THE UNIVERSITY OF SOUTH ALABAMA

School of Computing

# Multi-Script Handwriting Identification using Stroke Decomposition

BY

Joshua Jude Thomas

A Thesis Proposal

Submitted to the Graduate Faculty of the

University of South Alabama

in partial fulfillment of the

requirements for the degree of

Master of Science in Computer Science

in

Electrical and Computer Engineering

May 2024

Approved: Date:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Chair of Thesis Committee: Dr. First Name, Middle Initial, Last Name

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Committee Member: Dr. First Name, Middle Initial, Last Name

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Committee Member: Dr. First Name, Middle Initial, Last Name

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Chair of Department: Dr. First Name, Middle Initial, Last Name

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Director of Graduate Studies: Dr. First Name, Middle Initial, Last Name

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Dean of the Graduate School: Dr. J. Harold Pardue

# Multi-Script Handwriting Identification using Stroke Decomposition

A Thesis

Submitted to the Graduate Faculty of the

University of South Alabama

in partial fulfillment of the

requirements for the degree of

Master of Science

in

Computer Science

by

Joshua Jude Thomas

B.S., University of South Alabama, 2021

A.S., Coastal Alabama Community College, 2017

April 2023

**LIST OF ABBREVIATIONS**

SVM Support Vector Machine

LLR Log Likelihood Ratio

CNN Convolutional Neural Network

HIM Handwriting Identification Model

HVM Handwriting Verification Model

SD Stroke Decomposition

# ABSTRACT

Joshua Jude Thomas, M. S., University of South Alabama, April 2023. Multi-Script Handwriting Identification using Handwritten Stroke Decomposition Chair of Committee: Ryan Benton, Ph.D.

This study will implement and test the effectiveness of adding a preprocessing step, named **Stroke Decomposition**, to a typical Handwriting Identification model. Stroke Decomposition works by approximating the different writing strokes that make up the characters in a handwritten document; The idea being that the characters of different writing scripts (English, Arabic, Mandarin, etc.) have common strokes which can be extracted and used in the handwriting identification process. The problem domain of this study is **Multi-Script Writer Identification** problem, which tries to identify the writer of a questioned document in a setting where a writer can possibly write in multiple writing scripts or languages. The Stroke Decomposition step will be added to two different handwriting identification models (using the **Handwriting Identification Model** (HIM) and the **Handwriting Verification Model** (HVM)) and the performance will be measured on various multi-script and single-script handwriting datasets.

# CHAPTER I

**Introduction**

**Handwriting Identification** is the process of classifying the writer of a handwritten Questioned Document based on handwriting habits contained in that document. Handwriting Identification predates computational processing by a fair bit and is considered fair evidence in civil and criminal court {citation needed}. According Harralson and Miller in *Huber and Headrick’s Handwriting Identification: Facts and Fundamentals*, forensics experts commonly compare twenty one “discriminating elements of handwriting” that deal with properties such as word size, word placement, margin sizes, abbreviation choices, etc. [1].

Computationally assisted Handwriting Identification is typically done by extracting visual features from a set of known documents and questioned documents. Those features are usually aggregated into a **feature vector** and are then used to compare the known documents to the questioned documents, and the writer is predicted by a similarity score or a likelihood [2]. Handcrafted features and Deeply Learned features are two broad categories which handwriting features roughly fall into; Handcrafted features share similarities with the twenty-one features described in Huber and Hendricks, while Deeply Learned features use statistical and machine learning techniques in their extraction.

## 1.1 Verification and Identification Model

There are two main architectures used in computationally assisted Handwriting Identification: the **Handwriting Identification Model** and the **Handwriting Verification Model** [2].

**The Handwriting Verification Model** estimates the likelihood that two documents were written by the same person (two-class classification) [2]. The extracted feature vectors of a document are passed to a decision function that can either directly classify the writer such as with a Support Vector Machine (SVM) [3], or output a similarity/difference score such as the Log Likelihood Ratio (LLR) [4].

Graphical user interface

Description automatically generated with medium confidence

Figure 1. The Handwriting Verification Model

The **Handwriting Identification Model**, on the other hand, matches a Questioned Document to a database of known writers (n-class classification). Here the extracted features of a document are typically a multi-class classifier such as Nearest Neighbor [5], [6]. The Writer Identification Model seems to be the more popular model overall.

