OPTIMIZED MARATHI HANDWRITTEN CHARACTER RECOGNITION USING CONVOLUTION NEURAL NETWORK

Prof. Darshan Mhapasekar
Dept. of Computer Engg.
SSPM's College of Engg., Sindhudurg
University of Mumbai
dmhapasekar@gmail.com

Ms. Siddhi Shintre
Dept. of Computer Sci. Engg.(AIML)
SSPM's College of Engg., Sindhudurg
University of Mumbai
siddhishintre16@gmail.com

Ms. Sakshi Shetye
Dept. of Computer Sci. Engg.(AIML)
SSPM's College of Engg., Sindhudurg
University of Mumbai
sakshishetye01@gmail.com

Ms. Komal Dichavalkar
Dept. of Computer Sci. Engg. (AIML)
SSPM's College of Engg., Sindhudurg
University of Mumbai
komaldichavalkar1802@gmail.com

Ms.Sanjana Naik
Dept. of Computer Sci. Engg. (AIML)
SSPM's College of Engg., Sindhudurg
University of Mumbai
1sanjananaik@gmail.com

***Abstract*—Marathi handwritten character recognition, enabled by image processing and deep learning, is a groundbreaking development in the field of Devanagari OCR. It unlocks the vast potential of Devanagari heritage and culture. This paper aims to identify Marathi characters in a sentence or a line. The challenges such as individual having their own style of writing, representing words with blocks of letters, and identification of punctuation, leading to errors and reducing accuracy. To overcome this, we use a holistic approach that combines image processing technology and deep learning. It starts with a study of language and character segmentation.**

We have used Convolutional Neural Networks for Marathi Handwritten Character Recognition. We have developed five configurations of CNN models and conducted experiments with different epoches and optimizers. Our system relies on Python's diverse ecosystem, using OpenCV for image processing, TensorFlow/Keras for neural networks, and sci-kit-learn for machine learning. We collaborate with Google for interactive development and use GPU for rapid training. The experimental results show that CNN with 3 blocks of convolution layer, 3 dense layers, and Adam optimizer gives an accuracy of 85% . Our system is incredibly sensitive, flexible, and progressive. It blends accuracy, adaptability, and efficiency to revolutionize Marathi handwritten text recognition and will transform Devanagari text.

***Index Terms*—Deep Learning, CNN, Character recognition, Optimization, Regularization, HACR**

I. Introduction

Marathi is the official language of India and is spoken by over 80 million people worldwide. The language uses the Devanagari script, which is also used by other Indian languages such as Hindi and Sanskrit. [1] However, recognizing Marathi handwriting (HMCR) can be challenging due to various factors, such as the presence of noise in handwritten images, variations in handwriting styles, and the large number of characters in the Devanagari script. [2]

A. Key Problems

There are some of the key problems that need to be addressed in designing a Marathi handwritten recognition system:

1) Character Set Size: Devanagari script has a large character set of over 50 characters. 15 consonants and vowels. This makes it difficult to train a classifier that can identify all characters. [3]
2) Handwriting variability: Handwriting styles can vary widely from person to person. This can make it difficult for a classifier to generalize to new handwriting styles.
3) Noise: Handwritten images can be noisy due to factors such as pen smudges. and paper folds. This can make it difficult for a classifier to extract meaningful features from the image [4]

B. Need for Marathi Handwritten Recognition System

Here are some specific areas where there is a need for good Marathi handwritten recognition systems:

1) Digitizing handwritten Marathi manuscripts and documents: There is a vast amount of handwritten Marathi content in the form of manuscripts, books, letters, and other documents. Digitizing this content would make it more accessible to researchers and scholars, and would also help to preserve this important cultural Heritage.
2) Developing Marathi handwriting tutorials and other educational applications: Marathi handwritten recognition systems could be used to develop Marathi handwriting tutorials and other educational applications. These would help people to learn Marathi handwriting and to improve their handwriting skills. In addition to these specific areas, there is also a need for Marathi handwritten recognition systems that are more accurate and efficient. Current Marathi handwritten recognition systems still have some limitations in terms of their accuracy and speed.

