holder. This is typically done using image recognition techniques to extract features from the signature and then comparing these features to those of the stored signature. However, traditional image recognition techniques often struggle with variations in handwriting and other factors that can affect the appearance of a signature.

GNNs offer a promising solution to this problem by allowing for more complex and nuanced analysis of signatures. In a GNN-based approach to bank cheque signature verification, the signature is first represented as a graph, with each stroke or line in the signature represented as a node in the graph. The edges between nodes represent the spatial relationships between strokes.

Once the signature has been represented as a graph, a GNN can be used to analyze it. The GNN takes as input the graph representation of the signature and learns to identify patterns and features that are indicative of a genuine or forged signature. This is done through a process of message passing between nodes in the graph, where each node updates its representation based on information from its neighbors.

One advantage of using GNNs for bank cheque signature verification is that they are able to capture more complex relationships between strokes than traditional image recognition techniques. For example, a GNN can learn to recognize that certain combinations of strokes are more likely to occur in genuine signatures than in forged ones.

Graph neural networks are a type of deep learning architecture that operates on graph-structured data. They have been successfully applied to various tasks, such as social network analysis, drug discovery, and natural language processing. In the context of signature verification, GNNs can

model the complex relationships between different parts of a signature, allowing them to capture the unique characteristics of an individual's handwriting.

The Role of Graph Neural Networks in Signature Verification

In signature verification, the input data is typically represented as a sequence of points or strokes. A GNN can be used to model the relationships between these points by constructing a graph, where each node represents a point, and the edges connect neighboring points. By incorporating this graph structure, GNNs can capture the spatial and temporal dependencies between points in a signature, leading to more accurate classification results.

GNN-based Signature Verification Models

There are several GNN-based models that have been proposed for signature verification. Some notable examples include Graph Convolutional Networks (GCN), which use a spectral convolution to aggregate information from neighboring nodes, and Message Passing Neural Networks (MPNN), which update node representations based on exchanged messages with neighboring nodes. These models have demonstrated improved performance over traditional methods in various benchmark datasets.

Benefits of GNNs in Signature Verification

The use of GNNs in signature verification offers several advantages over traditional methods. First, GNNs can capture the complex relationships between different parts of a signature, leading to more accurate classification results. Second, they can handle variations in pen pressure, stroke direction, and writing speed, which are common in handwritten signatures. Finally, GNNs can be easily trained on large datasets, enabling the development of more robust and generalizable models.

To implement signature verification using GNNs in Python, you can use popular deep learning libraries such as PyTorch, TensorFlow, or Keras. These libraries provide pre-trained GNN models that can be fine-tuned for signature verification.

Here is a general outline of the steps involved in implementing signature verification using GNNs in Python:

Graph Construction: Represent the preprocessed signature data as a graph, where each node represents a stroke and edges represent the connections between them.

Feature Extraction: Use GNNs to extract features from the graph representation of the signature. This can be done using pre-trained GNN models or by training a custom GNN model.

Verification: Compare the extracted features from the acquired signature with the reference signature to determine if they are from the same person.

Combining KNN AND GNN

One way to combine KNN and GNN is to use a KNN graph as input to a GNN. In this approach, each node in the graph represents a data point, and edges are added between nodes that are K-nearest neighbors of each other. The resulting graph is then fed into the GNN, which can learn representations of the nodes and edges based on their local and global topological relationships.

The GNN can then be used for various tasks on the graph, such as node classification, link prediction, or graph classification. For example, in node classification, the GNN can learn to predict the label or value of each node based on its local and global neighborhood information.

Overall, the combination of KNN and GNN can be a powerful approach for machine learning on graph-structured data, especially when the data has a natural notion of locality.