Diagram

Description automatically generated with medium confidence

Figure 2. The Handwriting Identification Model

## Multi-Script Writer Identification

The Multi-Script Writer Identification problem is like normal Handwriting Identification, except that a writer is not limited to a single writing script. This means that they can possibly write in multiple scripts such as English, Arabic, and Mandarin, or even write in different languages that share a writing script [7]. The ICFHR 2018 Competition on Multi-Script Writer Identification [7] is a popular reference in papers on Multi-Script Writer Identification [6]. The competition paper uses the CERUG [8], LAMIS-MSHD [9], and the WDAD [7] datasets: which contain Chinese/English, Arabic/French, and Farsi (Persian)/English Multi-Script datasets, respectively. There are six tasks introduced by the competition, each of which uses one of the datasets (above). Each task has the Writer Identification system train on one language in the dataset, and test on the other. The goal of Multi-Script Writer Identification, according to the ICFHR 2018 competition paper, is to find “… writing patterns that are common across different scripts [and] may be exploited to identify the writer”. This problem is based on the assumption that there are ingrained patterns in a person’s handwriting that are stable across different writing scripts [7].

## Stroke Decomposition Algorithm

The purpose of the **Stroke Decomposition** (SD) Preprocessing algorithm, which will be developed in this study, is to improve the performance of Multi-Script Writer Identification. The idea of this study is that the SD preprocessing step will transform the handwriting characters into simpler sub-strokes that will be more comparable. Figure 3 demonstrates the importance of this idea; The characters of the four different writing scripts can look very different from one another and may confuse some of the feature extraction algorithms due to the overall visual differences in the characters.

A picture containing text, document

Description automatically generated

Figure 3. (From left to right) Chinese, Bengali, Tamil, and English. Writing Scripts of three different origins that look very different on a document-level scale.

The SD algorithm is based on Morphological Processing and Graph Theory. With Morphological Processing, a binary image is processed via spatial filtering with a Structuring Element SE to identify shapes an patterns in the image. Uses include: preprocessing and post processing on the image, finding the boundary or convex hull of an image, and so on [10]. Morphological Processing is used to remove noise and unnecessary components of document images as well as to preprocess the document images for graph theory processing. Graph Theory is the study of graph data structures, which can be described as a set of observations (vertices) and their relation to one another (edges). Here a processed document image, binarized and thinned by morphology, is converted into an Attributed Relational Graph: in which the vertices are based on critical and branching points, and the edges are based on connections between those points. Inside the edge relations are the pixel coordinates to be extracted.

# CHAPTER II

**Literature Review**

## Handwriting Identification

The ICFHR 2018 Multi-Script Handwriting Identification competition is a popular reference point for more recent literature. However, the competition paper itself reports on the successes of four different systems submitted to the competition. These systems are the LIMPAF-I, LIMPAF-II, Tokyo System, and the Nuremberg System [7]. The LIMPAF-I and LIMPAF-II were submitted by the same group; LIMPAF-I uses Uniform Complete Local Binary Patterns (U-LBP) [11] for its feature extraction while the LIMPAF-II uses Oriented Basic Image Features (oBIF) [12]. For classification, both systems used a multi-class SVM. The Tokyo system used two CNNs to extract features from randomly selected sub-images of a writing sample. Features extracted from writing samples were passed into a “Transfer Neural Net” to transform the extracted features, of writing samples of possibly different writing scripts, into a more uniform representation. These transformed features were then finally classified by K-Nearest Neighbors [13]. The Nuremberg system was actually based on another work ([14]) which extracted features by a pre-trained CNN. The extracted features were then “PCA-Whitened” and encoded in a visual bag of words algorithm called VLAD. The feature vectors of the Nuremberg systems were classified by measuring the Cosine-Distance between each sample.

Abbas *et al* combines both LBP and oBIF, creating a histogram out of the LBP and oBIF descriptors over the whole range of the document and then classifying the writer with an multi-class SVM [15]. Semma, Hannad, Siddiqui, Lazrak, and Kettani extract features from sub-images obtained from a Harris Corner Detector. The sub-images are then fed into a CNN for feature extraction, and then transformed with VLAD. The VLAD vectors are then used to classify the writer with Nearest Neighbor [16].

In much of the literature, the input into the feature extraction algorithms are random sub-images of the documents (like the Tokyo system in the competition paper) [7], using key point detectors such as the Harris Corner Detector [16] as guides. One study aggregates local features generated from sub images into global features through processes like max-pooling and average-pooling [13]. This study plans to test the effect of separating the characters of a handwritten document into their approximate individual strokes that make them up; With the idea that breaking the characters of a document into sub-strokes will provide more common features for use in Writer Identification.

