Few-Shot Hindi Handwritten Text Recognition Using Meta-Optimized Res-Net-Transformer Networks

[A] **Md Tahseen Equbal**

M.Tech Student, Department of Computer Science and Engineering
All Saints College of Technology, Bhopal, India
Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya (RGPV)
mdtahseen278@gmail.com

[B] **Prof. Sarwesh Site**

Associate Professor, Department of Computer Science and Engineering
All Saints College of Technology, Bhopal, India
Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya (RGPV)
er.sarwesh@gmail.com

ABSTRACT

Handwritten Text Recognition (HTR) in Indic scripts such as Hindi presents significant challenges due to diverse handwriting styles, limited annotated data, and the inherent complexity of Devanagari script. This dissertation proposes a novel few-shot learning framework titled ***RT-MAML (Res-Net-Transformer with Model-Agnostic Meta-Learning)*** *to address these challenges. The architecture combines a Res-Net18 encoder for effective visual feature extraction and a Transformer-based decoder for sequence modeling, trained using a meta-learning strategy to enable rapid adaptation to new writers with minimal data. Unlike conventional CTC-based HTR systems, our model incorporates a cross-entropy- based sequence decoder with beam search, enhancing recognition accuracy. The approach is evaluated on the IIIT-HW Hindi handwritten word dataset and demonstrates a* ***Character Error Rate (CER) of 6%*** *and* ***Word Error Rate (WER) of 13.0%****, outperforming several existing baselines, including CNN- BiLSTM-CTC and Vision Transformer models. Meta-optimization through MAML improves the model's generalization to unseen writing styles, while techniques such as orthogonality regularization and curriculum learning further refine the learning process. The study highlights the potential of combining deep visual encoders, attention-based decoding, and meta-learning for low-resource script recognition tasks. Future work will explore expanding this framework to multilingual handwritten datasets and integrating large language models for semantic-aware decoding. This research contributes to the advancement of intelligent handwriting recognition systems in Indian languages under limited- resource settings.*

Keywords: *Handwritten Text Recognition (HTR), Hindi Script, Res-Net,Transformer Decoder, Connectionist Temporal Classification (CTC), Beam Search Decoding, Character Error Rate*

(CER), Word Error Rate (WER), Indic NLP, Deep Learning, Regularization, Script Complexity,

1 Introduction

1.1. Background

From intelligent tutoring systems and document retrieval to the digitalization of historical manuscripts and postal automation, Handwritten Text Recognition (HTR) is an essential problem in computer vision and pattern recognition. Although deep learning has made great strides in HTR for Latin and English scripts, identifying intricate Indic scripts, such as Hindi (Devanagari), is still a difficult task because of the script's intrinsic complexity. Complex symbols, diacritics, modifiers (matras), and a top-line (Shirorekha) that joins letters are some of the distinctive structural features of Devanagari-written Hindi. Because of these features, traditional optical character recognition (OCR) and deep learning methods are inadequate; they produce visual ambiguities and spatial dependencies that are uncommon in Latin scripts. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers are just a few examples of the deep neural architectures that have lately made strides in solving script complexity. Unfortunately, low-resource languages like Hindi frequently lack the massive amounts of labelled data needed by these algorithms. Also, when the model encounters unseen writers or even minor differences in handwriting styles, its performance degrades. This problem has led to the introduction of meta-learning frameworks like Model-Agnostic Meta-Learning (MAML). Since MAML can quickly learn new jobs with little training data, it is well-suited to few-shot HTR situations in which it would be impossible to gather large labelled samples for each writer. For the purpose of solving the Hindi HTR issue with little data, this dissertation investigates a meta-optimized pipeline based on Res-Net-Transformers (RT-MAML). In order to guarantee effective training and adaptability to writer- specific styles, the architecture integrates beam search decoding, orthogonality regularization, and a curriculum learning technique.

