JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 34 (2025) 98–107
Received: 2 November 2024
Accepted: 8 December 2024

Optical character recognition for ancient scripts: A case study on Syloti Nagri using deep learning models

Optyczne rozpoznawanie znaków w starożytnych pismach: studium przypadku Syloti Nagri z wykorzystaniem modeli głębokiego uczenia

Tanzidul Islam[a], Sheikh Kamrul Hasan Omur[a], Nafiz Nahid[a], Lukman Chowdhury[a], Gourab Roy[a], Md. Abu Naser Mojumder[a*], Md. Janibul Alam Soeb[b], Md. Fahad Jubayer[c*]

[a]*Department of Computer Science and Engineering, Sylhet Engineering College, Sylhet, Bangladesh*
[b]*Department of Farm Power and Machinery, Sylhet Agricultural University, Sylhet, Bangladesh*
[c]*Department of Food Engineering and Technology, Sylhet Agricultural University, Sylhet, Bangladesh*

Abstract

This study proposes a novel OCR system for the Syloti Nagri script, addressing the complexities of segmenting and classifying its unique characters. Given the limited annotated data and intricate structures, a tri-level segmentation approach was applied, involving line, word, and character segmentation. Deep learning models-VGG16, VGG-19, ResNet-50, MobileNet-v3, and Xception, were evaluated, with VGG-19 achieving the highest scores in accuracy, precision, recall, and F1 metrics. The segmentation process attained 98% accuracy for word and 94% for character segmentation, while the classification model reached a test accuracy of 99.78%, demonstrating robust recognition of complex patterns. This work advances OCR technology and supports the preservation and accessibility of the Nagri script.

Keywords: Text recognition; OCR; VGG-19; feature extraction

Streszczenie

W pracy zaproponowano nowatorski system OCR dla pisma Syloti Nagri, uwzględniający złożoność segmentacji i klasyfikacji jego unikalnych znaków. Biorąc pod uwagę ograniczoną liczbę danych z adnotacjami i skomplikowane struktury, zastosowano trójpoziomowe podejście do segmentacji, obejmujące segmentację linii, słów i znaków. Oceniono modele głębokiego uczenia – VGG16, VGG-19, ResNet-50, MobileNet-v3 i Xception, przy czym VGG-19 uzyskał najwyższe wyniki w zakresie dokładności, precyzji, zapamiętywania i wskaźników F1. Proces segmentacji osiągnął 98% dokładności w przypadku słów i 94% w przypadku segmentacji znaków, podczas gdy model klasyfikacji osiągnął dokładność testową na poziomie 99,78%, co świadczy o solidnym rozpoznawaniu złożonych wzorców. Ta praca w znaczący sposób rozwija technologię OCR i wspiera dostępność pisma Nagri.

Słowa kluczowe: Rozpoznawanie tekstu; OCR; VGG-19; ekstrakcja cech

[*]Corresponding author(s)

Email address: abu.naser84@gmail.com (M.A.N. Mojumder), fahadbau21@hotmail.com (M.F. Jubayer)

1. Introduction

Optical character recognition (OCR) is an essential technology for automating the conversion of printed and handwritten text into machine-readable formats, facilitating numerous applications in document digitization, information retrieval, and linguistic analysis [1]. OCR plays a crucial role in the digitization of handwritten ancient manuscripts, as well as in the conversion of typewritten materials into digital formats [2]. The implementation of this technology has facilitated the process of accessing necessary information, eliminating the need to sift through extensive collections of documents and files in order to locate the desired information [3]. Developing an OCR system for a particular language necessitates the identification and classification of its characters, which demands the use of language-specific OCR models. The absence of electronic transcriptions for text graphics in different languages poses a significant obstacle to their immediate conversion. Hence, efficient OCR technologies serve to close this disparity [4].

The digitization of ancient scripts has emerged as a critical area of research within the broader field of OCR. Among these, the ‘Nagri’ script, an ancient and historically significant language, presents unique challenges due to its complex character structures and limited available resources. The Syloti Nagri script, alternatively referred to as Sylheti Nagri or Sylheti Nagari, is an ancient Indic script belonging to the Brahmic family. It was historically employed in the eastern regions of Bengal and Assam, specifically in the eastern zone of the Sylhet region, situated east of the Padma River. The script played a crucial role in the documentation of Muslim religious poetry, commonly referred to as puthis. Notwithstanding its historical importance, Sylheti Nagri did not possess a formal presence in the documents and has gradually been supplanted by the standardized Eastern Nagari script over the 20th century [5]. OCR systems have shown considerable progress in recognizing widely used scripts, yet ancient scripts like Nagri remain underexplored, largely due to the scarcity of annotated data and the script's intricate glyphs.

The Nagri script, once widely used in historical documents across various regions, now faces the threat of becoming obsolete. Its preservation and accessibility through digital means are crucial for historical research and cultural heritage [5]. However, the development of OCR systems for Nagri is fraught with challenges, including the segmentation of complex characters, handling of noise in aged manuscripts, and the accurate classification of similar-looking glyphs. These challenges are compounded by the limited availability of digitized resources and annotated datasets specific to Nagri. Nagri characters resemble both Bangla and Hindi characters. Therefore, in this present research we looked for character recognition systems that were customized to these related languages, which helped us make our decision. Previous research in OCR has predominantly focused on more commonly used scripts such as Latin, Arabic, and Devanagari, with methods ranging from traditional pattern recognition techniques to more recent deep learning approaches [6,7].