C. key Challenges

In the development of a Marathi handwritten recognition system, several key challenges and limitations must be considered, including

1) Character Set Complexity: Recognizing the extensive Devanagari character set poses accuracy challenges.
2) Handwriting Variability: Adapting to diverse handwriting styles remains a significant hurdle.
3) Noisy Input: The presence of noise, such as smudges, affects recognition accuracy.
4) Confusing Similar Characters: Distinguishing similar-looking Marathi characters can be challenging.
5) Real-time Processing Demands: Achieving real-time recognition may require substantial computational resources. [5]

II. LITERATURE REVIEW

Yawalkar and colleagues [6] undertook an analysis of the applicability of Deep Convolutional Neural Networks (DCNN) through transfer learning approaches. Their study utilized a novel extended edition of the presented database. The outcomes of their research demonstrated acceptable identification accuracies, surpassing other well-known conventional techniques in the field of Handwritten Marathi Character Recognition (HACR). The adopted scheme proved to be more effective than conventional techniques, as confirmed by the study's results.

Garg and his team [7] explored the effectiveness of online and offline character recognition, feature extraction, classification, and deep learning techniques. The study achieved high identification accuracies, showcasing superiority over various conventional methods in the realm of Handwritten Marathi Character Recognition (HACR). The study's success underlines the potential for advanced deep-learning applications in Marathi handwriting recognition.

Dandawate and the team [8] engaged in comprehensive research involving data acquisition, pre-processing, skew correction, and feature extraction using Convolutional Neural Network (CNN) architecture. The study emphasized the importance of deep learning, particularly CNNs, in Optical Character Recognition (OCR) systems. The future scope of their work includes refining accuracy, addressing character variations, and expanding the system's capabilities for improved Handwritten Marathi Character Recognition (HACR).

Jindal and collaborators [9] executed a multi-stage process involving image acquisition, digitization, pre-processing, segmentation, feature extraction, and classification. While achieving an accuracy of 88.95%, the study outlined plans to expand the character recognition system, including modifiers and conjuncts, and enhance recognition accuracy through more efficient feature extraction and exploration of additional classification techniques.

Munish Kumar et al. [10] implemented various features, such as Matra/Shirorekha, segmentation-based features, foreground-background transition features, and convex-hull-based features. The study achieved high accuracies of 99.27% in script identification schemes. Future recommendations involve the development of models for the identification and recognition of Marathi letters and sentences, contributing to the broader scope of Handwritten Marathi Character Recognition (HACR).

M. K. Jindal et al. [11]applied fine-tuning, classification, dataset creation, pre-processing, and data augmentation to achieve an accuracy of 96.55%. Their focus for future work is on deploying more compact deep convolutional networks for character classification and exploring different network architectures for compound Devanagari characters and words. The study underscores the significance of structural features like concavity to enhance recognition accuracy in Handwritten Marathi Character Recognition (HACR).

Prashanth and the team [12] proposed a Convolutional Neural Network (CNN)-based Optical Character Recognition (OCR) system for Devanagari ancient manuscripts. The study achieved an accuracy of 93.73% and highlighted the need for a system capable of recognizing characters from different handwriting styles, degraded or damaged characters, and characters from various languages. The research addressed the challenges in recognizing diverse and historical scripts within the field of Handwritten Marathi Character Recognition (HACR).

III. METHODOLOGIES

Converting a scanned Devanagari handwritten document into a digitized version involves several key steps, ensuring that the final output accurately represents the original document. This process can be outlined as follows:

1.Scanned Devanagari Handwritten Document: The initial step involves the utilization of a scanned Devanagari handwritten document as the input. [4]

2.Preprocessing: The scanned document undergoes preprocessing to enhance image quality. This phase includes tasks such as noise reduction, contrast adjustment, and image resizing. The aim is to optimize the document's visual quality for precise character recognition.
[5]

3.Segmentation: The document is then divided into individual characters or words. Character segmentation is a critical process for distinguishing and recognizing each unit accurately.
[13]

4.Feature Extraction: After characters are isolated, feature extraction is performed. This step involves the extraction of relevant characteristics from each character. Commonly extracted features include pixel intensity values, edges, and gradients.
[14]

5.Character Recognition: Following feature extraction, the system processes the extracted features using a character recognition model. This model can be a Convolutional Neural Network (CNN) or other machine learning algorithms.
[15]

6.Classification: The recognized features are subsequently classified into specific characters. Classification assigns each character to its corresponding class or symbol within the Devanagari script.
[16]