1.2. Motivation

There are more than 600 million people who speak Hindi, making it an essential language for inclusive digitalization and cultural heritage preservation efforts to recognize handwritten writing in Hindi. In spite of its significance, Hindi is still underrepresented in HTR's high-quality labelled datasets. Attempts to apply existing systems to Devanagari script that were trained on English or other Latin-based scripts typically result in failure because of the underlying structural differences between the two. Moreover, there is a great deal of variation in handwritten text both within and between writers. People's writing styles, pen pressure, spacing, and flow all play a role in how characters are fashioned. When entering data from a new writer is difficult, as is often the case in

real-world applications such as mail forms or handwritten applications, this unpredictability becomes exceptionally challenging. With their heavy dependence on supervised training data, traditional deep learning models have a hard time performing well in these limited settings. The few-shot learning and meta-learning paradigms, in particular MAML, provide a workable answer to this problem. This approach allows a model to rapidly learn new activities or writing styles with few labeled instances. An effective and scalable framework for Hindi HTR in low-resource situations may be achieved by combining

these meta-learning approaches with Sequential models like Transformers and sophisticated feature extractors like Res-Net.

1.3. Problem Statement

Despite progress in handwritten text recognition using deep learning, existing models suffer from three major limitations in the context of Hindi script:

1.3.1. **Poor Generalization to New Writers:** Models trained on limited data from a specific set of writers fail to generalize well to unseen handwriting styles.

1.3.2. **Complex Script Challenges:** Devanagari's character set includes compound characters

and visually similar glyphs, increasing inter-class confusion.

1.3.3. **Lack of Efficient Low-Resource Learning:** Existing models require extensive retraining to accommodate new data, which is infeasible for many real-world applications. This dissertation addresses the above limitations by proposing a meta-learned Res-Net-Transformer architecture (RT-MAML) capable of learning from 5-shot labeled samples per class using a 5-way meta-learning setup, achieving rapid adaptation and high accuracy even with minimal data.

1.4. Research Objectives

1.4.1. Using the IIIT-HW dataset as a baseline, this research aims to construct an optimal deep learning pipeline for handwritten Hindi text recognition.

1.4.2. To develop a **meta-learning optimized architecture (RT-MAML)** for efficient few- shot handwritten text recognition in Hindi.

1.4.3. To design an **encoder-decoder model** that integrates **Res-Net-18** for visual feature extraction and a **Transformer decoder** for sequence prediction.

1.4.4. To incorporate **orthogonality regularization** to reduce confusion between visually similar characters in Devanagari script.

1.4.5. To apply **beam search decoding** to improve the accuracy of predicted text sequences over greedy methods.

1.5. Scope and significance of study

The intricate Devanagari script is used to write the Hindi language; this work aims to create a reliable and flexible handwritten text recognition (HTR) system for this script. An optimized deep neural network utilizing Model-Agnostic Meta-Learning (MAML) is the main focus, with a focus on integrating a Res-Net-18 encoder and Transformer decoder into a meta-learning framework. With just a few number of labelled samples, the model can quickly adapt to new handwriting styles since it is trained in a few-shot learning environment. The model employs orthogonality regularization to reduce the likelihood of mistaken identification of visually identical Hindi characters, and it employs beam search decoding to improve the accuracy of sequence predictions. When it comes to low-resource HTR tasks, where data is few and writers are unpredictable, conventional deep learning models aren't able to generalize, which is why this study is important. Automated processing of handwritten forms, archiving of historical documents, and digitalization of educational or governmental data are real-world applications that highlight the importance of this technique. Furthermore, this study adds to the little- studied field of Indic-script HTR, paving the way for future work in this area and perhaps other complicated scripts such as Bengali, Tamil, and Gujarati, which need the creation of scalable and flexible models. Without the luxury of retraining on vast datasets, the suggested technique not only increases accuracy but also provides a practical solution for real-world deployment, where novel handwriting styles are constantly encountered.

2. LITERATURE REVIEW

2.1. Key Insights from Relevant Research

The Over the last 20 years, deep learning-driven techniques have largely replaced more conventional, manually-crafted feature-based methods in the area of Handwritten Text Recognition (HTR). Systems in their early stages used SVMs, HMMs, and statistical feature extractors including zoning, HOG, and projection profiles, among others. These approaches performed adequately on limited datasets, but they failed miserably when asked to generalize to new writers, scripts, or noise levels. With the development of deep learning, HTR models were able to learn rich hierarchical characteristics from raw picture inputs, which led to revolutionary advancements. Improving the performance of sequence prediction models, particularly in Latin scripts, has been made possible by

