





## Enhanced Recognition of Offline Marathi Handwriting via a Self-Additive Attention Mechanism

Diptee Vishwanath Chikmurge<sup>1,2\*</sup>, Shriram Raghunathan<sup>1</sup>

<sup>1</sup> School of Computing Science and Engineering, VIT Bhopal University, Madhya Pradesh 466114, India

<sup>2</sup> School of Computer Engineering, MIT Academy of Engineering, Alandi, Maharashtra 412105, India

Corresponding Author Email: [diptee.chikmurge2018@vitbhopal.ac.in](mailto:diptee.chikmurge2018@vitbhopal.ac.in)

Copyright: ©2023 IIETA. This article is published by IIETA and is licensed under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

<https://doi.org/10.18280/ts.400646>

### ABSTRACT

**Received:** 23 April 2023

**Revised:** 18 August 2023

**Accepted:** 12 September 2023

**Available online:** 30 December 2023

#### Keywords:

*optical character recognition, bidirectional long short-term memory, convolutional neural network, encoder-decoder, and connectionist temporal classifier*

**Background:** Offline Marathi handwriting recognition presents a significant challenge due to the script's complexity and the variability of individual writing styles. **Methods:** In the present work, an advanced encoder-decoder framework is introduced, wherein a novel self-additive-attention mechanism is integrated. This model capitalizes on a Convolutional Neural Network paired with a Joint Scale Feature Extractor (CNN-JSE) to discern low-level image features within the IIIT-HW-Dev dataset. Such features are subsequently fed into an encoder model that employs a dual-phase fusion process: initially leveraging a Bidirectional Long Short-Term Memory (BiLSTM) network, followed by the self-additive-attention mechanism to maintain dependencies over extensive sequences. **Results:** The fusion output, comprising BiLSTM and self-additive attention data, is conveyed to a Connectionist Temporal Classifier (CTC) decoder. This decoder adeptly identifies character sequences within Marathi words. The introduction of self-additive attention alongside BiLSTM is instrumental in preserving dependencies that are both long-range and multi-stage. **Performance Evaluation:** The efficacy of the proposed system was rigorously evaluated on the multi-author IIIT-HW-Dev dataset. Performance metrics, specifically Character Error Rate (CER) and Word Error Rate (WER), were employed for benchmarking against various established models. **Conclusion:** The proposed methodology demonstrates a significant enhancement in the recognition of handwritten Marathi text, thus facilitating the conversion of handwritten documents into machine-editable text. The utilization of self-additive attention within the BiLSTM framework underscores its potential in capturing complex dependencies, setting a new precedent in automated handwriting recognition technologies. **Significance:** This study not only paves the way for increased accuracy in handwritten text recognition but also contributes a novel approach to the encoding of feature sequences, which may be applicable to a broad range of pattern recognition applications.

## 1. INTRODUCTION

Offline handwritten word recognition constitutes a predominant segment of the Optical Character Recognition (OCR) domain [1]. This task is extensively utilized across various applications, including the scanning of medical reports, recognition of words in doctor prescriptions [2], conversion of ancient manuscripts [3] into machine-editable formats, automated assessment of answer sheets, digit identification on cheques, and more. The surge in handwritten content during the COVID pandemic has underscored the necessity of processing this data and transitioning it into a format amenable to machine editing. Consequently, the digitization of documents has garnered substantial attention within the research community, with a particular focus on the automatization of word recognition.

The digitization of the Marathi script—the language predominantly used in the Indian state of Maharashtra—presents a formidable challenge that demands increased scholarly focus to render it into an editable format. Historically, several algorithms have been developed to analyze and

comprehend the recognition tasks of individual characters and digits within Marathi manuscripts [3-5]. The Devanagari script, which encompasses the Marathi language, consists of 10 numerals, 11 vowels, and 37 consonants. A distinctive feature of this script is the 'Shirorekha'—a horizontal line to which characters adhere. This line extends as characters congregate to form meaningful words. Additionally, the script employs a systematic concatenation of consonants and vowels, giving rise to modified characters and compound characters when two consonants are juxtaposed.

The potential of Marathi Handwritten Text Recognition (MHTR) to serve as a conduit between archival materials, cultural preservation, and contemporary technological advancements is immense. Marathi, with its esteemed literary history, spans several centuries. By translating handwritten Marathi texts into digital formats, recognition algorithms can facilitate the preservation of this heritage, ensuring its easy accessibility and wider dissemination. The ability to digitize these texts enables scholars, historians, and linguists to engage with these documents more thoroughly. Moreover, this technological advancement fosters the integration of the

Marathi language into modern communication technologies, thus benefiting the broader Marathi-speaking populace. MHTR, by mechanizing the transcription process, not only enhances linguistic comprehension but also promotes cultural appreciation and the exchange of knowledge, contributing significantly to the field of digital preservation.

Recognizing words and sentences in the local Devanagari script is inherently complex. This complexity escalates when moving from the recognition of individual characters or digits to words or sentences, owing to the expansive vocabulary inherent to the Devanagari language. Attempts at implementing Marathi handwritten numeral recognition using segmentation methods [5] have encountered performance degradation due to elevated error rates. The Hidden Markov Model (HMM) [4], a machine learning approach, has been adapted to distinguish Malayalam vowels and capture long-range correlations in input feature sequences, yet exceptions are noted in its efficacy.