K-Nearest Neighbors (KNN) and Graph Neural Networks (GNN) are two distinct machine learning techniques that have been widely used in various applications. KNN is a non-parametric and instance-based learning algorithm that relies on the proximity of data points, while GNN is a deep learning technique that leverages the graph structure to extract information from nodes and edges. Combining the K-Nearest Neighbors (KNN) algorithm and Graph Neural Networks (GNN) involves leveraging the strengths of both methods to enhance the overall performance of a predictive model. The following is a general procedure for combining KNN and GNN:

- Data representation: The first step is to represent the data in a graph structure. This involves creating nodes for data points and edges to connect them based on their similarity or relationship.

- Feature extraction: Once the data is represented as a graph, there is a need to extract features from the graph structure. GNNs are designed to learn from the graph structure, so in this study it is used GNNs to extract features from the graph.
- KNN classification: After extracting the features, KNN can be used KNN to classify the data points. In this step, we need to determine the appropriate value of K (number of nearest neighbors) should be determined for the KNN algorithm.
- Combining predictions: The final step is to combine the predictions from the KNN model with the information from the GNN model. This can be done using different methods, such as averaging, voting, or weighting the predictions.

3.2.6. Evaluation Methods

Once the model has been created, its performance can be evaluated on the testing set. For this purpose metrics of error rates such as False Acceptance Rate (FAR), False Rejection Rate (FRR), and Average Error Rate (AER) score can be employed to evaluate the model's performance.

False Acceptance Rate (FAR)

False Acceptance Rate (FAR) is a metric used to measure the performance of a biometric system or algorithm. It represents the probability of incorrectly accepting an impostor as an authorized user [32]. In simpler terms, it is the likelihood of a false positive occurring in the system. The FAR is typically expressed as a percentage and is calculated by dividing the number of incorrect acceptances by the total number of impostor attempts. For example, if a biometric system has a FAR of 1%, it means that, on average, 1 out of every 100 impostors will be falsely accepted as a legitimate user. The FAR is an important metric to consider when evaluating the security and reliability of a biometric system, as a high FAR can lead to compromised security and increased costs associated with false acceptances.

False Rejection Rate (FRR)

False Rejection Rate (FRR) is another metric used to evaluate the performance of a biometric system or algorithm. It represents the probability of incorrectly rejecting an authorized user as an impostor [32]. The FRR is typically expressed as a percentage and is calculated by dividing the number of incorrect rejections by the total number of legitimate user attempts. For example, if a biometric system has an FRR of 5%, it means that, on average, 5 out of every 100 authorized

users will be falsely rejected. The FRR is an important metric to consider when evaluating the usability and convenience of a biometric system, as a high FRR can lead to frustrated and inconvenienced users.

Average Error Rate (AER)

Average Error Rate (AER) is a metric that combines both the False Acceptance Rate (FAR) and False Rejection Rate (FRR) to provide an overall measure of the performance of a biometric system or algorithm. It is calculated by averaging the FAR and FRR, which can be expressed as a single percentage [32]. For example, if a biometric system has an AER of 2.5%, it means that, on average, there is a 2.5% chance of a false acceptance or false rejection occurring. The AER is a useful metric for comparing different biometric systems or algorithms, as it provides a comprehensive view of their overall performance.

In signature verification, False Acceptance Rate (FAR) refers to the rate at which an unauthorized signature is accepted as genuine. FAR is computed using the following formula.

$$FAR = \frac{Number\ of\ Falsely\ Accepted\ Images}{Total\ number\ of\ person\ in\ the\ database}$$

As compared to FAR, False Rejection Rate (FRR) refers to the rate at which a genuine signature is rejected as unauthorized. FRR is computed using the following formula.

$$FRR = \frac{Number\ of\ Falsely\ Rejected\ Images}{Total\ number\ of\ person\ in\ the\ database}$$

Average Error Rate (AER) is calculated by taking the average of FAR and FRR.

$$AER = \frac{FAR + FRR}{2}$$

CHAPTER FOUR

EXPERIMENTAL RESULTS

4.1.Overview

The foundation of any signature verification system lies in the quality and diversity of the dataset. This experimental result commence with a detailed description of the dataset, encompassing the number of genuine and forged signatures, and addressing variations in writing styles, acquisition conditions, and potential sources of noise. Preprocessing steps, including resizing, Freeman Chain Code application, and other feature extraction methods, set the stage for subsequent analyses. Experimental setups are meticulously outlined, shedding light on the division of datasets into training and validation sets. Insights into the training process, including hyperparameter tuning and optimization techniques, contribute to a comprehensive understanding of the model's learning dynamics.

