الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية الرياضيات و الإعلام الآلي و الاتصالات السلكية و اللاسلكية

قسم: الإعلام الآلي

Mémoire de Master en informatique

Spécialité : Intelligence artificielle principes et applications

T h è m e

Text independent writer verification using offline handwriting images

- **Présenté par :**

  -Mostepha SELLAM

- **Dirigé par :**

  -Dr.Tayeb BAHRAM

Année universitaire 2024-2025

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Preface

This thesis marks the culmination of a challenging yet rewarding academic journey. It reflects not only months of research, experimentation, and writing, but also years of learning, curiosity, and personal growth. This project would not have been possible without the guidance, support, and encouragement of many individuals. I would first like to express my sincere gratitude to my supervisor, **Dr.Tayeb Bahram**, for his continuous support, valuable insights, and constructive feedback throughout this research. His expertise and mentorship played a vital role in shaping the direction of this work. I also extend my thanks to the faculty and staff of the **Computer Science Department**, whose instruction and support provided me with the foundation necessary to undertake this research. I am grateful to the creators of the IAM and KHATT datasets, whose publicly available resources were essential for evaluating the proposed system. A special word of thanks goes to my family and friends. Their patience, understanding, and moral support were a constant source of strength. Their belief in me inspired me to persevere even during the most difficult stages of this work. Finally, I dedicate this thesis to all aspiring researchers who, like me, are passionate about contributing to the advancement of knowledge and technology through persistence, curiosity, and critical thinking.

Saïda, June 2025
Mostepha Sellam

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Truth is what stands the test of experience.

Albert Einstein

1.1 Introduction

This thesis addresses the problem of automatic person verification using scanned images of handwriting. verify the writer of a questioned document using automatic image-based methods is an interesting pattern recognition problem with direct applicability in the forensic and historic document analysis fields. Approaching this challenging problem raises a number of important research themes in computer vision:

- How can individual handwriting style be characterized using computer algorithms?
- What representations or features are most appropriate and how can they be combined?
- What performance can be achieved using automatic methods?

The current study describes a new and very effective techniques that we have developed for automatic writer verification. The goal of our research was to design state-of-the-art automatic methods involving only a reduced number of adjustable parameters and to create a robust writer verification system capable of managing hundreds to thousands of writers (named *USAWRS*, University of Saïda Automatic Writer Recognition System).

There are two distinguishing characteristics of our approach: human intervention is minimized in the writer recognition process and we encode individual handwriting style using features designed to be independent of the textual content of the handwritten sample.

Writer individuality is encoded using probability distribution functions extracted from handwritten text blocks and, in our methods, the computer is completely unaware of what has been written in the samples. The development of our writer verification

techniques takes place at a time when many biometric modalities undergo a transition from research to real full-scale deployment.

Our methods also have practical feasibility and hold the promise of concrete applicability. The writer verification techniques proposed in this thesis have possible impact in forensic science. Our methods are statistically evaluated using large datasets with handwriting samples collected from up to $1,000$ subjects.

1.1.1 Motivation and Context

In our digital world, which is characterized by rapid technological evolution, the widespread of connected systems, and the continuous need for security and identity protection, the demand for reliable, robust, and automatic identity verification systems has become not only critical but indispensable. With the rise of online services, e-governance, digital banking, and remote access to confidential information, traditional methods of verification are proving increasingly inadequate and vulnerable.

Historically, identity verification has relied on three main strategies:

1. *knowledge-based methods* (e.g., passwords, secret questions);
2. *possession-based methods* (e.g., physical tokens, smart cards, badges);
3. *Inherence-based methods*, more commonly known as biometrics.

The first two approaches, while widespread, suffer from severe limitations. Passwords can be forgotten, guessed, shared, or stolen; cards and tokens can be lost, duplicated, or counterfeited.Such Defects expose systems to risks including security breaches, identity theft, unauthorized access, and reduced operational efficiency.

Biometric technologies have completely changed the way we manage and verify identity. Biometric systems authenticate or identify individuals based on distinctive, permanent, and measurable biological or behavioral traits. These include fingerprints, iris patterns, facial features, palm geometry, voice, gait, keystroke dynamics, and handwriting, among others. Because such characteristics are unique to each individual, they provide a far higher level of security, resistance to forgery, and user convenience than traditional methods.

Furthermore, biometric recognition systems offer several key advantages: they eliminate the need to remember passwords or carry physical tokens; they reduce administrative overhead in managing access credentials; and they offer scalable solutions that can operate across diverse domains, from border control and national ID programs to smartphone authentication and forensic analysis. As global security challenges continue to evolve driven by cybercrime, terrorism, and the expansion of digital ecosystems the role of biometrics in reinforcing trust, privacy, and authentication integrity becomes increasingly prominent.

In the face of these challenges, the exploration of new biometric modalities and the enhancement of existing ones represent a frontier of scientific inquiry and technological innovation. In particular, behavioral biometrics such as handwriting offer promising avenues due to their non-intrusive nature, cultural acceptance, and applicability in both online and offline contexts. This research contributes to this evolving field by focusing on the development of systems for writer verification based on offline handwriting samples—an approach that seeks to recognize individuals based on the unique characteristics of their handwritten script.

1.2 Overview of Biometrics

Biometrics refers to the automated recognition of individuals based on their physiological and/or behavioral traits. It is grounded in the principle that each person possesses unique and measurable characteristics that can be used to establish verify with a high degree of confidence. These biometric traits are generally divided into two categories:

1. **Physiological Biometrics**: These are based on physical characteristics that remain relatively stable over time. (e.g. fingerprints, iris patterns, facial structure, palm geometry, and DNA).

2. **Behavioral Biometrics**: These are based on patterns in human activity and behavior, which can vary slightly over time but are still unique to individuals (e.g. voice recognition, gait analysis, typing dynamics, and handwriting).

Biometric systems operate in two main modes: *verification* (1:1 comparison) and *identification* (1:N comparison). In the verification mode, the system confirms a claimed identity by comparing the input biometric data with a stored template. In identification mode, the system determines an individual's identity by comparing the input data with multiple stored templates, aiming to find the best match [13].

1.2.1 Handwriting as a Behavioral Biometric

Among various behavioral biometric modalities, handwriting particularly *offline handwriting* has attracted significant interest due to its natural use in many real world scenarios, such as legal documents, forms, and historical archives. Handwriting contains a wealth of personalized features that can reflect an individual's motor habits, cognitive style, and neuromuscular characteristics.

Writer verification through handwriting analysis involves confirming whether a given handwritten sample was written by a specific claimed individual, based on comparison with that individual's known handwriting samples.This process is challenging due to the intra-writer variability (variations in an individual's writing over time or under different conditions) and the inter-writer similarity (similarities between different individuals' writing styles). Despite these challenges, handwriting offers an important advantage: it can be acquired non-intrusively using simple devices like scanners or cameras, making it ideal for both forensic and commercial applications.

1.2.2 Factors causing variability in handwriting

Figure 1.1 shows four factors causing variability in handwriting [20]:

1. *Affine transforms*: Affine transforms are under voluntary control. However, writing slant constitutes a habitual parameter which may be exploited in writer recognition;

2. *Neuro-biomechanical variability*: Neuro-biomechanical variability refers to the amount of effort which is spent on overcoming the low-pass characteristics of the biomechanical limb by conscious cognitive motor control;

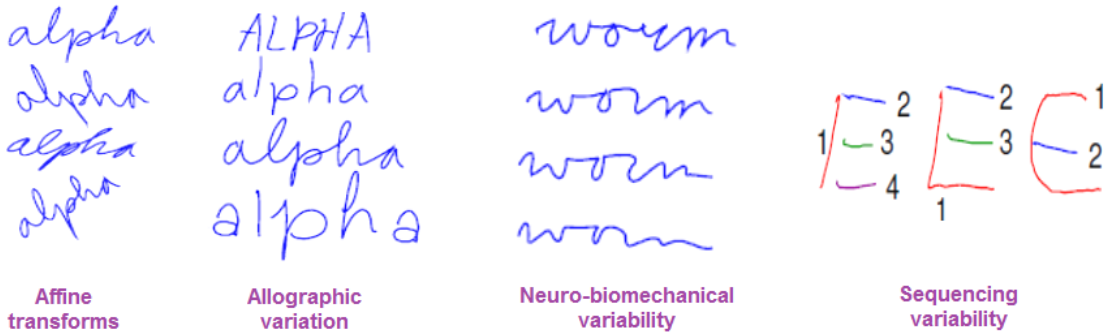

Figure 1.1: Factors causing handwriting variability.

3. *Sequencing variability*: Sequencing variability becomes evident from stochastic variations in the production of the strokes in a capital E or of strokes in Chinese characters, as well as stroke variations due to slips of the pen;

4. *Allographic variation*: Allographic variation refers to individual use of character shapes.

1.3 Writer identification and Verification in Handwriting Biometrics

Handwriting, as a form of behavioral biometric trait, encapsulates the neuromuscular patterns and cognitive processes unique to each individual. The act of writing is influenced by a combination of motor coordination, learned habits, psychological state, and physiological conditions, making it a rich source of biometric data. These characteristics are particularly useful for tasks such as *writer identification* and *writer verification*, which are two core problems in the domain of handwriting biometrics.

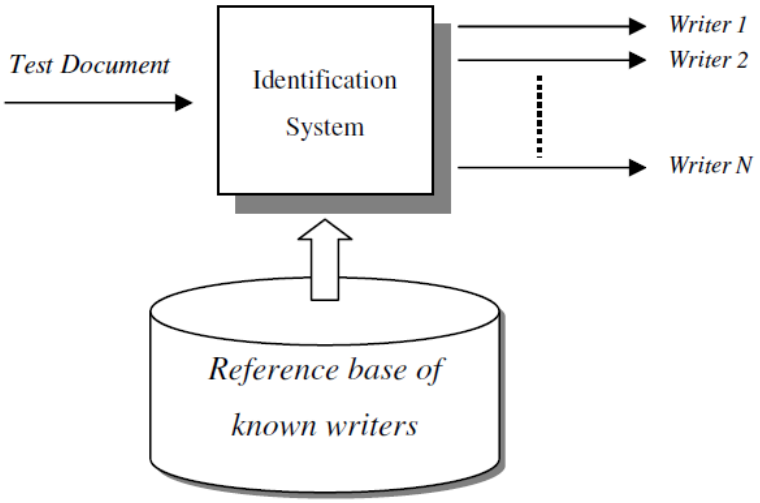

Figure 1.2: Writer identification model.

1.3.1 Writer Identification

Writer identification refers to the process of determining the identity of an individual solely based on the analysis of a handwriting sample, without any claim of identity from the user. It is a **1: N comparison** task, where the system attempts to match the sample against a database of known handwriting profiles (also called templates or models). If a match is found with sufficient similarity, the system outputs the corresponding identity (see Figure 1.2).

Writer identification can be further categorized as:

- **Closed-set identification:** The writer of the sample is assumed to be among the enrolled users. The goal is to rank all templates and select the top candidate with the highest similarity score.
- **Open-set identification:** The writer may or may not be enrolled in the system. The system must not only identify the most similar template but also determine whether the sample belongs to an enrolled writer or should be rejected as unknown.

This mode is highly relevant in forensic and law enforcement scenarios where, for example, a handwritten note must be attributed to one of several suspects.

1.3.2 Writer Verification

In contrast, writer verification involves confirming a claimed identity based on handwriting. The user presents a handwriting sample along with an identity claim (e.g., user ID), and the system verifies whether the handwriting matches the stored template for that specific individual. This is a **1 : 1 comparison** task, and the outcome is a binary decision: accept or reject the claim (see Figure 1.3).

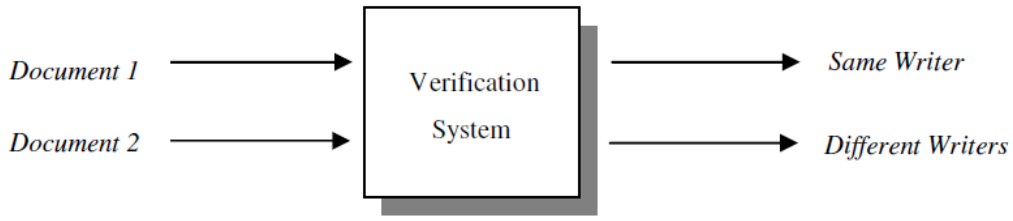

Figure 1.3: Writer verification model.

Writer verification may be performed in two ways:

- **Static (Offline) Verification:** The handwriting sample is acquired after writing, typically via scanning. Features are extracted from the image of the writing, such as stroke width, slant, curvature, baseline deviation, and spacing between characters and words.
- **Dynamic (Online) Verification:** The writing is captured in real-time using digital devices like tablets or styluses, recording temporal data such as pen pressure, velocity, acceleration, and stroke order. This method is generally more accurate but requires specialized acquisition devices.

1.4 Relevance of this Research

The objective of this work is to investigate and develop techniques for **writer verification based on offline handwriting**. This biometric approach combines image processing, feature extraction and pattern recognition to analyze handwriting samples and attribute them to the correct writer. Such systems have wide-ranging applications including forensic analysis, access control, fraud detection, and archival document indexing.

The present work has a particular focus on *text-independent recognition*. In the broader context of biometric authentication, handwriting is classified as a behavioral biometric trait, as it reflects individual neuromotor and cognitive processes. Unlike physiological traits (e.g., fingerprints or iris patterns), behavioral biometrics such as handwriting exhibit greater intra-personal variability, requiring sophisticated modeling to achieve accurate recognition.

In writer verification, in this context, aims to determine whether two handwriting samples originate from the same author. This process involves comparing the features of the handwriting samples that are invariant to the content written, the language or script used (Arabic and English).

1.4.1 Motivation and Challenges

While writer identification in monolingual contexts has matured significantly, with numerous contributions addressing major world scripts, the domain of **writer verification in mono-script settings** remains relatively under-explored. Few existing studies address how features behave in this field.

This research confronts these challenges by focusing on the development of a *robust and scalable writer verification system* capable of handling:

- Handwriting samples written in a single language and script.
- Text-independent verification, where the actual content of the writing is unknown or irrelevant.
- High similarity between different writers' handwriting and significant variability within the same writer's samples.

1.4.2 Objectives of the Research

The specific objectives of this project are as follows:

1. **Competitive Performance Metrics:** The system aims to achieve low FAR, FRR, and EER values, demonstrating high accuracy and reliability in writer verification across both Arabic and English scripts.
2. **Multilingual Robustness:** By leveraging the IAM and KHATT datasets, the system will validate its ability to handle diverse writing systems, addressing the challenges of multilingual handwriting analysis.
3. **Scientific Contribution:** Results will be documented in a potential publication submitted to ICDAR or ICFHR, contributing to the field of biometric biometrics and document forensics.