Both feature extraction and classification play a critical role in OCR systems, significantly impacting the total accuracy. Previous studies have investigated several techniques, including Euclidean distance, HOG, Fourier descriptor, and directional chain code, for the purpose of feature extraction. Additionally, classification models such as CNN, KNN, ANN, and SVM have been employed in these investigations [8]. The recognition of Sanskrit characters poses unique challenges as a result of the intricate nature of the script and the similarities among symbols [9]. Recent developments in OCR technology have involved the incorporation of machine learning and artificial intelligence (AI) techniques, specifically aimed at comprehending ancient scripts and transforming them into contemporary languages [8]. Deep learning architectures, such as CNN-Bidirectional LSTM, have demonstrated promise for detecting Sanskrit manuscripts [9]. The advancement of OCR systems customized for Indian scripts holds considerable implications in the conservation of historical manuscripts and their subsequent accessibility via digital libraries [10]. Using CNN architectures such as ResNet and VGG, researchers achieved a recognition accuracy of 97.009% on ancient character datasets, highlighting the potential of deep learning for digitizing historical manuscripts [11]. Various approaches, such as Euclidean distance and HOG, are used to improve classification accuracy, especially for ancient scripts [8].

Despite the success in these domains, applying these techniques to Nagri is not straightforward due to the script's distinct morphological characteristics and the lack of standardized orthography in historical texts. Therefore, there is a pressing need for a specialized approach that can effectively segment and classify Nagri characters, taking into account the script's unique attributes. This paper proposes a novel segmentation and classification approach tailored to the specific requirements of Nagri OCR. This study aims to address the existing gap by proposing a comprehensive approach to Nagri OCR that includes both segmentation and classification of Nagri text images. Utilizing the dataset, we employ a tri-level segmentation method—line, word, and character segmentation—followed by classification using deep learning-based Convolutional Neural Network (CNN) models. By integrating these models, we seek to create a functional OCR system that can accurately convert Nagri text into electronic formats, thereby contributing to the digitization and preservation of the Nagri language. This study not only contributes to the field of OCR by addressing the challenges associated with Nagri but also plays a pivotal role in the preservation of a valuable cultural asset. To guide this research, we hypothesize that among the evaluated deep learning models (VGG-16, VGG-19, ResNet-50, MobileNet-v3, and Xception), VGG-19 will achieve the highest classification accuracy, precision, recall, and F1-score. This is based on prior studies highlighting the effectiveness of deeper architectures like VGG-19 in handling complex patterns of ancient scripts. The study aims to validate this by systematically comparing these models on a custom Nagri script dataset.

2. Materials and methods

2.1. Dataset

The Nagri script consists of 32 letters, including 5 vowels and 27 consonants. It also incorporates a variety of compound characters. The Sylheti Nagri script supports Unicode, which allows for the digital representation of consonants, vowels, and vowel marks. However, it is important to note that compound characters currently lack a Unicode equivalent. The absence of a Unicode standard for these compound characters presents an opportunity for future advancements and research in the digital representation of Sylheti Nagri. The Nagri alphabet is illustrated in Figure 1.

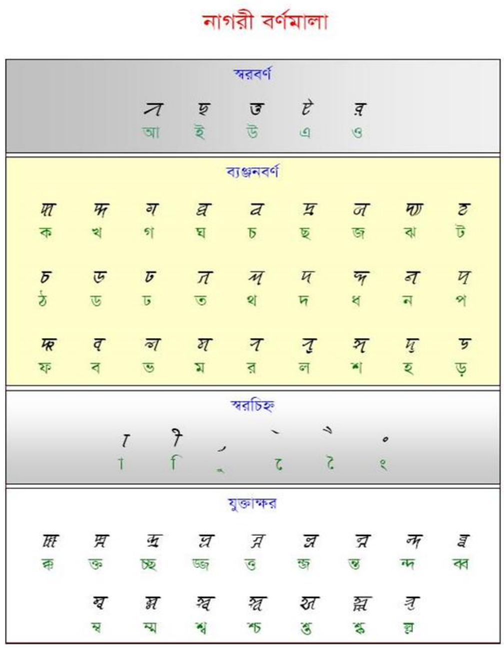

Figure 1: The Nagri alphabet: An illustration of 32 letters including vowels and consonants.

Accessing open-source datasets related to print Nagri materials is challenging. Therefore, the development of our Nagri OCR system relied on a custom dataset, "NagriOCRbd" (Figure 2), which comprises 1,486 printed text images of Nagri. This dataset includes pages with printed Nagri text alongside their corresponding Bangla translations. To prepare the dataset for OCR, we conducted extensive pre-processing steps, including noise removal, skew correction, and digitization. A tri-level segmentation process: line segmentation, word segmentation, and character segmentation was then employed. For line segmentation, we utilized connected component (CC) analysis, Hough transform, filling, and smoothing techniques to identify word components within the text. After successfully segmenting the lines, we performed word segmentation, creating a word dataset from the "NagriOCRbd" dataset. This step facilitated the separation of characters for model training. The resulting character dataset contains 18,411 segmented words and 58,343 segmented characters, organized into 54 distinct categories, with a total of 11,473 individual characters. This meticulous organization and categorization of segmented characters into distinct folders provided a solid foundation for developing our Nagri OCR system. The classification into 54 unique categories is predicated on individual characters (5 vowels and 27 consonants), compound characters, and special symbols and signs. These are crucial for precisely replicating the script and preserving linguistic integrity during digitization. By categorizing the dataset into 54 classifications, we guarantee that all unique components of the script are included. This classification facilitates efficient segmentation and categorization during model training, ensuring the OCR system can accommodate the extensive diversity intrinsic to the script.

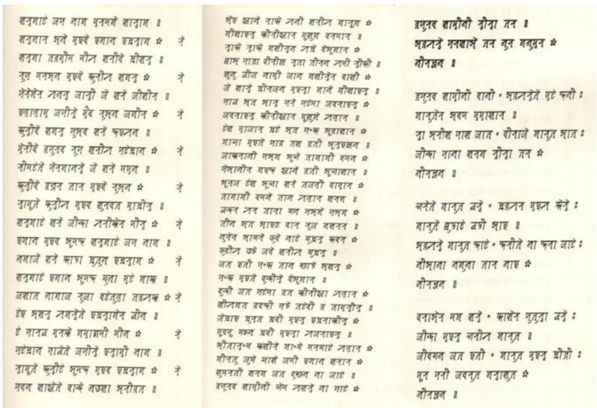

Figure 2: Sample images from the NagriOCRbd dataset: printed Nagri text.

