

Revolutionizing Historical Manuscript Analysis: A Deep Learning Approach with Intelligent Feature Extraction for Script Classification

Merouane Boudraa¹ , Akram Bennour¹ , Tahar Mekhaznia¹ , Abdulrahman Alqarafi²,
Rashiq Rafiq Marie², Mohammed Al-Sarem² , Ayush Dogra³ 

¹ Laboratory of Mathematics, Informatics and Systems, Larbi Tebessi University – Tebessa, Algeria

² College of Computer Science and Engineering, Taibah University, Madinah, Saudi Arabia

³ Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India

Corresponding author: Merouane Boudraa (merouane.boudraa@univ-tebessa.dz)

Abstract

The automated classification of historical document scripts holds profound implications for historians, providing unprecedented insights into the contexts of ancient manuscripts. This study introduces a robust deep learning system integrating an intelligent feature selection method, elevating the script classification process. Our methodology, applied to the CLaMM dataset, involves preprocessing steps such as advanced denoising through non-local means and binarization using Canny edge detection. These steps, pivotal for image cleaning and segmentation, set the stage for subsequent in-depth analysis. To enhance feature detection, we employ the Harris corner detector, followed by a k-means clustering process to eliminate redundancy and outliers. This process facilitates the extraction of consistently sized patches, capturing distinctive features of various scripts in historical manuscripts. The dataset undergoes rigorous training using precise convolutional neural network (CNN) models, empowering our system to discern intricate patterns and features for informed decision-making during the classification process. Ultimately, for the definitive script classification of an entire document, we employ a majority voting mechanism on the patches. The results highlight the effectiveness of this comprehensive approach, with the system achieving an impressive accuracy rate of 89.2%. This underscores the system proficiency in accurately classifying historical document scripts, offering a reliable and efficient solution for historians and researchers. The robustness of our methodology positions it as a compelling tool for meticulous analysis of historical manuscripts, contributing significantly to the field of historical document research and preservation.

Keywords

Script classification; Historical manuscripts; Intelligent features; Feature extraction; CNNs; Transfer learning.

Citation: Boudraa, M., Bennour, A., Mekhaznia, T., Alqarafi, A., Marie, R. R., Al-Sarem, M., & Dogra, A. (2024). Revolutionizing Historical Manuscript Analysis: A Deep Learning Approach with Intelligent Feature Extraction for Script Classification. *Acta Informatica Pragensia*, 13(2), 251–272. <https://doi.org/10.18267/j.aip.239>

Special Issue Editors: Hakim Bendjenna, Larbi Tebessi University – Tebessa, Algeria
Lawrence Chung, University of Texas at Dallas, USA
Abdallah Meraoumia, Larbi Tebessi University – Tebessa, Algeria

Academic Editor: Zdenek Smutny, Prague University of Economics and Business, Czech Republic

Copyright: © 2024 by the author(s). Licensee Prague University of Economics and Business, Czech Republic.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution License (CC BY 4.0).

1 Introduction

Historical manuscripts are repositories of knowledge and windows to the past that reveal the intricacies of bygone eras. Understanding the script types used in these manuscripts is of paramount importance to historians, as it not only provides invaluable insights into historical contexts but also assists in unravelling the mysteries that time has shrouded. For years, historians and researchers have meticulously examined these artifacts, labouring to decipher their content and glean a better understanding of the past using manual methods (Parkes, 2016). Palaeographers, experts in this field, employ a keen eye for detail to discern subtle nuances in handwriting that can be indicative of specific script types or distinct periods. By studying the intricacies of scripts, palaeographers contribute significantly to the attribution of manuscripts to particular regions, cultures and historical eras. The expertise of palaeographers is crucial for historians seeking to unlock the rich content embedded in historical manuscripts, providing a bridge between the written word and the historical contexts in which it emerged. Through their expertise, palaeographers play a key role in preserving and interpreting the cultural heritage encapsulated in the diverse array of scripts found in ancient manuscripts.

With the increasing prominence of computer-based methods, the classification of historical manuscript scripts has undergone a transformative shift towards automation, empowering historians with a formidable analytical tool. The conventional approach, initially reliant on traditional feature extraction and assisted by early machine learning techniques (Peake & Tan, 1997; Tan, 1998; Singh et al., 2016), marked a notable improvement over entirely manual methods. This transition from manual to semi-automated processes represented a significant stride in efficiency and scalability, laying the groundwork for more sophisticated and nuanced methodologies in subsequent developments. Despite the improvements brought about by traditional machine learning methods, the pursuit of more robust and advanced methodologies has persisted. Notably, convolutional neural networks (CNNs) (LeCun et al., 2015) have emerged as a transformative force, showcasing superior performance in comparison to their traditional counterparts (Samai et al., 2017; Meraoumia et al., 2018; El Bahi & Zatni, 2019; Demilew & Sekeroglu, 2019). It is this exceptional capability of CNNs that serves as the driving force behind our study. In response to the growing demand for enhanced efficiency and effectiveness in historical manuscript script classification, this research delves into the realm of deep learning, aiming to harness the potential of CNNs to address the intricate challenges posed by the diverse scripts found in historical manuscripts.

The crux of our investigation revolves around convolutional neural networks (CNNs) and their variants such as ResNet and DenseNet. They are renowned for their prowess in various computer vision tasks and adeptness in pattern recognition. In our study, we rigorously evaluate the performance of these architectures to determine which one is best suited for the task of historical manuscript script classification. Our methodological approach, illustrated in Figure 1 and inspired by our prior work on dating tasks (Boudraa & Bennour, 2023), starts with manuscript preprocessing—an essential phase aimed at improving document quality for subsequent analysis. Notably, denoising and binarization procedures play a central role in this preprocessing stage, employing non-local means (NLM) and the Canny edge threshold method to eliminate noise and standardize visual attributes. These enhancements lay the groundwork for feature detection using the Harris corner detector (HCD), a crucial step in extracting uniform patches of significance, a prerequisite for the application of deep learning models. In the classification process, we employ a variety of models sourced from diverse architectures, including CNNs such as ResNet and DenseNet. Ultimately, for the definitive script classification of an entire document, we adopt a majority voting mechanism—an approach that reinforces the accuracy and reliability of our classifications. Our objective is to discern the model most apt for our system. The assessment of our system hinges on the ClaMM dataset (Stutzmann, 2016), recognized as a benchmark in historical manuscript analysis. Figure 2 is an illustration of a manuscript from the ClaMM dataset. The results gleaned from our experiments

vividly illustrate the remarkable capabilities of our deep learning-based system. The main highlights of this research can be summarized as follows:

1. We introduce an innovative approach that integrates machine-learned features with a deep learning model, utilizing an intelligent feature selection method. This method significantly contributes to the accurate classification of scripts in old manuscripts.
2. Employing effective pre-processing techniques, we ensure the acquisition of clean and denoised data for improved analysis.
3. To address the challenge of a limited number of samples, we implement efficient data augmentation strategies, enhancing the diversity and richness of our dataset.
4. Automatic keypoint detection, combined with the k -means clustering technique, is employed to eliminate manual selection of the region of interest. This ensures a well-distributed selection of regions across the entire image while minimizing redundancies and irrelevant information that could affect the training process and overall classification system.
5. Multiple experimentation scenarios are provided for each phase of the framework, offering a thorough justification for the choices made during the development process.

The remainder of this article unfolds as follows. In the next section (1.1), we delve into the related works pertinent to this study, providing insights into existing research in the field. Following that, in Section 2, we present a detailed overview of our methodology, outlining the steps and approaches employed in our research. Subsequently, Section 3 comprises our experiments, where we detail the procedures, scenarios and results obtained, fostering a comprehensive understanding of our system's performance. In Section 4, we engage in a discussion, offering insights, interpretations and implications arising from our findings. Finally, Section 5 concludes the paper, summarizing key discoveries and potential avenues for future research.

1.1 Literature review

The investigation into historical manuscript analysis has traced a diverse path in research, ranging from traditional methodologies to contemporary deep-learning approaches. Many of these methodologies have featured prominently in competitions such as ICFHR 2016 (Cloppet et al., 2016), ICDAR 2017 (Cloppet et al., 2017) and ICDAR 2021 (Seuret et al., 2021).