Handwritten word recognition algorithms that circumvent segmentation have been developed, showing promise for languages such as Arabic and English. These leverage advanced deep learning algorithms that do not require segmentation, including CNN-Gated Recurrent Unit with CTC, Recurrent Neural Network (RNN) with CTC, and CNN-LSTM with CTC. These models have demonstrated superior performance compared to HMM-based approaches. However, despite the structure of RNNs, which includes fixed hidden states and layers, past information is encapsulated within fixed-span vectors. Consequently, RNNs, along with LSTM and GRU architectures, struggle to manage the long-range context required for self-adding attention in word inputs.

Variations in individual handwriting and diverse writing styles have historically impeded accurate recognition of Marathi handwritten text. The intricate ligatures and cursive nature of Marathi scripts pose significant challenges for existing recognition algorithms. Additionally, the limited size of datasets composed of handwritten Marathi samples restricts the effectiveness of training processes, leading to suboptimal performance. Factors such as inconsistent line spacing, slant, and size variations further complicate recognition efforts. The absence of reliable preprocessing techniques for text augmentation and noise reduction further influences the overall quality of recognition. These challenges underscore the urgent need for more advanced and adaptable MHTR techniques.

Within this paper, Section 2 provides a comprehensive literature survey, identifying research gaps and discussing pertinent theories and prior work. Section 3 details the methodology, elucidating the proposed system design and various relevant approaches. Section 4 presents an analysis of experimental results and contrasts the proposed model with existing ones. Finally, Section 5 offers a conclusion, summarizing the proposed work, and delineates future endeavors aimed at resolving the issues identified in the proposed methods.

## 2. RELATED WORK

Previous research endeavors in Devanagari script recognition were predominantly confined to the domain of printed text. For the task of recognition, various methodologies such as K-nearest neighbors (KNN) and multi-layer perceptrons have been employed, with a comprehensive

review of these methods provided in the research [6]. The current project shifts the focus to handwritten word images, which present greater challenges than their printed counterparts.

In the realm of scene text recognition, which has recently seen a surge in interest, comprehensive surveys on the technology can be found in the researches [7, 8]. According to earlier studies, scene text can be categorized into two types: regular and irregular. Regular text generally exhibits a near-horizontal orientation [9, 10], while irregular text is characterized by unpredictable and sometimes distorted patterns.

Various neural network approaches have been explored for the development of an automatic handwritten character recognition system, as reviewed in the researches [11-13]. The objective functions of Recurrent Neural Networks (RNNs) [14] serve as the basis for analyzing typical OCR tasks. For instance, an LSTM network employing a Connectionist Temporal Classification (CTC) objective function has been devised for recognizing Arabic phrases in images, as introduced through the ADOCR benchmark datasets. Moreover, a model that integrates Convolutional Neural Networks (CNNs) with LSTM networks has been proposed for the identification of Amharic text forms.

Recent advancements suggest that attention-based sequence-to-sequence models are becoming the forefront of handwritten text recognition technology [15, 16]. RNNs are commonly employed to model the temporal dynamics of text, and when coupled with an attention mechanism, they excel at pinpointing critical details at each timestep [13]. Persian handwritten words are recognized using both CNNs and RNNs, with a CTC loss function to obviate the need for the segmentation step required by traditional methods [17].

The recognition of handwritten Chinese text without the necessity for character segmentation is achievable through the use of Multi-Dimensional Long-Short-Term Memory Recurrent Neural Networks (MDLSTM-RNNs) [18]. Many Chinese text recognition systems discussed in the literature rely on segmenting text images into individual characters. This segmentation is potentially problematic, as it is a preliminary step to recognition, and any errors incurred can significantly impact overall system performance. Hybrid CNN-RNN models [19] have been introduced for lexicon-free handwriting recognition in Devanagari and Bangla scripts. Employing synthetic data for pre-training has shown promise in enhancing recognition capabilities for languages with limited datasets. Additionally, the attention-based sequence-to-sequence model [20] addresses transfer learning challenges in handwritten word recognition by initially training on scene text images and subsequently adapting to refine handwriting recognition models in light of scarce training data. A recurrent network model incorporating CNN, embedding layers, and LSTM has been developed for encoding linguistic features from images [21]. It features a joint decoder for visual and linguistic features and leverages VGG16 and DEEP CNN models [22] for offline signature verification.

## 3. METHODOLOGY

### 3.1 Feature extraction using CNN-JSE

In the proposed work, CNN-JSE, as shown in Figure 1, is employed to extract collaboratively scaled features from

images. CNN-JSE architecture employs a Multi-Scale Patch (MSP) [23], with heterogeneous dimensions of Kernels. The feature extraction task is completed in two steps: First, Convolutional features are retrieved with conventional Convolutional layers, which contain the sole kernel dimension, and second, MSP retrieves the discrete features with respect to multiple filters having divergent kernel dimensions. The feature maps generated by MSP at various ranges represent the distinct patterns in the image, and at the end, all feature maps are concatenated to pass to the next stage. Feature matrix maps are computed in the first step as per the following Eq. (1), where the input image is shown by  $I$  and kernel by  $x$  and  $y$  are indexes of rows and columns of the resultant feature matrix map, respectively.

$$F_{(x,y)} = (I * K)[x, y] = \sum_a \sum_b K[k_h, k_w] I[X - j, Y - k] \quad (1)$$

The  $F_{(x,y)}$  results are fed to second step using the MSE approach and  $F_j$  are CNN-JSE features used for further sequence feature maps.