7.Output: The recognized characters are provided as the output, resulting in the creation of a digitized version of the scanned Devanagari handwritten document.
[17]

8.End Result: The final outcome is a precise, digital representation of the original handwritten document. This digitized version makes the document accessible and searchable, facilitating its use in various applications. [18]

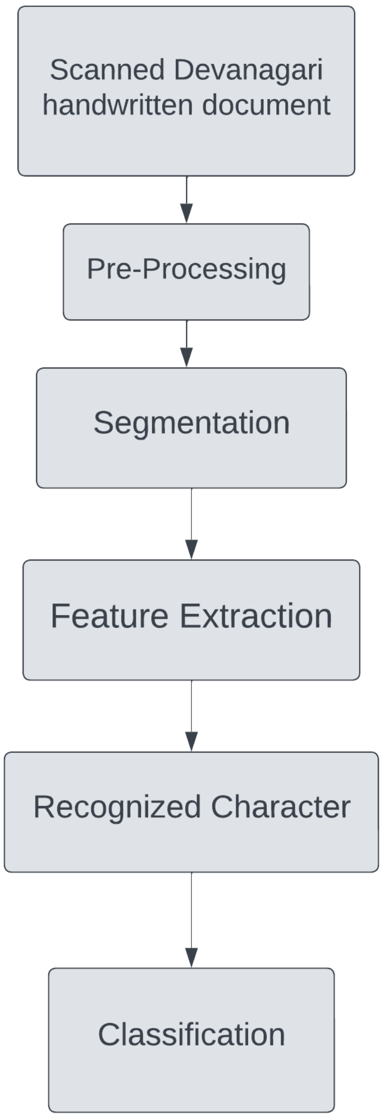

Fig. 1. Design Details

A. Algorithms

*1) **Convolutional Neural Network (CNN):*** In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks. It is used for image recognition and processing tasks. The Convolution Neural Network consists of a total of 4 layers, such as Input Layer, Convolution Layer, Pooling Layer, and Dense Layer(Fully connected layer). [19] In CNN, filters are applied to the input image to extract features like edges, textures, and shapes. The output of the convolutional layers is input to pooling layers, which is used to down-sample the feature maps, reducing the spatial dimensions and keeping the most important information. The output of the pooling layers is given to one or more fully connected layers, which are used to make a prediction or classify the image. Functioning of convolutional neural network is shown in Fig 9

- Kernels: also known as kernel techniques or kernel functions, are a collection of distinct forms of pattern analysis algorithms, using a linear classifier, that solve an existing non-linear problem. SVM (Support Vector Machines) uses Kernels Methods in ML to solve classification and regression issues
- Stride: It is a parameter of the neural network's filter that modifies the amount of movement over the image or video. For example, if a neural network's stride is set to 1, the filter will move one pixel, or unit, at a time [19]
- Padding : It is a technique employed that guarantees that the input data has the exact size and shape that the model anticipates after every convolutional operation at each stage of the deep learning model.

*2) **CNN Layers:***

1) Input Layer: The input layer is the entry point of the convolutional neural network. In image processing neural networks, it captures the image's pixel matrix.

2) Convolution Layer: This is the second layer that is used to extract the various features from the input images. This layer performs the mathematical operation of convolution between the input image and a filter of a size MxM. By putting the filter over the input image, the dot product is taken between the filter and the parts of the input image. The most commonly used convolution is the 2D convolution layer and it is written as as conv2D. A filter or a kernel in a conv2D layer "slides" over the input data, performing a dot product. All the values will be summed up to get the output as a single output pixel. The filter will perform the dot location it slides over.

3) Pooling Layer: The pooling layer is used to reduce the dimensions of the feature maps. Thus, it reduces the number of parameters and the amount of computation in the network. The pooling layer reduces the number of features and parameters so that further processing will be easy.

4) Dense Layer
Dense Layer is simply a layer of neurons in where each neuron take input from previous layer neuron, so it is called as dense. Dense Layer is used for classifying image based on output from convolutional layers. Dense layer is also known as fully connected layer as it takes input from all neurons of previous layers. One can add as many dense layers as required in application.