4. **Future Research Opportunities:** The evaluation will identify areas for improvement, paving the way for further research, potentially leading to a doctoral thesis focused on advanced feature extraction and verification techniques.

1.5 Overview of the thesis

The methods for writer recognition (verification) that are presented in this thesis operate at two levels of analysis: the image texture and its contours. The body of the thesis is therefore divided into two main parts treating these two important aspects. The rest of the thesis is arranged as follows. In Section 2, we summarize related work and feature extraction techniques from handwriting images. The proposed method for offline writer verification is presented in detail in Section 3. Next, Section 4 introduces eight well-known handwritten databases considered in this study with their details of experimental results. Finally, Section 5 provides a summary and future works.

Chapter 2

Writer verification: SOTA

> Science is what we understand well enough to explain to a computer. Art is everything else we do.
>
> Donald Knuth

2.1 Introduction

Despite the development of electronic documents and predictions of a paperless world, the importance of handwritten documents has retained its place and the problems of identification and authentication of writers have been an active area of research over the past few years. A wide variety of systems that are based on the use of computer image processing and pattern recognition techniques have been proposed to solve the problems encountered in automatic analysis of handwriting and recognition of the writer of a questioned document. A comprehensive survey of the work in writer recognition until 2022 has been presented in [9]. Here, we will be more interested in surveying the approaches developed in the last several years, thanks to the renewed interest of the document analysis community for this domain. The techniques for writer verification are traditionally categorized into two broad classes: *text-dependent* and *text-independent* methods.

In text dependent methods the writing samples to be compared require to contain the same fixed text. Signature verification, for example, can be considered in this category. These methods normally use the comparison between individual characters or words of known transcription and thus require the text to be recognized or segmented (manually or automatically) into characters or words prior to writer recognition [33]. The text independent methods on the other hand verify the writer of a document independent of its semantic content. These methods use features extracted from the entire image of a text or from a region of interest. A minimal amount of handwriting is necessary in order to derive stable features insensitive to the text content of the samples [11].

We will present in this chapter, an overview of significant contributions to the field of text-independent writer verification.

2.2 Text-independent writer verification

In the last twenty years, a variety of different features (attributes or descriptors) could be found in the literature of writer verification based on handwritten documents. In

general, the presented offline writer verification work has been based on three main kinds of features: **(1)** codebook-based, **(2)** texture-based, and **(3)** contour-based.

2.2.1 Codebook-based approaches

The codebook or bag-of-shapes can be used as a distinctive measure (i.e., feature vector) to characterize the writing style of each writer [11, 20]. The shape occurrence probability is a characteristic for a given writer and may be employed to distinguish between intra-cluster and inter-cluster similarity [3]. Several methods are proposed in the literature to create the codebook, such as the Graphemes (writing fragments or small patterns) [20, 51] and Implicit Shape (Patches around the key points) [15]. Regarding the allographic features, the writer is considered as a stochastic generator of graphemes and a set of emission shape probabilities is computed by building a normalized histogram.

In 2007, Bulacu et al. (2007) proposed the use of allographic features to characterize writer individuality thus making the approach quite similar to Bensefia et al. [17]. The probability distribution of each writer is computed using a common codebook of 20×20 graphemes generated with the ksom2D clustering algorithm for each document individually. Figure 2.1 shows examples of codebooks that have been obtained using the three clustering methods.

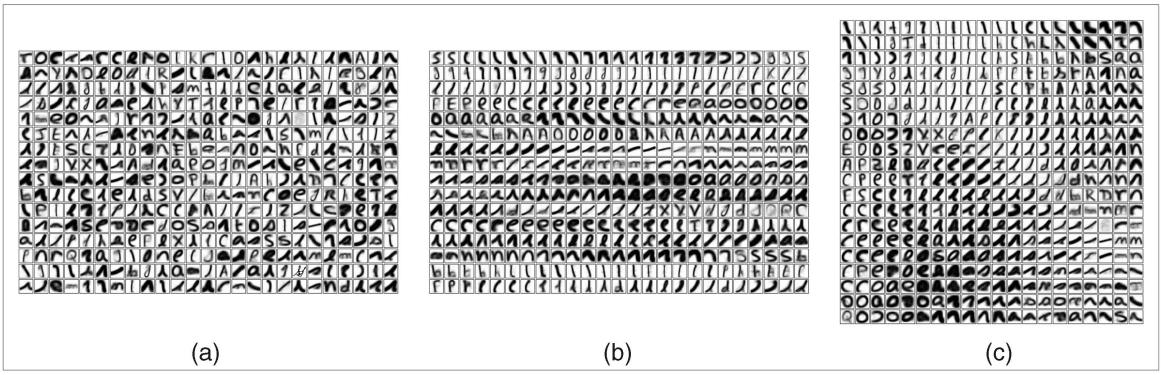

Figure 2.1: Examples of codebooks with 400 graphemes. For **(a)** kmeans and **(b)** ksom1D the graphemes have been placed 25 in a row, while, for **(c)** ksom2D, the 20×20 original SOM organization has been maintained.

In 2010, Siddiqi and Vincent proposed the use of redundancy of graphemes to characterize writer individuality [51]. A universal codebook, computed with the k-means clustering algorithm, and additionally, a local codebook, computed for each document individually by hierarchical clustering, is formed from image patches.

In In 2019, Bennour et al. (2019) [15] exploited the key points in handwriting to generate the implicit shape codebook and identify the right writers. The problem with this type of methods is the complexity of the segmentation and feature extraction algorithms, which are highly expensive in terms of execution time.

2.2.2 Texture-based approaches

Image texture-based techniques for offline writer verification consider each digitized image of handwriting as a different texture and extract features from the whole document (Entire Image or EI) ([20]), Regions of Interest (or ROIs like blocks, grid cells,

connected-components, words, etc.) ([18], or Writing Fragments (WFs) ([29]. Probability Distribution Functions (PDFs) are calculated and employed to characterize the writer of a given sample. Since the content of the handwritten document is seen as a digitized image, a set of textural features can be extracted from handwriting images (EI, ROIs, or WFs) of each writer [10].

In 2013, Bertolini et al. (2013) [18] published a texture-based descriptor for writer identification. The authors used Local Binary Patterns (LBP) in a comparative study with a Local Phase Quantization (LPQ) and concluded that LPQ performed better than their LBP variant.

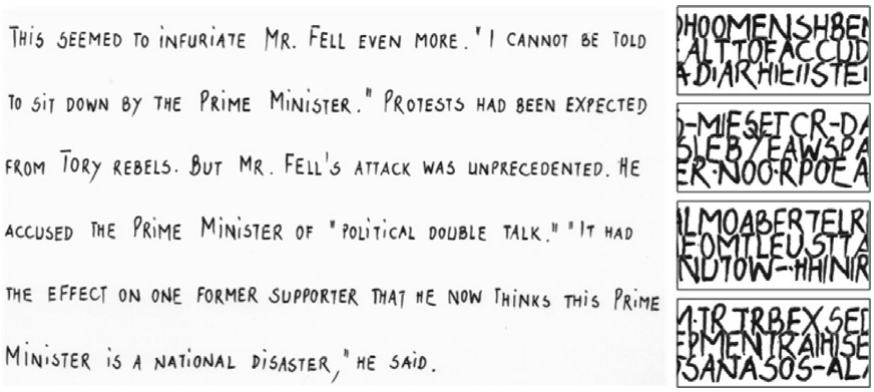

Figure 2.2: Original letter and texture blocks – sample from the IAM database [40]

2.2.3 Contour-based approaches

Contour-based methodologies represent a distinct category within the realm of offline writer verification [20, 51, 11]. In these approaches, Probability Distribution Functions (PDFs) of local attributes or features are derived from the inner and outer contour pixels, as opposed to the conventional image pixels. These features are specifically engineered to encapsulate writer-specific characteristics, including angle, direction, curvature, slant, ink-trace widths, and letter shapes [9, 10].

In 2007, Bulacu et al. [20] proposed to perform handwriting writer identification by using the contour-based features. Probability Distribution Functions (PDFs) are calculated using the entire handwriting image (document) and its contours for contour direction, contour-Hinge, direction co-occurrence and Run-Length on white distributions (RL). In 2010, Siddiqi and Vincent [51] proposed the use of visual attributes of orientation, and curvature to characterize writer individuality. In 2018, Kessentini et al. combined Edge-Hinge with a fragment length of 6 and 7 pixels and Run-length features in the Dempster-Shafer Theory (DST) model to improve the verification rate. Notably, Bahram [9] published a prominent writer recognition system in 2022 that utilizes texture contour-based features. This system leverages the joint distribution of the Modified Local Binary Pattern (MLBP) and the Ink-trace Width and Shape Letters (IWSL) measurements (MLBP-IWSL) to capture intricate handwriting details, including direction, curvature, slant, and shapes, thereby characterizing individual writing styles.

It is important to mention that, in general, the texture-based and contour-based approaches are proven to be efficient in terms of execution time and are generally preferred

when only a certain minimum amount of handwriting data is available (i.e., with an important number of writers) [9]. In this research investigation, we originally, spotlighted on the third category, which is hinged on the contour-based descriptors (computed from binary connected components and their contours).

2.2.4 Methodology and Technologies

To achieve these objectives, the following methodology and tools will be employed:

- **Programming Languages:** C++ and C# for system development and performance optimization.
- **Platforms:** using both Windows operating system.
- **Development Tools:** Microsoft Visual Studio, CPPunit for unit testing.
- **Modeling and Design:** StarUML for system architecture modeling; application of software engineering best practices using Design Patterns.
- **Data Preparation and Visualization:** Adobe Photoshop and GIMP for image preprocessing and annotation.
- **Build Acceleration and Version Control:** IncrediBuild for distributed build acceleration, and TortoiseSVN for source code versioning.
- **Documentation:** TeXStudio and MiKTeX for LaTeX writing, with Beamer used for presentation preparation.

2.3 Project Timeline and Work Plan

The development of the writer verification systems involves several critical stages to ensure scientific robustness and practical applicability. This chapter outlines the detailed timeline, objectives, and methodologies used throughout the project. Each phase is designed to build upon the previous one, culminating in a functional system and a comprehensive academic report.

2.3.1 Phase 1: Literature Review on contour-Based Writer Verification Systems

Duration: 1 month

The first phase focuses on conducting an in-depth literature review of writer verification systems, emphasizing texture-based approaches. This phase includes:

- Reviewing classical and modern writer verification methods, including codebook-based, contour-based, and texture-based approaches.
- Analyzing benchmark competitions such as ICDAR 2011 and ICDAR 2013 to understand common datasets, evaluation protocols, and performance baselines for verification tasks.

- Investigating the challenges of verifying writers in mono-script environment.
- Summarizing the strengths and limitations of each method and identifying knowledge gaps.

This foundational knowledge guided the project's direction and the design of new discriminative features for writer verification.

2.4 Phase 2: Proposal of Discriminative Features

Duration: 2 months

The second phase centers on proposing new features for writer verification, inspired by expert analysis in forensic document examination and modern image processing techniques. This phase includes:

- Extracting low-level and mid-level features capturing texture, stroke direction, curvature, and edge statistics.
- Introducing two original features tailored for multi-script verification environments.
- Performing feature normalization and dimensionality reduction for improved classification.
- Designing a modular pipeline to test different feature combinations for verification.

2.5 Phase 3: Presentation of Results to Evaluate Proposed Features

Duration: 1 month

The third phase aims to experimentally evaluate the proposed features using real datasets, specifically the IAM and KHATT databases. It includes:

- Benchmarking on the IAM and KHATT multilingual datasets for writer verification.
- Evaluating performance using metrics such as False Acceptance Rate (FAR), Equal Error Rate (EER), and False Rejection Rate (FRR).
- Comparing the proposed system's performance with state-of-the-art writer verification methods.

The results of these experiments are presented and analyzed in the Results Analysis chapter.

2.6 Phase 4: Preparation of the Final Thesis

Duration: 1 month

The final phase involves compiling the entire work into a well-structured thesis document. Activities include:

- Structuring the manuscript according to academic standards.
- Integrating all diagrams, tables, and results into a unified format.
- Final proofreading, formatting in LaTeX, and preparing for submission.
- Preparing presentation slides and defense materials.

The documentation also discusses potential extensions of this research and prospects for Ph.D. work in biometric verification.

Keywords

- Behavioral biometrics
- Offline handwritten text
- Texture
- Writer verification
- Feature extraction/combination
- Forensic document examination

2.7 Objectives (Prioritized)

1. Achieve the primary goal of the internship: develop a robust writer verification system.
2. Design, develop, and test a writer verification system (unit and integration tests).
3. Produce a scientific publication based on the results (ICDAR or ICFHR competitions).
4. Explore opportunities for continuation in a Ph.D. program in biometric verification.

This chapter presented the detailed timeline and methodology for the multilingual writer verification system project. Each phase was designed to incrementally build knowledge, tools, and experimental insights, ensuring a rigorous and scientifically sound outcome. The following chapters detail the technical implementation and empirical evaluation using the IAM and KHATT databases.

Chapter 3

Proposed technique

> Computer Science is no more about computers than astronomy is about telescopes.
>
> Edsger Dijkstra

3.1 Introduction

In this section, two primary representations of the scanned handwriting images are employed during the feature extraction step: the binary connected-components and their exterior contours. The contour contains a sequence of pixels located on the ink-background boundary of a connected-component. This is a very effective vectorial representation that will allow a fast computation of the proposed features. In addition, contours are important cues to describing writing shapes. For writer characterization, the Modified Local Binary Pattern (MLBP) and angle histograms (contour-based descriptors) can provide more information about the distribution of the attributes of writing (i.e., direction, slant, and curvature) while the overall information about the ink-trace width and letters' shapes can also be captured by using the IWSL and Run-length distributions. In this work, we propose the $MLBP - IWSL$ and $AC - RL$ as inspired by the co-occurrence features. As illustrated in Figure 3.1, our proposed approach consists of three main processing steps: **1.** pre-processing, **2.** feature extraction, and **3.** classification (verification). Each step is detailed in the following subsections.

3.2 Image pre-processing

Like most of the pattern recognition problems, the preprocessing step plays an important role in the performance of the writer identification system. For our experiments, the input data are in black connected-components and their contours, so the pre-processing step is necessary and it consists in binarizing the document images (foreground/background separation), extracting the connected-components, removing the nonsignificant connected components (i.e., small components, dots, etc.) and extracting the interior and exterior contours of connected components. In this section, a very effective vectorial representation is considered which will subsequently allow a fast calculation of the proposed feature. Before proceeding to the connected components' extraction, we binarize the handwritten grayscale images in a manner that preserves the foreground

(ink pixels), using the global thresholding Otsu's method (Otsu, 1979) [41]. From the black-and-white image, each binary region of interest (or connected component) is detected using 8-connected pixels. As stated in Bahram et al. (2016) [11], a connected component can be defined as an ink blob or complete region of ink trace, determined by a set of pixels, i.e., all connected to each other, drawn without lifting the pen from paper. In binary images of handwriting, the foreground (black) pixels correspond to the black ink trace and the white pixels correspond to the background. In order to capture more details of the directional strokes, the slant and skew corrections of the handwritten document are not used in this work. Subsequently, some non-significant connected components like diacritics, small-components with a width or height smaller than one strokewidth, scatter noise, periods, commas, and dots, are discarded in this step. Next, from the remaining (labeled) connected components, we compute and extract the interior and exterior contours using Moore's algorithm.