recent advancements in Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). In HTR pipelines, architectures like as CRNN (Convolutional Recurrent Neural Network) have been popular, particularly when combined with CTC loss. This type of network combines CNNs for spatial feature extraction and Bidirectional LSTMs for temporal modeling. When it comes to scripts like Devanagari, which are quite intricate, these models typically fail because they don't take spatial attention or context into account. Over the past few years, RNNs have been supplanted by attention-based encoder- decoder models and Transformer topologies in HTR and other sequence modeling applications. These models have demonstrated potential in difficult script identification tasks and are very good at capturing long-range dependencies. But they need a lot of data and can easily overfit when resources are scarce, therefore researchers are looking into meta- learning methods like Model-Agnostic Meta-Learning (MAML) to help them get better generalization results with less data. To further improve contemporary systems' capacity to manage writer heterogeneity and script complexity, beam search decoding, orthogonality regularization, and curriculum learning are integrated.

2.2. CNN- and RNN-Based Approaches

An early focus of deep learning models for HTR was CNN-RNN combinations, particularly with LSTM units. Time-series modeling was handled by RNNs, while convolutional neural networks (CNNs) were tasked with gleaning spatial information from word graphics. In particular, Shi et al.'s (2017) Convolutional Recurrent Neural Network (CRNN) was a groundbreaking innovation in this area. Central Tension Reduction (CTN), BiLSTMs, and CTC loss were all included into this fully trainable system. In contrast to their success with English and numerically organized datasets, CRNNs have a hard time adapting to more complicated scripts like Devanagari.

For example, Jain et al. (2020) used synthetic datasets to train a CRNN-based system for Hindi word recognition; the results were acceptable Character Error Rates (CER), but the system exhibited restricted performance on genuine handwriting because to the lack of miscellaneous data. Similar to how Sharma et al. (2021) used a CNN-BiLSTM model on the IIIT-HW dataset, it frequently made inaccurate predictions since it couldn't grasp the contextual information included in complicated conjunct characters and matras (diacritics). Even while these models did a good job of character recognition, they struggled with global context and long- range dependencies, two factors that are essential for accurate HTR in languages like Hindi. As a result of their reliance on sequential processing, CNN-RNN architectures are often slower during training and inference than parallelizable models like as Transformers. Handwritten Hindi documents sometimes have noisy or

extremely cursive input, and these models generally fail when dealing with such inputs and require extensive hyperparameter adjustment.

2.3. Transformer-Based Architectures

Veaswani et al. (2017) pioneered the use of transformers in NLP, which was a paradigm change since they did away with recurrence altogether and relied on self- attention mechanisms to detect series relationships. They had immediate success in text processing, which led them to focus on vision and, eventually, HTR. With the use of attention processes, Baek et al. (2019) shown that scene text recognition utilizing a Transformer decoder can increase recognition of deformed and unevenly shaped text.

Regarding Indic scripts, Rajeswaran et al. (2022) investigated the application of Transformer decoders with CNN-based encoders for Devanagari and Tamil HTR. Due to improved modeling of contextual dependencies, their model outperformed typical RNNs in terms of accuracy. The convolutional neural network (CNN) encoder did, however, limit the receptive field and global feature extraction.

Findings also highlighted the need of script-aware vocabulary and hierarchical decoding structures for better recognition on scripts rich in morphology. A number of studies have demonstrated that when it comes to multilingual and multi-script HTR scenarios, Transformer-based encoder-decoder models perform better than conventional CRNNs. To generate text, architectures like TrOCR (Transformer OCR) and ViT-Hybrid HTR have used autoregressive decoders that rely on visual tokens obtained from CNNs or Vision Transformers (ViTs). Notwithstanding these advantages, Transformers often overfit in low-resource environments and need massive amounts of data for proper training. Consequently, methods like few-shot learning and meta-learning have been investigated as potential solutions to enhance performance in the face of sparse data, particularly for languages with minimal resources, such as Hindi.

2.4. Hindi Script-Specific HTR Studies

Hindi, written in the Devanagari script, presents unique challenges not addressed in most Latin-script HTR models. These include a large number of characters (including vowels, consonants, numerals, and various

matras), complex rules for character positioning, and ligature formation (conjuncts). Unlike Latin alphabets, where characters are linearly aligned, Hindi characters can appear above, below, or around

consonants, creating spatially complex patterns.