The CNN-JSE features are represented by  $F_j$  in Eq. (2):

$$F_j = \text{Concat} [F_j^p, F_j^q, F_j^r, F_j^t] \quad (2)$$

Let's  $p, q, r$  and  $t$  represent divergent dimension kernels with sizes of  $3 \times 3, 5 \times 5, 7 \times 7$ , with  $j$  representing the total number of kernels used for each filter. In this work, a total 32 number of kernels are embedded for each size of the kernel. The relevant features are extracted using CNN-JSE concatenated using spatial max pooling and convolution layer, whereas spatial max pooling is mounted inside  $9 \times 9$  regions with a stride of one. The final convolutional layer reacted with 36 kernels of size  $5 \times 5 \times 36$  kernel bank. In the end, the ReLu activation function is non-linearly fired to activate the relevant feature matrix.

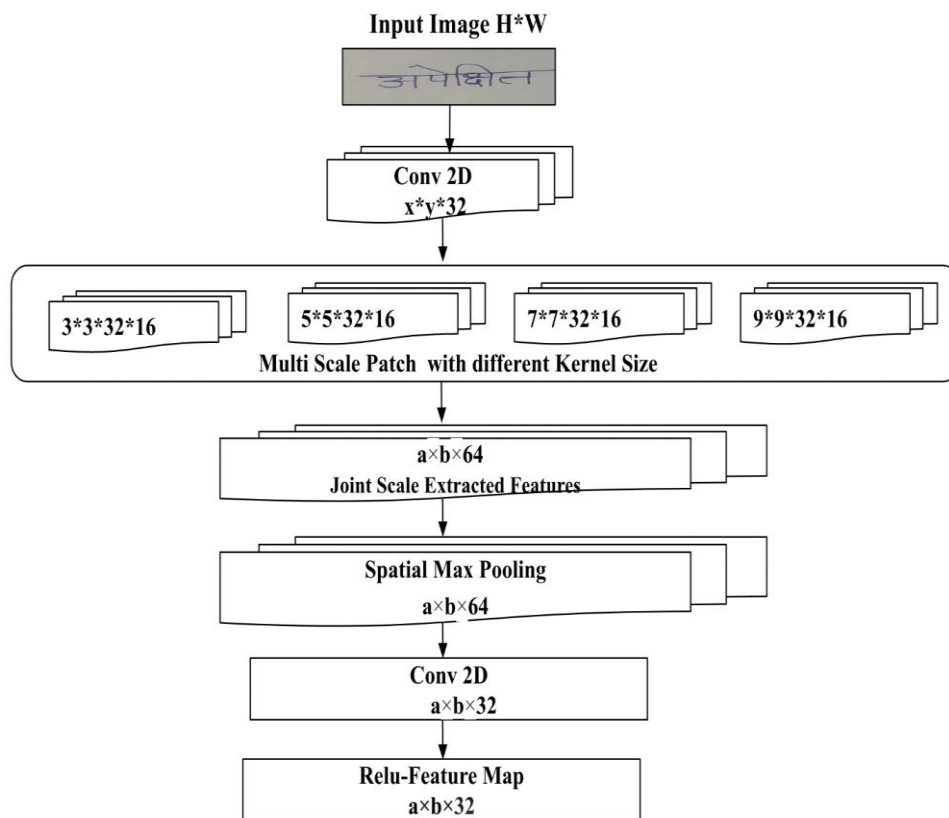


Figure 1. CNN-JSE Architecture

The input image, as shown in Figure 2a, is passed through the CNN-JSE algorithm to extract the low level features that are visualized for 36 filters, as shown in Figure 2b.

In order to extract hierarchical information from input images, CNN-JSE feature maps are essential in MHTR. Convolutional neural networks with various receptive field sizes are used to create these feature maps. The model gathers contextual and fine-grained information from handwritten text by examining several scales, which improves the model's ability to detect letters precisely. By using a multiscale approach, the model is able to understand both regional text patterns and global character forms. The features that CNN-JSE learned to visualize are those in Figure 2b. be passed as input to encoder framework.

### 3.2 Encoder-decoder structure

The encoder-decoder structure in the proposed work accepts input in the form of a series of attribute vectors, so it is necessary to map the feature set that the CNN-JSE model generates into sequences.

So the Feature set is  $FS = \{f_{s_1}, f_{s_2}, f_{s_3}, f_{s_4} \dots \dots \dots, f_{s_n}\}$  where each  $f_s$  feature. The set or channel undergoes a process of reshaping in which each feature of the set, denoted as  $f_s$ , has a size of  $a \times b$ . Here,  $a$  and  $b$  represents the height and breadth of the feature map, respectively. Additionally, the variable  $n$  represents the total number of filters present in the CNN-JSE collaborative model [24]. Consequently, the feature set  $FS$  consists of  $n$  rows. The feature set is arranged in rows

and structured in sequences. MHTR tasks often include a sequence-to-sequence mapping, where the input consists of a sequence of attribute vectors that are converted into a sequence of characters.