3.2.1 Image Processing Stages

As covered in Bahram [9], there are several image processing stages that generally need to be applied in order to make use of the image data in writer recognition. For these experiments, the input data format is in connected-components, so the following stages are necessary and are briefly summarised below: binarization, connected-component extraction, and contours (outer and inner) extraction.

3.2.1.1 Binarization

The vast majority of offline writer recognition techniques work with binary input images, rather than colour or grayscale. Although it appears that no comprehensive studies have been done, available results suggest that despite the information loss, this is the most useful format to work with. Although grayscale may be appropriate for some features, e.g. attempting to reconstruct pen pressure, for the most part it makes the resulting images too complicated to process further.

Binarization is a key step in preparing images for analysis. It takes a grayscale or color image and simplifies it by turning it into a black and white picture, where only two colors exist: black for important details like text or handwriting, and white for the background, like the paper. This simplification makes it much easier to perform tasks such as recognizing characters, verifying writers, or detecting shapes and outlines. Depending on how this threshold is determined, binarization methods are commonly divided into two main categories:

- **Global Binarization :** Global binarization applies a single threshold value to the entire image. Each pixel is classified as foreground or background based on this one global threshold.
  Advantages: Simple, fast, and effective for images with uniform lighting and contrast.
  Common method: Otsu's method, which automatically selects an optimal threshold by maximizing inter-class variance.

- **Local Binarization :** Local binarization, also known as adaptive binarization, computes a different threshold for each pixel based on the local neighborhood (a small window around each pixel). This makes it more suitable for documents with

irregular lighting, shadows, and low quality.
Advantages: Durable to variations in lighting and background noise.
Common methods: Niblack, Sauvola, and Wolf algorithms, which dynamically adjust thresholds using local mean and standard deviation.

Aspect	Global Thresholding	Local Thresholding
Characteristics	Uses a single threshold value for the entire image. It is less effective for images with uneven illumination.	Computes thresholds based on neighboring pixel intensities, adapting to local illumination and contrast variations.
Threshold Value Determination	Threshold is constant across the image.	Thresholds vary across different image regions, based on local statistics.
Illumination Handling	Performs poorly under varying lighting; suitable for uniformly illuminated images.	Robust to lighting variation; ideal for non-uniformly lit scenes.
Computational Complexity	Computationally efficient and fast.	Requires more processing due to per-pixel or per-region computations.
Use Cases	Best suited for clean, scanned documents or uniformly lit environments.	Preferred for natural scenes or images with shadows and inconsistent lighting.

Table 3.1: Comparison between global and local thresholding.

3.2.1.2 Binarization Methods :

1. Otsu's Method (Global Thresholding) : Otsu's method is a nonparametric, unsupervised technique for automatic image thresholding. It selects the optimal threshold by maximizing the between-class variance using histogram moments. The method is simple, efficient, and extendable to multilevel thresholding.[41]

$$\sigma_b^2(t) = \omega_1(t)\omega_2(t)\left[\mu_1(t) - \mu_2(t)\right]^2$$

Where:

- ω_1, ω_2 are the class probabilities
- μ_1, μ_2 are the class means

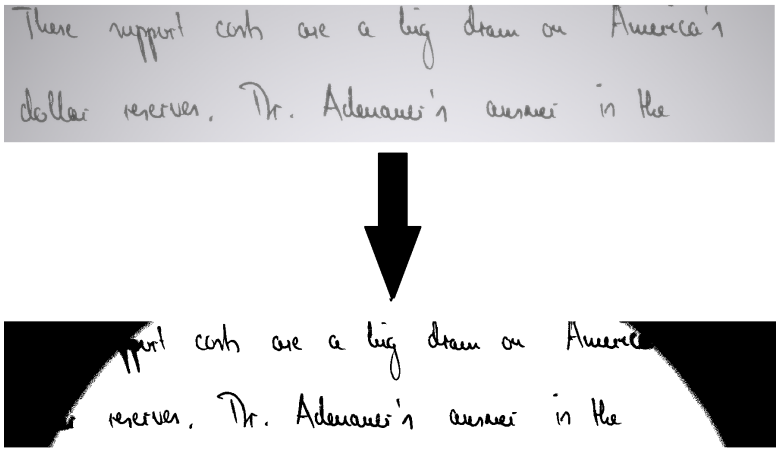

Figure 3.2: Global Image Thresholding using Otsu's Method

2. Niblack's Method (Local Thresholding) : Niblack's method is a local thresholding technique used in image processing to segment an image into foreground and background regions. It computes the threshold for each pixel based on the mean and standard deviation of the pixel intensities in a local neighborhood[38].

$$T(x, y) = m(x, y) + k \cdot s(x, y)$$

Where:

- $m(x, y)$ is the local mean
- $s(x, y)$ is the local standard deviation
- k is a tunable parameter, usually between -0.5 and 0.5

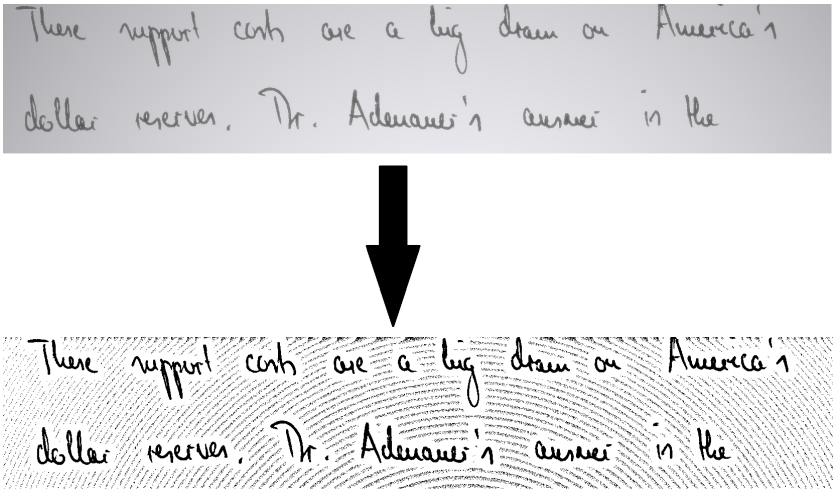

Figure 3.3: local Image Thresholding using Niblack's Method

3. Sauvola's Method (Improved Local Thresholding) : Sauvola's method improves upon Niblack's adaptive thresholding by handling uneven illumination. It uses the local mean and standard deviation with a dynamic range parameter for better adaptability ,This makes it effective for binarizing document images with text and background noise[38].

The formula for the threshold is:

$$T(x, y) = \mu(x, y) \left[1 + k \left(\frac{\sigma(x, y)}{R} - 1\right)\right]$$

Where:

- R is the dynamic range of standard deviation (typically 128)
- k is typically in the range [0.3, 0.5]

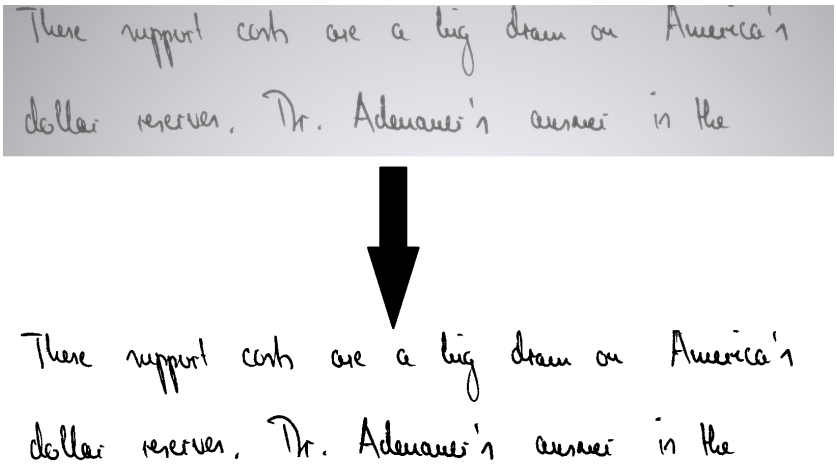

Figure 3.4: local Image Thresholding using sauvola's Method

4. K-means Clustering : K-meansis an unsupervised algorithm and it is used to segment the interest area from the background. It clusters, or partitions the given data into K-clusters or parts based on the K-centroids(typicaly k=2).

Method	Type	Lighting	Parameters	Noise Robustness	Best For
Otsu	Global	Uniform	No	Low	Clean scanned documents
Niblack	Local	Irregular	Yes	Medium	Historical, noisy text
Sauvola	Local	Irregular	Yes	High	Noisy/low contrast docs
K-means	Clustering	Any	Yes	High	Degraded/irregular docs

Table 3.2: Comparison of Binarization Methods

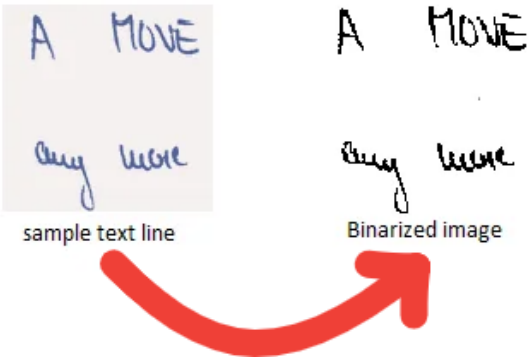

Figure 3.5: Binarization Process

For our work, we decided to use Otsu's method to binarise the handwriting images. The main reason is that our images are mostly clean and well-scanned, with good lighting and a clear difference between the text and the background. Otsu's method is great in this kind of situation because it automatically finds the best threshold to separate the black writing from the white background. It's also fast and simple to use, which made

it a practical choice. By using Otsu's method, we were able to get clean binary images that made the next steps, like contour detection and writer analysis, much easier and more reliable.

3.2.2 Connected Components Extraxtion :

Connected components extraction is a key step in processing binary images where the goal is to identify and separate individual objects made up of connected pixels. In a black-and-white image, for example, the black pixels usually represent meaningful content such as handwritten letters, symbols, or shapes, while the white pixels represent the background.

The extraction process works by examining the image to find groups of black pixels that are connected to each other based on a connectivity rule. Typically, this rule considers either 4-connectivity (pixels connected horizontally or vertically) or 8-connectivity (including diagonal neighbors). Pixels that meet this connectivity condition are grouped together as one connected component.

Once these groups are identified, each connected component is assigned a unique label. This labeling allows us to treat each component as an individual object for further analysis. For example, in handwriting recognition or writer identification, connected components often correspond to letters, strokes, or words that need to be analyzed separately.

This step is crucial because it simplifies complex images by breaking them down into smaller, meaningful parts. After extracting connected components, other important tasks such as contour tracing, feature extraction, or classification become more manageable and accurate[22].

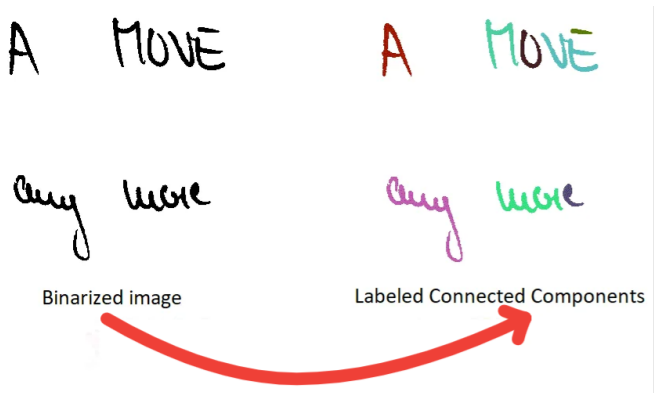

Figure 3.6: Connected Components Extraxtion Process

3.2.3 Removal of Small Connected Components :

This is an important cleaning step in image processing, especially after connected components extraction. When working with binary images, small groups of connected pixels often appear due to noise, dust, or minor imperfections in the scanning process. These tiny components usually do not carry useful information—for example, random dots, specks, or small artifacts—and can interfere with later analysis steps.

To improve the quality of the data, these small connected components are identified

and removed based on their size, typically by setting a minimum pixel area threshold. Any connected component smaller than this threshold is discarded from the image. This helps to reduce noise and focus on the meaningful parts of the image, such as actual letters or handwriting strokes.

Removing small connected components is essential to improve the accuracy of tasks like writer identification, character recognition, or contour extraction, because it ensures that only significant and relevant parts of the image are processed.

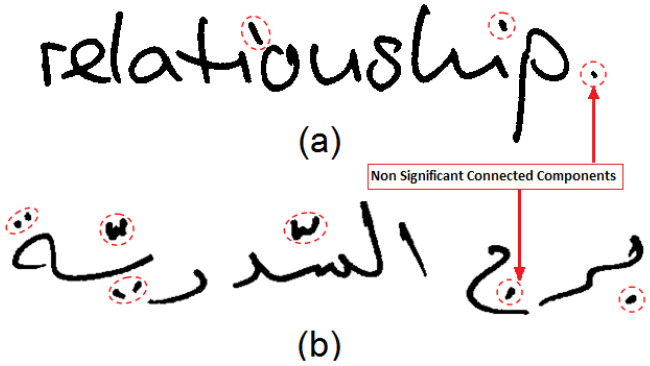

Figure 3.7: Non significant Connected Components Detection

3.2.4 Contour Detection :

Contour detection is the process of tracing the outlines or borders of shapes in a binary image. In the context of handwriting analysis, these shapes are the connected components representing individual letters or strokes. This step helps us understand the exact form and structure of each component, which is essential for detailed analysis. There are two types of contours we usually extract:

- **Outer contours**, which trace the external boundary of a shape. For example, in the letter "O", the outer edge forms the main visible border.
- **Inner contours**, which trace holes or enclosed spaces inside a shape like the circular gap inside the letter "O" or "P".

To perform contour detection, one commonly used method is **the Moore-Neighbor Tracing Algorithm**, which is simple and effective for binary images[28].