4.2.Data collection

The first step in any experimental design is data collection. In this study, we need to collect a dataset of bank cheque signatures. This dataset should include both genuine and forged signatures. The dataset should be large enough to ensure that the model can learn the patterns in the signatures.

Type of image data	Size	Proportion (In percent)
Genuine signatures	528	67%
forged signatures	264	33%
Total	792	100%

Table 1: Summary of the data set

4.3.Data Preprocessing

Once signature image have been collected the, they are preprocessed to make them ready for experimetntion. This involves cleaning the data, removing any noise or outliers, and normalizing the data. Finally the data is split the dataset into training and testing sets.

4.3.1. Normalizing the data

In signature verification, the normalization of data involves preprocessing steps to ensure uniformity and consistency across signature images. The normalization process in this research typically included resizing the signature images to a standardized size and applying scaling techniques to the pixel values. For resizing, a common practice is to standardize the signature images to a specific dimension; a common size for signature images are 128x128 pixels and 256x256 pixels, in order for the signature to be used for other steps without any constraint 256x256 pixels is chosen and is applied to each signature.

4.3.2. Noise removal

In this research the two most commonly used noise removal techniques are used namely median filter and morphological filter. Median filter is used for effectiveness in removing impulse noise (salt-and-pepper noise). It replaces each pixel's value with the median value of the neighborhood. Also morphological filter is used based on the mathematical morphology theory, which involves the analysis and manipulation of shapes. Sample result image of median filter and morphological filter is presented hereunder.

Figure 11: Median filter result

Figure 12: Morphological filter result

4.4.Feature Extraction

After the data is well prepared we extracted features from the signatures that are used as input to our KNN then GNN model. Some possible features include stroke width, curvature, and direction.

In this research, implementing the Freeman Chain Code involved several key steps. Initially, the signatures were preprocessed using techniques such as resizing, grayscale conversion, and normalization to ensure consistency and enhance feature extraction. Subsequently, the Freeman Chain Code algorithm was applied to obtain the directional sequence that represents the signature's contour. The implementation of the Freeman Chain Code in this research for signature recognition has proven to be a valuable approach. By leveraging its rotational invariance and compact representation, the chain code serves as a robust and efficient method for encoding the spatial features of handwritten signatures. The integration of this technique with advanced recognition algorithms enhances the accuracy and reliability of signature recognition systems, contributing to the efficiency and security of bank cheque processing in the digital age. Hereunder there is sample freeman chain code.

Freeman Chain Code: [1, 6, 2, 1, 7, 6, 7, 1, 7, 0, 1, 7, 7, 1, 7, 0, 1, 7, 7, 0, 1, 0, 7, 1, 7, 1, 7, 5, 4, 0, 1, 3, 1, 7, 6, 0, 1, 5, 4, 6, 7, 5, 1, 7, 5, 0, 5, 4, 0, 1, 7, 1, 0, 7, 0, 1, 7, 6, 7, 5, 7, 1, 7, 1, 5, 3, 1, 7, 0, 7, 0, 1, 3, 2, 5, 4, 3, 5, 3, 1, 2, 2, 1, 3, 5, 3, 2, 3, 1, 3, 5, 3, 2, 7, 3, 1, 5, 4, 2, 1, 4, 3, 1, 5, 3, 2, 7, 3, 2, 1, 2, 3, 5, 4, 0, 1, 7, 6, 3, 4, 3, 1, 0, 2, 5, 0, 7, 5, 5, 3, 6, 5, 7, 1, 7, 6, 5, 1, 7, 0, 4, 3, 1, 5, 5, 3, 5, 4, 3, 1, 3, 2, 1, 3, 1, 5, 1, 5, 3, 2, 1, 1, 2, 6, 5, 3, 5, 4, 3, 1, 5, 3, 1, 7, 1, 2, 3, 7, 0, 4, 3, 2, 1, 7, 3,