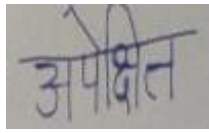


Figure 2a. Input image

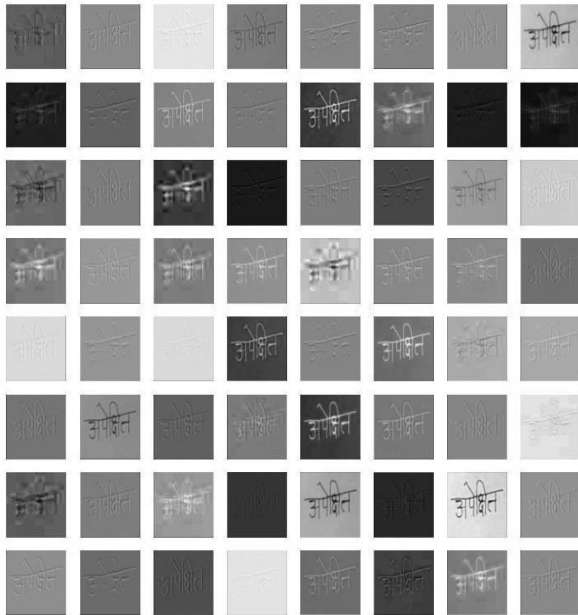


Figure 2b. The visualization of CNN-JSE feature samples

decoder framework using a self-additive-attention mechanism will take the place of segmenting handwritten words. In short, the encoder model is responsible for converting CNN-JSE feature sets into a series of character tokens. In the BiLSTM encoder, the image features are sequentially modeled, allowing for the representation of the input from the CNN-JSE feature set  $FS$ . This process involves resizing the feature set  $FS$  into a series of feature  $FS$ , where each column in the feature set  $fs_i$  corresponds to a member of the series  $FS$  and is employed as a frame within the series. The BiLSTM can be a sufficiently accomplished encoder task but is unable to manage long-scale and multi-extent dependencies over input series. If the number of characters in the image is greater, then BiLSTM is not sufficient to preserve long-term dependency (Figure 3).

So, to handle long-scale and multi-extent dependencies, an extra self-additive-attention layer is built into the encoder framework along with BiLSTM and then followed by a decoder. The decoder comes after the encoder framework in the proposed work, which also integrates BiLSTM and self-additive attention.

### 3.2.1 Bidirectional Long Short-Term Memory (BiLSTM)

The feature series from the previous CNN-JSE module passed through the BiLSTM and Self-additive-attention mechanisms. The feature set size  $a \times b \times 32$  passed through BiLSTM. For sequence matching, a BiLSTM [18, 20] is used. This makes it possible to store facts that depend on each other in a two-way sequence and contributes to the recognition of character series, which is more stable and good for perceiving characters on their own. Ultimately, two LSTMs are assembled in forward and backward directions to train a high degree of extraction. At each time step, the BiLSTM network successfully generates predictions from label space.

The following formula represents the sequence labeling using BiLSTM:

The BiLSTM result expressed as in Eq. (3)

$$ypred_t = (w_{hsypred} hs_t) \tag{3}$$

where,  $ypred_t$  is predicted output,  $w_{hsypred}$  is a weight parameter and  $hs_t$  is the hidden layer description at  $t^{th}$  time step and computed in Eq. (4)

$$hs_t = f(hs_{t-1}, fs_t) = \tanh(w_{hsh}hs_{t-1} + w_{fh}fs_t) \tag{4}$$

where,  $fs_t$  is the input at  $t^{th}$  time step,  $hs_{t-1}$  is hidden layer description at  $(t-1)^{th}$  time step,  $w_{fh}$  and  $w_{hsh}$  are the weight parameters.

Basically, Multiscale CN-JSE provided the BiLSTM module with a feature vector of 32 sequences, each of length  $a \times b$ , as input. The BiLSTM model generates the token characters for the words to carry out the sequence labeling for word recognition.

### 3.2.2 Self-additive-attention

The self-additive-attention mechanism, as presented in the work [13, 20], is included in the encoder to effectively preserve long-range and multi-level dependencies in the encoder's output. Its output is added to BiLSTM output for passing to the decoder. The self-additive-attention mechanism employed ahead can cover the challenges mentioned by using a variable-length weighted context structure in the BiLSTM model. The features set in sequence propagate through the

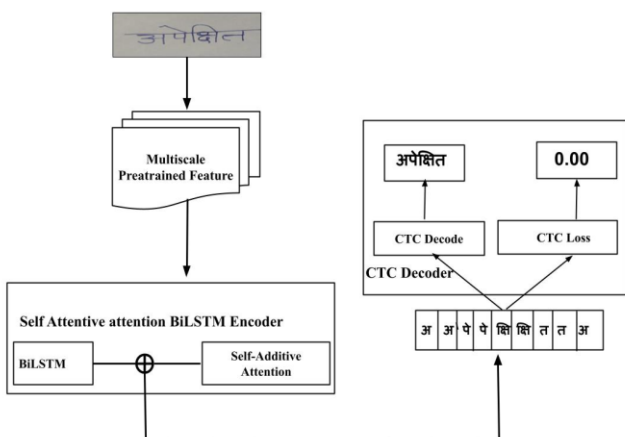


Figure 3. The encoder-decoder structure for MHTR

The CNN-JSE algorithm extracts observable, relevant feature sequences from the input data. These sequences are then used to construct a target sequence, which consists of tokens representing the characters seen in the input handwritten text image. The primary aim of our study was to ascertain a suitable correlation between two input and output sequences of varied lengths, with the purpose of effectively transforming one sequence into the other. The efficiency of the encoder-decoder structure diminishes rapidly as the length of the input word increases. In the proposed work, an encoder-