Approaches to script classification centre on identifying script features within manuscripts, typically extracted using conventional machine learning techniques. The FAU system, as detailed by Cloppet et al. (2016), introduced a writer style analysis method combining speaker verification, computer vision and statistical modelling. Utilizing i -vectors (Dehak, 2011) for representation of handwriting styles and employing session compensation through within-class covariance normalization (WCCN), it merges concepts from text and speaker analysis. The adaptation of scale invariant feature transform (SIFT) descriptors, transformed into RootSIFT, enhances feature robustness, with feature comparison employing Hellinger distance (Arandjelovic & Zisserman, 2012). The final classification is accomplished using latent Dirichlet analysis (LDA), offering a unique approach to capturing and classifying distinctive handwriting styles. The S2-CCMV system of Cloppet et al. (2017) focuses on colour, luminance and geometrical features for document image classification. It starts by extracting connected components (CCs) through global thresholding on a difference-of-gaussian (DOG) enhanced image (Fischer et al., 2012). Patterns are formed using k -means clustering on CCs for each script class, and class probabilities are estimated based on Euclidean distances. Criteria such as match frequency, class correspondence and confusion matrix analysis are applied to model patterns, followed by a majority voting scheme for page-level classification. Additionally, CK2, detailed by Cloppet et al. (2017), draws inspiration from effective writer identification techniques (Christlein et al., 2015; Christlein et al., 2017; Bennour et al., 2019), employing a traditional bag-of-(visual)-words approach with RootSIFT descriptors. These descriptors undergo PCA whitening and

dimensionality reduction, utilizing vectors of locally aggregated descriptors (VLAD) for encoding. Feature correlation is minimized through multiple background models and PCA whitening. The classification is executed using linear support vector machines (SVMs), with adjustments for the second test set involving a smaller image crop to reduce background interference.

The rise of deep learning, particularly convolutional neural networks (CNNs), has revolutionized script classification, as showcased by prominent studies such as the DeepScript method of Cloppet et al. (2016). This neural network architecture, a variation on the VGG architecture, incorporates fully connected dense layers and a softmax layer for class label prediction. Training involves batch size 30, stochastic gradient descent and cross-entropy loss, with regularization through dropout layers and Glorot initialization. Testing entails averaging predictions over random crops. Similarly, the FRDC-OCR system (Cloppet et al., 2016), operating as a CNN-based classifier, extracts fixed-size patches evenly across images. These patches undergo individual processing by a trained CNN classifier, with the output comprising a feature vector and recognition results with confidence scores. The system computes average feature vectors and confidences for comprehensive image representation. The NNML system (Cloppet et al., 2016) utilizes deep CNNs with residual learning and batch normalization for script classification. Extracting and classifying 227x227 sub-windows, it employs stochastic gradient descent, scale-specific models in an ensemble and averages predictions for the final result. The T-DeepCNN system (Cloppet et al., 2017) relies on deep CNNs with residual learning, classifying 227x227 sub-windows for script or date classes. Training incorporates stochastic transformations, downsampling and scale-specific models in an ensemble. The P-CNN system (Cloppet et al., 2017) adopts CNNs, using a ResNet152 pre-trained on ImageNet for script classification. Training includes a random crop of 256x256, with results showcasing improved performance. The Pero system (Seuret et al., 2021) utilizes a CNN on text lines, employing global average pooling and the Lmin loss function for classification. Page-level outputs are obtained by calculating mean or median vectors for specific text line lengths. The LTU method (Seuret et al., 2021) undergoes preprocessing with cropping and downsizing before training. Crops of 224x224 with a stride of 224 are used, employing an 18-layer pre-trained ResNet for classification.

Recent studies in the field have unveiled diverse methods, ranging from traditional machine learning to advanced deep learning techniques. Building on the foundation laid by preceding studies, our research goes a step further by proposing a hybrid system. This brings a fresh methodology to the realm of script classification for historical manuscripts, contributing to the ongoing advancements in this dynamic and evolving field.

2 Research Methods

Classifying historical scripts presents a complex challenge requiring a robust methodology. In response to this need, we engineered a meticulously designed system that incorporates multiple sequential steps as presented in Figure 1. This section will expound on distinct phases, including preprocessing, feature detection and classification, while also providing insights into the dataset employed for the analysis.

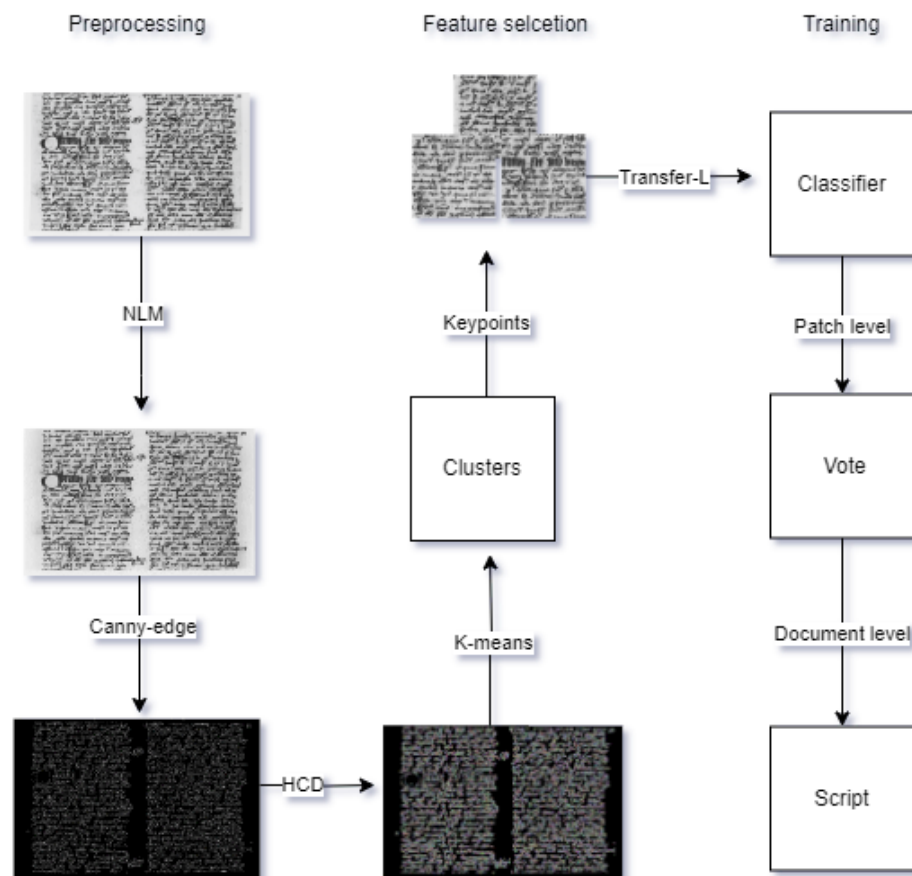


Figure 1. Script classification system design

2.1 Dataset

The dataset utilized in this study is referred to as the CLaMM dataset, which made its appearance in the Competition on the Classification of Medieval Handwritings in Latin Script (Cloppet et al., 2017). Comprising 3540 samples showcasing diverse script types, as illustrated in Figure 2, the dataset originates from the French catalogues of dated and datable manuscripts (Stutzmann et al., 2016). Table 1 outlines the dataset composition, consisting of 12 distinct script classes, each contributing to the comprehensive scope of historical manuscript script classification undertaken in this research.

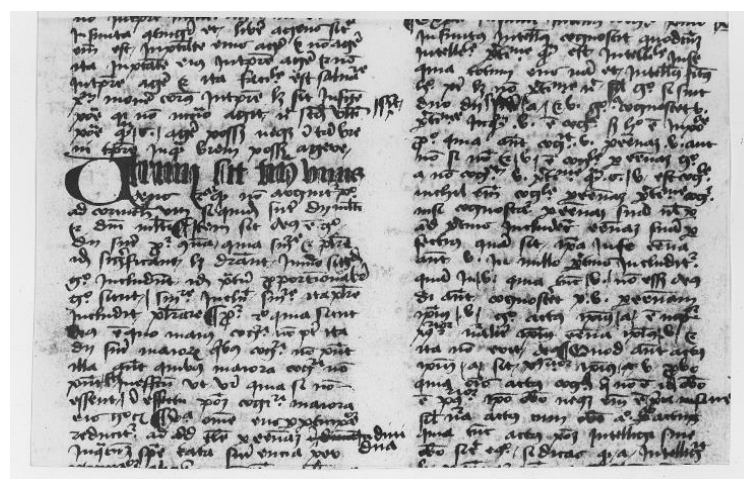


Figure 2. Script samples from CLaMM dataset.

Table 1. CLaMM scripts per class.

Class	1	2	3	4	5	6	7	8	9	10	11	12
Samples	356	395	200	207	217	341	411	265	293	202	442	211

2.2 Preprocessing

The intricacy of analysing historical manuscripts arises from the inherent challenges in their quality, characterized by various imperfections such as worn and torn papers (Figure 3 (a)), residual ink traces (Figure 3 (b)) and indistinct handwritten text (Figure 3 (c)). These factors present difficulties in identifying keypoints and crucial regions of interest for the classifier. Consequently, our system is purposefully crafted to mitigate these challenges through the implementation of two primary preprocessing techniques: denoising through non-local means algorithm and binarization using Canny edge detection. These procedures are elaborated in detail in the subsequent section.

