International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Exploring ResNet101, InceptionV3, and Xception for Modi Script Character Classification

[1]Ravindra Sonavane, [2]Dr. Pandharinath Ghonge, [3]Sandip Umajirao Patil, [4]Kaustubh Shivaji Sagale, [5]Anand Arvind Maha

Submitted: 22/12/2023 **Revised**: 28/01/2024 **Accepted**: 08/02/2024

Abstract: The "MODI lipi" script, which was used historically in Maharashtra, Western India, to record religious writings, and which was the official script used by the Maratha administration from the 17th century until the mid-1900s, has a rich cultural legacy. Even though the "Manuscript" is a valuable source of inspiration and knowledge from a bygone age, modern audiences are not as familiar with it. Recognizing its potential to inspire and educate the present generation, there is a need to develop a sophisticated recognition system for MODI within handwritten character recognition. Deep learning-based algorithms, such as ResNet101, InceptionV3, and Xception, have demonstrated remarkable efficacy in various pattern identification applications, including character recognition. In the current landscape, transfer learning algorithms, especially those leveraging ResNet101, InceptionV3, and Xception architectures, have gained prominence for significantly enhancing recognition task outcomes. This study specifically proposes implementing these advanced deep-learning models to classify MODI characters. By harnessing the power of ResNet101, InceptionV3, and Xception algorithms, the aim is to optimize the recognition accuracy and efficiency of the MODI script, making it more accessible and comprehensible for today's youth. This research endeavors to unlock the untapped potential of MODI script as a valuable cultural and educational resource by utilizing state-of-the-art deep learning methodologies.

Keywords*: Character recognition, Modi script, segmentation, recognition, classification*

1. Introduction

The 'MODI' script is a distinctively archaic style of writing when compared to other old Indian languages. The MODI script was only used for writing and was derived from a cursive style called "Marathi," which is the primary language of the state of Maharashtra in western India. There are several theories as to where this style came from. A widely held belief is that "Hemadpant" or "Hemadri" invented MODI in the twelfth century. Since documents are the most common type of information in modern civilization, document image analysis is an important area of image processing and pattern recognition research.

The modern world operates rapidly, characterized by high mechanization and a strong connection between automation and technology. This trend arises from our innate preference for completing tasks swiftly and efficiently. As automation becomes more integral to our processes, tasks are streamlined and expedited. Digitalization, a cornerstone of today's fast-paced society, seeks to transform all accessible information into a digitized format stored on computers with enhanced computational capabilities. However, the challenge lies in imparting specific real-world knowledge to computers to facilitate accurate translation into the digital domain.

This discussion delves into Handwritten Character Recognition (HCR) technology, a topic that has gained significant attention in the past decade. HCR is crucial because it allows computers to learn and understand regional languages effectively, opening up limitless possibilities. While existing HCR research has predominantly focused on widely spoken languages such as English, French, Hindi, and even foreign languages like Chinese and Japanese, our attention shifts to the unique challenges posed by the MODI script.

Handwritten Character Recognition involves obtaining an input image from a scanner for an offline recognition system. Subsequent preprocessing of the scanned digital image enhances its quality and prepares it for segmentation. Erosion, dilation, opening, closure, and smoothing are all part of this preprocessing. The grayscale image is converted to a binary image by the binarization process, which uses the global thresholding technique. The last two steps provide a preprocessed picture that is prepared for segmentation. They comprise hole filling, dilatation of the image, and edge identification using the Sobel technique. The unique

[1]*Dept. of Artificial Intelligence & Data Science, Thakur College of Engineering & Technology, Mumbai, India*
ravi.mergit16@gmail.com
[2]*Dept. of Electronics and Telecommunication Engineering, SJCEM, Palghar, Mumbai, India*
pandharinathg@sjcem.edu.in
[3]*Dept.of Electronics and Computer Science Department, St. John College of Engineering and Management, Palghar, Mumbai, India*
sandipp@sjcem.edu.in
[4]*Dept. Electronics and Telecommunication Engineering, R.C. Patel Institute of Technology, Shirpur, Dhule, India*
Kaustubh.sagale@rcpit.ac.in
[5]*Dept. of Artificial Intelligence and Machine Learning, TCET, Kandivali, Mumbai, India*
anand.maha@tcetmumbai.in

characteristics of the MODI script present specific challenges and opportunities for applying advanced deep learning models, including ResNet101, InceptionV3, and Xception, in character classification.