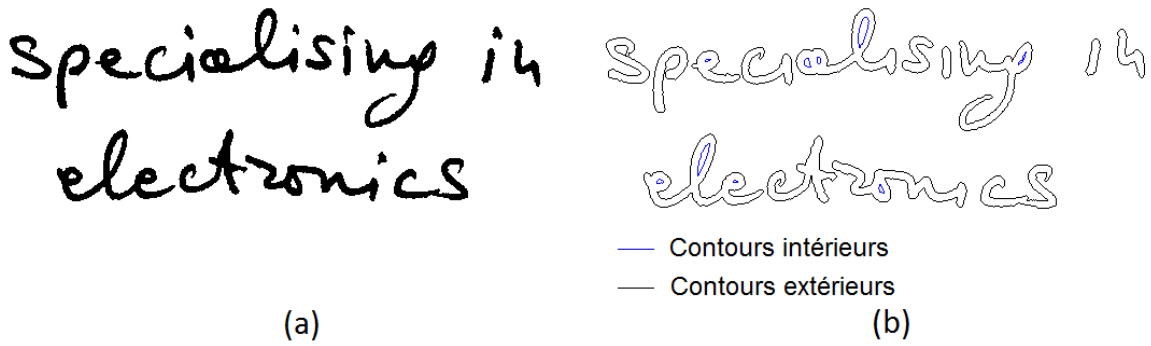

Figure 3.8: Outer-Iner Contour Detection

The Moore-Neighbor Tracing Algorithm :

The Moore-Neighbor Tracing algorithm is a classical method used to extract the boundary (or contour) of a shape in a binary image. It is especially useful when the shape consists of connected pixels, such as characters in handwriting, and we want to trace its exact border[28].

3.2.4.1 Moore Neighborhood :

The Moore neighborhood of a pixel (also known as the 8-neighbors), P, is the set of 8 pixels that share either a vertex or an edge with P. These neighboring pixels are denoted as $P_1, P_2, ..., P_8$ as illustrated in the figure below[28].

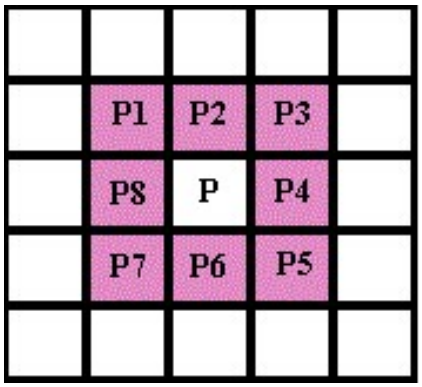

Figure 3.9: The Moore neighborhood of a pixel P[28]

3.2.4.2 Algorithm Idea :

Given a digital pattern (a group of black pixels) on a white background arranged in a grid, we first locate a black pixel and declare it as the *start* pixel. This can be done by scanning from the bottom to the top and left to right until a black pixel is found. Imagine a ladybug placed on this start pixel. The goal is to trace the contour by walking around the shape in a clockwise direction. At every step, when a black pixel P is reached, we backtrack to the white pixel from which we entered P, and then explore all pixels in the Moore neighborhood of P in a clockwise order until another black pixel is found. The tracing continues until the start pixel is visited again in the same manner.

All the black pixels visited during this process form the boundary or contour of the shape[28].

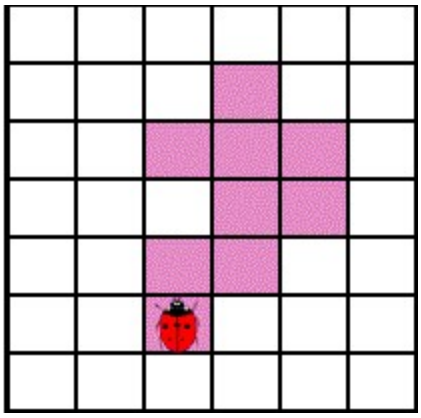

Figure 3.10: standing on the start pixel.

3.2.4.3 Formal Algorithm :

Input: A square grid pattern T containing a connected component P of black cells.
Output: A sequence $B = (b_1, b_2, ..., b_k)$ of boundary pixels (the contour).

1. Initialize B as an empty list.
2. Scan T from bottom to top and left to right until a black pixel $s \in P$ is found.
3. Insert s into B.
4. Set the current boundary point $p := s$.
5. Backtrack to the pixel from which s was entered.
6. Set c to be the next clockwise pixel in the Moore neighborhood $M(p)$.
7. **While** $c \neq s$:
   - If c is black:
     - Insert c into B
     - Set $p := c$
     - Backtrack to the pixel from which p was entered
   - Else:
     - Advance c to the next clockwise pixel in $M(p)$
8. End While

Result: The sequence B contains the coordinates of the contour pixels[28].

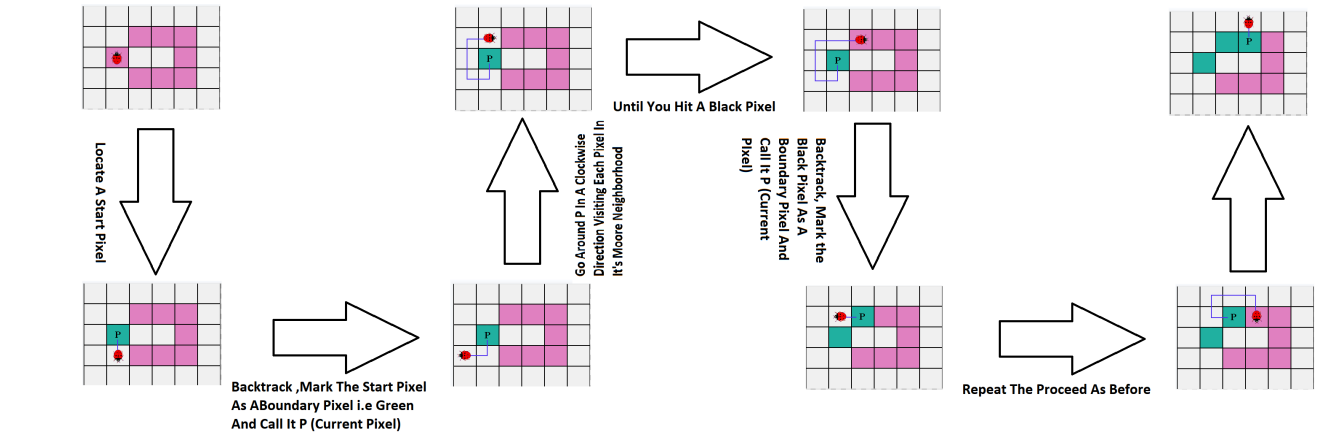

Figure 3.11: description of the Moore-Neighbor tracing algorithm[28]

3.3 Feature Extraction

3.3.1 Run-Length

The concept of Run-Length (RL) distribution was first introduced by Arazi in 1977[6] as one of the earliest features for automatic writer recognition. Since then, it has gained recognition as an effective global descriptor for capturing the unique stylistic traits of handwritten text. The fundamental idea behind this method is to analyze sequences of pixels called *runs* that share similar visual properties across specified directions in the image.

A run is defined as a series of consecutive, connected pixels that exhibit the same characteristic, such as pixel intensity[25]. In the context of binary images (i.e., black-and-white images where black denotes ink and white represents the paper background), run-length analysis is used to measure the lengths of such sequences. These sequences are measured separately for black pixels (foreground) and white pixels (background).

The white pixel runs provide valuable insight into the spatial layout of the handwriting, including intra- and inter-letter spacing as well as the curvature and openness of character shapes. Conversely, the black pixel runs convey details about the writing instrument, particularly the thickness of the strokes or the width of the ink traces, which are highly characteristic of a writer's style.

Run-length measurements can be conducted in multiple orientations to capture directional textural features. The four commonly used directions are:

- Horizontal (0°)
- Vertical (90°)
- Diagonal (45°)
- Diagonal (135°)

In each of these directions, the lengths of the detected runs are compiled into histograms. These histograms are then normalized to form probability distribution functions (PDFs), which effectively describe the textural structure of the handwriting. These PDFs serve as discriminative features for identifying individual writers.

To further enhance the analysis beyond binary images, **Gray-Level Run-Length Matrices (GLRLMs)** can be employed. This extension allows the system to work directly with grayscale images without needing a binarization step[24]. In this case, a run is defined as a sequence of neighboring pixels that all share the same gray-level intensity g. The GLRLM stores this information in a matrix $M(g, h)$, where each element corresponds to the number of runs with gray level g and length h.

GLRLMs can also be computed in the four principal directions (0°, 45°, 90°, and 135°) to capture multi-directional textural information[43]. Once generated, these matrices are likewise converted into normalized histograms, providing compact statistical representations of the textural patterns found in handwriting. These descriptors are then used as part of the writer identification system to distinguish between different writing styles.

An example illustrating how run-lengths are computed across the four directions is provided in Fig. 4.8, which demonstrates the visual formation of both black and white run-length distributions in different orientations.

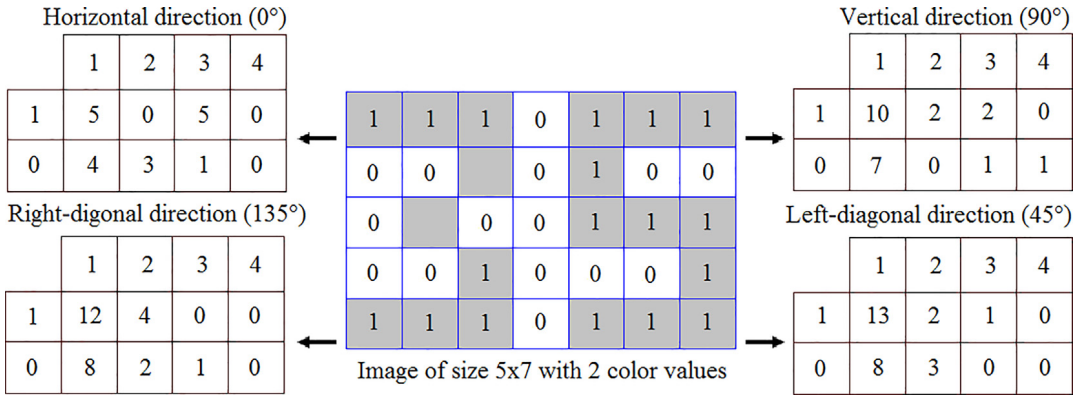

Figure 3.12: Calculation of run-length matrices

3.3.2 Contour Direction(Angle)

The function ϕ is defined as the angle of a local tangent line with respect to the horizontal straight line. This technique is based on similar ideas but applied specifically to connected components.

Two types of contour directions are used in this approach:

- **Right contour direction** ϕ_r $\left(-\frac{\pi}{2} < \phi_r < \frac{\pi}{2}\right)$
- **Upper contour direction** $\phi_u (\phi_u < \pi)$

The parameter r has been empirically set to a fixed number of pixels (depending on the ink trace thickness in dataset). Next, these angular values are normalized into 12 equal segments through experimentation. Each segment, representing an angle of 15°, provides a detailed yet robust description of handwriting suitable for writer recognition.

These features are illustrated in Fig. 4.14. The locally measured directions are accumulated into a histogram. This histogram is then normalized and interpreted as a probability distribution function $p(\phi)$, which serves as a descriptor of the writer's style.

Angles ϕ_r and ϕ_u are used to compute the probability distributions for the right contour direction and the upper contour direction, respectively[12].

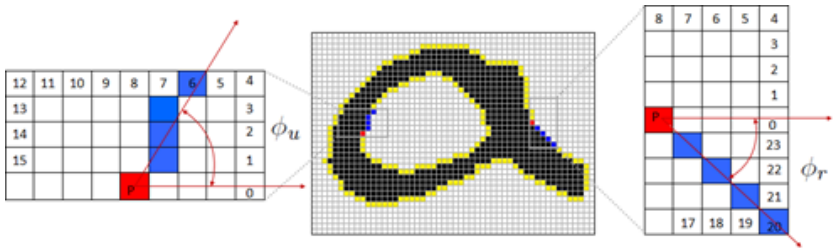

Figure 3.13: Extraction of the contour direction and angle features, (a) computing local right (Φ_r) and upper (Φ_r) directions

3.3.3 Ink-trace Width and Shape Letters (IWSL)

An IWSL measurement refers to the number of consecutive black and white pixels located between two exterior contour points along a specific direction in a binary image. Formally, the IWSL can be defined as follows:

Let S be a binary sequence of connected pixels in a bi-level (binary) image, where each pixel is represented as either 1 (black) or 0 (white). Let p_{cs} and p_{ce} denote the starting and ending positions of the sequence S, and let $p_1, \ldots, p_n$ represent the pixels lying between the two exterior contour points.

The IWSL is then computed as the Euclidean distance between the two exterior contour points:

$$p_{cs} = (x_{cs}, y_{cs}), \quad p_{ce} = (x_{ce}, y_{ce})$$
$$\text{IWSL} = \sqrt{(x_{ce} - x_{cs})^2 + (y_{ce} - y_{cs})^2} \tag{3.1}$$

This measurement provides a numeric estimation of the stroke length between two contour points in a specific direction[9].

The IWSL computation for exterior contour pixels is based on techniques similar to those described in [24, 25, 33]. The measurement is evaluated along four main directions to effectively capture structural writing details such as character shapes, average letter widths, and ink stroke width:

- Horizontal direction: IWSL_1
- Vertical direction: IWSL_2
- Left-diagonal direction: IWSL_3
- Right-diagonal direction: IWSL_4

These directional IWSL measurements serve as valuable features in handwriting analysis. An illustration of the IWSL computation on a binary image is shown in Fig. 4.9.

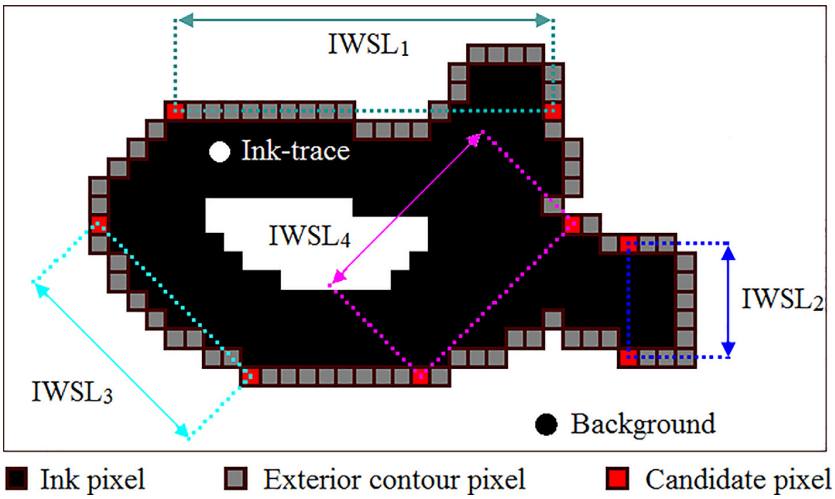

Figure 3.14: Computing local IWSL in the four directions

3.3.4 Modified Local Binary Pattern (MLBP)

The modified LBP (MLBP) operator labels each candidate pixel p_c at coordinates (x_c, y_c) in a binary image (connected components) I_B by examining its eight neighbors within a 3×3 neighborhood $(n_0, ..., n_7)$. It produces a binary pattern by concatenating eight binary digits (0s and 1s) and converting the result into a decimal number.