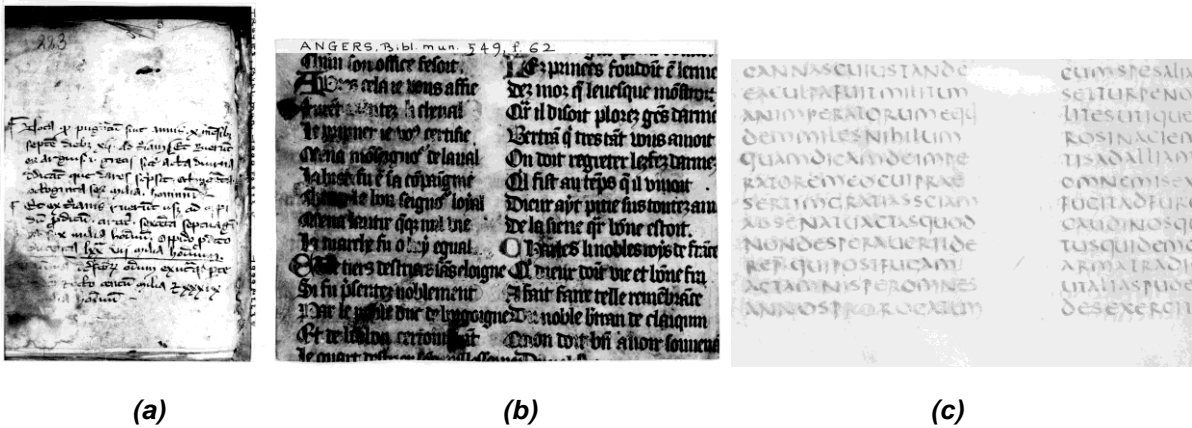
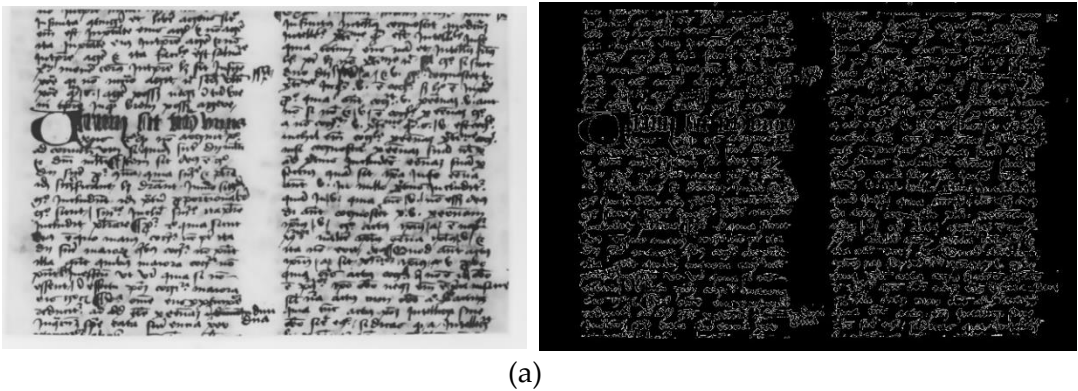


Figure 3. Problems with CLaMM samples: (a) worn paper, (b) ink traces, (c) indistinct text.

2.2.1 Non-local means denoising

Denoising plays a crucial role in our system as it helps improve the quality of the images before proceeding to the feature selection step. To ensure optimal denoising performance, we experimented with various denoising techniques, including non-local means (NLM) (Buades et al., 2011) (Figure 4 (a)) and bilateral filtering (Paris et al., 2009) (Figure 4 (b)). To further evaluate and compare their performance, we applied the Canny edge detection algorithm to the denoised images and the outputs are shown in Figure 4 as well. It was observed that the NLM denoising technique exhibited the best performance based on the clarity and accuracy of the edges detected after denoising, as shown in the final output. This highlights the importance of selecting the appropriate denoising method to ensure optimal results in subsequent steps of our system, particularly in the feature selection process.



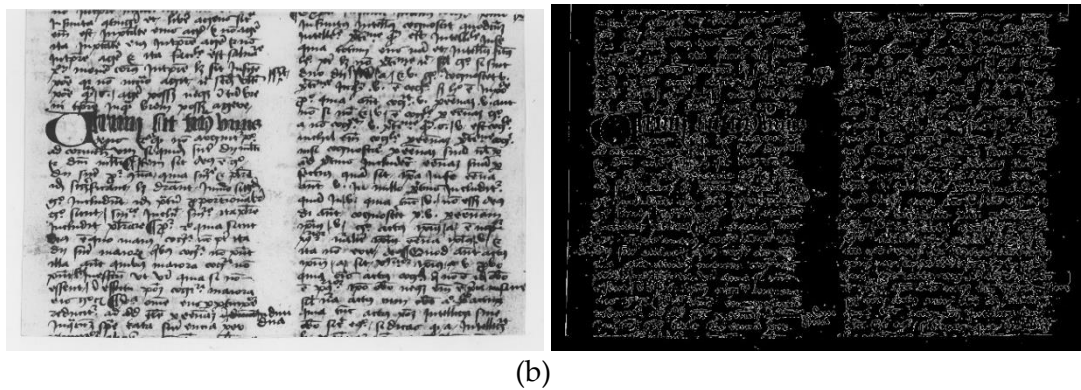


Figure 4. Denoising results, (a) non-local means, (b) bilateral filtering.

The non-local means (NLM) algorithm operates by averaging pixel values based on the similarity of image patches within the document. The denoising process involves patch extraction ($P(x)$), similarity calculation using the mean squared error (MSE) or Euclidean distance (Equation 1) and weighted averaging of pixel values (Equation 2). This methodology, applied to every pixel in the image, results in the denoised image ($I_d(x)$). The normalization factor ($C(x)$) ensures a balanced weighted average, enhancing the algorithm effectiveness in reducing noise while preserving intricate details crucial for historical script classification.

$$S(P(x), P(y)) = e^{-\frac{\|P(x) - P(y)\|^2}{2a^2}} \quad (1)$$

$$I_d(x) = \frac{1}{C(x)} \sum_y S(P(x), P(y)) \cdot I(y) \quad (2)$$

- **$P(x)$:** This refers to the image patch centred on the pixel x . In the NLM algorithm, $P(x)$ is a square or rectangular region of pixels extracted from the input image around the pixel x .
- **$P(y)$:** This refers to the image patch centred on the pixel y . In the NLM algorithm, $P(y)$ is a square or rectangular region of pixels extracted from the input image around the pixel y .
- **$S(P(x), P(y))$:** This represents the similarity between two image patches, $P(x)$ and $P(y)$, which are extracted from the input image around the pixels x and y respectively. The similarity is calculated using a Gaussian function based on the squared Euclidean distance between the patches.
- The variable **a** is a parameter that controls the extent of similarity; a smaller **a** value indicates stricter similarity criteria.
- **$I_d(x)$:** This denotes the denoised intensity or pixel value at the position x in the output image. It is computed as a weighted average of pixel values based on the similarities between the patches $P(x)$ and $P(y)$, where y ranges over all the pixels in the image. The weights are determined by the similarity function $S(P(x), P(y))$ and the normalization factor $C(x)$.
- **$C(x)$:** The normalization factor ensures a balanced weighted average in the denoising process. It is calculated based on the sum of similarities between the patch **$P(x)$** and all other patches in the image, ensuring that the weighted average does not overly favour certain patches.
- **$I(y)$:** This represents the intensity or pixel value at the position y in the input image. It is used in the calculation of the denoised intensity **$I_d(x)$** through the weighted averaging process.

2.2.2 Canny edge binarization

Selecting the right binarization method is crucial for optimizing analysis in historical script classification, focusing on distinguishing writing styles. We use the Canny edge detection method (Canny, 1986) for its ability to outline distinctive edges, which vary among scripts. Direct application of Canny edge to noisy original images yields poor results, but applying denoising techniques first, as shown in Figure 5,

enhances edge detection by removing noise and smoothing the image, resulting in clearer and more accurate edges.

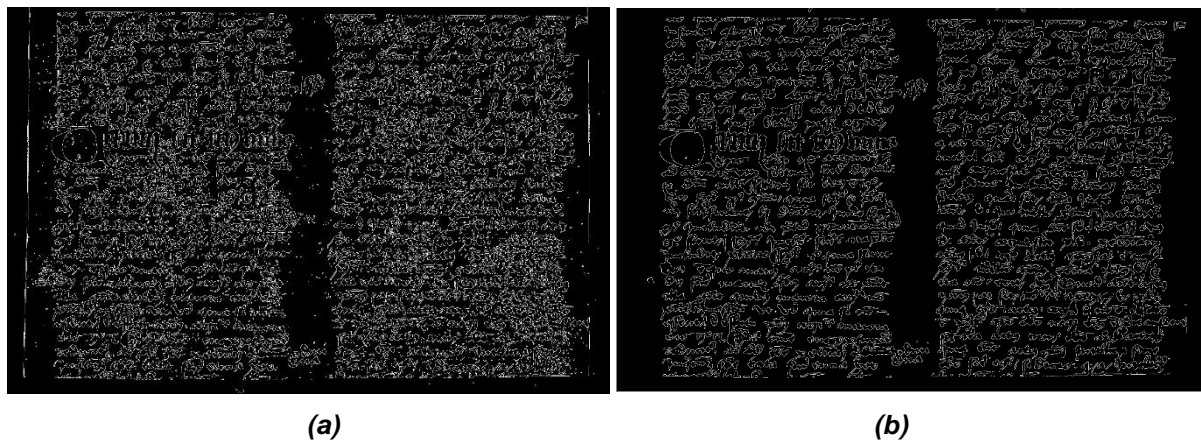


Figure 5. Canny edge binarization, (a) before denoising, (b) after denoising.