## Stroke Decomposition

## Stroke Decomposition is a method of reducing a handwritten character into approximate individual strokes. While similar concepts have been applied in Handwriting Identification, This technique typically appears in research on Optical Character Recognition (OCR) which is a similar field that attempts to convert images of text (handwritten or otherwise) into typed text. Kim et al. decompose Chinese characters into individual strokes by first performing a morphological thinning to reduce each character to a single pixel width [17]. They then segment the characters based on branching points (areas where strokes overlap) and excessively curved segments. The segments are grown morphologically using two modifications on a morphological dilation which use vectors both parallel and perpendicular (elongation and fattening) to the direction of each segment; A more standard dilation, named isotropic expansion, is performed on segments that are not long enough for the elongation step (less than five pixels in the paper). Both Fattening and Isotropic expansion are constrained by an approximate convexity measure. Finally, Grown stroke segments that have intersecting parts are then potentially merged using the same convexity measure as a conditional [17]. Chen et al. convert handwritten Chinese characters into stroke sequences (strings of numbers that indicate the type of stroke by number and the order the stroke was written by the position of that number). They use an encoder-decoder architecture to convert the character images to stroke sequences [18].

There is already some literature in sub-stroke decomposition. Kim, Kim, Choi, and Kim decompose Chinese character into their constituent sub-strokes using morphological processing [17]. However, to the best of our knowledge, there is no readily available software package that performs a technique similar to Stroke Decomposition.

# CHAPTER III

**Methodology of Study**

In this study, the Stroke Decomposition method is performed as a separate process from Writer Identification and is performed by constructing a set of filters for each handwritten document. A typical Handwriting Dataset consists of a set of digitized handwritten document images (scanned or photographed). A set of filters is constructed for each, which represent simplified stroke shapes extracted from the related document. Sub-images, containing the simplified stroke shape, are extracted from the document with the bounding box coordinates of the filters, and then unneeded pixel values (not corresponding to the filter pixels) are removed. Finally, the filtered sub-images are saved to disk, and then later loaded, via a data-loader, to be trained on, or predicted, by a modified ResNet50 CNN model. The final evaluation metrics for this study are split into Categorical Accuracy, Precision, and Recall for the overall performance metrics in both the training and testing stages. An additional Top-N Accuracy metric is used to evaluate the performance of the CNN model on individual documents (grouping all simplified strokes by the document they were extracted by) during testing.

## Stroke Decomposition Method

The main idea of stroke decomposition is to split handwritten characters into simpler shapes, which are hopefully more common between different writing scripts. Since there is no information about the order of strokes a writer makes when creating characters during writing, the strokes making up a character are approximated using **critical points**: branching points, and end points a character. The end goal of stroke decomposition is a collection of simplified shapes extracted from each document that could then be used with more conventional handwriting identification methods.

A mathematical graph representation is used to facilitate the finding of the critical points in a document. In this setting, the characters contained in a handwritten document are represented as a disconnected, planar, multi-graph: , where is a set containing all the critical points of the document and is a multi-set of pairs . Each pair in represents that the critical points are connected by a part of a stroke. The simplified shapes we want to extract can then be thought of as subgraphs of the connected-components of the larger document graph. It is of note that the actual pixel coordinates corresponding to the handwritten character in a document are not represented as the edges or nodes but are instead included as attributes in the nodes and edges of the graph. When extracting a simplified stroke shape, the pixel coordinate attributes are taken from the corresponding nodes and edges, and are combined to form the filter of that simplified shape.

General Steps of Stroke Decomposition:

1. Preprocessing: transform the document image into an inverted grayscale image for feature extraction, and an inverted binary image for forming the filters used in the extraction step.
2. Reduce the characters in the binary image to single pixel width through morphological skeletonization.
3. Convert the morphological skeleton into a graph representation to locate the critical points.
4. Apply an integer label to each edge and take the attributed pixel coordinates of the edge and it’s connecting nodes to form a skeleton of the filter
5. Perform region growing between the skeleton of the filter and the binary image to form different filters.
6. Remove any filter whose bounding box is too small or two large in either two dimensions (filtering criterion)
7. Use the filter(s) and its bounding box to extract the corresponding sub-image from the grayscale image, then remove any pixel values for pixels not coinciding with the pixels in the filter