The MLBP for a candidate pixel p_c is defined as:

$$\text{MLBP}_{8,1}(p_c) = \sum_{k=0}^{7} s(n_k) \cdot 2^k \tag{3.2}$$

where the function $s(n_k)$ is given by:

$$s(n_k) = \begin{cases} 1, & \text{if } n_k \in EC_{IB} \text{ and } p_k \in I_B \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

Here, EC_{IB} refers to the exterior contours of the binary image I_B. An illustration of the modified LBP operator is shown in Fig. 5.

Depending on the writing direction (i.e., directionality), languages can be classified as either Right-to-Left (RTL) or Left-to-Right (LTR) scripts. RTL scripts, such as Arabic, Farsi, Hebrew, and Urdu, are written from right to left and from top to bottom. In contrast, languages such as English, Greek, Dutch, and Latin are written from left to right.

Based on these directional properties, two variants of MLBP are proposed:

- **Left MLBP** (denoted by MLBP_1)
- **Upper MLBP** (denoted by MLBP_2)

They are computed as follows:

$$\text{MLBP}_1(p_c) = \sum_{k=0}^{1} s(n_{k+6}) \cdot 2^k + \sum_{k=2}^{4} s(n_{k-2}) \cdot 2^k \tag{3.4}$$

$$\mathrm{MLBP}_2(p_c) = \sum_{k=0}^{4} s(n_k) \cdot 2^k \tag{3.5}$$

In writer identification tasks, diagonal features are also important and have been shown to provide excellent results on widely used handwriting databases[25]. Therefore, two additional diagonal MLBP operators are proposed:

- **Left-Diagonal MLBP** (denoted by MLBP_3)
- **Right-Diagonal MLBP** (denoted by MLBP_4)

These diagonal MLBP values are computed as:

$$\mathrm{MLBP}_3(p_c) = \sum_{k=0}^{4} s(n_{k+1}) \cdot 2^k \tag{3.6}$$

$$\mathrm{MLBP}_4(p_c) = \sum_{k=0}^{4} s(n_{k+3}) \cdot 2^k \tag{3.7}$$

Figure 4.15 shows an example of the modified LBP pattern and its decomposition into the four MLBP codes described above.

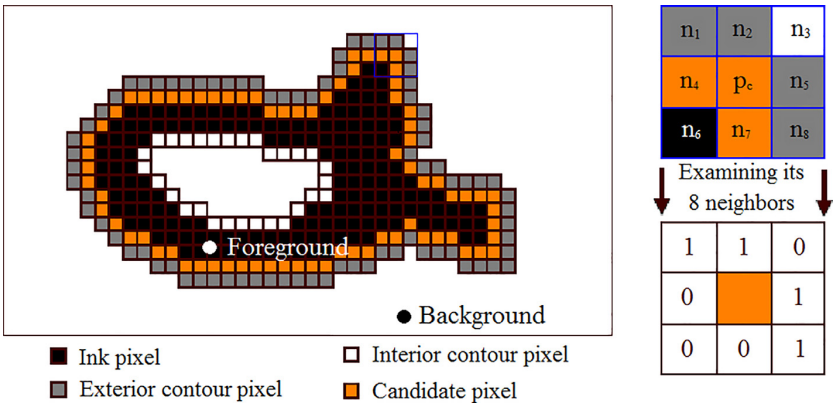

Figure 3.15: Local information used for MLBP code calculation[9]

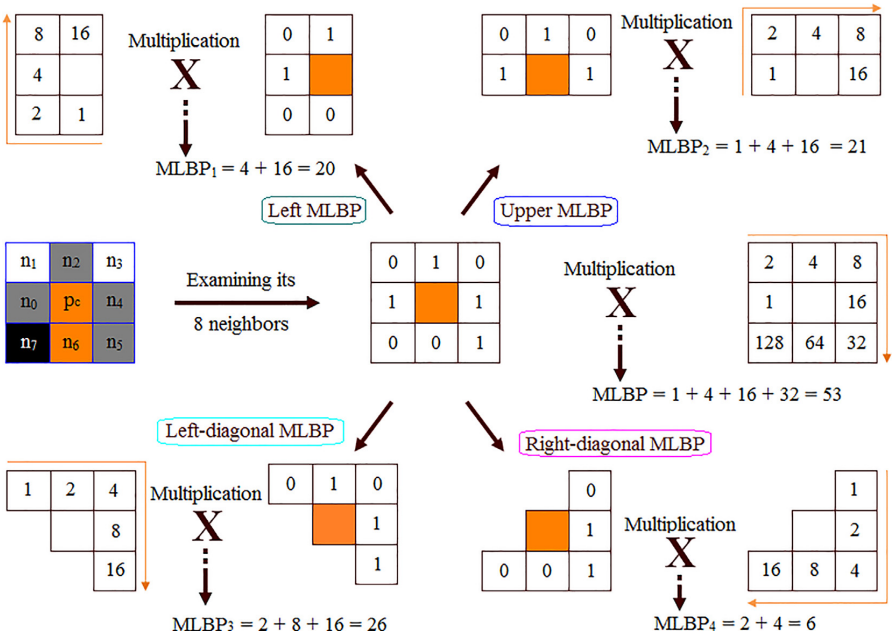

Figure 3.16: Computing the modified LBP code and its splitting into Left, Upper, Left- and Right-Diagonals MLBP codes (P $\frac{1}{4}$ 8 and R $\frac{1}{4}$ 1:0).[9]

3.3.5 Contour Direction-Length Feature (AG-RL)

The contour direction-length feature consists of a few simple parts that together form a powerful method for writer recognition: contour direction measurements (ϕ), length measurements (L), and the calculation of a probability distribution function (PDF).

Figures 4.18 illustrate the contour direction and length measurements, which form the core of the contour direction-length feature.

The features F_1 and F_2 correspond to the feature vectors of the right contour direction (ϕ_r) combined with horizontal length (L_h), and upper contour direction (ϕ_u) combined with vertical length (L_v), respectively[12].

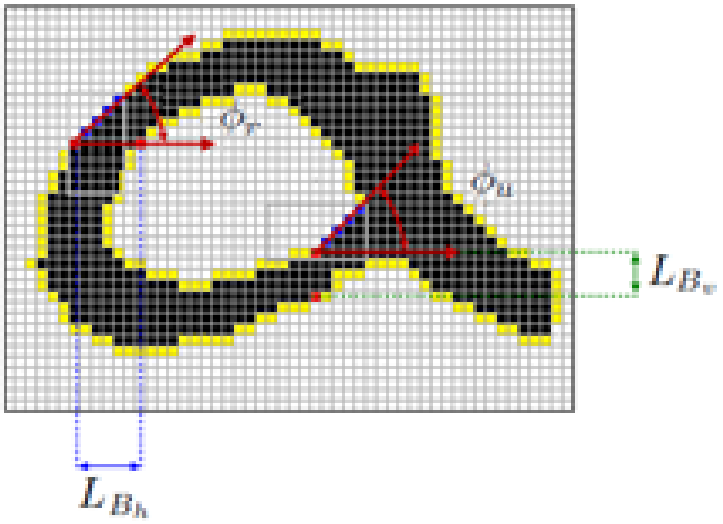

Figure 3.17: computing contour direction-black length

3.3.6 MLBP-IWSL

The joint distribution of the Modified Local Binary Pattern (MLBP_k) and the Ink-trace Width and Shape Letters (IWSL_k) measurements, where $k \in \{1, 2, 3, 4\}$.

For each input handwritten document image , the $F_{\text{MLBP}_k \times \text{IWSL}_k}$ feature vector is a probability distribution obtained by normalizing the histogram $H_{\text{MLBP}_k \times \text{IWSL}_k}$ and is defined as follows:

$$F_{\text{MLBP}_k \times \text{IWSL}_k} = \frac{H_{\text{MLBP}_k \times \text{IWSL}_k}}{\|H_{\text{MLBP}_k \times \text{IWSL}_k}\| + \varepsilon} \tag{3.8}$$

where ε is a very small value close to zero, and $H_{\text{MLBP}_k \times \text{IWSL}_k}$ is a histogram computed by using the following equation:

$$H_{\text{MLBP}_k \times \text{IWSL}_k}(l) = \sum_{i=1}^{N} \sum_{p_c \in C_i} \delta\left[l,\ ((\text{MLBP}_k - 1) \cdot N_{\text{IWSL}} + \text{IWSL}_k)\right], \quad l = 1, \ldots, (N_{\text{MLBP}} \cdot N_{\text{IWSL}}) \tag{3.9}$$

3.3.7 Feature Fusion

Fusion techniques are recognized for improving performance and have been utilized in various classification tasks in general [37], and in biometric applications specifically [27, 46, 45]. Fusion can be performed at the feature level [54, 42], where diverse features capturing different types of information are integrated, or at the decision level [55, 35], where outputs from multiple classifiers are combined to improve overall system performance.

This approach calculates a single numerical value, called a scalar score, which represents how different or similar the input sample is compared to the claimed identity.

In the context of writer verification, this score helps decide whether the handwriting sample belongs to the claimed writer or not.

To get this scalar score, the system first extracts multiple dissimilarity measures from the input sample. Each dissimilarity measure comes from a different textural descriptor—these are algorithms or features that capture various patterns and textures in the handwriting. Since different descriptors focus on different aspects of the handwriting (such as stroke thickness, curvature, or texture), combining their results provides a more comprehensive evaluation[30].

Once the individual dissimilarity scores are computed, they need to be combined into one final score. This is done using fusion rules, which are mathematical methods that integrate multiple values into a single representative number. Common fusion rules include:

- **Sum rule**: Adding all the individual scores together. This assumes that all descriptors contribute equally to the final decision[30].

  Let:

  - D_i denote the distance (or dissimilarity score) obtained from the i-th feature.
  - N be the total number of features.
  - $D_i \in \mathbb{R}$, and a special value (e.g., $D_i = \text{DBL\_MAX}$) indicates an invalid or undefined feature score.

  Then the normalized sum fusion distance D_{fused} is defined as:

$$D_{\text{fused}} = \begin{cases} \frac{1}{N} \sum_{i=1}^{N} D_i, & \text{if } D_i \neq \text{DBL\_MAX} \quad \forall i \\ \text{DBL\_MAX}, & \text{if any } D_i = \text{DBL\_MAX} \end{cases}$$

  Key Characteristics

  - **Robustness:** If any feature yields an invalid distance (DBL_MAX), the fused distance is set to DBL_MAX to flag unreliability.
  - **Normalization:** The sum of all valid distances is divided by the number of features to ensure scale consistency.
  - **Use Case:** This rule is typically applied when all features are expected to contribute equally and are on a comparable scale.

- **Product rule**: Multiplying all the scores. This method emphasizes agreement among descriptors since one very low score can reduce the total significantly[30].

  Let:

  - D_i be the distance from the i-th feature.
  - N be the total number of features.

$$D_{\text{fused}} = \begin{cases} \prod_{i=1}^{N} D_i, & \text{if } D_i \neq \text{DBL\_MAX} \quad \forall i \\ \text{DBL\_MAX}, & \text{if any } D_i = \text{DBL\_MAX} \end{cases}$$

  Key Characteristics

- **Amplifies large differences:** Sensitive to outliers and large distances.
- **Multiplicative:** Strongly penalizes any high distance.
- **Use Case:** Useful when features must all be jointly low to indicate similarity..

- **Min rule**: Taking the smallest (minimum) score among the descriptors. This could be useful when one strong match is enough to accept the sample[30].

$$D_{\text{fused}} = \begin{cases} \min\limits_{i=1}^{N} D_i, & \text{if } D_i \neq \text{DBL\_MAX} \quad \forall i \\ \text{DBL\_MAX}, & \text{if any } D_i = \text{DBL\_MAX} \end{cases}$$

Key Characteristics

- **Pessimistic:** Prioritizes the worst-case (highest) distance
- **Robust to low distances:** One bad score dominates.
- **Use Case:** Suitable when any dissimilarity should trigger rejection..

- **Max rule**: Taking the largest (maximum) score, which might reflect the worst-case dissimilarity[30].

$$D_{\text{fused}} = \begin{cases} \max\limits_{i=1}^{N} D_i, & \text{if } D_i \neq \text{DBL\_MAX} \quad \forall i \\ \text{DBL\_MAX}, & \text{if any } D_i = \text{DBL\_MAX} \end{cases}$$

Key Characteristics

- **Optimistic:** Considers the best-case (smallest) distance.
- **Normalization:** May ignore bad scores: One good match can dominate.
- **Use Case:** Suitable when any strong similarity is sufficient.

By applying these fusion rules, the system consolidates the diverse information from multiple descriptors into a single meaningful score, improving the robustness and accuracy of the verification process.

Fusion Rule	Formula	Intuition	Strengths	Weaknesses	Typical Use Case
Sum	$\sum_{i=1}^{l} c_{i,j}$	Aggregate all evidence additively	Simple, effective, smooth	Sensitive to scale, may dilute strong signals	When features complement each other
Product	$\prod_{i=1}^{l} c_{i,j}$	Emphasize consensus	Penalizes disagreement, more discriminative	Sensitive to low values, assumes independence	When all features must agree
Max	$\max_{i=1}^{l} c_{i,j}$	Take strongest evidence	Robust to weak features	Can be optimistic, ignores other sources	When one feature is highly reliable
Min	$\min_{i=1}^{l} c_{i,j}$	Take weakest evidence	Conservative, reduces false acceptance	Can be overly pessimistic	High-security scenarios requiring consensus

Table 3.3: Summary of Fusion Rules[5]

3.3.8 Distance Calculation

In this work ,we measure the degree of dissimilarity between two samples by applying various distance functions such as Chi-square (χ^2), Manhattan, Euclidean, Minkowski

3.3.8.1 Chi-square (χ^2) Distance

Given two feature vectors T_1 and T_2, the distance function computes a value that quantifies the difference between them while normalizing for the scale of their individual elements.

Formally, the Chi-square distance D between two vectors of equal length n is defined as:

$$D(T_1, T_2) = \sum_{i=1}^{n} \frac{(T_1[i] - T_2[i])^2}{T_1[i] + T_2[i]}$$

where $T_1[i]$ and $T_2[i]$ represent the i^{th} components of vectors T_1 and T_2, respectively.

The implementation follows these key steps:

Dimension Check: The function first verifies that both input vectors have the same dimension. This ensures that the comparison is valid and element-wise correspondence exists.