2.2. Overview of implementation stages

The process for developing an OCR system for the Nagri script is illustrated in Figure 3. This hybrid approach consists of two primary components: character segmentation and model training. The first phase of our methodology involves the segmentation of individual characters. We begin with word segmentation on text document images, which is then followed by character segmentation. This process ensures that each character is accurately identified and isolated for subsequent processing. In the next stage, we employ a deep learning-based convolutional neural network (CNN) model. Transfer learning techniques are applied to enhance the model's performance and efficiency. Each stage of our methodology is carefully designed to maximize precision and reliability in character recognition applications.

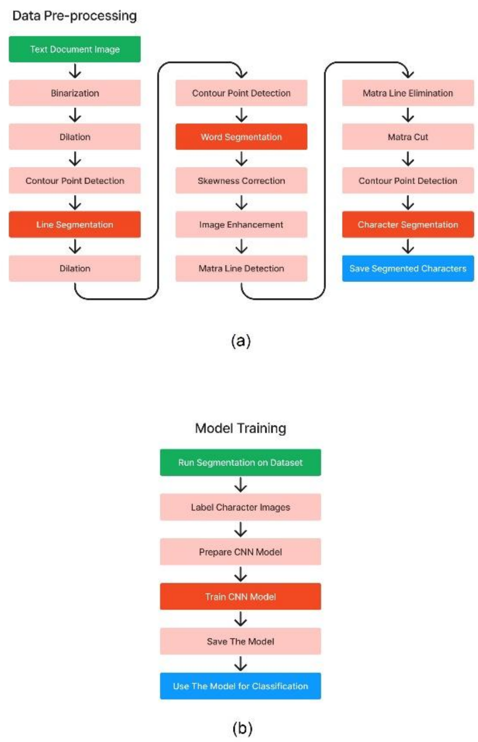

Figure 3: Implementation stages of the Nagri OCR system: character segmentation and model training process. (a) Data pre-processing, (b) Model training.

2.3. Data pre-processing

Effective OCR systems require precise segmentation of text into recognizable components. Our proposed methodology involves a sequential segmentation process that operates at the line, word, and character levels, with each phase subdivided into specific steps. Initially, the images are converted to binary format to clearly distinguish the text from the background. The processed images are then digitized to create a uniform dataset for further analysis.

2.3.1. Segmentation process

Line detection is facilitated through the application of the dilation technique, which aims to expand and connect text components. We utilize a large kernel size of (3, 85) of ones for dilation, ensuring the creation of new pixels that bridge adjacent words and maintain text continuity. After detecting contour points for each line, they are sorted based on their y-coordinates. Subsequently, bounding boxes are drawn to visually confirm the accuracy of the detected lines. It's important to note that the contour points are used solely for visualization purposes and are not employed for slicing or further processing. Figure 4a represents the steps of line detection. Words within each line are identified by analyzing the contour lines, using a dilation operation with a small kernel to accurately determine the boundaries of each word. The segmented words are then saved as individual images, forming a word dataset (Figure 4b).

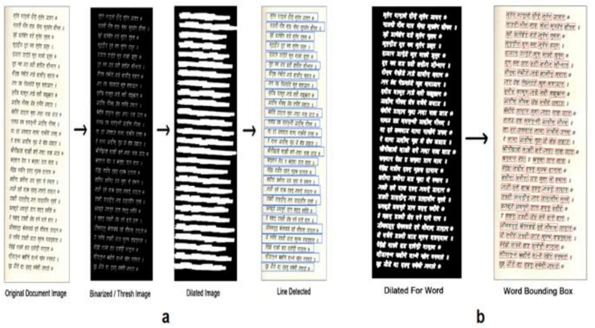

Figure 4: Steps in the segmentation process: (a) line detection and (b) word segmentation.

2.3.2. Image processing

Characters within each word are detected by enhancing the word images using Gaussian blur, followed by contour point detection. The Gaussian blur is applied with a kernel size of (3, 3) (as detailed in SM1). At this stage, skew correction is applied once again to ensure horizontal alignment, and a specialized technique is employed to eliminate matra lines (horizontal lines above characters) before the final character segmentation. We use the Canny edge detection and Hough line transform techniques for this purpose. By testing angles ranging from 0.1 to 180 degrees, the Hough line transform identifies pixels forming straight lines within the image, as shown in the supplementary material (SM1). Subsequently, lines with the highest pixel content are identified. These lines undergo angle testing to correct the skew in the word image, aiming to maximize the row histogram. The image is then skew-corrected based on the angle corresponding to the maximum row count. Once the skew correction is complete, we remove the matra line by determining the exact coordinates of the maximum row and applying a threshold value—in this case, 3. Pixels within the range from max_row_index − threshold to max_row_index + threshold are rendered white, effectively removing the entire matra from the word images. Following matra removal, contour point detection is used to identify the position of each character devoid of matras. A specific point for each matra, located just above the bounding box of a character without matra, is approximated. Based on these marked cut points, a strategy is implemented to separate characters from matras by removing a fixed length of pixels, typically 2 pixels, from the actual word image with matra. At the final stage, the word image undergoes matra cutting, resulting in individual characters each connected to their respective matras while being disconnected from one another. Finally, contour point detection is performed once more on the matra-cut word image, combined with binarization and dilation using a small kernel. As a result, each character becomes encapsulated within a bounding box. Using the coordinates of these bounding boxes, the individual characters are accurately sliced and saved for further processing. Initially, we executed the code on our dataset, saving each character separately for manual segmentation, as shown in Figure 5, to train our classification model. However, in real-world scenarios, we plan to feed the characters directly into our classification model for automatic classification, thereby generating their corresponding electronic forms.

Figure 5: Character segmentation process: From Gaussian blur to final Matra-cut character extraction.

2.4. Model training

For the classification task, we employed several CNN-based models: VGG-16, VGG-19, ResNet-50, MobileNet-v3, and Xception. The dataset was divided into three subsets: training (75%), validation (10%), and testing (15%). The VGG-19 model consists of 19 layers, organized sequentially with convolutional layers followed by max-pooling layers, as illustrated in Figure 6. The final output layer was customized to match the number of Nagri character classes in our dataset.