5, 4, 3, 5, 6, 5, 2, 3, 1, 3, 1, 0, 4, 4, 3, 4, 3, 7, 6, 5, 3, 4, 3, 1, 3, 3, 2, 1, 4, 3, 1, 3, 1, 2, 1, 2, 3, 5, 4, 0, 1, 7, 6, 3, 4, 3, 1, 0, 2, 5, 7, 2, 4, 3, 2, 1, 2, 1, 0, 7, 2, 6, 5, 3, 6, 5, 3, 5, 4, 7, 6, 7, 5, 4, 0, 1, 7, 6, 5, 4, 7, 7, 3, 7, 5, 1, 0, 7, 1, 0, 7, 3, 5, 3, 2, 4, 5, 3, 5, 3, 5, 3, 1, 4, 5, 6, 5, 3, 1, 5, 3, 1, 3, 4, 2, 1, 5, 3, 4, 7, 5, 6, 2, 1, 6, 5, 3, 2, 4, 3, 1, 5, 7, 5, 4, 3, 3, 1, 0, 1, 1, 2, 6, 5, 5, 4, 3, 1, 1, 5, 3, 4, 5, 1, 0, 7, 5, 5, 3, 2, 5, 0, 7, 5, 7, 3, 7, 6, 0, 7, 7, 5, 4, 0, 1, 3, 1, 7, 6, 0, 1, 7, 0, 7, 3, 0, 7, 3, 2, 7, 6, 2, 1, 7, 2, 1, 0, 7, 5, 7, 5, 3, 5, 3, 6, 5, 3, 7, 5, 7, 6, 2, 3, 7, 5, 4, 0, 7, 6, 5, 3, 5, 6, 5, 4, 7, 5, 4, 5, 4, 0, 7, 5, 3, 4, 6, 5, 3, 1, 3, 4, 0, 3, 7, 5, 1, 0, 7, 5, 1, 7, 6, 5, 3, 1, 2, 1, 5, 6, 5, 3, 7, 5, 1, 7, 5, 3, 5, 3, 7, 6]

4.5.Model Training

After feature extraction, KNN with GNN model is created on the training set. During training, we tuned the hyperparameters of the model are tuned to ensure optimal performance.

The training process for KNN is straightforward. The model essentially memorizes the training dataset, as there is no explicit training phase. When making predictions, the algorithm identifies the k-nearest neighbors of a test point and classifies or regresses based on the majority label or average value.

GNNs undergo a training phase where node and edge features are learned through a series of graph convolutions. This enables the model to understand the graph structure and make predictions based on the learned representations.

In our scenario, combining KNN and GNN algorithms lead to improved performance. For instance, incorporating KNN's instance-based learning with GNN's ability to capture graph structure.

Training Process:

In this research to train a hybrid model using both KNN and GNN (Graph Neural Network), we followed these steps:

Load Data: We loaded our data, including features and labels, from the Excel file and preprocessed the data as needed.

Train KNN Model: We trained the KNN model on the input data and use KNN to make predictions on the same dataset.

Generate GNN Input: We use the KNN predictions as additional features for each data point and create a graph structure based on the relationships between data points (e.g., using a similarity measure or predefined relationships).

Train GNN Model: We trained a GNN model using the original features and the KNN predictions as input and the GNN model take into account the graph structure.

Evaluate Hybrid Model: We evaluate the hybrid model on a separate test set and use the KNN model to make predictions on the test set. We combine KNN predictions with the original features and feed them into the GNN model and evaluate the overall performance of the hybrid model.