The Canny edge detection operates through a multi-stage algorithm. First, smoothing is achieved by convolving the image with a Gaussian filter, as shown in Equation 3. Next, gradients of the image are calculated using Sobel filters in both horizontal and vertical directions, illustrated in Equations 4 and 5. Subsequently, non-maximum suppression identifies local maxima in the gradient magnitude, thinning the edges, and Equation 6 represents edge tracking by hysteresis, connecting pixels with high gradient magnitude while suppressing weak edges below a certain threshold, ensuring the retention of strong edges and mitigating the impact of noise.

$$I_{smoothed}(x, y) = I(x, y) * G(x, y) \quad (3)$$

- $I_{smoothed}(x, y)$: Represents the smoothed image at the pixel coordinates (x, y) .
- $I(x, y)$: Denotes the original image intensity at the pixel coordinates (x, y) .
- $G(x, y)$: Refers to the Gaussian filter applied to the image at the pixel coordinates (x, y) .
- $*$: Represents the convolution operation between the original image and the Gaussian filter.

$$M(x, y) = \sqrt{Gx^2 + Gy^2} \quad (4)$$

- $M(x, y)$: Represents the magnitude of the gradient at the pixel coordinates (x, y) .
- Gx : Denotes the gradient magnitude in the horizontal direction at the pixel coordinates (x, y) .
- Gy : Denotes the gradient magnitude in the vertical direction at the pixel coordinates (x, y) .
- $\sqrt{\cdot}$: Represents the square root operation.

$$\theta(x, y) = \arctan\left(\frac{Gx}{Gy}\right) \quad (5)$$

- $\theta(x, y)$: Represents the orientation of the gradient at the pixel coordinates (x, y) in radians.
- **arctan**: Denotes the arctangent function used to calculate the angle of the gradient.

$$\text{Edge Tracking} = \begin{cases} 1 & \text{if } M(x, y) \text{ is above high threshold} \\ 0 & \text{if } M(x, y) \text{ is below low threshold} \\ \text{Undecided} & \text{otherwise} \end{cases} \quad (6)$$

2.3 Feature detection

In our investigation of keypoint detector techniques for our script classification system, we compared the performance of three popular methods: features from accelerated segment test (FAST) (Rosten et al., 2010), Harris corner detector (HCD) (Harris & Stephens, 1988) and scale-invariant feature transform (SIFT) (Lowe, 2004). Table 2, which presents the time taken by each technique to detect keypoints along with the number of keypoints detected, was generated from experimental evaluations conducted on one image

(Figure 6) from the CLaMM dataset. From the results, FAST is the fastest, taking 0.015 seconds and detecting 11,306 keypoints, while SIFT, though slower at 1.78 seconds, detects the most keypoints (15,018). HCD, with a detection time of 0.033 seconds and 1,312 keypoints, offers a balance between speed and number of keypoints. Despite the speed of FAST and the higher detection rate of SIFT, we chose the Harris corner detector for its efficiency in identifying key corners and interest points, lower computational costs and suitability for real-time or large-scale applications as Figure 6 shows.

Table 2. Keypoint detector techniques.

Feature detector	Time (seconds)	Keypoints
FAST	0.015	11,306
Harris	0.033	1,312
SIFT	1.78	15,018

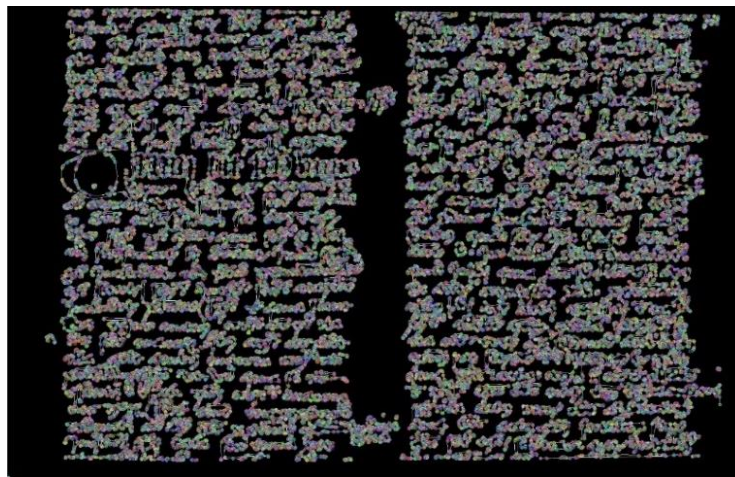


Figure 6. Harris corner keypoint detection.

2.3.1 Harris corner detector

The HCD method involves several steps: first, the calculation of gradients to emphasize edges in alignment with binarization (Equation 7). Subsequently, the computation of the structure tensor M (Equation 7) using these gradients, where I_x and I_y denote derivatives in the x and y directions and $w(x,y)$ is a weighting function. The corner response function R (Equation 8), relying on the eigenvalues of the structure tensor and an empirically determined constant k , is then calculated. Keypoints are localized by identifying pixels where the corner response R exceeds a predefined threshold. Non-maximum suppression is applied to retain only the local maxima in the corner response function, ensuring distinct keypoints. In the context of historical script classification, the Harris corner detection method excels in identifying significant points, such as corners or intersections in the handwritten content, crucial for subsequent stages in the classification.

$$M = \sum_{x,y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (7)$$

- M – structure tensor calculated by summing the components of the matrix;
- I_x and I_y – derivatives of the image intensity in the x and y directions, respectively.

$$R = \det(M) - k(\text{trace}(M))^2 \quad (8)$$

- $\det(M)$ – determinant of the structure tensor (M);

- $trace(M)$ – trace of the structure tensor M ;
- k – empirically determined constant used in the corner response function.

2.3.2 Clustering and patch extraction

After identifying features using the Harris corner detector, our next step is to utilize the k -means clustering method to structure these features. The key objective is to group keypoints, enabling the extraction of distinctive patches from the historical manuscript images. This clustering methodology is implemented to prevent the extraction of redundant or overlapping regions, ensuring that each patch provides unique information for the subsequent stages of analysis (Figure 7).

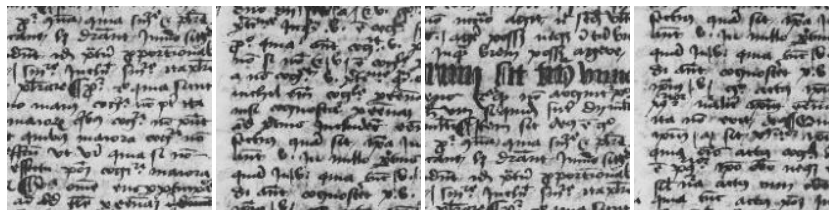


Figure 7. Patch extraction.

K -means clustering (Jin & Han, 2011), a widely used unsupervised learning algorithm, is employed in our research to partition the dataset of detected features (keypoints) obtained from the Harris corner detector into distinct clusters. The objective of the algorithm is to group similar features together, facilitating the extraction of unique patches and preventing redundancy. Given a set of features X representing the coordinates of keypoints and a desired number of clusters k , the algorithm initializes cluster centroids randomly and iteratively assigns features to the cluster with the nearest centroid based on the Euclidean distance (Equation 9). The centroids are updated by recalculating the mean of the features assigned to each cluster (Equation 10). The iterative process continues until convergence, where the centroids no longer change significantly or after a predetermined number of iterations. The objective function, minimized in k -means, is the sum of squared distances from each feature to its assigned centroid, known as "inertia" or "within-cluster sum of squares" (Equation 11). In our context, each feature corresponds to a Harris corner keypoint in binarized manuscript images. The k -means algorithm ensures that keypoints are assigned to clusters, yielding distinctive patches representing unique regions of interest in historical scripts, thereby enhancing the effectiveness of subsequent analysis stages.

$$distance(x_i, c_j) = \|x_i - c_j\| \quad (9)$$

- x_i represents a feature (keypoint) in the dataset and c_j denotes the centroid of the cluster j . The function $distance(x_i, c_j)$ calculates the Euclidean distance between a feature x_i and the centroid c_j , measuring their similarity.

$$c_j = \frac{1}{N_j} \sum_{x_i \text{ in cluster } j} x_i \quad (10)$$

- N_j – number of features in cluster j ;
- c_j represents the updated centroid of cluster j after reassignment of features;
- $\sum_{x_i \text{ in cluster } j} x_i$ calculates the sum of all features assigned to the cluster j and dividing by the number of features in that cluster gives the new centroid location.

$$J = \sum_{i=1}^n \|x_i - cluster(x_i)\|^2 \quad (11)$$

- x_i refers to a feature in the dataset and $cluster(x_i)$ indicates the cluster to which the feature x_i is assigned. The objective function J computes the sum of squared distances from each feature to its assigned centroid, representing the overall "inertia" or "within-cluster sum of squares" minimized by the k -means algorithm.