Selective Computation: To avoid division by zero or meaningless terms, the function only includes in the summation those indices where at least one of the vector components is non-zero.

Distance Accumulation: For each valid index, the squared difference of the components is divided by their sum, effectively normalizing the difference by the combined magnitude of the components. This emphasizes relative differences over absolute differences, making the metric robust to scale variations.

This distance metric is widely used in pattern recognition and image processingand is commonly used in hypothesis testing to determine if two histograms come from the same distribution[48], due to its sensitivity to relative changes rather than absolute magnitudes. By using this function, we ensure that the computed distances are meaningful even when component values vary widely in scale or distribution.

The method provides an efficient $O(n)$ time complexity, where n is the length of the vectors, making it suitable for high-dimensional data comparison in real-time or large-scale applications.

3.3.8.2 Manhattan Distance

Given two feature vectors T_1 and T_2, the Manhattan distance function calculates the total absolute difference between their corresponding elements.

Formally, the Manhattan distance D between two vectors of equal length n is defined as:

$$D(T_1, T_2) = \sum_{i=1}^{n} |T_1[i] - T_2[i]|$$

where $T_1[i]$ and $T_2[i]$ are the i^{th} components of the vectors T_1 and T_2, respectively.

The implementation follows these steps:

Dimension Check: The function first ensures that both vectors have the same length. This guarantees a valid one-to-one element comparison.

Distance Accumulation: For each element index i, the absolute difference between the corresponding elements of the two vectors is calculated using `Math::Abs()`, and then added to the cumulative distance.

This approach effectively measures how different the two vectors are in terms of individual element values.

The Manhattan distance is commonly used in machine learning, clustering, and pattern recognition tasks where the total deviation is more relevant than the squared deviation (as in Euclidean distance). It is particularly useful in high-dimensional spaces where the geometry of data behaves differently[26].

The algorithm has a linear time complexity of $O(n)$, where n is the length of the vectors, making it suitable for large-scale applications or real-time systems.

3.3.8.3 Euclidean Distance

The Euclidean distance function calculates the straight-line distance between two feature vectors T_1 and T_2 in multi-dimensional space.

Formally, the Euclidean distance D between two vectors of equal length n is defined as:

$$D(T_1, T_2) = \sqrt{\sum_{i=1}^{n} (T_1[i] - T_2[i])^2}$$

where $T_1[i]$ and $T_2[i]$ denote the i^{th} components of the vectors T_1 and T_2, respectively.

The implementation consists of the following steps:

Dimension Check: The function begins by confirming that both vectors have the same length to ensure a valid element-wise comparison.

Distance Accumulation: A loop iterates over each index i, computing the square of the difference between corresponding elements. These squared differences are accumulated into the variable `distance`.

Square Root Application: After the loop, the square root of the accumulated value is taken to obtain the final Euclidean distance, which reflects the true geometric distance between the two vectors.

It is one of the most commonly used distance metrics in pattern recognition, machine learning, and statistical analysis.

This method provides a reliable and interpretable measure of similarity or difference between vectors. It has a time complexity of $O(n)$, where n is the dimensionality of the input vectors.

3.4 Writer Verification

To evaluate the performance of the offline writer verification system, we compute the Equal Error Rate (EER) based on a range of thresholds (θ) applied to the similarity scores between handwriting samples. The verification process relies on feature vectors extracted from each document and compares their dissimilarity using a selected distance metric.

Each handwriting sample is represented by a normalized feature vector (FV), typically a histogram-based descriptor extracted using a predefined set of features (e.g., texture, shape). The system measures the degree of dissimilarity between two samples by applying various distance functions such as Chi-square (χ^2), Manhattan, Euclidean, Minkowski. In case multiple features are used, different fusion rules (defined previously) (e.g., sum, max, min, product) are applied to combine their individual distance scores into a single dissimilarity value.

The evaluation is carried out in four successive steps, where the threshold θ is iteratively refined to find the value that minimizes the absolute difference between False Acceptance Rate (FAR) and False Rejection Rate (FRR). The EER is the point at which FAR and FRR are equal:

$$\text{EER} = \frac{\text{FAR}(\theta^*) + \text{FRR}(\theta^*)}{2}$$

where θ^* is the threshold yielding the closest match between FAR and FRR.

During verification, each test document Q_j is compared to a set of reference documents $R_i \in TR$, and the corresponding distances $D(Q_j, R_i)$ are calculated. If the minimum dissimilarity between Q_j and the samples in TR is below the threshold θ, the sample is accepted as genuine; otherwise, it is rejected. Conversely, an imposter attempt is falsely accepted if the distance is incorrectly below the threshold.

The overall process is summarized as follows:

- **Step 1**: Broad θ sweep between a defined min and max, with a coarse increment.

- **Step 2–4**: Gradual refinement around the best θ found previously by reducing the increment at each step.
- For each θ, FAR, FRR are computed.
- The θ giving the smallest $|\text{FAR} - \text{FRR}|$ is selected, and the corresponding EER is reported.

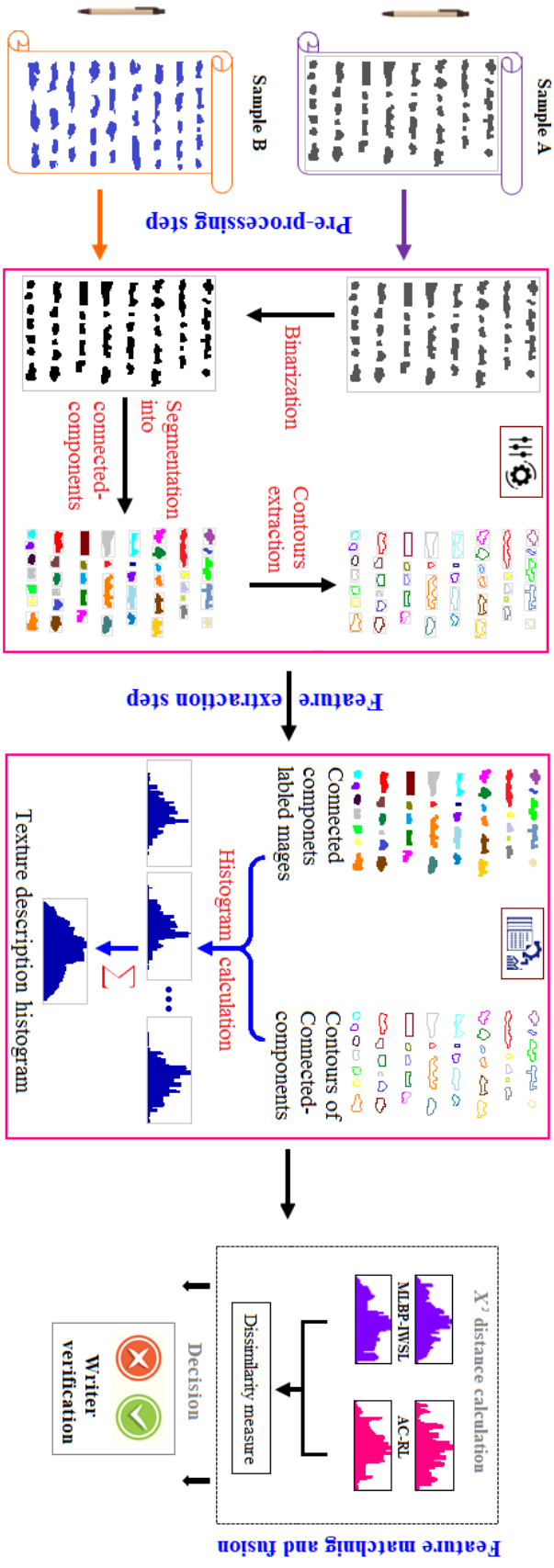

Figure 3.1: The framework of our writer verification system.

Chapter 4

Results Analysis

> If the brain were so simple we could understand it, we would be so simple we couldn't.
>
> Lyall Watson

4.1 Introduction

This section presents a comprehensive analysis of the writer verification experiments conducted on the IAM and KHAT datasets using two feature extraction methods MLBP-IWSL and AG-RL. Each feature was evaluated individually and in combination (fusion) using three distance metrics (Euclidean, Chi-squared, Manhattan) and four fusion rules (SUM, PRODUCT, MAX, MIN). The ROC curves below illustrate the performance for the Chi-squared distance with the SUM fusion rule, which yielded the most promising results.

4.2 Databases

We tested the performance of our proposed technique using two publicly available datasets, Arabic KHATT [47], English IAM [39].

Each featuring a different and challenging language: Arabic and English. We chose these datasets to make sure our method works well across diverse languages and their unique challenges. This approach gave us a thorough evaluation of how well the technique handles different language structures and styles

4.2.1 KHATT database

The KHATT (*King Saud University Handwritten Arabic Text*) database [47] is a large-scale, publicly available dataset developed for research in off-line Arabic handwriting analysis[9]. It consists of a total of 4,000 handwritten document images collected from 1,000 writers (both male and female) originating from 26 Arabic countries. Each participant contributed four paragraphs: two fixed text paragraphs and two free text paragraphs. This structure allows for both content-dependent and content-independent experiments in writer identification and verification.

The documents were scanned at 200, 300, and 600 dpi and stored in grayscale format, preserving rich handwriting details for feature extraction and recognition. This variation in resolution enables researchers to test algorithm performance under different image qualities. The wide geographical diversity of the writers also introduces a rich variety of handwriting styles and regional influences[47].

The KHATT database is commonly used in several research areas, including:

- Writer Identification
- Writer Verification
- Arabic Handwriting Recognition
- Line and Word Segmentation
- Style and Script Classification

A summary of the KHATT database features is presented in Table 4.2.1.

Feature	**Description**
Writers (Scribes)	1,000 (from 26 Arabic countries)
Total Samples	4,000 handwritten pages
Writing per Writer	4 paragraphs (2 fixed + 2 free text)
Scanning Resolutions	200, 300, and 600 dpi
Image Format	Grayscale
Language and Script	Modern Standard Arabic
Application Domains	Writer ID, Verification, Recognition, Segmentation

Table 4.1: Summary of the KHATT Database

Figure 4.1: Page Exemple From KHAT Database

4.2.2 IAM databse

The IAM (*Institute of Automation and Mathematics*) Handwriting Database [39] is a large-scale, publicly available dataset developed for research in offline English handwriting analysis[9]. It has become a standard benchmark for evaluating techniques in handwriting recognition, writer identification, and verification [20].

The dataset contains a total of 1,539 scanned forms written by 657 different writers. The number of forms per writer varies, ranging from 1 to 59, with an average of approximately 2.3 forms per writer. The text content is derived from the Lancaster-Oslo/Bergen (LOB) corpus, ensuring linguistic diversity. Each form contains between 2 and 13 lines of English handwritten text, resulting in over 13,000 text lines and more than 115,000 isolated words across the dataset. This diversity supports both small- and large-scale experiments in handwriting analysis.

All document images are scanned at 300 dpi and stored in 8-bit grayscale format. The IAM database is hierarchically structured, providing annotations at multiple levels: forms, lines, words, and characters. This structure facilitates research across a wide range of tasks, including segmentation, recognition, and stylistic analysis[39].

The IAM database is widely used in the following research areas:

- Handwriting Recognition (at word, line, and paragraph levels)
- Writer Identification
- Writer Verification
- Text Line and Word Segmentation
- Style and Script Analysis

A summary of the IAM database features is presented in Table 4.2.

Table 4.2: Summary of the IAM Database

Feature	**Description**
Writers	657 individuals
Total Samples	1,539 scanned handwritten forms
Writing per Writer	Between 1 and 59 forms
Text Source	Sentences from the LOB (Lancaster-Oslo/Bergen) Corpus
Text Length per Form	2 to 13 lines
Scanning Resolution	300 dpi
Image Format	8-bit Grayscale
Language and Script	English, Latin Script
Application Domains	Writer ID, Verification, Recognition, Segmentation

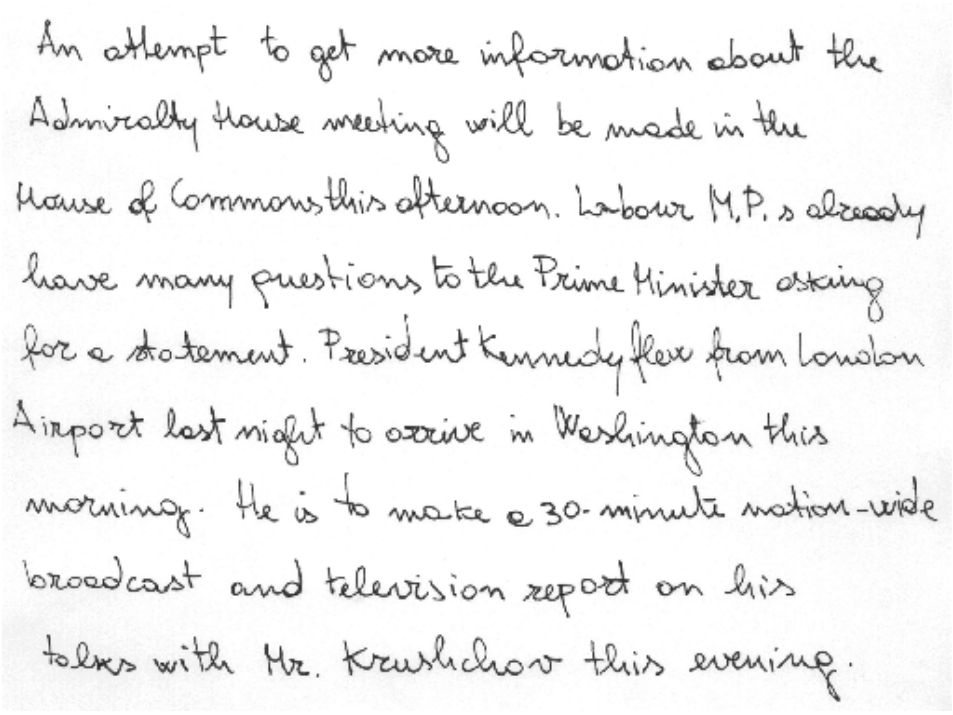

Figure 4.2: Page Exemple From IAM Database

4.3 Interpretation of Writer Verification Results

4.3.1 Individual Features

4.3.1.1 MLBP-IWSL

- **Performance Across Datasets:** MLBP-IWSL demonstrates robust performance on both datasets. On KHAT, it achieves an accuracy of 97.69% (θ = 0.054375) using the Khi-deux distance, with an EER of 2.31%. On IAM, the best result is 98.82% accuracy (θ = 0.0765) with an EER of 1.18% using the same distance metric. These results confirm that MLBP-IWSL is highly effective for writer verification, particularly on KHAT, where the feature seems to capture more distinctive handwriting patterns.