A hybrid model is trained by leveraging KNN for certain aspects and GNN for graph-specific features. The training process involves optimizing both sets of hyperparameters to find a synergistic combination.

Hyperparameters:

There are several hyperparameters optimized that influence the behavior and performance of our model. List of hyperparameters optimized in our hybrid model are listed hereunder:

- KNN Model Hyperparameter:

n_neighbors: Number of neighbors to consider for each data point in the KNN model. It is set to 3, meaning the model considers the labels of the three nearest neighbors.

- GNN Model Hyperparameters:

input_dim: Input dimension of the GNN model, determined by the number of features in the input data.

hidden_dim: Hidden dimension of the GNN model, specifying the number of neurons in the hidden layer.

output_dim: Output dimension of the GNN model, representing the number of classes or unique labels in the classification task.

learning_rate: The step size used by the optimizer during the training process. It controls the size of the steps taken to update the model parameters.

num_epochs: The number of training epochs, indicating how many times the entire training dataset is processed by the GNN model during training.

- Optimizer Hyperparameter:

Adam optimizer: The optimizer used for updating the parameters of the GNN model. It has its own hyperparameters like the learning rate, but these are set by the learning_rate parameter.

- Training Data Split Hyperparameter:

test_size: The proportion of the dataset to include in the test split when splitting the data into training and testing sets using train_test_split.

- Graph Construction Hyperparameter:

The graph structure used in the GNN is constructed using all possible combinations of nodes (excluding self-loops) as edges. This is done using torch.combinations(torch.arange(num_nodes), r=2).t().contiguous().

4.6.Experimental setup

To implement signature verification using KNN with GNN, we following steps are applied. First, we collected a dataset of signatures that includes both genuine and forged signatures. The dataset was diverse and representative of different writing styles and variations in signatures. Secondly, we preprocessed the dataset by extracting relevant features from each signature, such as stroke width, curvature, and direction. These features were normalized to ensure that they have similar scales and ranges.

Next, we splited the dataset into training and testing sets. The training set is used to create the KNN model, while the testing set is used to evaluate its performance. In this research 80% of the data are being used for training purposes, while the remaining are being used for testing.

After splitting the dataset, we applied KNN algorithm to train our model. In this step, we choose an appropriate value for K as 3, which determine the number of nearest neighbors to consider.

Once our model is trained, we used it to predict the class label of new signatures. To improve the performance of our model further, we also used Graph Neural Networks (GNNs). GNNs are a type of neural network that can operate on graph-structured data, such as signatures.

We also calculated various performance metrics such as FAR, FRR, and AER to evaluate our model's performance.

4.7.Experimental Result

In signature verification, precision and recall are two commonly used performance metrics. Precision measures the accuracy of the positive predictions made by the model, while recall measures the ability of the model to capture all the relevant positive instances. The formulas for precision and recall are as follows:

Precision:

Precision is calculated as the ratio of true positive predictions to the sum of true positive and false positive predictions. It is a measure of the accuracy of positive predictions made by the model.

$$\textbf{precision} = \frac{\textbf{true positives}}{\textbf{true positives} + \textbf{false positives}}$$

Recall:

Recall, also known as sensitivity or true positive rate, is calculated as the ratio of true positive predictions to the sum of true positive and false negative predictions. It measures the ability of the model to capture all relevant positive instances.

$$\textbf{Recall} = \frac{\textbf{true positives}}{\textbf{true positives} + \textbf{false negatives}}$$

Where:

True Positives (TP): Instances where the model correctly predicts positive cases.

False Positives (FP): Instances where the model incorrectly predicts positive cases (actual negative cases misclassified as positive).

False Negatives (FN): Instances where the model incorrectly predicts negative cases (actual positive cases misclassified as negative).