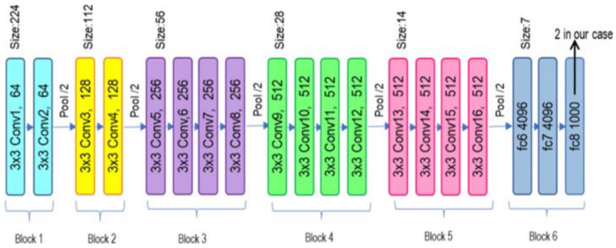

Figure 6: Architecture of the VGG-19 model: Sequential arrangement of convolutional and max-pooling layers.

We utilized an image size of 224 × 224 pixels for training, with "Adam" as the optimizer and "Sparse Categorical Cross Entropy" as the loss function. The final output layer was tailored to align with our specific classification needs, accommodating the required number of classes. Figure 7 shows the distribution of class counts across the training, validation, and test sets. After splitting, the training set contains 8,588 images, the testing set has 1,715 images, and the validation set includes 1,170 images.

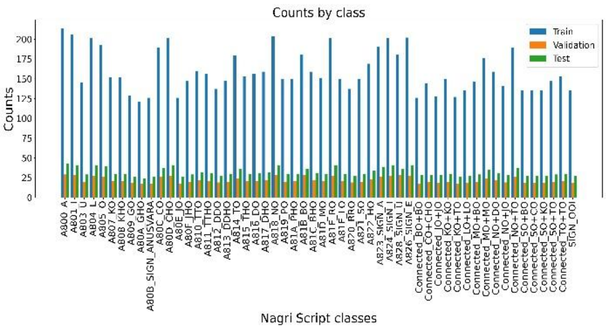

Figure 7: Distribution of class counts across training, validation, and test sets.

The models were trained using specific parameters, including batch size, learning rate, and the number of epochs, as detailed in Table 1. The selected image size of 224 × 224 pixels was consistently applied across all models during training.

Table 1: Training parameters for the CNN models

Model	Epoch	Batch size	Learning rate
VGG-16	7	32	0.0001
VGG-19	14	64	0.00001
ResNet-50	9	64	0.0001
MobileNet-v3	10	128	0.001
Xception	10	128	0.0001

2.5. Model pipelining

To integrate the segmentation and classification models into a cohesive OCR system, we employed a pipelining approach. In this system, the segmentation model first processes the entire document image, segmenting it into individual characters. These segmented characters are then fed into the classification model, which assigns the appropriate electronic form to each character based on the trained CNN model. The final output is a digitized version of the Nagri text, which can be used for various applications in document digitization and language preservation. Figure 8 provides an overview of the system architecture after deployment.

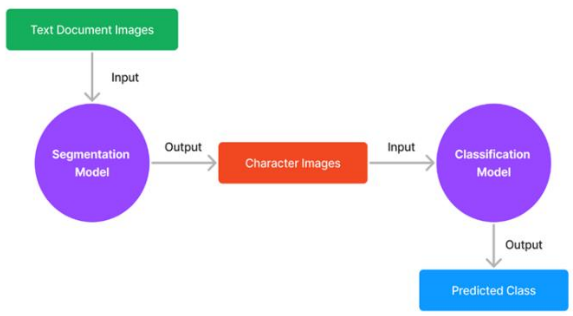

Figure 8: System architecture of the OCR pipeline: Integration of segmentation and classification models.

2.6. Evaluation metrics

In character segmentation, established algorithms for measuring accuracy are not readily available. To address this, we devised a technique to roughly measure the accuracy of our segmentation method. We randomly selected 2 document images and compared the number of words identified by our model against the actual count. The word segmentation accuracy was determined using equation 1. Similarly, we randomly selected 10-word images and assessed the number of distinct characters present, comparing our segmentation output with the actual count. The character segmentation accuracy was determined using equation 2. This methodology allowed us to evaluate the accuracy of our segmentation technique, providing insights into its performance in segmenting words and characters within documents.

$$Word\ segmentation\ accuracy = Segmented\ word\ count / Actual\ word\ count \tag{1}$$

Similarly, we randomly selected 10-word images and assessed the number of distinct characters present, comparing our segmentation output with the actual count. The character segmentation accuracy was determined using the formula.

$$Character\ segmentation\ accuracy = Segmented\ character\ count / Actual\ character\ count \tag{2}$$

Key statistical metrics are computed using equation 3 – equation 6 [12,13].

$$Precision = \frac{TP}{(TP+FP)} \tag{3}$$

$$Recall = \frac{TP}{(TP+TN)} \tag{4}$$

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{5}$$

$$F1-score = \frac{2}{\left(\frac{1}{precision}\right)+\left(\frac{1}{recall}\right)} \tag{6}$$

Here, TP = true positive, FP = false positive, TN = true negative, and FN = false negative.

3. Experimental results and analysis

3.1. Experimental environment

The model was created in the research lab server of Sylhet Engineering College, Bangladesh, compatible with a Windows server 2019 system. Python 3.10.14, Tensorflow 2.10 and PyTorch 1.11 were employed for the development. The experiment was executed using a single GPU NVIDIA RTX A4000 with 16 GB of memory and CPU RAM 64 GB.

3.2. Segmentation and model evaluation

Character segmentation, particularly for ancient and complex scripts like Nagri, presents unique challenges due to the absence of standardized evaluation metrics, especially for unsupervised segmentation techniques. To rigorously assess the performance of our segmentation model, we developed a tailored evaluation methodology. For word segmentation, two randomly selected document images were used to measure accuracy by comparing the number of segmented words to the actual word count.