- **Distance Metric Impact:** The Khi-deux distance consistently delivers the best results for MLBP-IWSL, with Manhattan distance closely following (accuracy 97.89%, EER 2.11% on KHAT; accuracy 98.79%, EER 1.21% on IAM). Euclidean distance lags behind, especially on KHAT (accuracy 96.19%, EER 3.81%), indicating that metrics emphasizing distributional differences are more suitable for this feature.

4.3.1.2 AG-RL

- **Performance Across Datasets:** AG-RL underperforms compared to MLBP-IWSL. On KHAT, its best accuracy is 96.27% (θ = 0.0315) with an EER of 3.73% using the Khi-deux distance. On IAM, the highest accuracy is 97.53% (θ = 0.03925) with an EER of 2.47%, also with Khi-deux.

- **Distance Metric Impact:** Manhattan distance provides comparable results to Khi-deux (accuracy 95.31%, EER 4.69% on KHAT; accuracy 97.22%, EER 2.78% on IAM). Euclidean distance is consistently the weakest (accuracy 92.03%, EER 7.97% on KHAT; accuracy 94.95%, EER 5.05% on IAM).

4.3.2 Fusion Strategies

4.3.2.1 Sum Fusion

- **Performance Across Datasets:** Sum fusion yields the highest performance among all strategies. On KHAT, it achieves an accuracy of 98.27% ($\theta = 0.0435$) with an EER of 1.73% using Khi-deux distance. On IAM, the accuracy rises to 98.8% ($\theta = 0.061$) with an EER of 1.20%. This demonstrates that combining AG-RL and MLBP-IWSL features via sum fusion effectively leverages their complementary strengths, especially for challenging datasets like IAM.

- **Distance Metric Impact:** Khi-deux distance remains the most effective for sum fusion, followed by Manhattan (accuracy 98.17%, EER 1.83% on KHAT; accuracy 98.77%, EER 1.23% on IAM). Euclidean distance again trails (accuracy 96.43%, EER 4.38% on KHAT; accuracy 97.56%, EER 2.44% on IAM), reinforcing the importance of selecting a metric that captures distributional nuances.

4.3.2.2 Product Fusion

- **Performance Across Datasets:** Product fusion also performs well. On KHAT, it achieves 98.22% accuracy ($\theta = 0.00175$) with an EER of 1.78%. On IAM, the accuracy is 98.72% ($\theta = 0.00325$) with an EER of 1.28%. While slightly less robust than sum fusion, it still significantly outperforms individual features.

- **Distance Metric Impact:** Khi-deux distance remains the most effective for product fusion, achieving the highest accuracy. Manhattan distance follows (97.99% accuracy, EER 2.01% on KHAT; 98.75% accuracy, EER 1.25% on IAM), while Euclidean distance consistently lags behind (96.73% accuracy, EER 3.27% on KHAT; 97.83% accuracy, EER 2.17% on IAM). This pattern reinforces that product fusion benefits most from distance metrics that emphasize distributional patterns, with Khi-deux and Manhattan providing superior discrimination compared to Euclidean.

4.3.2.3 Max Fusion

- **Performance Across Datasets:** Max fusion achieves an accuracy of 97.89% ($\theta = 0.054$) and EER of 2.11% on KHAT (Khi-deux), and 98.91% accuracy ($\theta = 0.28525$) with an EER of 1.09% on IAM. This shows it is effective at capturing the strongest evidence from either feature but is marginally less robust than sum or product fusion.

- **Distance Metric Impact:** For max fusion, Khi-deux distance again delivers the best results (97.89% accuracy, EER 2.11% on KHAT; 98.78% accuracy, EER 1.22% on IAM), with Manhattan distance performing comparably (98% accuracy, EER 2% on KHAT; 98.91% accuracy, EER 1.09% on IAM). Euclidean distance

is the least effective (92.12% accuracy, EER 7.88% on KHAT; 94.96% accuracy, EER 5.04% on IAM), confirming that max fusion's effectiveness is maximized with distance metrics tailored to capture feature distribution differences.

4.3.2.4 Min Fusion

- **Performance Across Datasets:** Min fusion underperforms all other strategies, with 96.27% accuracy ($\theta = 0.0315$) and EER of 3.73% on KHAT (Khi-deux), and 97.53% accuracy ($\theta = 0.03925$) with EER of 2.47% on IAM. This suggests that focusing on the weakest feature is overly conservative and not optimal for writer verification.

- **Distance Metric Impact:** Khi-deux and Manhattan distances provide similar, but relatively lower, performance for min fusion (Manhattan: 96.09% accuracy, EER 3.91% on KHAT; 97.42% accuracy, EER 2.58% on IAM). Euclidean distance trails further (96.19% accuracy, EER 3.81% on KHAT; 98.41% accuracy, EER 1.59% on IAM), illustrating that min fusion is generally less robust, but still benefits from metrics that highlight pattern differences over simple magnitude.

4.3.3 Comparaison with state of the art

Several notable systems have been proposed for offline writer verification using different types of features. Bulacu et al. (2007) employed contour-based descriptors such as direction histograms, hinge distributions, and run-length features, and achieved an EER of 2.80% on Arabic handwriting. Siddiqi and Vincent (2010) extracted visual attributes like orientation and curvature, considering global, local, and polygonal classes, reporting an EER of 2.46% on English handwriting. In a texture-based approach, Hanusiak et al. (2011) used GLCM (Haralick) texture features and reached a higher EER of 3.90% on Portuguese samples.

In comparison, our proposed method based on MLBP-IWSL features combined with AG-RL significantly improves verification performance. We obtained an EER of 1.20% on the KHATT (Arabic) dataset and 1.73% on the IAM (English) dataset, clearly outperforming the existing systems. These results confirm the effectiveness of our feature design and fusion strategy in capturing discriminative handwriting characteristics across different scripts.

Table 4.3: Comparison with State-of-the-Art Writer Verification Methods

Method	**Features Used**	**Dataset**	**EER (%)**
Bulacu et al. (2007) [21]	Contour PDFs (Direction, Hinge, RL)	Arabic	2.80
Siddiqi and Vincent (2010) [52]	Orientation, Curvature (Global, Local, Polygon)	English	2.46
Hanusiak et al. (2011) [32]	GLCM Texture Descriptors (Haralick)	Portuguese	3.90
Ours	MLBP-IWSL + AG-RL	Arabic	**1.20**
Ours	MLBP-IWSL + AG-RL	English	**1.73**

4.4 Conclusion

The ROC curves for both the IAM and KHAT datasets clearly demonstrate the superiority of the fusion approach over individual features. The fusion curve (green) consistently lies below those of AG-RL (blue) and MLBP-IWSL (red), indicating lower error rates at all operating points. This improvement is especially pronounced at low false positive rates, which are critical for reliable writer verification systems. MLBP-IWSL outperforms AG-RL as an individual feature, but both are surpassed by the fusion strategy, confirming that combining features leverages their complementary strengths. These results, supported by the quantitative metrics in the tables, highlight that feature fusion particularly when paired with an effective distance metric like Khi-deux significantly enhances verification accuracy and robustness across both English and Arabic handwriting datasets.

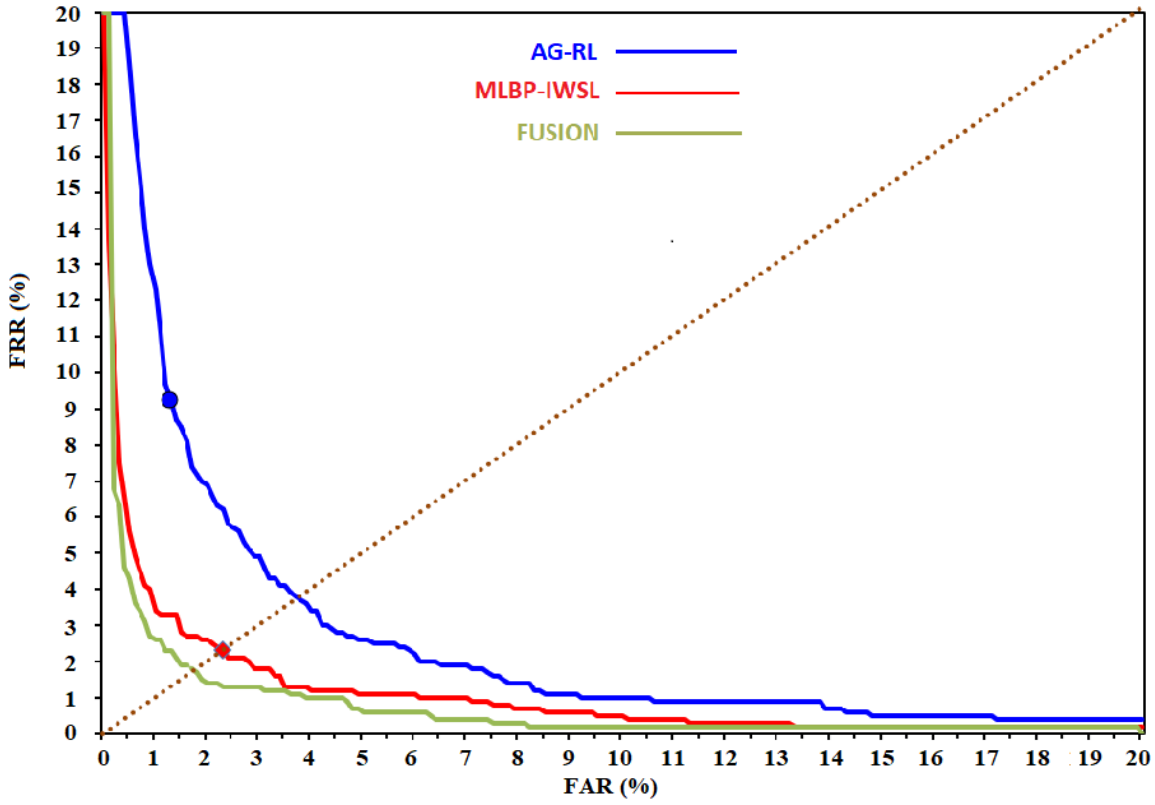

Figure 4.3: ROC Curve for KHAT Dataset

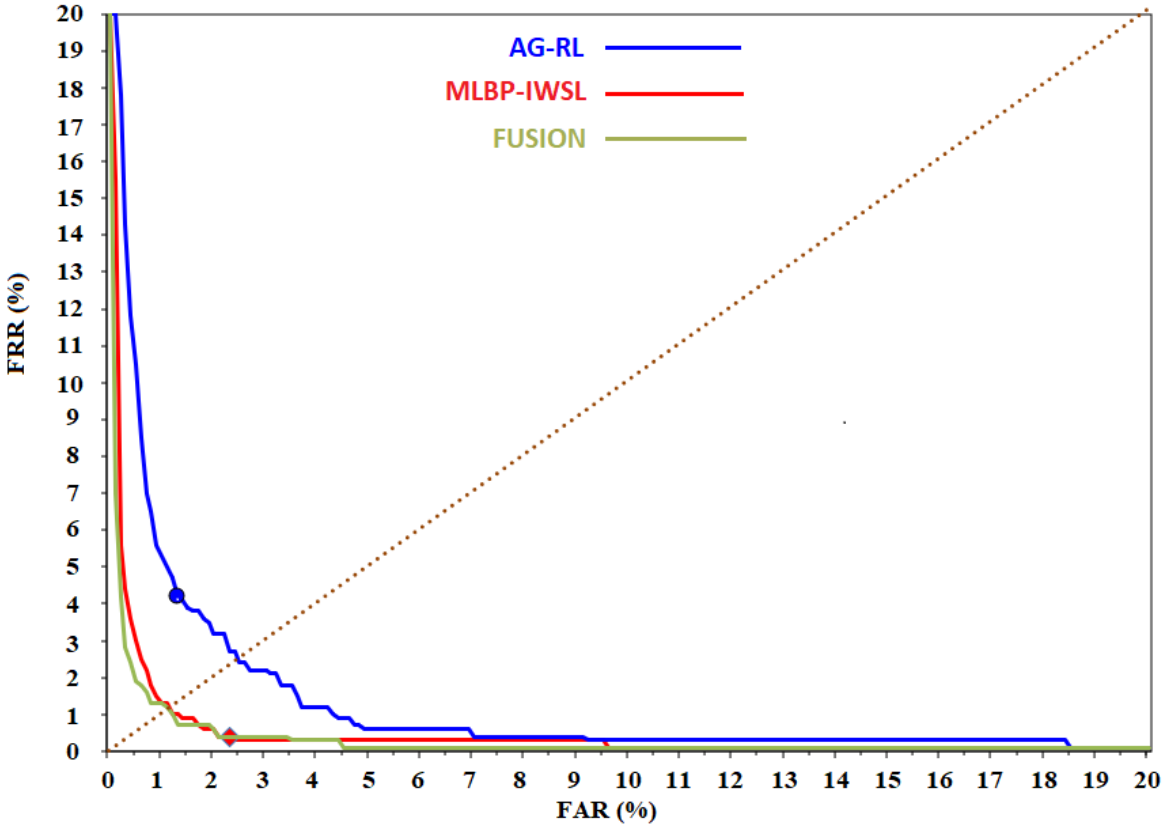

Figure 4.4: ROC Curve for IAM Dataset

Table 4.4: Performance of Individual Features and Fusion Strategies on KHAT and IAM Datasets

Dataset	**Feature/Fusion**	**Distance**	**Accuracy (%)**	$\boldsymbol{\theta}$	**EER (%)**
KHAT	MLBP-IWSL	Khi-deux	97.69	0.05375	2.31
		Manhattan	97.89	0.22675	2.11
		Euclidean	96.19	0.0145	3.81
	AG-RL	Khi-deux	96.27	0.0315	3.73
		Manhattan	95.31	0.18575	4.69
		Euclidean	92.03	0.0325	7.97
	Sum Fusion	Khi-deux	98.27	0.0435	1.73
		Manhattan	98.17	0.2095	1.83
		Euclidean	96.43	0.024	4.38
	Product Fusion	Khi-deux	98.22	0.00175	1.78
		Manhattan	97.99	0.04325	2.01
		Euclidean	96.73	0.00049	3.27
	Max Fusion	Khi-deux	97.89	0.054	2.11
		Manhattan	98	0.232	2
		Euclidean	92.12	0.0325	7.88
	Min Fusion	Khi-deux	96.27	0.0315	3.73
		Manhattan	96.09	0.18175	3.91
		Euclidean	96.19	0.0145	3.81
IAM	MLBP-IWSL	Khi-deux	98.82	0.0765	1.18
		Manhattan	98.79	0.282	1.21
		Euclidean	98.41	0.016	1.59
	AG-RL	Khi-deux	97.53	0.03925	2.47
		Manhattan	97.22	0.2115	2.78
		Euclidean	94.95	0.038	5.05
	Sum Fusion	Khi-deux	98.8	0.061	1.20
		Manhattan	98.77	0.25325	1.23
		Euclidean	97.56	0.02725	2.44
	Product Fusion	Khi-deux	98.72	0.00325	1.28
		Manhattan	98.75	0.0615	1.25
		Euclidean	97.83	0.00049	2.17
	Max Fusion	Khi-deux	98.78	0.0785	1.22
		Manhattan	98.91	0.28525	1.09
		Euclidean	94.96	0.03825	5.04
	Min Fusion	Khi-deux	97.53	0.03925	2.47
		Manhattan	97.42	0.2085	2.58
		Euclidean	98.41	0.016	1.59

Table 4.5: Performance of Individual Features and Fusion Strategies on KHAT and IAM Datasets

Chapter 5

Conclusions and Perspectives

Every accomplishment starts with the decision to try.