To evaluate the performance of KNN with GNN sample for signature verification, experimental results were conducted on a dataset of 792 signatures. The dataset was divided into two parts: 650 (82%) for training and 142 (18%) for testing. The KNN algorithm was implemented with a GNN sample to extract features from the signature images. The GNN sample was trained on the training set to learn the underlying patterns in the data. The KNN algorithm was then used to classify the test set based on the learned patterns. Summary of experimental result is presented in table 2 below.

Algorithms	Accuracy	Recall	precision	FAR (False Acceptance Rate)	FRR (False Rejection Rate)	AER (Average Error Rate)
KNN	60%	67%	71%	17%	21%	19%
KNN with GNN	91%	94%	92%	4.9%	3.5%	4.2.%

Table 2: Experimental result

The experimental results showed that KNN with GNN sample achieved an accuracy of 91% in classifying genuine signatures and detecting forged signatures. The precision and recall values were also 92% and 94% respectively; indicating that the algorithm has a low false positive and false negative rate. In general, KNN with GNN sample is an effective algorithm for signature verification, achieving high accuracy and precision values. This algorithm can be used by banks to prevent fraud and ensure secure transactions.

4.8.Discussion of the result

Several studies have reported promising results using KNN or GNN in separation for bank cheque signature verification. In a study by Prakash and Guru [21], the authors achieved an FAR of 0.5%, an FRR of 1.2%, and an AER of 17.36 % on a dataset of 1,000 genuine signatures and

1,000 forged signatures using Distance and orientation feature for feature extraction and k-Nearest Neighbor for classification.

Another study by Sabourin and Plamondon (1990) [33] used KNN on a dataset of 500 genuine signatures and 500 forged signatures and achieved an FAR of 14.66 %, an FRR of 25.11 %, and an AER of 19.89 %.

Finally, in a study by Wang et al. (2021) [34], the authors proposed an improved SVM (support vector machine) with Euclidean distance approach that achieved an FAR of 12.44 %, an FRR of 6.67 %, and an AER of 9.56 % on a dataset of 1,125 genuine signatures and 1,125 forged signatures.

KNN with GNN is a promising approach for bank cheque signature verification, with several studies reporting low FAR, FRR, and AER results.

In the realm of signature recognition and verification, the success of a study is often contingent on the clarity and effectiveness with which research questions are addressed. This study critically examines the extent to which the following research questions have been answered:

Suitable Image Processing Tasks and Methods:

The research begins by identifying suitable image processing tasks and methods for preparing the dataset. The clarity and precision with which this question is addressed dictate the robustness of the dataset for subsequent analysis. Image resizing, grayscale conversion, normalization, and preprocessing techniques are integral components, ensuring that the dataset is meticulously prepared for the signature recognition process.

Combination of K-Nearest Neighbor and Graph Neural Network:

Addressing the amalgamation of K-Nearest Neighbor (KNN) and Graph Neural Network (GNN) involves a methodological exploration. Here we evaluated the effectiveness of combining KNN and GNN in the context of signature recognition and verification. The integration of these two distinct methodologies signifies an attempt to leverage the simplicity of KNN alongside the relational insights offered by GNN. The extent to which this fusion is implemented and its impact on the overall recognition process is assessed.

Performance of the Proposed Model:

The essence of the evaluation lies in determining the extent to which the proposed model performs in signature recognition and verification. A comprehensive analysis of the model's performance metrics, including accuracy, precision and recall, provides a nuanced understanding of its efficacy. The comparison against traditional KNN methods serves as a benchmark for gauging the advancements achieved through the incorporation of GNN. The research scrutinizes the proposed model's strengths, limitations, and its overall contribution to the field of signature recognition.

Methodological Rigor and Alignment:

The success in addressing these research questions hinges on the methodological rigor employed throughout the study. The alignment of the chosen methods with the research questions ensures that each facet of the inquiry is systematically investigated. The seamless integration of image processing tasks, the strategic combination of KNN and GNN, and the meticulous evaluation of the proposed model collectively contribute to the overall methodological robustness.