*3) **Training:*** Training a deep neural network refers to the process of giving input data to the network, calculating the output and comparing it to the desired output, and adjusting the network's weights and biases to minimize the error. Error is nothing but a difference between the predicted value and desired outputs. Minimizing the loss is typically done using an optimization algorithm like gradient descent, where network parameters are updated based on a previously computed gradient at each iteration.

*4) **Optimization:*** Optimization in deep neural networks is the process of finding the set of weights and biases that will minimize the error. i.e. loss function. This is done through using optimization algorithms, such as gradient descent and Adam. These algorithms calculate the gradients of the loss function concerning the network parameters and update them in the direction that minimizes the loss. Various optimization techniques, like momentum, learning rate scheduling, and adaptive methods, can be applied to speed up convergence and improve performance.

Adam
The name Adam is inspired by the idea of adapting to change and making the most of every moment.An Adam combines the principles of Gradient Descent with Momentum and the RMSProp algorithm. The Momentum algorithm is employed to accelerate the Gradient Descent process by considering the exponentially weighted average of the gradient.RMSProp is an adaptive learning rate that rises to an improved degree. also, preserve per-parameter learning rates adjusted to the weight based on the average of recent magnitude. 4. here we control the rate of gradient descent in such a way that there is minimum oscillation when reaching the global minimum. it gives higher performance when there is noisy or sparse data

*5) **Activation functions** :* Activation functions decide whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it. The purpose of the activation function is to introduce non-linearity into the output of a neuron. The neural network has neurons that work in correspondence with weight, bias, and their respective activation function. Neural networks would update the weights and biases of neurons on the biases of error at output. This process is known as backpropagation. In an artificial neural network, each neuron forms a weighted sum of input and passes the resulting scalar value through a function as an activation function or transfer function. If a neuron has n input then the output or activation function of a neuron is
a = g(w1x1 + w2x2 + ...+ wnxn)

1. Relu: Relu is the most widely used activation function. It is implemented in a hidden layer. It gives output x if x is positive otherwise zero. It is non non-linear activation function. This means we can easily backpropagate the errors and have multiple layers of neurons being activated by the Relu function. Relu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. At a time only a few neurons are activated making the new network sparse and easy and efficient for computation.
2. Softmax: The softmax function is also the type of sigmoid function but it is handy when we are trying to handle multi-class classification problems. It is a non-linear activation function. It ranges between 0 to 1. The softmax function was commonly found in the output layer of image classification problems. The softmax function would squeeze the output for each class between 0 to 1 and also divide by the sum of the output.
*6) **Performance Evaluation Parameters**:* There are many Performance Evaluation Parameters like Accuracy, Precision, Recall, F1 score.

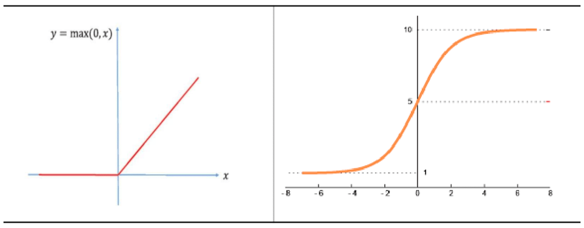

Fig. 2. relu and softmax

Accuracy: Accuracy is one of the performance evaluation criteria. formula : accuracy = (number of correct predictions) / (total number of predictions)

Precision: Precision is one of the criteria for model's performance – the quality of a positive prediction made by the model. formula : (number of true positives)/(total number of positive predictions)

Recall: Recall is also calle True Positive Rate(TPR) formula : True Positives / (True Positives + False Negatives) F1 Score : It is one of the models's performance evaluation criteria. It is a combination of Precision and Recall. formula : 2*((Precision * recall) / (Precision + recall))