2. Literature Survey

The structural similarities between MODI characters were investigated by Ramteke A.S. and Katkar G.S. [1]. They used structural similarity approaches to evaluate the quality of the images they used. recognition rates were from 91% to 97% as a consequence of the classification process, which was carried out utilising measured structural similarity (SSIM), KNN, and backpropagation NN. Using a zone-based approach, Besekar D.N. and Ramteke R.J. [2] concentrated on verifying offline handwritten numbers. Their variance table-based classification yielded a 93.5 percent identification rate after a pretreatment comprising median filtering, global thresholding, and size normalization.

A theoretical investigation of MODI script identification was carried out by Besekar D.N. and Ramteke R.J. [3], who looked at the structural characteristics of Roman, MODI, and Devanagari scripts. The study addressed internal and exterior segmentations and emphasised the difficulties in recovering MODI script because of its cursive form. The use of CNN autoencoder as a feature extractor for MODI script character recognition was suggested by this study. They used SVM for classification, shrinking the feature set from 3600 to 300, and achieved a remarkable 99.3 percent recognition rate beyond previously reported values [4].

Parag A. Tamhankar's research [5] focused on extracting individual characters from antique handwritten MODI Script writings. The study proposed a unique method for character extraction based on a dual thresholding criterion, demonstrating simplicity and cost-effectiveness in execution time efficiency. A supervised Transfer Learning-based classifier for handwritten MODI character recognition was presented by Savitri Chandure and colleagues [6]. After extracting features with a deep convolutional neural network, they classed the data with an SVM classifier. The research achieved an excellent accuracy rate of 92.32% by delving into the examination of feature properties and recognition accuracy.

A method for identifying offline handwritten Modi numerals was presented by Manisha Deshmukh et al. [7]. They demonstrated the usefulness of 5X5 grid divisions by achieving a maximum recognition rate of 85.21% on a database of 30,000 pictures using Modi numerical chain code feature extraction and a non-overlapping blocking method. Using the ANESP programme, Sanjay S. Gharde et al. [8] concentrated on detecting and recognizing handwritten MODI characters. The original MODI script document was gathered and preprocessed for the study. Affine and Moment Invariant approaches were used for feature extraction, and support vector machine (SVM) classification was used to achieve high recognition rates.

3. Proposed System

Fig. 2 depicts the proposed system's block diagram. Training and testing are the two sections of the block diagram.

A. MODI character and numeral Dataset

The envisioned system employs handwritten MODI characters sourced from diverse individuals. The dataset comprises handwritten MODI characters and numerals, with contributions from various writers exhibiting distinct writing styles. This deliberate inclusion of diverse writing styles enhances the system's accuracy by accounting for variations in style and image captures under different lighting conditions. Consequently, the dataset is meticulously curated to encompass a spectrum of writing types and lighting scenarios. To make training and evaluating the model easier, the whole dataset is divided into training and validation sets. In particular, eighty percent of the dataset is put aside for testing, which allows for a reliable evaluation of the model's performance in various scenarios, and the remaining twenty percent is set aside for training, allowing the system to learn from a sizable and diverse pool of samples.

B. Preprocessing

To ensure uniformity within the dataset, captured images, which may vary in size, are resized to a standardized dimension of 256x256 in the system. Additionally, various types of noise, such as salt and pepper and Gaussian noise, can affect the images. To address this, a median filter is employed to eliminate salt and pepper noise, while Gaussian filtering is utilized to mitigate the impact of Gaussian noise. This preprocessing step enhances the consistency and quality of the images for subsequent analysis and recognition tasks.