2.4 Training

2.4.1 Convolutional neural networks

Convolutional neural networks (CNNs) (LeCun et al., 2015) stand out as a potent class of deep neural networks crafted for processing structured grid data, particularly images. This architecture features convolutional layers, which meticulously scan input images for local patterns, pooling layers that adeptly downsample spatial dimensions, and fully connected layers catering to high-level reasoning. In the realm of script classification for historical manuscripts, CNNs present distinctive advantages. Their spatial hierarchy understanding proves well-suited for deciphering intricate scripts, showcasing translation invariance that accommodates variations in character placement. The automatic acquisition of hierarchical representations aligns seamlessly with the diverse and nuanced nature of historical manuscript features. Furthermore, CNNs demonstrate adaptability to varied data, ensuring robustness in handling diverse script styles, fonts and sizes. This versatility, combined with inherent capabilities in spatial and hierarchical feature learning, establishes CNNs as a popular and effective choice for script classification tasks in historical manuscript analysis. CNNs comprise several types of layers, each contributing uniquely to their image processing capabilities.

1. **Convolutional layers:** Convolutional layers in CNNs apply filters F_i (kernels) across the input image I to produce feature maps M_i . Each filter slides (convolves) over the input, computing the dot product of the filter weights and the input values, capturing local patterns. Mathematically, the feature map M_i is computed in Equation 12 as:

$$M_i = I * F_i + b_i \quad (12)$$

- where $*$ denotes the convolution operation and
- b_i is the bias term.
-

2. **Pooling layers:** Pooling layers perform downsampling on the feature maps to reduce spatial dimensions while preserving important features. The common operations are max pooling and average pooling.

For max pooling, the operation over a region R of the feature map M_i is presented in Equation 13:

$$P_{max} = \max(M_i[R]) \quad (13)$$

For average pooling, it is presented in Equation 14:

$$P_{avg} = \frac{1}{|R|} \sum_{x \in R} M_i[x] \quad (14)$$

- where P_{max} and P_{avg} are the pooled outputs.

3. **Fully connected layers:** Fully connected layers (dense layers) take the flattened output from the previous layers and transform it using weight matrices W and bias vectors b . The output O for a fully connected layer can be expressed in Equation 15 as:

$$O = W \cdot X + b \quad (15)$$

- where X is the input feature vector,
- W is the weight matrix and
- b is the bias vector.

CNNs have showcased remarkable success within the historical manuscripts domain, as evidenced by various studies (e.g., Lombardi & Marinai, 2020). These studies have effectively employed CNNs across a spectrum of tasks, including dating manuscripts, identifying writers and classifying scripts. In dating manuscripts, CNNs have been instrumental in capturing temporal patterns and writing styles indicative of specific historical periods (Boudraa et al., 2024). Furthermore, CNNs have proven invaluable in writer identification tasks by discerning unique handwriting features and characteristics, facilitating accurate

attribution to individual writers (Bennour et al., 2024). Additionally, CNNs have been employed in script classification endeavours, where they excel in distinguishing and categorizing different script types based on their visual attributes (Seuret et al., 2021). These applications highlight the versatility and effectiveness of CNNs in addressing diverse challenges within the domain of historical manuscripts, underscoring their significance as a cutting-edge technology in this field.

2.4.2 Transfer learning

Transfer learning is a machine learning technique that makes use of knowledge gained from one task to improve the performance of a model on another related task. In the context of script classification, transfer learning involves pre-training a model on a large dataset or a related task and then fine-tuning it on a smaller dataset specific to the script classification task, in our case convolutional neural networks (CNNs) such as ResNet, DenseNet and VGG as shown in Table 3. This approach capitalizes on the knowledge and features learned during the initial training, allowing the model to generalize better to the new task with limited labelled data. Capabilities of transfer learning in script classification include improved model convergence, faster training times and enhanced performance, especially in scenarios with limited labelled data. Transfer learning helps the model capture generic features and patterns that are transferrable to the specific nuances of script classification. It aids in overcoming the data scarcity challenge by providing a head start for the model to understand complex script structures and styles.

DenseNet-121 (Huang et al., 2017), VGG-16 (Simonyan & Zisserman 2015) and ResNet-50 (He et al., 2016) were chosen based on their unique design features and proven effectiveness in image-related tasks (Figure 8).

- **DenseNet-121**, with its dense connectivity, facilitates efficient feature reuse, making it suitable for capturing intricate patterns in historical scripts.
- **VGG-16**, known for its simplicity and uniform architecture, offers a straightforward and interpretable model, excelling in hierarchical feature extraction—a crucial aspect for capturing both low-level and high-level features in historical manuscripts.
- **ResNet-50**, using residual learning to address vanishing gradient issues, excels in handling very deep networks, making it well-suited for capturing nuanced patterns in diverse historical script datasets.

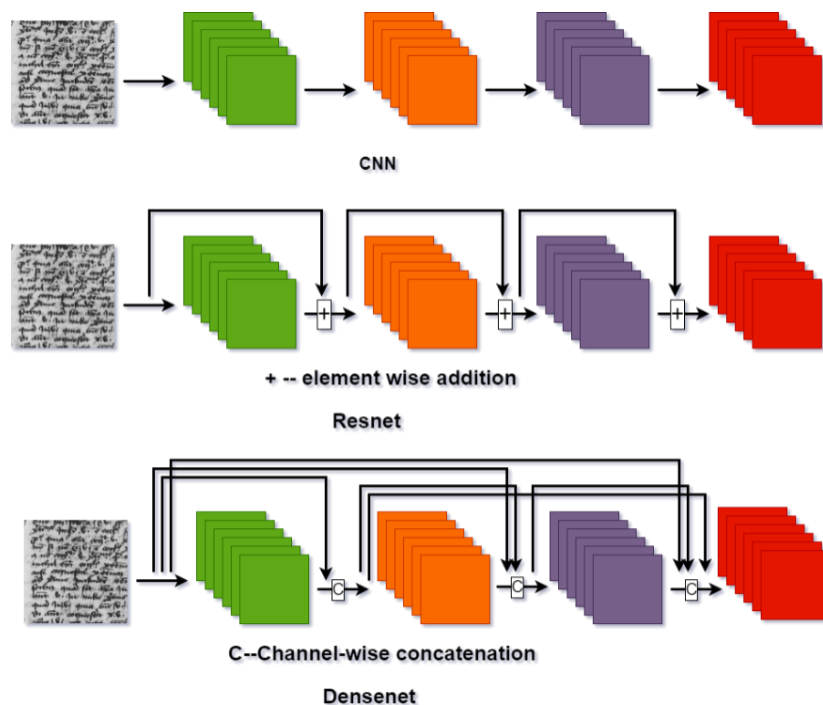


Figure 8. Traditional CNN, ResNet and DenseNet architectures.

Table 3. Pretrained model layers and parameters.

Pretrained model	Layers	Parameters
ResNet-50	50	23.9M
DenseNet-121	121	7.2M
VGG-16	16	138M

2.5 Majority vote

With the patches extracted from the historical documents through our trained models, we seize the opportunity to refine our classification methodology through the application of a majority vote mechanism, as illustrated in Figure 9. The majority vote technique involves aggregating the predictions of individual patches belonging to the same document and determining the final classification based on the most frequently predicted script type within that document. This ensemble approach makes use of the diversity of predictions from different patches, mitigating the impact of potential noise or inaccuracies in individual patch predictions (Equation 16). The rationale behind the majority vote is to enhance the robustness and reliability of the overall script classification for each historical manuscript, fostering a more accurate representation of the document's predominant script type.

$$C = \operatorname{argmax}_i \sum_{j=1}^n I(P_i = P_j) \tag{16}$$

- where C represents the final classification, which is determined through the majority vote mechanism;
- P_1, P_2, \dots, P_n are the predictions of individual patches for a given document. Each P_i represents the predicted script type for the i -th patch;
- n is the total number of individual patches considered for the document;
- argmax_i operator returns the index i that maximizes the expression following it. In this case, it finds the index of the most frequently predicted script type;
- $\sum_{j=1}^n$ represents the summation over all individual patches, where j ranges from 1 to n ;
- $I(P_i = P_j)$ is the indicator function that returns 1 if the predicted script type for the patch i is equal to the predicted script type for the patch j and 0 otherwise. It essentially checks if two patches have the same prediction.