Our model achieved an impressive word segmentation accuracy of 98%, successfully segmenting 262 words out of 266 actual words. Character segmentation was evaluated using 50 randomly selected word images, where the segmented characters were compared against the actual character count. The character segmentation process demonstrated an accuracy of 94%, accurately segmenting 358 characters out of 337 actual characters. The present performance is consistent with the results obtained from other research that emphasize the significance of employing efficient segmentation approaches. In the Devanagari script, the inclusion of modified and conjunct characters introduces complexities in the process of segmentation. The accuracy of segmentation might vary between 91.84% and 99.11%, contingent upon the specific technique employed [14]. In the study of Jindal and Ghosh [15], their proposed method segmented ancient Devanagari documents with 96.31% and 98.35% word and character accuracy. In Maithili script, the suggested method achieved 97.39% and 98.65% accuracy for word and character segmentation in ancient handwritten manuscripts.

3.3. Classification model performance

The classification of Nagri characters was achieved using five CNN-based models: VGG-16, VGG-19, ResNet-50, MobileNet-v3, and Xception. Among these, the VGG-19 model demonstrated the best performance across various metrics, including accuracy, precision, recall, and F1-score. Figure 9 presents the accuracy curves for the different deep learning models applied to the Nagri script classification task. The accuracy curves reflect how well each model performs over the training epochs. Notably, the VGG-19 model exhibits a consistently high accuracy, converging towards 99.78% by the end of the training period, outperforming the other models. This indicates its superior capability in recognizing and classifying the intricate characters of the Nagri script.

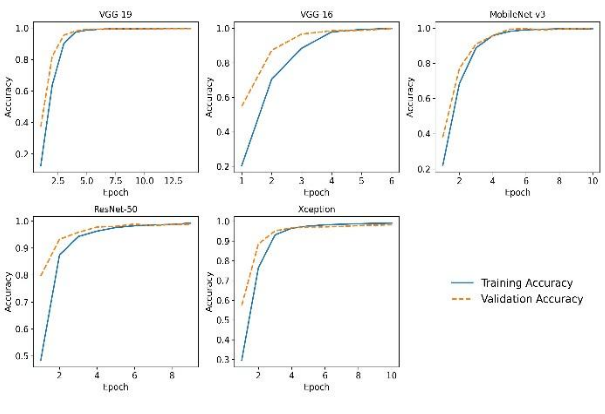

Figure 9: Comparison of model accuracy curves for Nagri script classification.

Figure 10 illustrates the loss curves for the same set of models, showing the reduction in loss (error) as the training progresses. The model's loss during training was minimal, indicating robust learning and generalization capabilities. Together with its remarkable accuracy, the model's minimized loss of 0.03724 demonstrated how well it could learn from and generalize the training set of data. The VGG-19 model demonstrates the most rapid decrease in loss, stabilizing at a lower value compared to the other models. This suggests that VGG-19 not only achieves higher accuracy but also learns more efficiently, minimizing errors effectively as it trains.

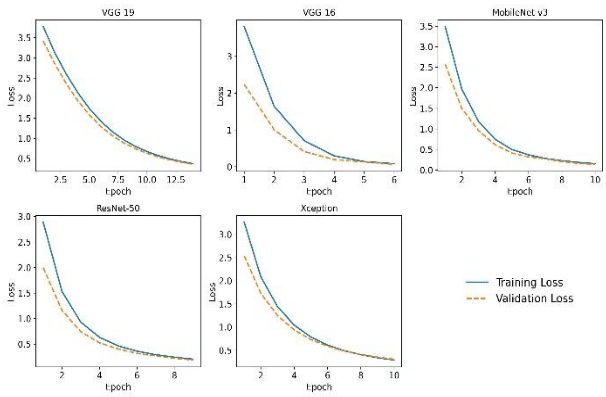

Figure 10: Comparison of model loss curves for Nagri script classification.

The VGG19 model demonstrates excellent performance in deepfake detection, surpassing both VGG16 and ResNet50 models in terms of accuracy, obtaining a robust 98% accuracy rate [16]. In the present study, its superior accuracy curve indicates that it learns the distinguishing features of the Nagri script more effectively than other models. This is particularly important given the complex nature of the script, which includes numerous intricate glyphs and character variations. Fig. 10 further supports these findings by demonstrating that the VGG-19 model not only achieves high accuracy but also does so efficiently, with a steep and steady decline in the loss curve. The VGG-19 model routinely outperforms other architectures in various classification tasks. For insect classification, the VGG-19 model demonstrated an accuracy of 97.07%, which exceeded the accuracy of 96.28% recorded by VGG-16 [17]. The VGG-19 model, when combined with feature selection techniques, achieved an accuracy of 98.34% in the identification of diabetic retinopathy [18]. The VGG-19 model exhibited exceptional performance, attaining an accuracy rate of 94.12% when applied to previously unknown data [19]. These results demonstrate that the deeper architecture of VGG-19 enables it to effectively capture intricate information, leading to enhanced classification accuracy across a wide range of domains. Despite the success of the VGG-19 model, the other models, such as ResNet-50 and Xception, also demonstrated competitive performance, with reasonably high accuracy and low loss values. However, their performance did not match the levels achieved by VGG-19, which may be due to differences in their architectural design, such as the number of layers and the type of operations performed within each layer. The VGG-19 architecture facilitates more intricate feature extraction in comparison to ResNet-50. ResNet-50, with its 50 layers, incorporates skip connections that may restrict feature learning in specific scenarios [16].

3.4. Test results

Table 2 presents the performance metrics—accuracy, precision, recall, and F1-score—of five different deep learning models applied to the Nagri script classification task. These outcomes provide valuable insights into the models' effectiveness in generalizing to new instances, essential for assessing their real-world applicability and robustness. Among the models, VGG-19 achieved the highest performance across all metrics, with an accuracy of 99.78%, precision of 99.99%, recall of 99.99%, and an F1-score of 99.99%. This indicates near-perfect classification capabilities for the Nagri script.