(step 1) When the stroke decomposition process begins, an image is converted into a grayscale representation, and then a binary representation. The grayscale representation of the document is kept for the stroke extraction stage at the end of the stroke decomposition process. A binary representation is obtained via Otsu’s method [19], which is then used to form the filters. When processing the binary image(s), an inherent amount of noise will exist in the images of the handwriting dataset, depending on both the method and quality of digitization, and may influence the graph representation. Possible examples are smudges on the paper, visible paper edges from poor digitization, etc. So a gaussian blur which is applied to the image document before binarization (though the blur is not present when actually extracting the strokes). Later a filtering criterion is used to remove filters whose width or height are too large or too small.

(step 2) The documents, now binarized and filtered, then go through morphological skeletonization. The skeletonization of the binary shapes in the image reduce them to a single pixel width and try to preserve the general shape of the character as much as possible.

(step 3) The skeletonized handwriting components can then be converted into a graph theory representation. The graph representing a handwriting component in a document is defined as such that the vertices of the graph correspond to pixels in the document representing critical points. A critical point is all the endpoints of the strokes in the document (pixels connected to only one other pixel) and the branching points of the document (pixels who are connected to three or more other pixels). Every pixel that does not fit the above criteria is considered an edge pixel; The edges of a graph represent critical points connected by a line of edge pixels. When the graph representation is made, the critical point pixels are stored in the corresponding vertices and the edge pixels are stored in the corresponding edge.

(step 4 and 5) During the construction of the filters, each edge is assigned an integer label . The attributed edge pixels of each edge, along with the attributed critical point pixels of each vertex are assigned said label; The union of the edge pixels and the node pixels form the skeleton of a filter in the document. Afterwards, region growing is performed, via KNN, between the filter skeletons and the binarized image to form different filters. That is, the filter skeletons are used as the training data to fit the KNN model, which is then used to segment the pixels of the binary image into the final set of filters.

(step 6) Some artefacts (noise) may remain in the image after all these steps. Large artefacts such as visible page edges will be viewed as handwriting strokes by the stroke decomposition algorithm will add noise into the output. In addition to large artefacts, some filters may end up being too small to give any meaningful information, and may result in a noise dataset as well. A **filtering criterion** is defined to remove artefacts and small filters from the final output. The bounding box of each filter is found and used for the feature extraction step (step 7). When the bounding box is found the width and height of the filter is taken. If either dimension is smaller than 3 pixels or larger than 80% the width/height of the image document, the filter is rejected and not included in the output.

(step 7) Finally, each filter is used to extract the corresponding pixels in the inverted grayscale image. The bounding box coordinates are used to define the location of the sub-image containing the desired stroke pixels. After the sub-image boundary is obtained any pixel not coinciding with the pixels in the filter are zeroed out.

The result of the Stroke Decomposition process is filtered sub-images, where is the number of found filters for a document and is the number of filtered rejected by the filter criterion. The filtered sub-images are grouped together into a directory of subdirectories. Each subdirectory corresponds to writer class; A writer may have a writer class for each different writing script collected in the dataset (or in some cases a class corresponding to a mix of multiple writing scripts). The directory of subdirectories are then used by a data-loader in the writer identification step to load the filtered sub-images and assign writer labels to them for classification.

## Writer Identification Method

In the Stroke Decomposition stage, the scanned handwritten documents of a Dataset are transformed into sets of filtered sub images containing the approximate strokes making up the characters of that document. The sets of the sub images are then loaded via a TensorFlow Data Loader. Each set of sub images (contained in a folder) are treated as the writer classes of a dataset. The Writer classes represent not only the writers of the document, but the writing script used in the document. The images are padded to standard size {haven’t decided yet. Will probably depend on the dataset} {and potentially augmented to produce more training examples}. The labels of the image classes are inferred from the names of the subdirectories and are then one-hot encoded to produce a vector of integer labels for each class. Training and Test splits will depend on the experiment being performed and will depend on the writing script represented in a writing class for multi-Script datasets.