CHAPTER FIVE

CONCLUSIONS AND FUTURE WORKS

5.1.Conclusion

Bank cheque signature recognition and verification is a crucial task in the banking industry to prevent fraud and ensure the authenticity of cheques. In recent years, there has been a significant increase in the use of automated systems for cheque signature verification due to their accuracy and efficiency. In this research we have used the combination of KNN with GNN that improved the accuracy of bank cheque signature verification by leveraging the strengths of both algorithms. KNN can provide baseline classification accuracy, while GNN can capture more complex relationships between different parts of a signature and improve the overall accuracy. Despite the progress made in automated bank cheque signature verification, there are still several challenges that need to be addressed. One of the main challenges is the large variability in signatures due to different writing styles, ages, and health conditions of signers. Another challenge is the presence of noise and distortions in signature images due to scanning or printing errors. Moreover, attackers can use sophisticated techniques such as tracing or copying to create convincing forgeries that can fool automated systems.

5.2.Recommendation

To overcome the challenges, observed in this study, we recommend the following as a way forward:

- Researchers need to explore new approaches that combine multiple machine learning algorithms for better performance or use advanced features such as dynamic time warping (DTW) or wavelet transforms.
- There is need to investigate new sources of information such as pressure sensors or accelerometers that can capture additional data about the signing process and improve the accuracy of signature verification.
- We recommend designing a system that can simplify users' and banks interaction with the proposed prototype for signature verification. There is a need to extend the research for the recognition and verification of online signature

References

1. Zaidi, S. F. A., & Mohammed, S. (2007). Biometric Handwritten Signature Recognition. *TDDD17: Information Security CourseLinköpings universitet*.
2. Chadha, A., Satam, N., & Wali, V. (2013). Biometric signature processing & recognition using radial basis function network. *arXiv preprint arXiv:1311.1694*.
3. Azmi, A. N., Nasien, D., & Omar, F. S. (2017). Biometric signature verification system based on freeman chain code and k-nearest neighbor. *Multimedia Tools and Applications*, *76*, 15341-15355.
4. Dongare, A. A., & Ghongade, R. D. (2016). Artificial intelligence based bank cheque signature verification system. *International Research Journal of Engineering and Technology*, *3*(01).
5. Kang, S. (2021). K-nearest neighbor learning with graph neural networks. *Mathematics*, *9*(8), 830.
6. Myint, S. M., Myint, M. M., & Cho, A. A. (2019). Handwritten Signature Verification System using Sobel Operator and KNN Classifier. In *Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN:2456–6470* (Vol. 3,No.5).
7. Tambade, V., Varma, P., & Sonawale, A. (2018). Bank Cheque Signature Verification System. *International Journal of Research in Engineering, Science and Management*, *1*(9), 265-267.
8. Azmi, A. N., Nasien, D., & Omar, F. S. (2017). Biometric signature verification system based on freeman chain code and k-nearest neighbor. *Multimedia Tools and Applications*, *76*, 15341-15355.
9. Morocho, D., Morales, A., Fierrez, J., & Vera-Rodriguez, R. (2016, February). Towards human-assisted signature recognition: improving biometric systems through attribute-based recognition. In *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)* (pp. 1-6). IEEE.
10. Doroz, R., & Wrobel, K. (2009, June). Method of signature recognition with the use of the mean differences. In *Proceedings of the ITI 2009 31st International Conference on Information Technology Interfaces* (pp. 231-236). IEEE.
11. Hussein, K. N. (2014). *Offline Signature Verification Based on Improved Extracted Features Using Neural Network* (Doctoral dissertation, Universiti Teknologi Malaysia).