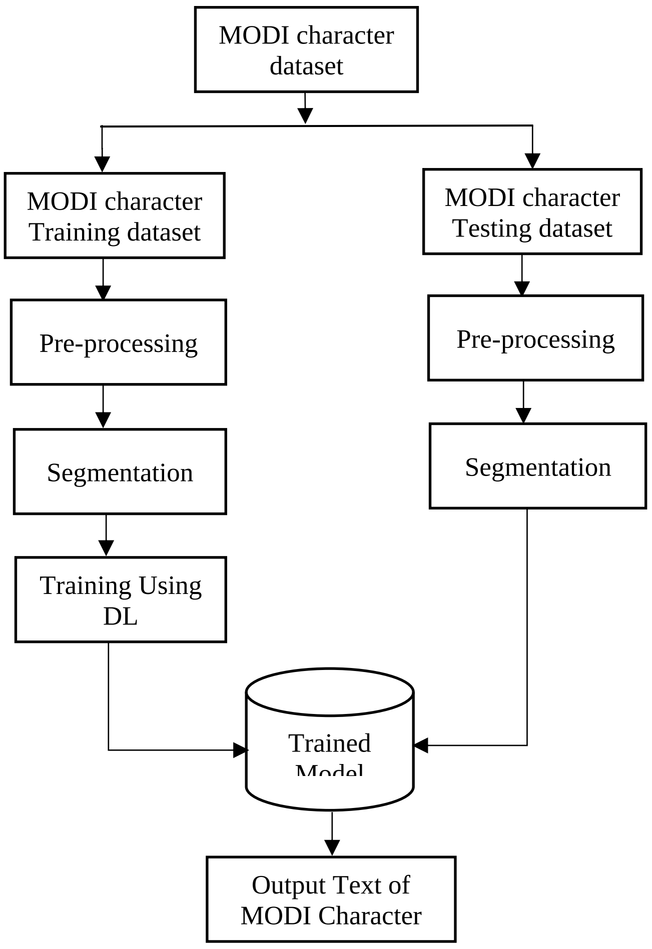


Fig.1. Block diagram of MODI character recognition system

C. *Segmentation*

During the first image processing steps, the picture's backdrop is carefully inspected, and character segmentation is then carried out using a thresholding method. This crucial phase separates the foreground from the background by converting the image into a binary representation. By setting an optimal threshold, the pixels in the image are categorized into two distinct intensity levels, facilitating the identification of the region of interest, namely, the characters within the image. Once this segmentation is accomplished, the individual characters are precisely isolated from the background clutter. The segmented characters are then cropped and saved for subsequent stages of further processing and analysis. This methodical approach to character segmentation through thresholding not only enhances the clarity and visibility of the characters but also serves as a fundamental preparatory step for subsequent advanced techniques applied in character recognition and feature extraction.

D. *Training and Testing*

The proposed approach used three different Resnet101, InceptionV3, and Xception algorithms to classify the Modi script characters. Each algorithm is explained below.

a. *Resnet101*

The ResNet101 architecture, short for Residual Network with 101 layers, is a deep convolutional neural network that has proven highly effective in various computer vision tasks, including image recognition. It is characterized by its use of residual blocks, which facilitate the training of very deep networks. The architecture of Modi script character recognition using ResNet101 involves building blocks and layers designed to extract hierarchical features from input images. Here is an overview of the ResNet101 architecture:

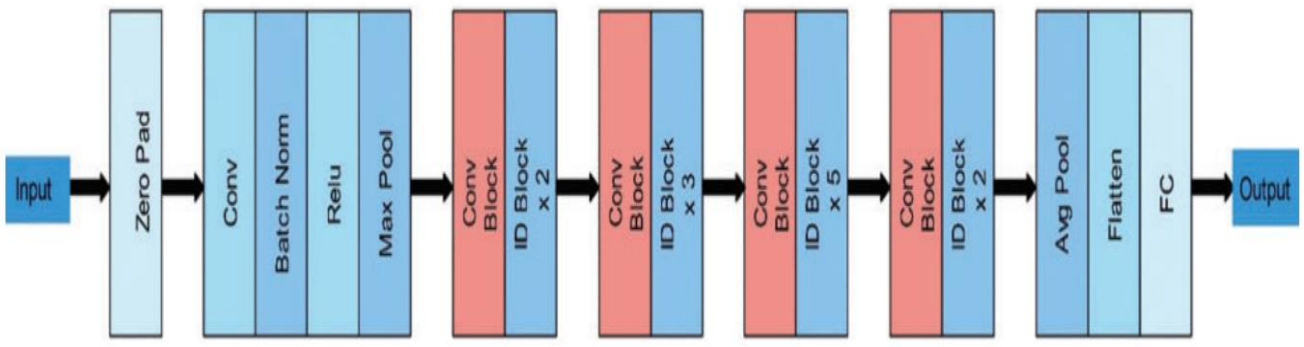


Fig.2. Architecture Diagram of Resnet101 algorithm

- Input Layer: The network starts with an input layer that takes in the grayscale or color images of Modi script characters.
- Convolutional Layers: Initial convolutional layers perform the task of extracting low-level features from the input images. These layers use small filters to capture basic patterns and textures.
- Residual Blocks: Residual blocks are one of the characteristics of ResNet101. A shortcut link that skips one or more layers can be found in every block. This design makes training very deep networks easier and mitigates the issue of vanishing gradients. Usually, two or more convolutional layers with rectified linear unit (ReLU) activation functions and batch normalisation make up the residual block.

- Stacked Blocks: Multiple residual blocks are stacked together to form a deep hierarchy. ResNet101 specifically includes 101 layers, making it a deep neural network capable of learning intricate features.
- Global Average Pooling (GAP) Layer: After the stacked blocks, a Global Average Pooling layer is frequently used. This layer extracts the global information from the whole feature map by fixing the spatial dimensions of the feature maps.
- Fully Connected Layer: The final categorization process makes use of a fully connected layer. The characteristics that were retrieved by the earlier layers are mapped to the output classes in this example, the various Modi script characters by this layer.
- Softmax Activation: The softmax activation function is applied to the output layer, providing normalized probabilities for each character class. This enables the network to make a confident prediction about the class of the input character.
- Training and Optimization: The entire network uses a suitable optimization algorithm, often with the cross-entropy loss function. To reduce the discrepancy between expected and actual character labels, the network weights are modified during training.

b. InceptionV3 algorithm

InceptionV3 is a deep CNN architecture for image classification. Researchers at Google developed it as part of the Inception family of models. The primary goal of InceptionV3 is to improve the efficiency and accuracy of image recognition tasks.

It's crucial to remember, though, that InceptionV3 was not created with Modi script character recognition in mind. In the Modi area of India, the Marathi language is written using the Modi script, an ancient script. Generally, Modi script character recognition is achieved by training a model using Modi script character-related data.