John F. Kennedy

This research has presented the development of an effective and scalable offline writer verification system designed to operate across multilingual handwritten documents, with a specific focus on Arabic and English scripts. By adopting a text-independent verification approach, the study successfully addressed key challenges commonly encountered in handwriting biometrics namely, the intra-writer variability (the natural fluctuations in an individual's handwriting over time) and inter-writer similarity (visual resemblance between different writers styles), especially in multilingual scenarios.

The proposed method is based in the use of handcrafted features, notably:

- **MLBP-IWSL (Modified Local Binary Pattern with Ink-trace Width and Shape of Letters)**
- **AG-RL (Angel with Run-Length)**

These features were evaluated both individually and in fusion, in combination with multiple distance metrics. The experimental analysis on the KHATT (Arabic) and IAM (English) datasets revealed several key findings:

- Fusion strategies, particularly the Sum rule, consistently outperformed individual features, highlighting the complementary nature of the descriptors.
- Among the tested distance measures, the Chi-square distance demonstrated superior performance compared to Euclidean and Manhattan distances.
- The system maintained high generalization capabilities across both datasets, underscoring its robustness in handling different scripts and writing styles.

This work contributes meaningfully to the field of behavioral biometrics by offering a viable solution for offline writer verification, independent of language and textual content. The outcomes have practical relevance in domains such as forensic document analysis, identity verification in secure systems, and historical manuscript authentication, where accurate writer attribution is essential.

While the proposed system has shown encouraging results, several opportunities exist for further enhancement and expansion:

- **Deep Learning Integration:** Future work could explore the use of Convolutional Neural Networks (CNNs) or attention mechanisms to enable automatic learning of feature representations. This could significantly improve the model's accuracy and adaptability, particularly on larger and more diverse handwriting datasets.
- **Smarter Feature Fusion:** Instead of static fusion strategies, adaptive or learnable fusion methods, such as weighted fusion using machine learning, could be employed to better exploit the relationships between different features and improve decision making.
- **Script and Language Expansion:** Extending the system to support additional scripts (e.g., Chinese, or Cyrillic) would enhance its multilingual generalizability and open up broader applications in global contexts.
- **Real-Time and Mobile Deployment:** Optimizing the system for real-time performance and deployment on mobile or embedded devices could make it suitable for practical applications, such as on real-time identity verification, tablet-based document signing, or security systems requiring fast and lightweight writer verification.

These directions open the door to continued progress toward building more intelligent, robust, and universally applicable writer verification systems.

Bibliography

[1] A contour-based feature extraction algorithm. *Open Access Journal of Artificial Intelligence and Machine Learning*, 2023.

[2] Contour-based diacritics detection for enhanced arabic handwritten text recognition. *Journal of Theoretical and Applied Information Technology*, 2024.

[3] M.N. Abdi and M. Khemakhem. A model-based approach to offline text-independent arabic writer identification and verification. *Pattern Recognition*, 48(5):1890–1903, 2015.

[4] B. Q. Ahmed, Y. F. Hassan, and A. S. Elsayed. Offline text-independent writer identification using a codebook with structural features. *PLoS ONE*, 18(4):e0284680, 2023.

[5] Anonymous. A survey on different levels of fusion in multimodal biometrics. *Indian Journal of Science and Technology*, 2019.

[6] B. Arazi. Handwriting identification by means of run-length measurements. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(12):878–881, 1977.

[7] Author(s). Improving codebook-based writer recognition. *World Scientific Publishing*, 2013.

[8] Author(s). Influence of codebook patterns on writer recognition. *Expert Systems*, 2020.

[9] T. Bahram. A texture-based approach for offline writer identification. *Journal of King Saud University - Computer and Information Sciences*, 34(8, Part A):5204–5222, 2022.

[10] T. Bahram and R. Adjoudj. Offline text-independent writer identification using local black pattern histograms. In M. Salem, J.J. Merelo, P. Siarry, R. Bachir Bouiadjra, M. Debakla, and F. Debbat, editors, *Artificial Intelligence: Theories and Applications. ICAITA 2022*, pages 241–254, Mascara, Algeria, 2023. Springer.

[11] T. Bahram, A. Benyettou, and D. Ziadi. A set of features for text-independent writer identification. *International Review on Computers and Software (I.RE.CO.S.)*, 56(2):898–906, 2016.

[12] Tayeb Bahram. Offline writer identification using texture features of connected-components. Correspondence: tayeb.bahram@univ-saida.dz.

[13] Tayeb Bahram. Cours 2 - systèmes biométriques, 2024.

[14] B. S. Baumer, D. T. Kaplan, and N. J. Horton. *Modern Data Science with R*. CRC Press, Boca Raton, FL, 2021.

[15] A. Bennour, C. Djeddi, A. Gattal, I. Siddiqi, and T. Mekhaznia. Handwriting based writer recognition using implicit shape codebook. *Forensic Science International*, 301:91–100, 2019.

[16] A. Bennour, C. Djeddi, and T. Mekhaznia. Using codebooks of fragmented connected-component contours in forensic and historic writer identification. *Pattern Recognition Letters*, 27(12):1443–1453, 2006.

[17] A. Bensefia, T. Paquet, and L. Heutte. A writer identification and verification system. *Pattern Recognition Letters*, 26(13):2080–2092, 2005.

[18] D. Bertolini, L.S. Oliveira, E. Justino, and R. Sabourin. Texture-based descriptors for writer identification and verification. *Expert Systems with Applications*, 40(6):2069–2080, 2013.

[19] Diego Bertolini, Luiz S. Oliveira, Edson Justino, and Robert Sabourin. Texture-based descriptors for writer identification and verification. *Expert Systems with Applications*, 40(6):2069–2080, 2013.

[20] M. Bulacu and L. Schomaker. Text-independent writer identification and verification using textural and allographic features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):701–717, 2007.

[21] M. Bulacu and L. Schomaker. Text-independent writer identification and verification using textural and allographic features. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 769–773, 2007.

[22] Dai Dehui and Li Zhiyong. A fast connected components analysis algorithm for object extraction. In Fatos Xhafa, Srikanta Patnaik, and Albert Y. Zomaya, editors, *Advances in Intelligent Systems and Interactive Applications*, pages 353–360, Cham, 2018. Springer International Publishing.

[23] C. Djeddi, M. Kherallah, N. Ben Amara, and A. Belaid. Arabic writer identification: A review of literature. *Journal of Theoretical and Applied Information Technology*, 69(3):474–484, 2014.

[24] C. Djeddi and L. Meslati. A texture based approach for arabic writer identification and verification. In *2010 International Conference on Machine and Web Intelligence (ICMWI)*, pages 115–120, Algiers, Algeria, 2010. IEEE.

[25] C. Djeddi, I. Siddiqi, L. Meslati, and A. Ennaji. Text-independent writer recognition using multi-script handwritten texts. *Pattern Recognition Letters*, 34(10):1196–1202, 2013.

[26] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley-Interscience, New York, NY, USA, 2nd edition, 2000.

[27] M. Faundez-Zanuy. Data fusion in biometrics. *IEEE Aerospace and Electronic Systems Magazine*, 20(1):34–38, 2005.

[28] Abeer George Ghuneim. Contour tracing algorithms. https://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/mmain.html#alg, n.d. Accessed: 2025-06-03.

[29] Y. Hannad, I. Siddiqi, and M.E. El-Kettani. Writer identification using texture descriptors of handwritten fragmen. *Expert Systems with Applications*, 47:14–22, 2016.

[30] Yaâcoub Hannad, Imran Siddiqi, Chawki Djeddi, and Mohamed El-Youssfi El-Kettani. Improving arabic writer identification using score-level fusion of textural descriptors. *IET Biometrics*, 8(1):3, 2019.

[31] R. K. Hanusiak, L. S. Oliveira, E. Justino, and R. Sabourin. Writer verification using texture-based features. *International Journal on Document Analysis and Recognition (IJDAR)*, 14(3):215–230, 2011.

[32] R. K. Hanusiak, Luiz S. Oliveira, Edson Justino, and Robert Sabourin. Writer verification using texture-based features. *International Journal on Document Analysis and Recognition (IJDAR)*, 14(3):215–230, 2011.

[33] S. He and L. Schomaker. Writer identification using curvature-free features. *Pattern Recognition*, 63:451–464, 2017.

[34] Inter-university Consortium for Political and Social Research (ICPSR). *Codebook Preparation Manual*. University of Michigan, 1983. Available at: https://www.icpsr.umich.edu/web/pages/ICPSR/guide.html.

[35] E. Khalifa, S. Al-Maadeed, M. A. Tahir, et al. Off-line writer identification using an ensemble of grapheme codebook features. *Pattern Recognition Letters*, 59:18–25, 2015.

[36] A. Kumar, S. S. S. R. Depuru, et al. Handwriting identification and verification using artificial intelligence and texture features. *Scientific Reports*, 2023.

[37] D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International Journal of Remote Sensing*, 28(5):823–870, 2007.

[38] Timothy M. What is thresholding in image processing? a guide, July 2024. Accessed: 2025-06-04.

[39] U.-V. Marti and H. Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, 5(1):39–46, 2002.

[40] U.V. Marti and H. Bunke. The IAM-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, 5:39–46, 2002.

[41] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[42] S. Prabhakar and A. K. Jain. Decision-level fusion in fingerprint verification. *Pattern Recognition*, 35(4):861–874, 2002.

[43] S. RahmanGour, C. KarmakerRobert, and J. Bignall. Improving image classification using extended run length features. In D. Huijsmans, D. P. Dionysius, and W. A. Smeulders, editors, *Visual Information and Information Systems*, pages 475–482. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[44] R. Raju and P. Reddy. Writer identification using texture features in kannada handwritten documents. *International Journal of Engineering Research and Technology*, 7(13):1–7, 2018.

[45] A. Ross and R. Govindarajan. Feature level fusion using hand and face biometrics. In *Proceedings of SPIE Conference on Biometric Technology for Human Identification II*, volume 5779, pages 196–204, Orlando, USA, 2005. SPIE.

[46] A. Ross and A. Jain. Information fusion in biometrics. *Pattern Recognition Letters*, 24(13):2115–2125, 2003.

[47] A. Sabri, A. Irfan, G. Wasfi, M. Alshayeb, M. Parvez, M. Märgner, and G. Fink. Writer identification for arabic and latin scripts using codebook features. *Pattern Recognition*, 47(3):1096–1112, 2014.

[48] Krystian Safjan. Metrics used to compare histograms. `https://safjan.com/metrics-to-compare-histograms/`, 2021. Accessed: 2025-06-05.

[49] S. Said et al. Writer identification using texture descriptors of handwritten fragments. *Pattern Recognition Letters*, 2000.

[50] et al. Shaikh. Attention based writer independent handwriting verification. *arXiv:2009.04532*, 2020.

[51] I. Siddiqi and N. Vincent. Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features. *Pattern Recognition*, 43(11):3853–3865, 2010.

[52] Imran Siddiqi and Nicole Vincent. Combining contour based orientation and curvature features for writer recognition. In *Proceedings of CAIP 2009*, 2009.

[53] Substance Abuse and Mental Health Services Administration (SAMHSA). *Codebook Development and Usage*, 2024. Available at: `https://www.samhsa.gov/data/codebook-development`.

[54] Y.-F. Yao, X.-Y. Jing, and H.-S. Wong. Face and palmprint feature level fusion for single sample biometrics recognition. *Neurocomputing*, 70(7):1582–1586, 2007.

[55] D. Zhang, F. Song, Y. Xu, et al. Decision level fusion. In *Advanced Pattern Recognition Technologies with Applications to Biometrics*, pages 328–348. IGI Global, USA, 2009.

ملخص

تقدم هذه الرسالة تطوير نظام تحقق تلقائي من الكاتب باستخدام عينات الخط اليدوي غير المتصلة بالإنترنت .تتناول الحاجة المتزايدة للتحقق الآمن من الهوية من خلال استخدام الخط اليدوي كخاصية بيومترية سلوكية .يقترح البحث تقنيات مبتكرة لاستخراج الميزات ودمجها لتحسين دقة التحقق .أظهرت التجارب على مجموعات بيانات معيارية مثل IAMو KHATTفعالية النظام .تسهم النتائج في تطوير طرق التوثيق البيومتري مع تطبيقات في الأمن والطب الشرعي.

الكلمات المفتاحية: التحقق من الهوية، القياسات الحيوية السلوكية، الكتابة اليدوية غير المتصلة، استخراج السمات، دمج السمات، التعرف على الكاتب، الأمان، المصادقة البيومترية، الطب الشرعي.

Abstract

This thesis presents the development of an automatic writer verification system using offline handwriting samples. It addresses the growing need for secure identity verification by leveraging handwriting as a behavioral biometric. The proposed approach introduces innovative feature extraction and fusion techniques to improve verification accuracy. Experiments on standard datasets such as IAM and KHATT demonstrate the effectiveness of the system. The results contribute to advancing biometric authentication methods with applications in security and forensics.

Keywords: Identity verification ,Behavioral biometrics ,Offline handwriting ,Feature extraction ,Feature fusion ,Writer recognition ,Security ,Biometric authentication , Forensics.

Résumé

Cette thèse présente le développement d'un système automatique de vérification d'écrivain à partir d'échantillons d'écriture manuscrite hors ligne. Elle répond au besoin croissant de vérification sécurisée de l'identité en exploitant l'écriture manuscrite comme biométrie comportementale. L'approche proposée introduit des techniques innovantes d'extraction et de fusion des caractéristiques pour améliorer la précision de la vérification. Les expérimentations sur des bases de données standard telles que IAM et KHATT démontrent l'efficacité du système. Les résultats contribuent à l'avancement des méthodes d'authentification biométrique avec des applications en sécurité et en médecine légale.

Mot clés: Vérification d'identité ,Biométrie comportementale ,Écriture manuscrite hors ligne ,Extraction de caractéristiques ,Fusion de caractéristiques ,Reconnaissance de l'écriture ,Sécurité ,Authentification biométrique ,Forensique.