Table 2: Performance metrics of different deep learning models for Nagri script classification

Models	Accuracy	Precision	Recall	F1-score
VGG-16	97.47%	97%	97%	97%
VGG-19	99.78%	99.99%	99.99%	99.99%
ResNet-50	98.71%	99%	99%	99%
MobileNet-v3	99.48%	99%	99.99%	99%
Xception	98.43%	98%	98%	98%

The results clearly indicate that the VGG-19 model outperformed the other models in every metric, making it the most effective for classifying Nagri script characters. The high accuracy, precision, recall, and F1-score metrics achieved by VGG-19 underscore its robustness in handling the complex and intricate features of the Nagri script. The near-perfect scores across all metrics reflect the model's capability to accurately distinguish between the subtle differences in character shapes, which is crucial for effective OCR. According to Shaha and Pawar [20], VGG-19 has demonstrated exceptional efficacy in the domain of image classification, surpassing alternative CNN designs and hybrid methodologies. VGG-19 demonstrated superior effectiveness in Tamil character recognition when compared to HOG-based feature extraction [21]. According to Zahoor et al. [22], the VGG-19 model demonstrated a remarkable accuracy rate of 99.03% in the recognition of Pashto ligatures. The application of the Show, Attend, and Read (SAR) approach has demonstrated efficacy in the recognition of Syloti Nagri text, effectively tackling the difficulties presented by complex character features [5]. These findings underscore the resilience of deep learning models in effectively managing intricate scripts. Notably, VGG-19 continually exhibits exceptional levels of accuracy, precision, recall, and F1-scores when applied to diverse Indic and cursive script datasets. MobileNet-v3's performance is also noteworthy, as it provides a balance between accuracy and efficiency, achieving high performance metrics while being computationally less demanding than VGG-19. This makes it a viable alternative in scenarios where computational resources are limited. ResNet-50 and Xception, while demonstrating strong performance, did not reach the same level of accuracy as VGG-19. This could be attributed to the specific architectural designs of these models, which may not be as well-suited for capturing the unique features of the Nagri script as VGG-19. VGG-19 has demonstrated outstanding efficacy in multiple Indic and cursive scripts. VGG-19 attained an exceptional accuracy of 99.03% for Pashto ligatures [22], surpassing previous CNN architectures. In Tamil character recognition, VGG-19 exhibited greater efficacy than HOG-based feature extraction [21]. The Show, Attend, and Read (SAR) methodology has demonstrated efficacy in Syloti Nagri text recognition, tackling difficulties associated with complex character attributes [5]. These findings underscore the resilience of deep learning algorithms in managing intricate scripts. Transfer learning and data augmentation approaches have been effectively utilised to improve recognition rates. For example, DenseNet attained a 99.31% recognition rate for Pashto ligatures with the utilisation of fine-tuned pre-trained models and data augmentation [22].

Figure 11 and Table 3 provide a comprehensive breakdown of the accuracy, precision, recall, and F1-score metrics for individual classes, culminating in an overview of Average, Macro-Average, and Weighted-Average scores. This empirical analysis underscores VGG-19's superior ability to accurately classify data across multiple classes, reaffirming its efficacy in complex classification tasks.

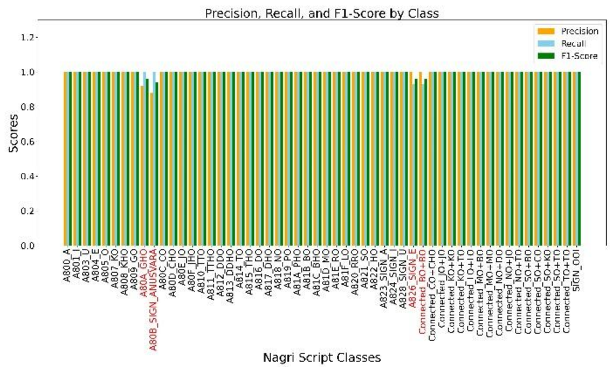

Figure 11: Class-wise performance metrics for Nagri script classification using VGG-19.

Figure 11 complements these results by providing a class-wise breakdown of these metrics, revealing that the model maintains high performance even at the individual character level. This uniformity in performance across classes demonstrates the model's ability to handle the variability and complexity of the Nagri script.

Table 3: Summary of accuracy, macro-average, and weighted-average metrics for Nagri script classification

Class Name	Precision	Recall	F1-score	Support
Accuracy	---	---	99.78	1715
Macro average	99.99	99.99	99.99	1715
Weighted average	99.99	99.99	99.99	1715

The high accuracy of 99.78% reflects the model's robustness in distinguishing between various characters, a crucial feature given the intricate nature of the Nagri script. The macro-average and weighted-average metrics, both at 99.99%, further emphasize the model's balanced performance across all classes. This balance is significant, particularly in OCR systems, where certain classes may have fewer samples or be more challenging to classify due to subtle differences in character shapes. The results from Figure 11 suggest that the VGG-19 model not only excels in aggregate metrics but also performs consistently well at a granular level, accurately classifying individual characters with minimal deviation in performance. This consistent performance across classes is essential for practical OCR applications, ensuring that all characters, regardless of frequency or complexity, are accurately recognized. In Devanagari handwritten character recognition, a fine-tuned Deep Convolutional Neural Network (D-CNN) attained 99.94% training accuracy and 99.57% testing accuracy [23]. A CNN-based OCR system exhibited resilience to picture quality, contrast, font style, and size for Sanskrit manuscripts in Devanagari script, rendering it effective for digitizing inadequately preserved ancient texts [1].

The confusion matrix in Figure 12 offers critical insights into the VGG-19 model's classification capabilities for Nagri script characters. The confusion matrix provides a detailed overview of the model's performance by displaying the number of correct and incorrect predictions for each class [24]. Diagonal elements of the matrix represent the number of instances correctly classified by the model, while off-diagonal elements indicate misclassifications.