Several research efforts have attempted to address these script-specific challenges. Sharma et al. (2022) developed a CNN-LSTM-CTC model trained on the IIIT-HW dataset, which achieved moderate CER but struggled with compound characters and rare conjuncts. Mishra et al. (2021) used handcrafted features combined with a shallow neural network to improve interpretability, but the model lacked scalability for large vocabulary settings. Others have focused on synthetic data generation to increase data availability, but synthetic datasets often lack the variability of natural handwriting.A Several works have attempted to address these challenges using CNNs and LSTMs. For instance, the IIIT-HW dataset was introduced as a benchmark for Hindi handwritten word recognition, and researchers have used it to train CRNN models with moderate success. Some studies have proposed character- level recognition followed by language modeling to improve accuracy. Others have explored spatial transformer networks and attention-based decoders to handle the variability in character positioning.Despite these efforts, existing models often fail to generalize well to unseen handwriting styles or low-data scenarios. This limitation highlights the need for **meta-learning-based models** that can adapt quickly using few examples, making them more suitable for real-world applications involving diverse handwriting inputs.

2.5. Decoding, Regularization, and Optimization Strategies

Decoding recognizing handwritten writing relies heavily on decoding algorithms, particularly when it comes to generating sequences. The most popular approach is greedy decoding, which selects the most likely character at each time step. But when character probabilities are close, greedy decoding can produce less- than-ideal outcomes. This is remedied by using beam search decoding, which keeps track of numerous candidate sequences and chooses the best one using cumulative probabilities. Both CTC-based pipelines and autoregressive decoder-based models benefit greatly from beam search's enhanced performance. To avoid overfitting and enhance model generalization, regularization techniques are just as important as decoding. Popular methods include data augmentation, label smoothing, and dropout. We also use Orthogonality Regularization (OrthoReg) in this study; it makes sure that the learnt feature embeddings are orthogonal, which helps to distinguish between characters that look similar ("उ" vs. "ढ"). This is especially helpful for scripts like Devanagari, where even little changes can have a big impact on meaning. Model-Agnostic Meta-Learning (MAML) is an optimization technique that allows for rapid task adaption with less

data by learning a good initialization of model parameters. For low-resource HTR tasks, this is especially helpful because it enables the model to generalize across various writers and scripts. To tackle the difficult challenge of Hindi handwritten text recognition, a strong and versatile technique is to combine MAML with beam search decoding and Transformer-based decoders.

3. DATASET DESCRIPTION

3.1. Dataset Description

We have utilized the IIIT-HW (Indian Institute of Information Technology
- Handwritten Words) dataset as our main benchmark to design and evaluate the suggested handwritten Hindi text recognition system. This dataset is specifically designed for handwritten word-level recognition in Devanagari script, making it highly suitable for research focused on Hindi Optical Character Recognition (OCR) tasks. IIIT-HW contains a large collection of offline handwritten word images representing a wide range of vocabulary used in real-world Hindi texts, including educational and administrative contexts.

3.2. Dataset Structure

Three main categories make up the dataset:

a. **Training Set**: Comes with labels that go with about 70,000-word pictures. Feature representations and character sequences are taught to the model using this subset during training.

b. **Validation Set:** Consists of around 15,000-word images. It is used for hyperparameter tuning and early stopping to avoid overfitting.

c. **Test Set**: Made up of almost 15,000-word images. The purpose of this set is to test how well the final model does in a hypothetical situation.

Each image in the dataset is grayscale, and most samples are centered on a uniform aspect ratio, though the actual dimensions vary depending on the word length. The words include a variety of Hindi vocabulary— common nouns, verbs, numerals, and grammatical forms—providing comprehensive linguistic coverage for real-world applications.

3.3. Vocabulary and Script Coverage.

The IIIT-HW dataset includes words composed of characters from the Devanagari script, covering:

a. **Consonants and vowels** (स्वर और व्यंजन)
b. **Dependent vowel signs** (मात्राएँ)
c. **Anusvara** (ं)**,Visarga** (ः)**, and Chandrabindu** (ँ)
d. **Numerals** (०–९)
e. **Conjunct characters** (संयुक्ताक्षर), which are particularly challenging due to their shape and positioning.

The character set includes over 200 unique graphemes, including base characters and diacritical forms. This diversity enables robust training of character recognition models and ensures broad applicability of the method to various types of Hindi handwriting.