General Steps of the Writer Identification Part:

1. Training
   1. Load training set via data-loader
   2. Initialize modified ResNet50 CNN Model (transfer learning) and set hyperparameters
   3. Train modified CNN on training set and return metrics
      1. Categorical Accuracy
      2. Precision
      3. Recall
2. Testing
   1. Load test set in such a way that each document sub-image is grouped together
   2. Make model predictions on each group
   3. Aggregate sub-image groups together to get overall metrics (same as training metrics)
   4. Measure Top-N accuracy for each document
      1. Per document Top-N accuracy

### Model and Method of Analysis

A Convolutional Neural Network (CNN) is fit to the data for use in the analysis. The CNN takes in a single handwriting stroke extracted from a handwritten document (stored in an image) from the processed dataset and outputs a SoftMax vector corresponding to a normalized probability score. The probability score represents the chances that a handwriting stroke was produced by one of the writers in the dataset.

The data being analyzed in this study represents approximated strokes in a handwriting document. Classifiers trained on this granularity will, potentially, classify individual strokes of a handwritten document as being produced by one writer or another, meaning that the set of strokes in a single handwritten document may be attributed to more than one writer. An assumption of this study is that there is only one writer per handwritten document for any given document in the dataset, so a combination of fine-grained metrics and an overall metric is needed.

The fine-grained metrics of this study represent the performance of the model at classifying individual strokes of a writer. The metrics selected for the fine-grained analysis are a standard combination of accuracy, precision, recall{, and potentially f1-score}. The fine-grained metrics will be evaluated on the training and test sets of a given dataset’s and set aside for analysis. The outputs of the model, the SoftMax score of each individual stroke in a handwritten document, will be passed on to the overall metrics.

Following the ICFHR 2018 Competition on Multi-Script Writer Identification, a Top-N strategy is used as an overall evaluation protocol [7]. Top-N in the paper is defined as “… the scenario where the genuine writer of a query document is present within the list of N most probable writers received by the system”.

NOTE to Dr. Benton: The overall metric for this study needs more fleshing out. The set of SoftMax scores for a given document image must be aggregated in some way. The two ways I see this happening is either with a vote-based scheme where each SoftMax score is considered as a vote with the most probable writer, or an averaged probability scheme where every SoftMax score is averaged (and then renormalized). The averaging scheme may end up working better depending on the number of strokes classified in a document as it guarantees that each writer has some amount of representation in the document, whereas a voting-scheme will suppress every writer except the most probable writer of a given stroke. The top-N metric as presented in the competition paper is unclear to me, and I may need to diverge from it. If I do diverge, then I still plan to use some Top-N strategy for the evaluation metric.

# CHAPTER IV

## Experiments

### CERUG

The Chinese-English Database of the University of Groningen (CERUG) is a Multi-Script Writer Identification dataset produced in “Junction Detection in Handwritten Documents and its Application to Writer Identification” [8]. The dataset has three different writing script classes per writer: CERUG-CN for Chinese writing in page 1 and 2, CERUG-EN in page 3 (split over two images), and CERUG-MIXED in page 4 which consists of a mix of Chinese and English. There are three possible experiments that can be performed with this dataset in this study. Experiment 1 and 2 will correspond to task 1 and 2 of the ICFHR 2018 Competition on Multi-Script Writer Identification. Experiment 1 will use the CERUG-CN portion of each writer as a training set and use the CERUG-EN portion as a test set. Experiment 2 will be the reverse of experiment 1: using CERUG-EN as the training set and CERUG-CN as the test set.

# CHAPTER VI

**Conclusion**

Multi-Script Handwriting Identification attempts to classify the writer of a handwritten document in a setting where there can be multiple writing scripts or languages in use, and with the possibility that a writer can create documents in more than one writing script. As such, Multi-Script Handwriting Analysis seeks common features between the different writing scripts that are both effective and consistent. Stroke Decomposition may provide a more robust comparison between characters by isolating common sub-strokes found in all, or many, writing scripts. While the use of Stroke Decomposition may result in a small performance penalty due to the loss of information about how the sub-strokes affect each other, it may provide more comparable handwriting samples between questioned documents written in different writing scripts or languages. This will hopefully result in a net positive impact on the performance of Multi-Script Handwriting Identification.

# REFERENCES

[1] H. Harralson and L. Miller, *Huber and Headrick’s Handwriting Identification: Facts and Fundamentals*, 2nd ed. CRC PRess, 2017.

[2] S. N. Srihari, “Individuality of Handwriting,” *Journal of Forensic Science*, vol. 47, no. 4, p. 17, Jul. 2002.