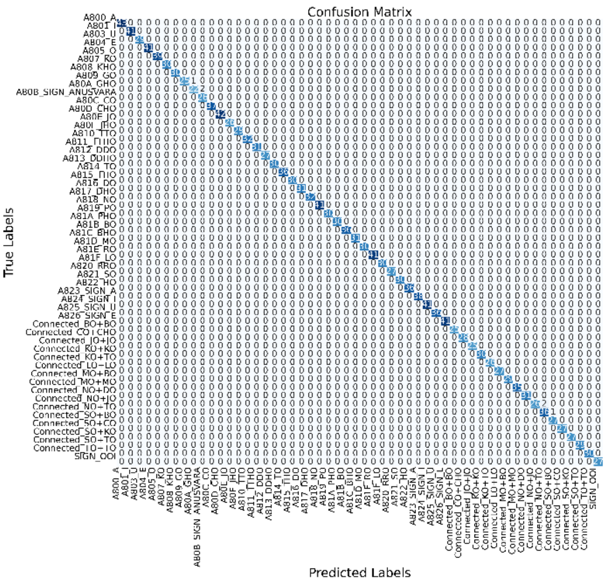

Figure 12: Confusion matrix for Nagri script classification.

The majority of the predictions are concentrated along the diagonal of the matrix, reflecting the model's high accuracy in correctly identifying Nagri script characters. The minimal number of off-diagonal entries suggests that misclassifications are rare and typically involve characters with subtle visual similarities. The strong diagonal dominance in the matrix highlights the model's effectiveness in accurately distinguishing between different characters. This accuracy is particularly notable given the complex and intricate nature of the Nagri script, where characters often have similar shapes and structures. Misclassifications, as indicated by the off-diagonal elements, are relatively sparse, suggesting that the model is generally robust in its predictions. However, the few misclassifications that do occur likely involve characters with similar visual features, which could be challenging to differentiate even for human observers [25]. These instances underscore the importance of further fine-tuning the model, possibly by augmenting the training data or employing more advanced pre-processing techniques to enhance character distinctiveness. The confusion matrix also reveals the model's performance across different character classes, offering insights into any potential biases or weaknesses [26]. For example, if certain characters are more frequently misclassified, this could indicate areas where the model requires additional training or where specific character features need to be more effectively captured by the model's architecture.

4. Limitations

While the VGG-19 model demonstrated exceptional performance in classifying the Nagri script, several limitations were encountered in this study. One of the primary challenges was the limited availability of annotated datasets specific to the Nagri script, which constrained the model's ability to generalize across a wider variety of text images. Another limitation is the model's occasional difficulty in distinguishing between visually similar characters, as evidenced by the sparse misclassifications in the confusion matrix. These errors highlight the need for more sophisticated feature extraction techniques or enhanced pre-processing steps to improve the model's ability to differentiate between subtle variations in character shapes. Moreover, the computational resources required for training and fine-tuning deep learning models like VGG-19 are substantial. This limitation may restrict the deployment of such models in resource-constrained environments or for real-time applications, where faster and more lightweight models might be preferable.

5. Conclusions and future works

This study presents a novel approach to optical character recognition (OCR) for the Nagri script, employing deep learning models, particularly VGG-19, to achieve high accuracy in character classification. The results demonstrate that the VGG-19 model is highly effective in recognizing and digitizing Nagri script characters, with performance metrics surpassing those of other contemporary models like ResNet-50, MobileNet-v3, and Xception. The model's accuracy, precision, recall, and F1-score all approached near-perfect levels, underscoring its robustness in handling the intricate features of the Nagri script. The successful implementation of this OCR system marks a significant step forward in the digitization and preservation of the Nagri script, contributing to the broader field of cultural heritage preservation. By converting ancient texts into machine-readable formats, this research not only aids in preserving a valuable cultural asset but also facilitates linguistic analysis and historical research. However, the limitations identified in this study pave the way for future work. One avenue for improvement involves expanding the dataset to include a more diverse range of Nagri script texts, particularly those that are rare or exhibit significant variation in character forms. Additionally, future research could explore the integration of more advanced pre-processing techniques to further reduce misclassification rates, particularly for visually similar characters. Another important direction for future work is the adaptation of the model for real-world applications, such as digitizing ancient manuscripts that may be degraded or contain noise. This could involve the development of more robust models that can handle these challenges, possibly through the use of data augmentation techniques or the incorporation of additional contextual information during classification. Exploring the use of more lightweight and efficient models could be beneficial for deploying OCR systems in resource-constrained environments or for real-time applications. Balancing accuracy with computational efficiency will be key to expanding the practical applicability of this research. In conclusion, while this study makes significant strides in Nagri script OCR, ongoing research and development will be essential to address the remaining challenges and to fully realize the potential of OCR technology in preserving and accessing ancient scripts.

Supplementary materials

SM1. Data pre-processing: Image enhancement, skewness correction, Marta elimination and cutting, and apply contour point detection.

Funding

No funding received for this work.

Acknowledgement

Nothing to declare

Data availability

Data will be made available upon reasonable request.

Declaration of competing interest

No potential conflict of interest was reported by the authors. All authors have read and agreed to the published version of the manuscript.

References

[1] M. Avadesh, N. Goyal, Optical character recognition for Sanskrit using convolution neural networks, In 2018 13th IAPR international workshop on document analysis systems (DAS) (2018) 447-452, https://doi.org/10.1109/DAS.2018.50

[2] A. Qaroush, A. Awad, M. Modallal, M. Ziq, Segmentation-based, omnifont printed Arabic character recognition without font identification, Journal of King Saud University-Computer and Information Sciences 34(6) (2022) 3025-3039, https://doi.org/10.1016/j.jksuci.2020.10.001

[3] M. A. Radwan, M. I. Khalil, H. M. Abbas, Neural networks pipeline for offline machine printed Arabic OCR, Neural Processing Letters 48(2) (2018) 769-787, https://doi.org/10.1007/s11063-017-9727-y

[4] M. A. Mojumder, A. Islam, T. Islam, S. Das, Z. Chowdhury, A. Rahman, Bangla and Nagri Language Detection and Classification Using Deep Learning Methods on Text Document Image, In 2023 International Conference on Advances in Electronics, Communication, Computing and Intelligent Information Systems (ICAECIS) (2023) 480-485. https://doi.org/10.1109/ICAECIS58353.2023.10170588

[5] M. Nihal Baig, N. Alam Owishi, M. Shahidur Rahman, A SAR-Based Approach to Recognize Text in Syloti Nagri, in 2023 6th International Conference on Electrical Information and Communication Technology (EICT) (2023) 1-6, https://doi.org/10.1109/EICT61409.2023.10427969