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

B. Experimental Result

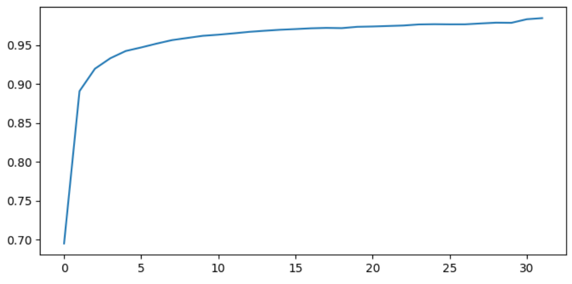

Fig. 3. model1

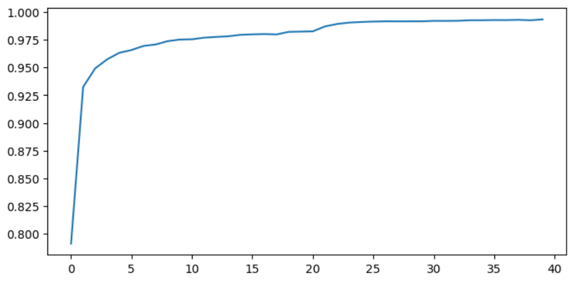

Fig. 4. model2

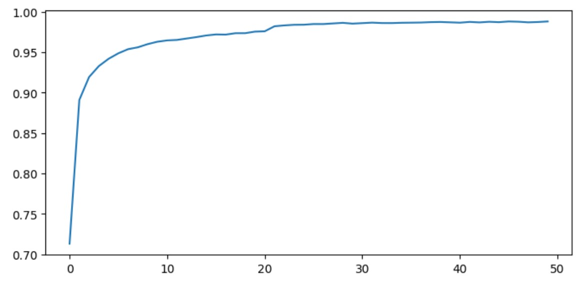
Fig. 5. model3

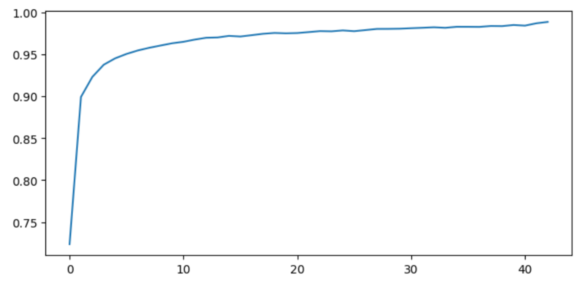
Fig. 6. model4

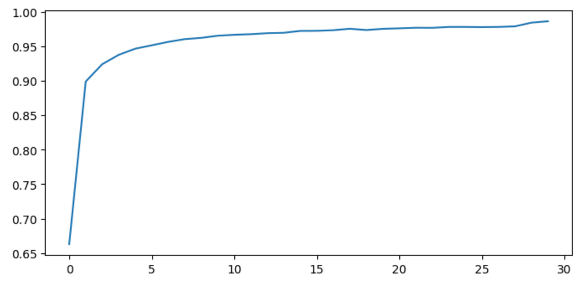
Fig. 7. model5

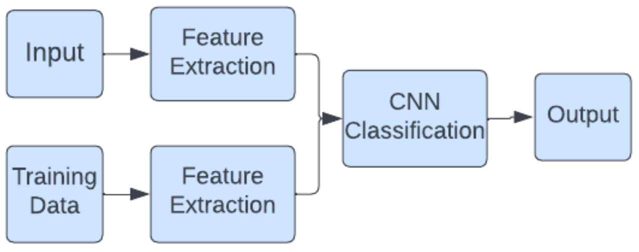

Fig. 8. Convolutional Neural Network

C. Design Details

The process of converting a scanned Devanagari handwritten document into a digitized version involves several key steps as shown in Fig. 1, ensuring that the final output is an accurate representation of the original document. This process can be outlined as follows:
1.Scanned Devanagari Handwritten Document: The initial step involves the utilization of a scanned Devanagari handwritten document as the input.

2.Preprocessing: The scanned document undergoes preprocessing to enhance image quality. This phase includes tasks such as noise reduction, contrast adjustment, and image resizing. The aim is to optimize the document's visual quality for precise character recognition.

3.Segmentation: The document is then divided into individual characters or words. Character segmentation is a critical process for distinguishing and recognizing each unit accurately.

4.Feature Extraction: After characters are isolated, feature extraction is performed. This step involves the extraction of relevant characteristics from each character. Commonly extracted features include pixel intensity values, edges, and gradients.

5.Character Recognition: Following feature extraction, the system processes the extracted features using a character recognition model. This model can be a Convolutional Neural Network (CNN) or other machine learning algorithms.

6.Classification: The recognized features are subsequently classified into specific characters. Classification assigns each character to its corresponding class or symbol within the Devanagari script.