[3] A. Foroozandeh, A. Askari Hemmat, and H. Rabbani, “Offline Handwritten Signature Verification and Recognition Based on Deep Transfer Learning,” in *2020 International Conference on Machine Vision and Image Processing (MVIP)*, Feb. 2020, pp. 1–7. doi: 10.1109/MVIP49855.2020.9187481.

[4] M. A. Shaikh, M. Chauhan, J. Chu, and S. Srihari, “Hybrid Feature Learning for Handwriting Verification,” Aug. 2018, pp. 187–192. doi: 10.1109/ICFHR-2018.2018.00041.

[5] B. V. Dhandra and M. B. Vijayalaxmi, “A Novel Approach to Text Dependent Writer Identification of Kannada Handwriting,” *Procedia Computer Science*, vol. 49, pp. 33–41, Jan. 2015, doi: 10.1016/j.procs.2015.04.224.

[6] X. Wu, Y. Tang, and W. Bu, “Offline Text-Independent Writer Identification Based on Scale Invariant Feature Transform,” *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 3, pp. 526–536, Mar. 2014, doi: 10.1109/TIFS.2014.2301274.

[7] C. Djeddi, S. Al-Maadeed, I. Siddiqi, G. Abdeljalil, S. He, and Y. Akbari, “ICFHR 2018 Competition on Multi-Script Writer Identification,” in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Aug. 2018, pp. 506–510. doi: 10.1109/ICFHR-2018.2018.00094.

[8] S. He, M. Wiering, and L. Schomaker, “Junction detection in handwritten documents and its application to writer identification,” *Pattern Recognition*, vol. 48, no. 12, pp. 4036–4048, Dec. 2015, doi: 10.1016/j.patcog.2015.05.022.

[9] C. Djeddi, A. Gattal, L. Souici-Meslati, I. Siddiqi, Y. Chibani, and H. El Abed, “LAMIS-MSHD: A Multi-script Offline Handwriting Database,” in *2014 14th International Conference on Frontiers in Handwriting Recognition*, Sep. 2014, pp. 93–97. doi: 10.1109/ICFHR.2014.23.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd edition. Upper Saddle River, NJ: Pearson, 2007.

[11] Z. Guo, L. Zhang, and D. Zhang, “A Completed Modeling of Local Binary Pattern Operator for Texture Classification,” *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1657–1663, Jun. 2010, doi: 10.1109/TIP.2010.2044957.

[12] A. J. Newell and L. D. Griffin, “Writer identification using oriented Basic Image Features and the Delta encoding,” *Pattern Recognition*, vol. 47, no. 6, pp. 2255–2265, Jun. 2014, doi: 10.1016/j.patcog.2013.11.029.

[13] H. T. Nguyen, C. T. Nguyen, T. Ino, B. Indurkhya, and M. Nakagawa, “Text-independent writer identification using convolutional neural network,” *Pattern Recognition Letters*, vol. 121, pp. 104–112, Apr. 2019, doi: 10.1016/j.patrec.2018.07.022.

[14] V. Christlein and A. Maier, “Encoding CNN Activations for Writer Recognition,” in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, Apr. 2018, pp. 169–174. doi: 10.1109/DAS.2018.9.

[15] F. Abbas, A. Gattal, C. Djeddi, I. Siddiqi, A. Bensefia, and K. Saoudi, “Texture feature column scheme for single- and multi-script writer identification,” *IET Biometrics*, vol. 10, no. 2, pp. 179–193, 2021, doi: 10.1049/bme2.12010.

[16] A. Semma, Y. Hannad, I. Siddiqi, S. Lazrak, and M. E. Y. E. Kettani, “Feature learning and encoding for multi-script writer identification,” *IJDAR*, vol. 25, no. 2, pp. 79–93, Jun. 2022, doi: 10.1007/s10032-022-00394-8.

[17] J. W. Kim, K. I. Kim, B. J. Choi, and H. J. Kim, “Decomposition of Chinese character into strokes using mathematical morphology,” *Pattern Recognition Letters*, vol. 20, no. 3, pp. 285–292, Mar. 1999, doi: 10.1016/S0167-8655(98)00147-0.

[18] J. Chen, B. Li, and X. Xue, “Zero-Shot Chinese Character Recognition with Stroke-Level Decomposition.” arXiv, Jun. 22, 2021. doi: 10.48550/arXiv.2106.11613.

[19] N. Otsu, “A Threshold Selection Method from Gray-Level Histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, Art. no. 1, Jan. 1979, doi: 10.1109/TSMC.1979.4310076.