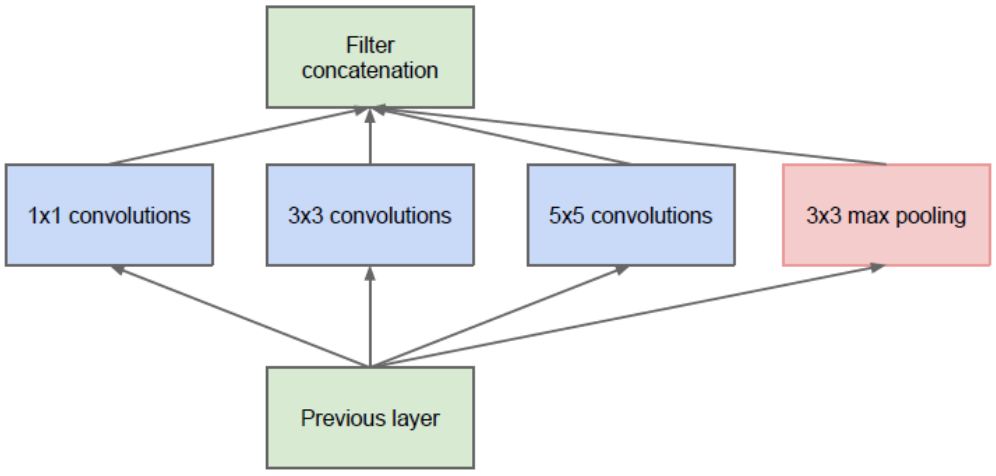


Fig.3. Architecture Diagram of InceptionV3 algorithm

- Input Layer: InceptionV3 takes an input image with a standard size (299x299 pixels).
- Convolutional Layers: The network starts with several convolutional layers that extract features from the input image. These layers apply convolutional filters to detect patterns and features in different image parts.
- Inception Modules: Using Inception modules is one of the main advancements found in InceptionV3. These modules are made up of several concurrent convolutional layers that have pooling functions and various filter sizes (1x1, 3x3, and 5x5). The outputs of these parallel operations are then concatenated, allowing the model to capture features at multiple scales.
- Reduction Blocks: InceptionV3 includes reduction blocks that help decrease the spatial dimensions of the input volume while increasing the depth. This helps reduce the computational load and capture more abstract features.
- Fully Connected Layers: The extracted features are fed into fully connected layers, which perform the final classification. These layers combine the information from the previous layers to produce class probabilities.
- Softmax Activation: Usually, a softmax activation function is used in the last layer to translate the network's output into probabilities for each class.
- Output Layer: The output layer produces the final predictions for the image's class.

c. Xception

Xception is a convolutional neural network (CNN) architecture that François Chollet introduced in the paper

"Xception: Deep Learning with Depthwise Separable Convolutions" in 2017. The architecture is an extension of the Inception architecture, incorporating depthwise separable convolutions.

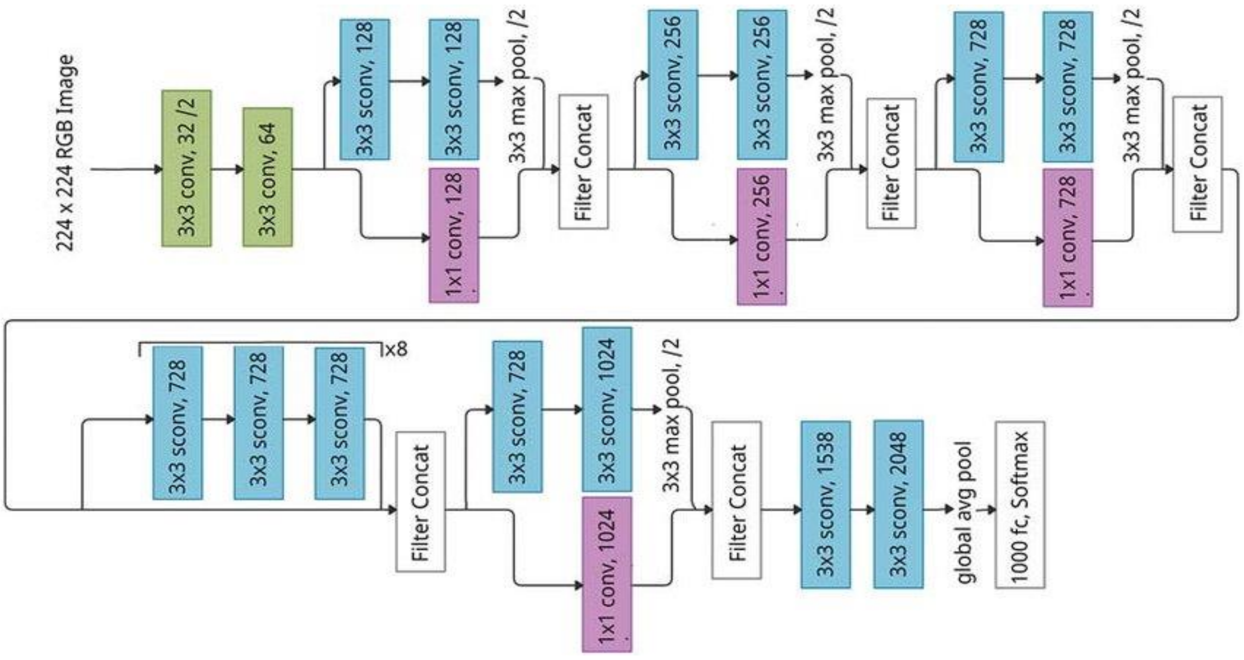
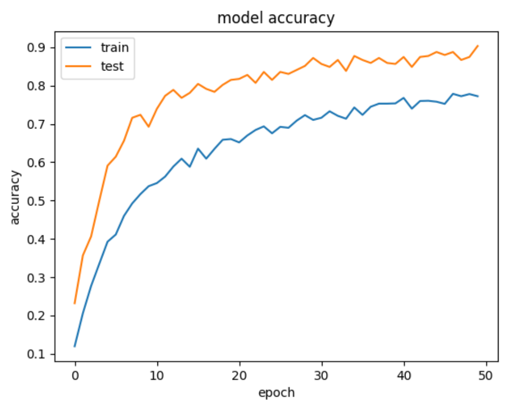