input embedding and orientation/positional encoding layers of the self-additive attention model to capture knowledge about the orientation of each character in the input image. The self-additive-attention technique is carried out on the output of the CNN-JSE layer to obtain context through various layers.

At each point in the CNN-JSE-produced input feature sequence, the first layer uses a positional encoding technique to implement an attention mechanism. The second layer utilizes a fully linked layer to perform feature reduction. In the third layer, a self-additive mechanism is used to abstract the relationship between the arrangement of a single sequence and the representation of the sequence.

The self-additive-attention technique [25] involves the evaluation of a context vector. This vector is then concatenated with the output of a BiLSTM and passed through a fully connected network and a softmax activation function as formulated in Eq. (5). The resulting output is fed into a decoder, which contains the most important information from all the hidden states of the encoder.

$$E_o = \text{softmax}((\text{BiLSTM}(FS), \text{selfaddatten}(FS))) \quad (5)$$

where,  $E_o = (E_{o1}, E_{o2} \dots \dots E_n)$  to denote a sequence of probabilities of sequence labels and whereas  $(FS = (f_{s1}f_{s2}, \dots \dots f_{sn}))$  features map sequence generated by CNN-JSE.

### 3.3 Decoder

The decoder parses the encoder's sequence of label likelihood into the final target sequence at the top of the encoder and also computes the CTC [17, 26] loss. The decoding process utilizes a mathematical approach to calculate the final label sequence with the greatest probability, based on a series of label probabilities. In this study, the decoder is constructed using the CTC method in order to calculate the conditional probability.

To reduce the CTC loss [26, 27], Adam and a stochastic gradient descent optimizer are used to train the entire model from beginning to end. The way CTC views input-output positioning is crucial to determining this likelihood. The CTC algorithm does not require input and output positioning; it is a positioning-free algorithm. To calculate the likelihood of an output given an input, CTC instead adds up the likelihood of all potential positionings between the two. Understanding these positions will help us comprehend how the loss function is eventually calculated. In order to avoid not understanding the positioning between the input and the output, CTC was developed. With the help of the CTC positionings, there is a straightforward path from the probabilities at each time step to the likelihood of an output sequence.

An encoder is frequently employed by models trained with CTC to calculate the probability for each time step. They may apply any learning method that generates a grouping across output classes given a constant chunk of the input, but an encoder often performs well since it takes context into account in the input.

Try to maximize the labeling probability given a sequence of inputs  $E_o$  and labeling  $L_o$ , where  $|E_o| \leq |L_o|$ , given the sequence using the maximum likelihood estimate.

More specifically, the CTC goal for only one  $(E_o, L_o)$  pair is as shown in Eq. (6):

$$p(L_o | E_o) = \sum_{POS \in POS_{E_o L_o}}^n \left( \prod_{ts=0}^{T_s} p_{ts}(POS_{ts} | E_o) \right) \quad (6)$$

When determining the probability for one positioning, the CTC conditional probability progressively diminishes across the set of suitable positions  $POS$ .

## 4. EXPERIMENTAL RESULTS AND ANALYSIS

Marathi uses the Devanagari script for written communication, making the Devanagari Dataset (IIIT-HW-Dev) crucial for retrieving handwritten Marathi text. This dataset emphasizes Devanagari handwriting, making it suitable for training and assessing models for Marathi handwritten text retrieval tasks.

For the purpose of determining the impact of the suggested model, a number of experiments were deployed on the Devanagari Dataset (IIIT-HW-Dev). more than 95K handwritten words in a Devanagari dataset [28]. The datasets for handwritten text recognition that are currently available have a significantly limited scope and originate from 12 contributors in the IIT-HW-Dev dataset.

The dataset contained 95,381 word samples from 12 individuals, representing a range of ages and educational backgrounds. Writers were free to use whichever pen they chose and to create their own unique style of writing.

Forms with clearly defined boxes were used to collect the information. In each box, there was a space to write the displayed word image as well as a reference word image. The gathered data was captured in JPEG format with a resolution of 600 DPI for color using a flatbed scanner. Using morphological techniques, the handwritten word images were recovered and given labels. As input photographs, a few example dataset images are shown in Table 1.