[6] S. Alghyaline, Arabic Optical Character Recognition: A Review, CMES-Computer Modelling in Engineering & Sciences 135(3) (2023) 1825-1861, http://dx.doi.org/10.32604/cmes.2022.024555

[7] S. Faizullah, M. Sohaib Ayub, S. Hussain, M. Asad Khan, A survey of OCR in Arabic language: applications, techniques, and challenges, Applied Sciences 13(7) (2023) 4584, https://doi.org/10.3390/app13074584

[8] T. Kumar, C. Khosla, K. Vashistha, Character recognition techniques using machine learning: A comprehensive study, In 2023 International Conference on Applied Intelligence and Sustainable Computing (ICAISC), (2023) 1-6, https://doi.org/10.1109/ICAISC58445.2023.10199173

[9] B. Kataria, H. B. Jethva, CNN-bidirectional LSTM based optical character recognition of Sanskrit manuscripts: A comprehensive systematic literature review, Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol 5(2) (2019) 2456-3307, https://doi.org//10.32628/CSEIT2064126

[10] A. Raj, R. Srivastava, T. Patnaik, B. Kumar, A survey of feature extraction and classification techniques used in character recognition for Indian scripts, International Journal of Engineering and Advanced Technology 2(3) (2013) 566-568.

[11] Z. Wang, Ancient character recognition with deep learning techniques, Applied and Computational Engineering 9 (2023) 194-203, https://doi.org/10.54254/2755-2721/9/20230089

[12] M. Fahad Jubayer, F. Mahafuz Ruhad, M. Shahidullah Kayshar, Z. Rizve, M. Janibul Alam Soeb, S. Izlal, I. Md Meftaul, Detection and Identification of Honey Pollens by YOLOv7: A Novel Framework toward Honey Authenticity, ACS Agricultural Science & Technology 4(7) (2024) 747-758, https://doi.org/10.1021/acsagscitech.4c00220

[13] S. Rabby, F. Hossain, S. Das, I. Rahman, S. Das, J. Soeb, M. Fahad Jubayer, An automated approach for the kidney segmentation and detection of kidney stones on computed tomography using YOLO algorithms, Journal of Ideas in Health 6(4) (2023) 963-970, https://doi.org/10.47108/jidhealth.Vol6.Iss4.313

[14] P. Patil, K. Noronha, Character Segmentation of Devnagari Script in Printed Document Images using Projection Profiles, In 2023 International Conference on Communication System, Computing and IT Applications (CSCITA), (2023) 158-163, https://doi.org/10.1109/CSCITA55725.2023.10104858

[15] A. Jindal, R. Ghosh, Word and character segmentation in ancient handwritten documents in Devanagari and Maithili scripts using horizontal zoning, Expert Systems with

Applications 225 (2023) 120127, https://doi.org/10.1016/j.eswa.2023.120127

[16] Z. N. Ashani, I. Syfidza Che Ilias, K. Yap Ng, M. Rezal Kamel Ariffin, A. Dahari Jarno, N. Zarina Zamri, Comparative analysis of deepfake image detection method using vgg16, vgg19 and resnet50, Journal of Advanced Research in Applied Sciences and Engineering Technology 47(1) (2024) 16-28, https://doi.org/10.37934/araset.47.1.1628

[17] D. Hindarto, N. Afarini, Comparison Efficacy of VGG16 and VGG19 Insect Classification Models, Jurnal Informatika dan Komputer 6(3) (2023) 189-195, https://doi.org/10.33387/jiko.v6i3.7008

[18] M. Mateen, J. Wen, Nasrullah, S. Song, Z. Huang, Fundus image classification using VGG-19 architecture with PCA and SVD, Symmetry 11(1) (2018) 1, https://doi.org/10.3390/sym11010001

[19] M. Manataki, N. Papadopoulos, N. Schetakis, A. Di Iorio, Exploring Deep Learning Models on GPR Data: A Comparative Study of AlexNet and VGG on a Dataset from Archaeological Sites, Remote Sensing 15(12) (2023) 3193, https://doi.org/10.3390/rs15123193

[20] M. Shaha, M. Pawar, Transfer learning for image classification, In 2018 second international conference on electronics, communication and aerospace technology (ICECA) (2018) 656-660, https://doi.org/10.1109/ICECA.2018.8474802

[21] S. Devendiran, P. Nedungadi, R. Raman, Handwritten Text Recognition using VGG19 and HOG Feature Descriptors, In 2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI) (2024) 1-4, https://doi.org/10.1109/IATMSI60426.2024.10502872

[22] S. Zahoor, S. Naz, N. H. Khan, M. I. Razzak, Deep optical character recognition: a case of Pashto language, Journal of Electronic Imaging 29(2) (2020) 023002, https://doi.org/10.1117/1.JEI.29.2.023002

[23] R. Gautam, R. Parajuli, and G. Sharma, Devanagari Handwritten Character Recognition using fine-tuned Deep Convolution Neural Network, International Journal of Research Publications 146(1) (2024) 327-335, https://doi.org/10.47119/IJRP1001461420246272

[24] F. Jubayer, J. Alam Soeb, A. Naser Mojumder, M. Kanti Paul, P. Barua, S. Kayshar, S. Sabrina Akter, M. Rahman, A. Islam, Detection of mold on the food surface using YOLOv5, Current Research in Food Science 4 (2021) 724-728, https://doi.org/10.1016/j.crfs.2021.10.003

[25] K. Liang, L. Jin, Z. Xie, X. Xiao, and W. Huang, A comprehensive analysis of misclassified handwritten Chinese character samples by incorporating human recognition, In 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR) 1 (2017) 460-465, https://doi.org/10.1109/ICDAR.2017.82

[26] I. Markoulidakis, and G. Markoulidakis, Probabilistic Confusion Matrix: A Novel Method for Machine Learning Algorithm Generalized Performance Analysis, Technologies 12(7) (2024) 113, https://doi.org/10.3390/technologies12070113