Fig.4. Architecture Diagram of Xception algorithm

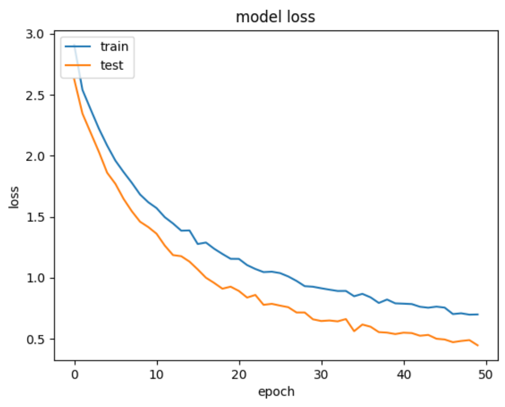
- Depthwise Separable Convolutions: Depthwise separable convolutions are used by Xception in place of conventional convolutional layers. The convolutional procedure is divided into depthwise convolution and pointwise convolution in a depthwise separable convolution.
- Depthwise convolution individually applies one filter to each of the input channels. Using 1x1 convolutions, pointwise convolution combines the outcomes of depthwise convolution.
- Feature Learning: With a more effective design, Xception seeks to capture increasingly intricate details. While maintaining expressive capacity, the depthwise separable convolutions aid in lowering the number of parameters.
- Linear Unit Activation: The activation function used in Xception is the linear unit (ReLU) after each convolution operation. Regarding Modi script character recognition, the application of Xception or any other deep learning architecture involves adapting the network to the characteristics of the script and the dataset. Here are general steps you might consider:
- Data Preparation: Collect and preprocess a dataset of Modi script characters. This involves obtaining labeled data and preparing it for training.
- Model Architecture: Adapt the Xception architecture or a similar architecture to Modi script character recognition specifics. This may involve adjusting input size, the number of classes, and other parameters.
- Training: Utilizing the provided dataset, train the model. To prevent overfitting, adjust the settings and keep an eye on performance on a validation set.
- Evaluation: Test the trained model's performance and ability to generalise to new data on a different test set. .

4. Results

The implementation of the system in question was carried out using the Python programming language. The dataset was trained through the application of Resnet101, InceptionV3, and Xception algorithm, leveraging the capabilities of the Google Colab platform. The results of the system are explicated through a thorough analysis encompassing both qualitative and quantitative assessments. The accuracy and loss of the proposed system using Resnet101, InceptionV3, and the Xception algorithm are presented in Fig. 5, Fig.6, and Fig. 7, respectively.

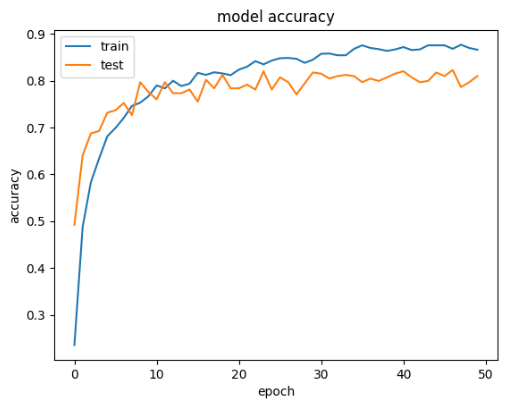


(a)

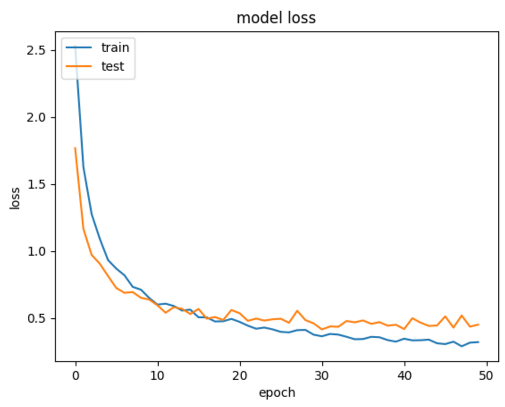


(b)