**Table 1.** Experimental results of proposed model

Input Marathi Handwritten Image	Ground Truth and Prediction	Prediction Outcome
	Ground Truth Text - काळा Predicted Text - काका	Wrong
	Ground Truth Text - अंधार Predicted Text - अंधार	Wrong
	Ground Truth Text - शिर Predicted Text - शिर	Correct
	Ground Truth Text - चिंता Predicted Text - चिंता	Correct
	Ground Truth Text - निराशा Predicted Text - निराशा	Correct

Training began with the architecture being initialized and pictures being processed sequentially via BiLSTM layers with incorporated Self additive attention. Gradients were

backpropagated, iterations were made across epochs, the performance of model was verified, and test data were used to assess correctness. The model was able to efficiently encode the input image thanks to this procedure, which created representations for the following decoding tasks. Training data diversity and size are crucial for performance; augmentation techniques like rotation, scaling, and noise addition enhance model handling. Hyperparameters, including LSTM layers, attention mechanisms, and CTC parameters, can be optimized using grid or random search. The validation analysis of an encoder-decoder HTR model evaluates its performance and generalization on unseen data. The model extracts meaningful features from input handwritten text images and generates transcriptions. Metrics like CER and WER quantify transcription precision. The analysis helps identify areas for improvement, such as handling variable handwriting styles, handling noisy images, and balancing local character details and global text structure comprehension. The recognized outputs for a few example images from the IIIT-HW-Dev utilizing the suggested model network are shown in Table 1. The presence of many changes in the underlying text patterns resulted in ambiguity within the original handwritten picture, hence contributing to the bulk of the mistakes seen. Furthermore, if there are errors in the image segmentation, the recognition system may easily misinterpret the updated character.

The present study focuses on the analysis of handwritten Devanagari text, thereby necessitating the use of appropriate assessment metrics such as CER and WER.

CER measures how much of a handwritten text the used HTR model failed to correctly read and works on character rather than word.

CER rate can be computed as depicted in Eq. (7):

$$CER = (R + W + E)/N \tag{7}$$

where,

$R$ =Number of replacement characters.

$W$ =Number of withdrawals of characters.

$E$ =The estimation of character embedding and its frequency.

$N$ =The total count of characters present in the ground truth word.

The outcome of this mathematical expression yields the ratio of characters in the reference text that were inaccurately evaluated by the suggested model. When there are several insertions, CER's output is less likely to be a value between 0 and 1. This number is frequently linked to the proportion of characters whose guesses are wrong. The HTR system performs better when the value is smaller, with a zero CER being a perfect result. If the encoding of paragraphs and sentences containing meaningful words is involved, WER might be more useful. The Eq. (8) demonstrates that the formulation of Word Error Rate (WER) is similar to that of Character Error Rate (CER), with the distinction that the WER algorithm operates at the level of words. It stands for the quantity of word embedding, withdrawals, or withdrawals required to change one sentence into another.

$$WER = (R_w + W_w + E_w)/N_w \tag{8}$$

where,

$R_w$ =Number of replacements of words.

$W_w$ =Number of withdrawals of words.

$E_w$ =Number of embeddings of words.

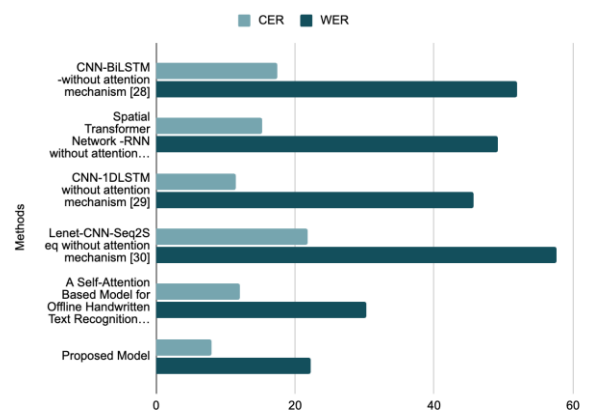
$N_w$ =Number of words in a ground truth sentence.

The performance of the proposed model is evaluated by comparing it with an existing model for text recognition, as seen in Table 2 and Figure 4. The performance of the Devanagari handwritten text recognition system improves when the CER and WER decrease. It is crucial to acknowledge that the values of CER and WER may exhibit variations based on factors such as the particular dataset used, the architecture of the model, the chosen hyperparameters, and the evaluation metrics employed. Hence, it is vital to meticulously devise and assess the model in order to get optimal performance. Table 2 presents a comparative analysis of several methodologies used for the recognition of handwritten text. The findings indicate that the suggested model outperforms other techniques in terms of performance. The parameters are involved more due to the high level features being extracted using the CNN-JSE algorithm. The self additive attention mechanism incorporated with BiLSTM for encoding image feature sequence significantly reduces the CER and WER of the model as compared to the encoder without the self additive attention mechanism.

**Table 2.** Comparative results

Methods	Number of Parameters	CER	WER
CNN-BiLSTM -without attention mechanism [28]	2.1M	17.5	51.93
Spatial Transformer Network -RNN without attention mechanism [28]	2.9M	15.3	49.23
CNN-1DLSTM without attention mechanism [29]	1.5M	11.5	45.76
Lenet-CNN-Seq2Seq without attention mechanism [30]	2.1M	21.8	57.66
A Self-Attention Based Model for Offline Handwritten Text Recognition [25, 31]	4.4M	12.0	30.22
Proposed Model	5.3M	8.02	22.23

**Comparative Analysis**



**Figure 4.** Comparative performance of proposed work with other existing model

**Table 3.** Encoder performance with various numbers of hidden layers in the HTR model

Encoder Framework Model	Hidden Layers	CER	WER
BiLSTM	128	17.5	51.93
LSTM -Self attention	220	16.1	43.44
BiLSTM -Self additive attention (Proposed Model)	128	11.3	34.22
BiLSTM -Self additive attention (Proposed Model)	256	8.02	22.23

The performance of the encoder model with various numbers of hidden layers in BiLSTM and LSTM models is analyzed and compared in Table 3. The encoder model incorporates a CTC decoder for recognition of handwritten text. The optimal number of hidden layers is 256 in the encoder's proposed model. The WER and CER are improved when optimal numbers of hidden layers are embedded in BiLSTM.