7.Output: The recognized characters are provided as the output, resulting in the creation of a digitized version of the scanned Devanagari handwritten document.

8.End Result: The final outcome is a precise, digital representation of the original handwritten document. This digitized version makes the document accessible and searchable, facilitating its use in various applications. The whole functioning is shown in Fig. 8

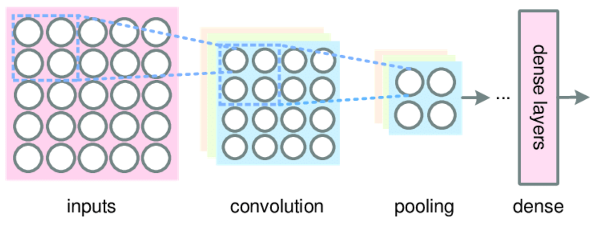

Fig. 9. Convolutional Neural Network

IV. Experimental Setup

Research Implementation Details:
Data Collection and Preprocessing: Collect a dataset of characters in the Devanagari script. Apply data augmentation techniques using ImageDataGenerator for diverse training data.

Model Architecture: Implement a Convolutional Neural

Model Performance :

Table 1 : Classification report on validation data

	Accuracy Score	Precision Score	Recall Score	F1 Score
Model_1	0.9856	0.9858	0.9856	0.9856
Model_2	0.9888	0.9889	0.9888	0.9888
Model_3	0.9889	0.9890	0.9889	0.9889
Model_4	0.9892	0.9894	0.9892	0.9892
Model_5	0.9836	0.9838	0.9836	0.9836
Boosting	0.9932	0.9933	0.9932	0.9932

Table 2 : Classification report on sample data

	Accuracy Score	Precision Score	Recall Score	F1 Score
Model_1	0.7857	0.8733	0.7857	0.8153
Model_2	0.8333	0.9415	0.8333	0.8701
Model_3	0.7262	0.8333	0.7262	0.7485
Model_4	0.7857	0.8175	0.7857	0.7895
Model_5	0.8095	0.9200	0.8095	0.8386
Boosting	0.8571	0.9444	0.8571	0.8862

Network (CNN) using TensorFlow/Keras. Use a Sequential model with convolutional layers, max-pooling layers, and dense layers. Increase the number of filters in convolutional layers from 32 to 64. Employ ReLU activation and max-pooling for feature extraction. End the model with a flatten layer, a dense layer with 256 units, and dropout for regularization. Use softmax activation in the output layer for multi-class classification.

Model Compilation and Training: Compile the model using the Adam optimizer, categorical cross-entropy loss, and accuracy as a metric. Implement callbacks such as ReduceLROnPlateau, EarlyStopping, CSVLogger, and ModelCheckpoint during training. Train the model for 50 epochs using the training and validation data generators.

Character Image Prediction: Implement a function to predict characters given a list of character images. Preprocess each character image by converting to grayscale, applying morphological operations, normalizing, thresholding, and resizing. Iterate through pre-trained models, normalize pixel values, and predict class labels for character images. Consolidate predictions using mode aggregation.

Word Segmentation: Preprocess word images by converting to grayscale, applying morphological operations, normalizing, thresholding, and dilation. Analyze horizontal and vertical projections for trimming and identifying character bounds. Extract characters using contours, handling overlapping or merged cases. Return a list of segmented characters for each word.

Model Evaluation: Load validation data, normalize images, predict characters, and calculate classification metrics. Load sample words, split into characters, predict characters, and evaluate model performance. Implement Boosting by aggregating predictions across multiple models.

Research Metrics: Evaluate model performance using metrics such as accuracy, precision, recall, and F1-score. Implement Boosting to improve overall predictions. Analyze results using aggregated metrics and individual model performance.

Conclusion: Summarize research findings and highlight the effectiveness of the proposed model and techniques in character recognition and word segmentation for the Devanagari script.

V. Conclusion

The process of converting scanned Devanagari handwritten documents into digitized versions involves a meticulous series of steps, integrating image processing and deep learning techniques. Through pre-processing, segmentation, feature extraction, and character recognition, accurate representation of the original document is achieved. The utilization of Convolutional Neural Networks (CNNs) and Python's diverse ecosystem underscores the synergy between advanced technologies in this endeavor. The culmination is a precise digital rendition of Marathi handwritten characters, unlocking the rich heritage and cultural potential of Devanagari script. This groundbreaking approach, exemplified by our system's adaptability and progressive accuracy, promises to revolutionize Marathi handwritten text recognition, enhancing accessibility and facilitating its widespread application.