Fig.5. Training progress graph of Resnet101 algorithm in terms of (a) Accuracy, (b) Loss

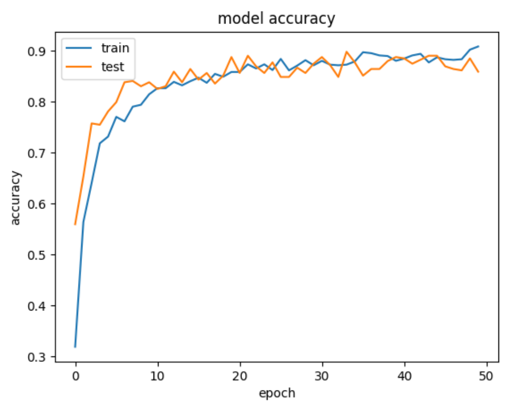


(a)

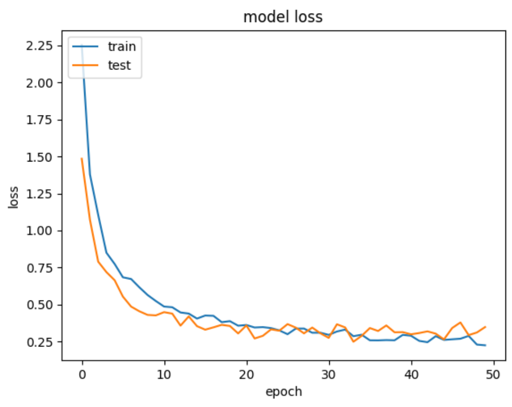


(b)

Fig.6. Training progress graph of Inception algorithm in terms of (a) Accuracy, (b) Loss



(a)



(b)

Fig.7. Training progress graph of Xception algorithm in terms of (a) Accuracy, (b) Loss

Table I presents a tabular comparison of the accuracy and loss of the Resnet101, InceptionV3, and Xception algorithms during training and validation.

TABLE I. QUANTITATIVE ANALYSIS

Algorithm	***Training***		***Validation***	
	Accuracy	***Loss***	***Accura cy***	***Loss***
Resnet101	0.7725	0.6996	0.9036	0.446 6
InceptionV3	0.8662	0.3185	0.8099	0.448 9
Xception	0.9087	0.2242	0.8594	0.347 4

The presented table outlines the performance metrics of three distinct algorithms—ResNet101, InceptionV3, and Xception—applied to a specific task. The metrics are reported for the training and validation phases, encompassing accuracy and loss values.

In the training phase, ResNet101 achieved an accuracy of 77.25%, indicating that approximately 77.25% of the training dataset was correctly classified. The corresponding training loss was 0.6996, measuring the algorithm's error during the learning process. Moving to the validation phase, ResNet101 exhibited an improved accuracy of 90.36%, with a reduced validation loss of 0.4466, signifying enhanced generalization to new, unseen data.

Similarly, InceptionV3 and Xception were evaluated using the same metrics. InceptionV3 demonstrated a training accuracy of 86.62% and a training loss of 0.3185. Its validation accuracy was 80.99%, with a validation loss of 0.4489. As for Xception, it achieved a training accuracy of 90.87% with a training loss of 0.2242. In the validation phase, Xception attained an accuracy of 85.94%, accompanied by a validation loss of 0.3474.

These metrics comprehensively understand how well each algorithm learned from the training data and how effectively they generalize to new data during the validation phase. The values presented in the table serve as quantitative benchmarks for assessing the relative performance of ResNet101, InceptionV3, and Xception in the specified task.

Fig. 8 displays the qualitative analysis of the proposed system.