Suppose the ground truth Marathi text is "अच्छा" (meaning "good" in English) and a handwritten input image. A MHTR system is used to predict the text from the input image, and the system predicts the text as "अच्ा". To calculate the CER, compare the recognized text to the ground truth text, character by character. In this case, the predicted text has a missing character "छ" compared to the ground truth. This means that the MHTR system has recognized 75% of the characters correctly in the predicted text, and 25% of the characters were incorrect or missing. A lower CER indicates better performance of the recognition system. und truth text. Therefore, the number of deletions is 1. The CER value is 0.25.

## 5. CONCLUSIONS

The encoder uses a CNN-JSE mechanism with multiple variable-sized kernels to extract low-level features from handwritten text scans. The encoder enhances sequence encoding by deploying the self-additive-attention mechanism and BiLSTM to capture long-scale and multi-degree dependencies. A CTC decoder processes the encoded feature sequence. For sequence-to-sequence tasks like text recognition, CTC is frequently employed. Taking into account the varying alignment between the input and output sequences, it aids in mapping the encoded sequence to the actual text labels. The encoder-decoder model combines CNN-JSE, BiLSTM Self-additive Attention processes, and CTC to improve recognition difficulty. The performance is better than existing algorithms, with a 8.02 CER and 22.23 WER on the IIIT-HW-Dev dataset. By setting up the BiLSTM with multiple head self-additive attention processes, the model might be improved even further. This change tries to concurrently capture different patterns and interactions, possibly enhancing the model's comprehension of complicated dependencies.

## REFERENCES

[1] Holambe, A.N., Thool, R.C., Jagade, S.M. (2012). A brief review and survey of feature extraction methods for Devnagari OCR. In 2011 Ninth International Conference on ICT and Knowledge Engineering, Bangkok, Thailand,

pp. 99-104. <https://doi.org/10.1109/ICTKE.2012.6152421>

[2] Dhande, P.S., Kharat, R. (2017). Character recognition for cursive English handwriting to recognize medicine name from doctor's prescription. In 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA), Pune, India, pp. 1-5. <https://doi.org/10.1109/ICCUBEA.2017.8463842>

[3] Shah, K.R., Badgular, D.D. (2013). Devnagari handwritten character recognition (DHCR) for ancient documents: A review. In 2013 IEEE Conference on Information & Communication Technologies, Thuckalay, India, pp. 656-660. <https://doi.org/10.1109/CICT.2013.6558176>

[4] Gayathri, P., Ayyappan, S. (2014). Off-line handwritten character recognition using Hidden Markov Model. In 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Delhi, India, pp. 518-523. <https://doi.org/10.1109/ICACCI.2014.6968488>

[5] Jinturkar, A.A., Khanale, P.B. (2020). Segmentation & recognition of Marathi handwritten numeral strings using multiple features. International Journal of Emerging Technology and Advanced Technology, 10(5): 18-21.

[6] Pal, U., Chaudhuri, B.B. (2004). Indian script character recognition: a survey. pattern Recognition, 37(9): 1887-1899. <https://doi.org/10.1016/j.patcog.2004.02.003>

[7] Zhu, Y., Yao, C., Bai, X. (2016). Scene text detection and recognition: Recent advances and future trends. Frontiers of Computer Science, 10: 19-36. <https://doi.org/10.1007/s11704-015-4488-0>

[8] Ye, Q., Doermann, D. (2014). Text detection and recognition in imagery: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(7): 1480-1500. <https://doi.org/10.1109/TPAMI.2014.2366765>

[9] Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Lee, H. (2019). What is wrong with scene text recognition model comparisons? Dataset and Model Analysis. <https://doi.org/10.48550/arXiv.1904.01906>

[10] Wang, P., Yang, L., Li, H., Deng, Y., Shen, C., Zhang, Y. (2019). A simple and robust convolutional-attention network for irregular text recognition. arXiv preprint arXiv:1904.01375. <https://doi.org/10.48550/arXiv.1904.01375>

[11] Bhujade, R.K., Pandit, A. (2018). A survey on various methodologies of automatic handwritten character recognition using neural network. International Journal of Emerging Technology and Advanced Technology, 8(2): 328-332.