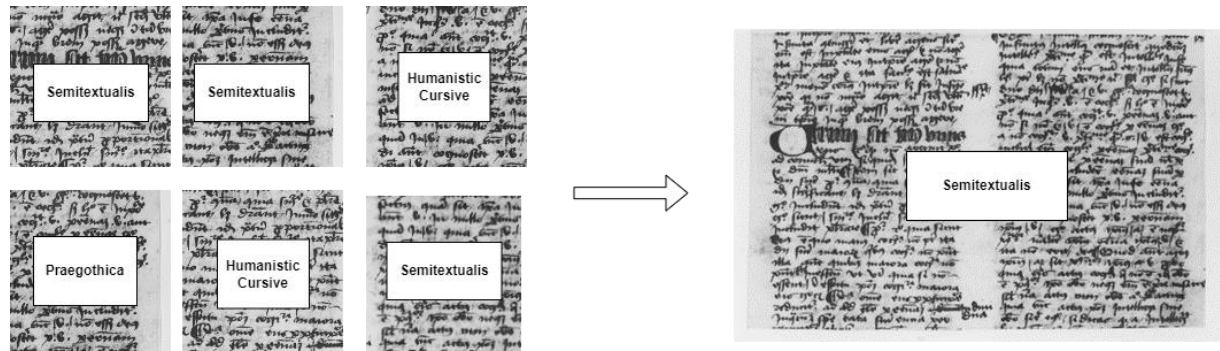


Figure 9. Majority vote technique.

3 Solutions and Results

In this section, we embark on the implementation of our historical manuscript script classification system, aiming to optimize its performance through a systematic exploration of various parameters. Our objective is to unveil the most effective combination of parameters that yields superior results. Through a series of carefully designed experiments on our dataset, we scrutinize the impact of different configurations,

enhancing our understanding of the system behaviour. Our experiments follow a rigorous protocol, encompassing diverse scenarios to comprehensively evaluate the system performance. We meticulously document the outcomes of each experiment, facilitating a nuanced analysis of the system behaviour under varying conditions. This section serves as a dynamic platform for iterative improvement. By meticulously analysing the results obtained from different experiments, we iteratively refine our system. The continuous feedback loop between experimentation, analysis and refinement is essential for achieving optimal performance. Our findings are systematically organized and presented in tabular format, capturing the essence of each experiment and its corresponding experiment configuration. Tables serve as visual aids, allowing clear comparisons and facilitating the identification of trends across different experiments.

To thoroughly evaluate our historical manuscript script classification system, we utilize a set of stringent evaluation metrics specifically tailored to the nuances of our dataset. We meticulously partition our dataset, dedicating 80% for training and reserving 20% for validation.

Taking inspiration from the evaluation methodology employed in the ICDAR 2017 for a comparable task, our primary focus lies on the metric of global accuracy (GA), as defined by Equation 17:

$$GA = \frac{1}{N} \sum_{i=1}^N \delta(Dli, GTLi) \quad (17)$$

- where **GA** is global accuracy, which is the metric used to measure the overall accuracy of the classification system;
- **N** is the total number of samples or instances in the dataset;
- **i** is the index representing each sample in the dataset, ranging from 1 to **N**;
- $\delta(Dli, GTLi)$ is the delta function that evaluates whether the predicted class **Dli** matches the ground truth class **GTli** for the sample **i**;
- **Dli** is the predicted class or label for the sample **i**; and
- **GTli** is the ground truth class or correct label for the sample **i**.

3.1 Preprocessing impact

The experiment titled "preprocessing impact" delves into the effectiveness of various preprocessing steps employed in our system, which include denoising, binarization, feature detection and patch extraction. The objective is to ascertain the advantages gained from implementing these preprocessing techniques. Table 4 presents the results obtained from this experiment, specifically focusing on the identification performance measured using the GA metric (ResNet-50 model). The experiment is divided into three scenarios: without preprocessing using the whole manuscript as input by only resizing it; random patches, where we extract small patches from each manuscript randomly; and finally with our preprocessing, which includes denoising, binarization, feature detection and patch extraction.

Table 4. Identification performance with and without preprocessing steps.

Experiment	GA (ResNet-50)
Without preprocessing (only resizing)	27.6
Random patches	55.6
With preprocessing	71.16

3.2 Patch size determination

Regarding the patch size parameter, we conducted an experiment involving various patch sizes, specifically 150x150, 224x224, 550x550 and 750x750. To accelerate the process and ensure a fair

comparison, we exclusively utilized one patch from each sample and maintained consistency by training on the same model, ResNet-50. This allowed a comprehensive evaluation of the impact of different patch sizes on the model performance. Table 5 clearly illustrates that a patch size of 550x550 yielded the most favourable results compared to other patch sizes. The observation suggests that this particular patch size contributes to optimal model performance in the context of our experiment.

Table 5. Patch size experiment results.

Patches size	GA (ResNet-50)
150 x 150	39.83
224 x 224	49.64
550 x 550	71.16
750 x 750	68.14

3.3 Cluster number choice

In our subsequent analysis, we scrutinized the influence of varying the number of clusters on the classification rate using same model (ResNet-50), starting with 3, 9, 16 and 25 clusters respectively. To accelerate the process and maintain consistency, we trained on the same model, ResNet-50. Notably, our findings revealed that opting for 16 clusters produced the highest accuracy among the different cluster choices, as Table 6 demonstrates.

Table 6. Cluster number experiment results.

Clusters	GA (ResNet-50)
3	71.16
9	76.42
16	78.28
25	77.32

3.4 Model selection

Furthermore, as shown in Table 7, we conducted a thorough comparative assessment of diverse pre-trained deep learning models to evaluate their efficacy in script classification. We investigated different pretrained models, namely VGG, ResNet and DenseNet. To accelerate the process and ensure a fair comparison, we exclusively utilized one patch from each sample. The results reveal that the DenseNet-121 model achieved the most favourable outcome, highlighting the superior performance of its architecture.

Table 7. Model experiment results.

Model	GA
ResNet-50	71.16
VGG-16	68.5
DenseNet-121	73.02

3.5 Final system result

Following the optimization of parameters, our training involved 20 epochs utilizing patches of size 550x550 with 16 clusters, employing the DenseNet-121 model. Figure 10 shows the confusion matrix for both patch and document level. The achieved results, as presented in Table 8, demonstrate optimal performance. The comprehensive approach, involving parameter tuning and model selection, culminated

in an effective and refined system for the classification of historical manuscript scripts. Additionally, we employ the majority vote technique as a last step in our system, ensuring a robust and accurate classification of historical manuscript scripts.

Table 8. Final system experiment results.

Results	CNN (DenseNet-121)
Patch level	83.66
Document level (majority vote)	89.2

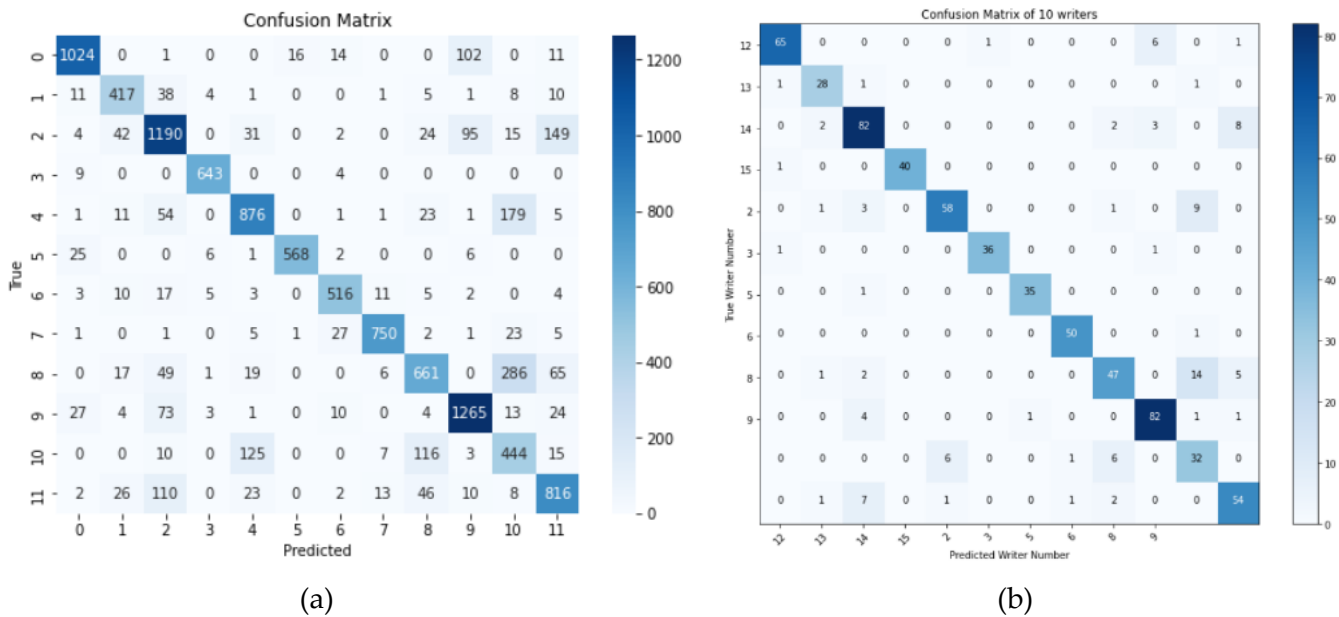


Figure 10. Confusion matrix for (a) patch level, (b) document level.

4 Discussion

In this section, we delve into a comprehensive analysis of the results garnered from our series of experiments, offering nuanced interpretations and insights. The focal points of discussion encompass the outcomes of individual experiments, the implications of parameter choices, the performance of the final system and a comparative analysis with existing systems in historical script classification.