VI. Acknowledgement

We would like to express our sincere gratitude to ours supervisor, Prof. Darshan Mhapasekar , and Dr. N. M. Shivsharan his invaluable guidance, patience, and support throughout the course of this research. His insightful feedback and construc- tive criticism have been instrumental in shaping this work. I would also like to thank the participants who generously gave their time and shared their experiences for this study.Finally, I want to thank my family and friends for their unwavering support and encouragement.

References

[1] Jamshed Memon, Maira Sami, Rizwan Ahmed Khan, and Mueen Uddin. Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). *IEEE Access*, 8:142642–142668, 2020.

[2] Umapada Pal and BB Chaudhuri. Indian script character recognition: a survey. *pattern Recognition*, 37(9):1887–1899, 2004.

[3] Vikas J Dongre and Vijay H Mankar. A review of research on devnagari character recognition. *arXiv preprint arXiv:1101.2491*, 2011.

[4] Mohamed Cheriet, Nawwaf Kharma, Cheng-Lin Liu, and Ching Suen. Character recognition systems: a guide for students and practitioners. 2007.

[5] Ruwei Dai, Chenglin Liu, and Baihua Xiao. Chinese character recognition: history, status and prospects. *Frontiers of Computer Science in China*, 1:126–136, 2007.

[6] Prashant Yawalkar and Madan Kharat. Automatic handwritten character recognition of devanagari language: a hybrid training algorithm for neural network. *Evolutionary Intelligence*, 15, 04 2021.

[7] Munish Kumar, M. Jindal, and Sonika Narang. Devanagari ancient documents recognition using statistical feature extraction techniques. *Sadhana*, 44:1–8, 06 2019.

[8] Sukhjinder Singh, Naresh Garg, and Munish Kumar. Feature extraction and classification techniques for handwritten devanagari text recognition: a survey. *Multimedia Tools and Applications*, 82, 06 2022.

[9] Ambadas Shinde and Yogesh Dandawate. Convolutional neural network based handwritten marathi text recognition. 08 2020.

[10] Munish Kumar, M. Jindal, and Sonika Narang. Devanagari ancient documents recognition using statistical feature extraction techniques. *Sadhana*, 44:1–8, 06 2019.

[11] Shalaka Deore and Pravin Albert. Devanagari handwritten character recognition using fine-tuned deep convolutional neural network on trivial dataset. *Sādhanā*, 45:243, 09 2020.

[12] Duddela Prashanth, R Vasanth Kumar Mehta, and Ramana Kadiyala. Handwritten devanagari character recognition using modified lenet and alexnet convolution neural networks. *Wireless Personal Communications*, 122, 01 2022.

[13] Teófilo E de Campos, Bodla Rakesh Babu, and Manik Varma. Character recognition in natural images. 1:273–280, 2009.

[14] Diptee Chikmurge and Raghunathan Shriram. Marathi handwritten character recognition using svm and knn classifier. 2019.

[15] Ranadeep Dey, Pranav Gajanan Gawade, Ria Sigtia, Shrushti Naikare, Atharva Gadre, and Diptee Chikmurge. A comparative study of handwritten devanagari script character recognition techniques. *2022 IEEE World Conference on Applied Intelligence and Computing (AIC)*, pages 431–436, 2022.

[16] Anupama Thakur and Amrit Kaur. Devanagari handwritten character recognition using neural network. *International Journal of Scientific & Technology Research*, 8:1736–1744, 2019.

[17] Shailesh Acharya, Ashok Kumar Pant, and Prashnna Kumar Gyawali. Deep learning based large scale handwritten devanagari character recognition. *2015 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, pages 1–6, 2015.

[18] Suprava Patnaik, Saloni Kumari, and S. Das Mahapatra. Comparison of deep cnn and resnet for handwritten devanagari character recognition. *2020 IEEE 1st International Conference for Convergence in Engineering (ICCE)*, pages 235–238, 2020.

[19] Deepak Chaudhary and Kaushal Sharma. Hindi handwritten character recognition using deep convolution neural network. *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 961–965, 2019.