12. Sharma, N., Gupta, S., & Mehta, P. (2021, July). A comprehensive study on offline signature verification. In *Journal of Physics: Conference Series* (Vol. 1969, No. 1, p. 012044). IOP Publishing.
13. Fadnavis, S. (2014). Image interpolation techniques in digital image processing: an overview. *International Journal of Engineering Research and Applications*, *4*(10), 70-73.
14. Varshney, P. K., Arora, M. K., Rao, R. M., & Arora, M. K. (2004). Overview of image processing. *Advanced image processing techniques for remotely sensed hyperspectral data*, 51-85.
15. Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255-260.
16. Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, *9*, 381-386.
17. Pratt, W. K. (2007). *Digital image processing: PIKS Scientific inside* (Vol. 4). Hoboken, New Jersey: Wiley-interscience.
18. Qidwai, U., & Chen, C. H. (2009). *Digital image processing: an algorithmic approach with MATLAB*. CRC press.
19. Takeda, H., Farsiu, S., & Milanfar, P. (2007). Kernel regression for image processing and reconstruction. *IEEE Transactions on image processing*, *16*(2), 349-366.
20. Zheng, N., Loizou, G., Jiang, X., Lan, X., & Li, X. (2007). Computer vision and pattern recognition.
21. Prakash, H. N., & Guru, D. S. (2010). Offline signature verification: An approach based on score level fusion. *International journal of computer applications*, *1*(18), 52-58.
22. Kaur, J., Singh, W. Tools, techniques, datasets and application areas for object detection in an image: a review. *Multimed Tools Appl* **81**, 38297–38351 (2022).
23. Gonzalez, R. C. (2009). Digital image processing. Pearson education india.
24. Szeliski, R. (2022). Computer vision: algorithms and applications. Springer Nature.
25. Zhao, G., Liu, J., Jiang, J., & Wang, W. (2018). A deep cascade of neural networks for image inpainting, deblurring and denoising. Multimedia Tools and Applications, 77, 29589-29604.
26. Kaur, D., & Kaur, Y. (2014). Various image segmentation techniques: a review. International Journal of Computer Science and Mobile Computing, 3(5), 809-814.

27. Khalid, S., Khalil, T., & Nasreen, S. (2014, August). A survey of feature selection and feature extraction techniques in machine learning. In 2014 science and information conference (pp. 372-378). IEEE.
28. Das, K., & Behera, R. N. (2017). A survey on machine learning: concept, algorithms and applications. International Journal of Innovative Research in Computer and Communication Engineering, 5(2), 1301-1309.
29. Bishop, C. (2006). Pattern recognition and machine learning. Springer google schola, 2, 35-42.
30. Freeman, H., & Garder, L. (1964). Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. IEEE Transactions on Electronic Computers, (2), 118-127.
31. Rosso, O. A., Ospina, R., & Frery, A. C. (2016). Classification and verification of handwritten signatures with time causal information theory quantifiers. PloS one, 11(12), e0166868.
32. Nguyen, V., Blumenstein, M., & Leedham, G. (2009, July). Global features for the off-line signature verification problem. In 2009 10th International Conference on Document Analysis and Recognition (pp. 1300-1304). IEEE.
33. Plamondon, R., Lorette, G., & Sabourin, R. (1990). Automatic processing of signature images: Static techniques and methods. Computer Processing of Handwriting, 49-63.
34. Zhou, Y., Zheng, J., Hu, H., & Wang, Y. (2021). Handwritten signature verification method based on improved combined features. Applied Sciences, 11(13), 5867.
35. Wu, X., & Bhanu, B. (1997). Gabor wavelet representation for 3-D object recognition. IEEE transactions on image processing, 6(1), 47-64.
36. Malik, M. I., Liwicki, M., Alewijnse, L., Ohyama, W., Blumenstein, M., & Found, B. (2013, August). ICDAR 2013 competitions on signature verification and writer identification for on- and offline skilled forgeries (SigWiComp 2013). In 2013 12th international conference on document analysis and recognition (pp. 1477-1483). IEEE.
37. Salinca, A., Rusu, S. M., & Diaconescu, Ş. (2012, April). An Approach to data collection in an online signature verification system. In International Conference on Web Information Systems and Technologies (Vol. 2, pp. 251-254). SCITEPRESS.