[12] Hemanth, G.R., Jayasree, M., Venii, S.K., Akshaya, P., Saranya, R. (2021). CNN-RNN based handwritten text recognition. ICTACT Journal on Soft Computing, 12(1): 2457-2463. <https://doi.org/10.21917/ijsc.2021.0351>

[13] Kang, L., Toledo, J.I., Riba, P., Villegas, M., Fornés, A., Rusinol, M. (2019). Convolve, attend and spell: An attention-based sequence-to-sequence model for handwritten word recognition. In Pattern Recognition: 40th German Conference, GCPR 2018, Stuttgart, Germany, pp. 459-472. [https://doi.org/10.1007/978-3-030-12939-2\\_32](https://doi.org/10.1007/978-3-030-12939-2_32)

[14] Belay, B.H., Habtegebirial, T., Liwicki, M., Belay, G., Stricker, D. (2019). Amharic text image recognition: database, algorithm, and analysis. In 2019 International conference on document analysis and recognition

- (ICDAR), Sydney, NSW, Australia, pp. 1268-1273. <https://doi.org/10.1109/ICDAR.2019.00205>
- [15] Kang, L., Riba, P., Villegas, M., Fornés, A., Rusiñol, M. (2021). Candidate fusion: Integrating language modelling into a sequence-to-sequence handwritten word recognition architecture. *Pattern Recognition*, 112: 107790. <https://doi.org/10.1016/j.patcog.2020.107790>
- [16] Michael, J., Labahn, R., Grüning, T., Zöllner, J. (2019). Evaluating sequence-to-sequence models for handwritten text recognition. In 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, pp. 1286-1293. <https://doi.org/10.1109/ICDAR.2019.00208>
- [17] Safarzadeh, V.M., Jafarzadeh, P. (2020). Offline Persian handwriting recognition with CNN and RNN-CTC. In 2020 25th international computer conference, computer society of Iran (CSICC), Tehran, Iran, pp. 1-10. <https://doi.org/10.1109/CSICC49403.2020.9050073>
- [18] Messina, R., Louradour, J. (2015). Segmentation-free handwritten Chinese text recognition with LSTM-RNN. In 2015 13th International conference on document analysis and recognition (ICDAR), Tunis, Tunisia, pp. 171-175. <https://doi.org/10.1109/ICDAR.2015.7333746>
- [19] Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V. (2018). Towards accurate handwritten word recognition for Hindi and Bangla. In *Computer Vision, Pattern Recognition, Image Processing, and Graphics: 6th National Conference, NCVPRIPG 2017*, Mandi, India, pp. 470-480. [https://doi.org/10.1007/978-981-13-0020-2\\_41](https://doi.org/10.1007/978-981-13-0020-2_41)
- [20] Kass, D., Vats, E. (2022). AttentionHTR: handwritten text recognition based on attention encoder-decoder networks. In *International Workshop on Document Analysis Systems*, pp. 507-522. [https://doi.org/10.1007/978-3-031-06555-2\\_34](https://doi.org/10.1007/978-3-031-06555-2_34)
- [21] Ngo, T.T., Nguyen, H.T., Ly, N.T., Nakagawa, M. (2021). Recurrent neural network transducer for Japanese and Chinese offline handwritten text recognition. In *International Conference on Document Analysis and Recognition*, pp. 364-376. [https://doi.org/10.1007/978-3-030-86159-9\\_26](https://doi.org/10.1007/978-3-030-86159-9_26)
- [22] Gadre, A., Pund, P., Ajmire, G., Kale, S. (2021). Signature recognition models: Performance comparison. In 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, pp. 1-6. <https://doi.org/10.1109/ICAECA52838.2021.9675598>
- [23] Zeng, L., Xu, X., Cai, B., Qiu, S., Zhang, T. (2017). Multi-scale convolutional neural networks for crowd counting. In 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, pp. 465-469. <https://doi.org/10.1109/ICIP.2017.8296324>
- [24] Li, S., Zhu, X., Bao, J. (2019). Hierarchical multi-scale convolutional neural networks for hyperspectral image classification. *Sensors*, 19(7): 1714. <https://doi.org/10.3390/s19071714>
- [25] Ly, N.T., Ngo, T.T., Nakagawa, M. (2021). A self-attention based model for offline handwritten text recognition. In *Asian Conference on Pattern Recognition*, pp. 356-369. [https://doi.org/10.1007/978-3-031-02444-3\\_27](https://doi.org/10.1007/978-3-031-02444-3_27)
- [26] Liwicki, M., Graves, A., Fernández, S., Bunke, H., Schmidhuber, J. (2007). A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In *Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007*.
- [27] Liu, H., Zhu, Z., Li, X., Satheesh, S. (2017). Gram-CTC: Automatic unit selection and target decomposition for sequence labelling. In *International Conference on Machine Learning*, pp. 2188-2197.
- [28] Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V. (2018). Offline handwriting recognition on Devanagari using a new benchmark dataset. In 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), Vienna, Austria, pp. 25-30. <https://doi.org/10.1109/DAS.2018.69>
- [29] Moysset, B., Messina, R. (2019). Are 2D-LSTM really dead for offline text recognition? *International Journal on Document Analysis and Recognition (IJDAR)*, 22(3): 193-208. <https://doi.org/10.1007/s10032-019-00325-0>
- [30] Sueiras, J., Ruiz, V., Sanchez, A., Velez, J.F. (2018). Offline continuous handwriting recognition using sequence to sequence neural networks. *Neurocomputing*, 289: 119-128. <https://doi.org/10.1016/j.neucom.2018.02.008>
- [31] Bluche, T., Louradour, J., Messina, R. (2017). Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention. In 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, pp. 1050-1055. <https://doi.org/10.1109/ICDAR.2017.174>