4.1 Impact of preprocessing analysis

The results of the "preprocessing impact" experiment showcase a clear trend indicating the importance of preprocessing steps in enhancing the identification performance of the system. Here is a detailed analysis of the results presented in Table 4:

- Without preprocessing (only resizing):** The lowest identification performance is observed in this scenario, with a GA (ResNet-50) score of 27.6. This outcome underscores the limited effectiveness of merely resizing the images without employing additional preprocessing techniques. It suggests that basic resizing alone is insufficient for capturing the intricate details and features necessary for accurate script classification.
- Random patches:** Introducing random patch extraction as a preprocessing step leads to a notable improvement in identification performance, as evidenced by the GA (ResNet-50) score increasing to 55.6. This improvement indicates that extracting patches from the images helps in capturing

diverse features from different regions, thereby enhancing the ability to discern unique handwriting characteristics associated with individual script types.

3. **With preprocessing:** The most significant improvement in identification performance is observed when complete preprocessing is implemented. The GA (ResNet-50) metric achieves the highest score of 71.16, highlighting the substantial impact of preprocessing steps such as denoising, binarization, feature detection and patch extraction. These preprocessing techniques collectively contribute to improving the quality of the input data by reducing noise, enhancing contrast, extracting meaningful features and standardizing patch sizes. As a result, the identification system can more accurately discriminate between different writing styles and achieve higher accuracy in classifying different script types.

The analysis of these results emphasizes the critical role of preprocessing in optimizing the input data for deep learning models, ultimately leading to improved performance and more reliable outcomes of historical script classification.

4.2 Patch size analysis

Table 5 provides insights from the patch size experiment, offering crucial guidance on the optimal choice of patch size. Key observations include:

1. Starting with a patch size of 150x150, the results indicate low accuracy, falling below 40%. This suggests that a smaller patch size is insufficient for capturing meaningful features, affecting the overall classification performance.
2. As the patch size increases to 224x224, the results exhibit improvement, indicating that a larger patch size allows better feature representation. However, the subsequent increase to a size of 750x750 results in lower performance. This decline is attributed to the diminishing distinction of features as the size grows excessively.
3. The experiment highlights the significance of finding a balance in patch size. The optimal performance is achieved with a patch size of 550x550, providing the best results. This optimal size strikes a balance between capturing distinctive features and generating an adequate number of patches for a robust dataset.

In conclusion, the patch size analysis experiment guides the selection of an optimal patch size for enhanced performance. The chosen size of 550x550 proves to be the most effective, ensuring a balance between feature representation and dataset richness. This careful consideration contributes to the refinement of your system, fostering improved accuracy in historical script classification.

4.3 Cluster number analysis

Table 6 presents the outcomes of the cluster number experiment, shedding light on the impact of varying the number of clusters. Key findings include:

1. Beginning with a small number of clusters (3), the results indicate relatively low accuracy. This suggests that a minimal number of clusters may not capture the diversity of features adequately, affecting the classification performance.
2. As the number of clusters increases, the accuracy improves until reaching an optimal value at 16 clusters. This suggests that a moderate number of clusters enhances the system ability to generate diverse patches, contributing positively to the learning process.
3. However, the results show a decline in performance when the number of clusters further increases to 25. This decline is attributed to the generation of redundant patches, where multiple clusters may represent the same regions of interest. This redundancy negatively affects the training process, leading to diminished accuracy.

In conclusion, the cluster number analysis highlights the importance of finding the right balance. An optimal value, such as 16 clusters, proves effective in capturing diverse features without introducing redundancy. This careful selection contributes to the efficiency of our system, ensuring a robust dataset for historical script classification.

4.4 Model analysis

Table 7 outlines the experiment results for different models, including ResNet-50, VGG-16 and DenseNet-121, with the evaluation metric of global accuracy (GA). The findings indicate distinctive performance levels among the models:

1. **ResNet-50:** Achieves a global accuracy of 71.16%, demonstrating its capability in accurately classifying historical manuscript scripts.
2. **VGG-16:** Shows a global accuracy of 68.5%, representing a slightly lower accuracy compared to ResNet-50.
3. **DenseNet-121:** Outperforms both ResNet-50 and VGG-16, attaining a higher global accuracy of 73.02%. This superiority can be attributed to the complexity of the DenseNet architecture, facilitating more effective feature extraction.

The results highlight the significance of selecting an appropriate CNN architecture, with DenseNet-121 emerging as the most accurate model among those considered. These insights provide valuable guidance for the choice of models in historical manuscript script classification, emphasizing the importance of architectural considerations in achieving optimal performance.

4.5 Final system analysis

Table 8 presents the final experiment results for the CNN model, specifically DenseNet-121, in both patch-level and document-level classifications as the confusion matrix in Figure 10 demonstrates. At the patch level, the model achieves a commendable accuracy of 83.66%, indicating its proficiency in accurately classifying individual patches within historical manuscripts. Moving to the document level, where a majority vote mechanism is employed, the accuracy improves significantly to 89.2%. This implies that when aggregating predictions across multiple patches from the same document, the model attains a higher accuracy in determining the overall script classification for the entire document. The robust performance at both patch and document levels underscores the effectiveness of the DenseNet-121 architecture in capturing intricate patterns and features in historical manuscripts. These results suggest the potential practical utility of the developed system in automating historical manuscript script classification tasks with a high level of accuracy.

Our method for script classification from historical manuscripts performed well due to a combination of factors.

1. We utilized preprocessing techniques such as denoising and binarization to obtain clean data.
2. We extracted meaningful features from the handwritten style by clustering keypoints using k -means.
3. We trained patches on a deep learning model with transfer learning.
4. We adopted a majority voting mechanism for the definitive script classification of an entire document.
5. We experimented with different patch sizes, numbers of clusters and pretrained models and found optimal values for each hyperparameter.

4.6 Comparison

The results presented in Table 9 reveal the remarkable success of the proposed historical manuscript script classification method, integrating a convolutional neural network (CNN) and an intelligent feature selection technique, achieving an outstanding accuracy of 89.2%. This not only marks a discernible divergence from, but also a definitive enhancement over, various entrenched methodologies in the historical script classification domain up until this study. When considering exemplary benchmarks, it notably eclipses the performance metrics of NNML at 83.80%, FCDR-OCR at 79.80% and DeepScript at 76.79%. Additionally, it outshines FAU at 83.90%, showcasing a substantial progression vis-à-vis The North LTU at 73.96%, T-DeepCNN (85.20%), P-CNN (73.25%), CK2 (78.05%); PERO 4 at 88.77%, decisively outperforming S2-CCMV (56.55%). This comparison showcases not only the quantitative advantage but also highlights the significance of the intelligent feature selection method, emphasizing its crucial role in the success of our model.

Table 9. Comparison with prior studies.

Method	Accuracy
Proposed method (CNN)	89.2
NNML (Cloppet et al., 2016)	83.80
FCDR-OCR (Cloppet et al., 2016)	79.80
DeepScript (Cloppet et al., 2016)	76.79
FAU (Cloppet et al., 2016)	83.90
S2-CCMV (Cloppet et al., 2017)	56.55
T-DeepCNN (Cloppet et al., 2017)	85.20
P-CNN (Cloppet et al., 2017)	73.25
CK2 (Cloppet et al., 2017)	78.05
PERO 4 (Seuret et al., 2021)	88.77
The North LTU (Seuret et al., 2021)	73.96

Our research makes significant contributions to the field of script classification by addressing key limitations in existing methodologies and making use of a novel approach that combines traditional methods with deep learning techniques. One of the primary contributions lies in the development of an intelligent feature selection framework that extracts relevant features crucial for script classification tasks. This framework overcomes the drawbacks of traditional methods, such as limited adaptability and accuracy, by integrating advanced feature selection algorithms. These algorithms identify significant patterns and structures within the script data, enhancing the ability of the deep learning model to learn and generalize effectively. Additionally, our work introduces a comprehensive analysis of script characteristics, including curves, corners and other intricate details, further refining the feature extraction process. By employing deep learning models that learn from these intelligently selected features, we achieve notable improvements in classification accuracy and robustness, especially when dealing with diverse script styles and variations. Our contributions extend beyond mere algorithmic advancements; they pave the way for more sophisticated and efficient approaches to script classification, with implications for various applications in document analysis, historical manuscript interpretation and automated language processing systems.

5 Conclusion

In summary, our research offers a thorough investigation into the classification of scripts in historical manuscripts, integrating diverse methodologies and harnessing the capabilities of deep learning techniques with an intelligent feature selection approach. Utilizing the CLaMM dataset, our process began with document preprocessing, involving non-local means denoising for noise reduction and Canny edge for binarization. Feature detection was then carried out using the Harris corner detector on the binarized images, followed by keypoint clustering using the k -means algorithm to extract uniformly sized, meaningful patches. The dataset underwent training using a pre-trained CNN model. Ultimately, for the definitive script classification of an entire document, we adopted a majority voting mechanism. Through systematic experiments, we optimized critical parameters, including patch size, number of clusters and model selection, resulting in a refined system exhibiting superior performance and underscoring the effectiveness of CNNs in the context of script classification for historical manuscripts.