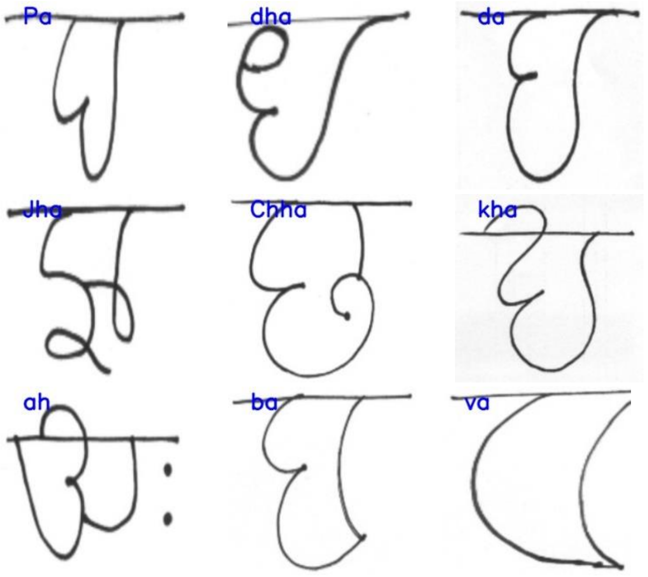


Fig. 6. Qualitative analysis of the proposed system

Through a qualitative analysis, it becomes evident that the Resnet101 algorithm exhibits precise and accurate recognition of MODI characters.

5. Conclusion

In conclusion, this research sheds light on the rich cultural heritage encapsulated within the "MODI lipi" script by implementing advanced deep learning models, ResNet101, InceptionV3, and Xception, to classify MODI characters. The study results showcase promising outcomes, with the three algorithms demonstrating varying degrees of effectiveness in recognizing MODI script characters. ResNet101, known for its deep learning capabilities, achieved a training accuracy of 77.25%, which improved to an impressive validation accuracy of 90.36%. InceptionV3, with its unique architecture, demonstrated a commendable training accuracy of 86.62% and a validation accuracy of 80.99%. Xception, leveraging depthwise separable convolutions, outperformed the others with a training accuracy of 90.87% and a validation accuracy of 85.94%. These findings underscore the potential of deep learning, especially transfer learning using ResNet101, InceptionV3, and Xception architectures, in enhancing the recognition accuracy and efficiency of the MODI script. The success of these algorithms in character recognition tasks signifies a significant step towards making MODI more accessible and comprehensible for contemporary audiences, particularly the youth. By unlocking the untapped potential of the MODI script as a valuable cultural and educational resource, this research contributes to preserving and revitalizing a script that holds historical significance, ultimately bridging the gap between the past and the present.

References

[1] Ramteke A.S., Katkar G.S., 2012, "Recognition of Offline MODI Script," International Journal of Research

in Engineering, IT and Social Science (IJREISS), Volume 2, Issue 11, ISSN 2250-0588, pp.102-109.

[2] Besekar D.N., Ramteke R.J., 2012, "Feature Extraction Algorithm for Handwritten Numerals Recognition of MODI Script using Zoning-based Approach", International Journal of Systems, Algorithms & Applications, Volume 2, Issue ICRASE12, ISSN 2277 2677, pp. 1-4.

[3] Besekar D.N., Ramteke R.J, 2013, "Study for Theoretical Analysis of Handwritten MODI Script – A Recognition Perspective," International Journal of Computer Applications, vol. 64, no. 3, ISSN 0975-8887, pp. 45-49.

[4] S. Joseph and J. George, "Handwritten Character Recognition of MODI Script using Convolutional Neural Network Based Feature Extraction Method and Support Vector Machine Classifier," 2020 IEEE 5th International Conference on Signal and Image Processing (ICSIP), 2020, pp. 32-36.\

[5] P. A. Tamhankar, K. D. Masalkar, and S. R. Kolhe, "A novel approach for character segmentation of offline handwritten Marathi documents written in Modi script," Procedia Computer Science, Third International Conference on Computing and Network Communications (CoCoNet'19), vol. 171, pp. 179–187, 2020.

[6] S. Chandure and V. Inamdar, "Handwritten Modi character recognition using transfer learning with discriminant feature analysis," IETE Journal of Research, pp. 1–11, 2021.COEP,

[7] Manisha s. Deshmukh, Manoj Patil, and Satish Kolhe," Offline handwritten Modi numerals recognition using chain code."

[8] Sanjay S. Gharde and Dr. R. J. Ramteke, "Indian Recognition of Characters in Modi Script," 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication,978-1-5090-0467-6/16/$31.00 ©2016 IEEE.