In our future endeavours, we aspire to enhance the versatility of our system, making it adaptable to a wide range of datasets. Our aim is to develop an end-to-end software solution capable of effectively handling diverse types of historical manuscripts. By doing so, we hope to provide a user-friendly and robust tool for historians and researchers working with varying collections of historical documents. This endeavour represents a commitment to advancing the field of historical manuscript analysis by underlining the significance of deep learning methodologies and the meticulous fine-tuning of parameters. Through these ongoing efforts, we seek to contribute to the continued evolution and refinement of techniques for achieving precise script classification in historical manuscript studies.

Additional Information and Declarations

Funding: This work was funded by the Deputyship for Research & Innovation, Ministry of education in Saudi Arabia, Grant No. 445-9-918.

Conflict of Interests: The authors declare no conflict of interest





Author Contributions: M.B.: Writing, Methodology, Software. A.B.: Supervision, Conceptualization. T.M.: Supervision. A.A.: Visualization. R.R.M.: Data curation. M.A.: Reviewing and Editing A.D.: Reviewing and Editing.

Data Availability: The data used in this study are openly available in Zenodo at <https://zenodo.org/records/5527690>.

References

- Arandjelovic, R., & Zisserman, A. (2012). Three things everyone should know to improve object retrieval. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 2911–2918). IEEE. <https://doi.org/10.1109/cvpr.2012.6248018>
- Bennour, A., Boudraa, M., Siddiqi, I., Al-Sarem, M., Al-Shabi, M., & Ghabban, F. (2024). A deep learning framework for historical manuscripts writer identification using data-driven features. *Multimedia Tools and Applications*, (in press). <https://doi.org/10.1007/s11042-024-18187-y>
- Bennour, A., Djeddi, C., Gattal, A., Siddiqi, I., & Mekhaznia, T. (2019). Handwriting based writer recognition using implicit shape codebook. *Forensic Science International*, 301, 91–100. <https://doi.org/10.1016/j.forsciint.2019.05.014>
- Boudraa, M., Bennour, A., Al-Sarem, M., Ghabban, F., & Bakhsh, O. A. (2024). Contribution to Historical Manuscript Dating: A Hybrid Approach Employing Hand-Crafted Features with Vision Transformers. *Digital Signal Processing*, 149, 104477. <https://doi.org/10.1016/j.dsp.2024.104477>
- Boudraa, M., & Bennour, A. (2023). Combination of local features and deep learning to historical manuscripts dating. In *Intelligent Systems and Pattern Recognition – ISPR 2023*, (pp. 129–143). Springer. https://doi.org/10.1007/978-3-031-46335-8_11
- Buades, A., Coll, B., & Morel, J. (2011). Non-Local means denoising. *Image Processing Online*, 1, 208–212. https://doi.org/10.5201/ipol.2011.bcm_nlm
- Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 679–698. <https://doi.org/10.1109/tpami.1986.4767851>

- Christlein, V., Bernecker, D., & Angelopoulou, E. (2015). Writer identification using VLAD encoded contour-Zernike moments. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)* (pp. 906–910). IEEE. <https://doi.org/10.1109/ICDAR.2015.7333893>
- Christlein, V., Bernecker, D., Hönig, F., Maier, A., & Angelopoulou, E. (2017). Writer identification using GMM Supervectors and Exemplar-SVMs. *Pattern Recognition*, 63, 258–267. <https://doi.org/10.1016/j.patcog.2016.10.005>
- Cloppet, F., Eglin, V., Kieu, V.C., Stutzmann, D., & Vincent, N. (2016). ICFHR2016 Competition on the classification of medieval handwritings in latin script. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE. <https://doi.org/10.1109/ICFHR.2016.0113>
- Cloppet, F., Eglin, V., Helias-Baron, M., Kieu, C., Vincent, N., & Stutzmann, D. (2017). ICDAR2017 Competition on the Classification of Medieval Handwritings in Latin Script. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, (pp. 1371–1376). IEEE. <https://doi.org/10.1109/ICDAR.2017.224>
- Dehak, N., Torres-Carrasquillo, P. A., Reynolds, D. A., & Dehak, R. (2011). Language recognition via i-Vectors and dimensionality reduction. In *Interspeech 2011*, (pp. 857–860). ISCA.
- Demilew, F. A., & Sekeroglu, B. (2019). Ancient Geez script recognition using deep learning. *SN Applied Sciences*, 1(11). <https://doi.org/10.1007/s42452-019-1340-4>
- El Bahi, H., & Zatni, A. (2019). Text recognition in document images obtained by a smartphone based on deep convolutional and recurrent neural network. *Multimedia Tools and Applications*, 78, 26453–26481. <https://doi.org/10.1007/s11042-019-07855-z>
- Fischer, A., Indermühle, E., Bunke, H., Viehhauser, G., & Stolz, M. (2010). Ground truth creation for handwriting recognition in historical documents. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems* (pp. 3–10). ACM. <https://doi.org/10.1145/1815330.1815331>
- Harris, C., & Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference* (pp. 147–152). BMVA. <https://doi.org/10.5244/C.2.23>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 770–778). IEEE. <https://doi.org/10.1109/cvpr.2016.90>
- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2261–2269). IEEE. <https://doi.org/10.1109/cvpr.2017.243>
- Jin, X., & Han, J. (2011). K-Means Clustering. In Sammut, C., Webb, G.I. (eds) *Encyclopedia of Machine Learning*. Springer. https://doi.org/10.1007/978-1-4899-7687-1_431
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Lombardi, F., & Marinai, S. (2020). Deep Learning for Historical Document Analysis and Recognition—A survey. *Journal of Imaging*, 6(10), 110. <https://doi.org/10.3390/jimaging6100110>
- Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2), 91–110. <https://doi.org/10.1023/b:visi.0000029664.99615.94>
- Meraoumia, A., Bendjenna, H., Amroune, M., & Dris, Y. (2018). Towards a Secure Online E-voting Protocol Based on Palmprint Features. In *2018 3rd International Conference on Pattern Analysis and Intelligent Systems*. IEEE. <https://doi.org/10.1109/pais.2018.8598520>
- Paris, S., Kornprobst, P., Tumblin, J., & Durand, F. (2009). Bilateral Filtering: Theory and applications. *Foundations and Trends in Computer Graphics and Vision*, 4(1), 1–75. <https://doi.org/10.1561/06000000020>
- Parkes, M. B. (2016). *Pause and effect: An introduction to the history of punctuation in the West*. Routledge.
- Peake, G. S., & Tan, T. N. (1997). Script and language identification from document images. In *Proceedings Workshop on Document Image Analysis (DIA'97)*, (pp. 10–17). IEEE. <https://doi.org/10.1109/DIA.1997.627086>
- Rosten, E., Porter, R., & Drummond, T. (2010). Faster and Better: A machine learning approach to corner detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1), 105–119. <https://doi.org/10.1109/tpami.2008.275>
- Samai, D., Meraoumia, A., Bendjenna, H., & Laimeche, L. (2017). Oriented Local Binary Pattern (LBP_θ): A new scheme for an efficient feature extraction technique. In *International Conference on Mathematics and Information Technology (ICMIT)*. IEEE. <https://doi.org/10.1109/MATHIT.2017.8259710>
- Seuret, M., Nicolaou, A., Rodríguez-Salas, D., Weichselbaumer, N., Stutzmann, D., Mayr, M., Maier, A., & Christlein, V. (2021). ICDAR 2021 Competition on Historical Document Classification. In *Document Analysis and Recognition – ICDAR 2021* (pp. 618–634). Springer. https://doi.org/10.1007/978-3-030-86337-1_41
- Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR 2015)*. ICLR.
- Singh, A. K., Mishra, A., Dabral, P., & Jawahar, C. V. (2016). A simple and effective solution for script identification in the wild. In *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, (pp. 428–433). IEEE. <https://doi.org/10.1109/DAS.2016.57>
- Stutzmann, D. (2016). Clustering of medieval scripts through computer image analysis: Towards an evaluation protocol. *Digital Medievalist*, 10. <https://doi.org/10.16995/dm.61>
- Tan, T. (1998). Rotation invariant texture features and their use in automatic script identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(7), 751–756. <https://doi.org/10.1109/34.689305>

Editorial record: The article has been peer-reviewed. First submission received on 5 February 2024. Revisions received on 9 April 2024, 14 May 2024 and 30 May 2024. Accepted for publication on 3 June 2024. The editors coordinating the peer-review of this manuscript were Hakim Bendjenna , Lawrence Chung , Abdallah Meraoumia , and Zdenek Smutny . The editor in charge of approving this manuscript for publication was Zdenek Smutny.

Special Issue: Future Trends of Machine Intelligence in Science and Industry. Selected papers from the National Conference on Artificial Intelligence: From Theory to Practice (NCAI'2023).

Acta Informatica Pragensia is published by Prague University of Economics and Business, Czech Republic.

ISSN: 1805-4951