3.4. Preprocessing and Augmentation

Various preprocessing processes are implemented prior to supplying photos to the model:

a) **Resizing:** By utilizing padding, all word images are enlarged to a consistent resolution (for instance, 384×384 pixels) while preserving their aspect ratio.

b) **Normalization:** In order to enhance convergence, the pixel values are transformed to fall inside the [0,1] range.

c) **Noise Reduction:** Basic image filtering (median blur or Gaussian blur) is optionally applied to reduce scanner noise or stroke irregularities.

To enhance generalization, especially for unseen handwriting styles, advanced data augmentation techniques are applied during training, including:

d) Elastic distortions to simulate pen pressure variations

e) Affine transformations to introduce slant and rotation

f) Perspective warping to mimic camera-based captures or skewed scanning

These augmentations help the model learn invariance to distortions commonly observed in natural handwriting.

4. PROPOSED METHODOLOGY

4.1. Proposed Methodology

The As a complicated Indic script with rich morphology and various writing styles, Hindi presents unique difficulties for handwritten text recognition (HTR), which the suggested architecture aims to solve. The architecture allows for strong recognition in situations with limited data by combining advanced sequence modeling with convolutional feature extraction. Its four main parts are an encoder that uses Res-Net, a projection head to align dimensions, a decoder that uses Transformer and is trained using Connectionist Temporal Classification (CTC) loss, and a beam search decoding module that can be added for better inference. Orthogonality Regularization (OrthoReg) and Model-Agnostic Meta-Learning (MAML) are used to optimize the training pipeline further, making it more adaptable and generalizable. Model-Agnostic Meta- Learning (MAML) is a meta- learning paradigm that, when embedded into the encoder-decoder framework, makes it possible for the model to quickly adapt to new character styles or unseen writers by mimicking low-

data training situations during meta-training. For difficult scripts like Hindi that have little annotated data, this meta-optimization technique greatly improves the system's generalizability. By applying Orthogonality Regularization (OrthoReg), we can make sure that the features between classes stay as distant as possible in the embedding space and significantly increase the learned features' discriminative ability. Furthermore, inference employs beam search decoding to enhance sequence predictions; this method, which avoids depending on greedy predictions in favor of considering numerous hypotheses at each decoding stage, is effective. By utilizing multiple paths, the Character Error Rate (CER) and Word Error Rate (WER) are considerably decreased in comparison to traditional decoding methods. A strong, flexible, and efficient solution for Hindi HTR is provided by this well-planned architecture.

4.2. Data preprocessing

Effective preprocessing is critical for preparing handwritten word images for training and inference. The following steps are applied to each input image:

4.2.1. **Resizing and Normalization:** The word graphics are all downsized to 384×384 pixels in size, and their aspect ratio is maintained by utilizing zero-padding. Prior to normalization, the pixel values are set to the interval [0, 1].

4.2.2. **Grayscale Conversion:** Although the IIIT-HW dataset is already grayscale,

additional scans or user data may require channel reduction.

4.2.3. **Noise Removal:** Median and Gaussian filters are optionally used to reduce background noise or ink artifacts from scanning.

4.2.4. **Advanced Data Augmentation:** To enhance generalization and model robustness, the following augmentations are applied during training:

a) Elastic distortions to simulate human writing pressure and curvature

b) Perspective warping to introduce skewed viewpoints

c) Affine transformations (scaling, rotation, shearing) to mimic natural variability

d) Random erasing to simulate missing strokes

e) Contrast and brightness jittering to emulate different pen/paper qualities

These augmentations help the model learn invariant features that generalize across diverse handwriting styles, including various matras and conjuncts in the Devanagari script.

4.3. Proposed Architecture

The As a complicated Indic script with rich morphology and various writing styles, Hindi presents unique difficulties for handwritten text recognition (HTR), which the suggested architecture aims to solve. The architecture allows for strong recognition in situations with limited data by combining advanced sequence modeling with convolutional feature extraction. Its four main parts are an encoder that uses Res-Net, a projection head to align dimensions, a decoder that uses Transformer and is trained using Connectionist Temporal Classification (CTC) loss, and a beam search decoding module that can be added for better

inference. Orthogonality Regularization (OrthoReg) and Model-Agnostic Meta-Learning (MAML) are used to optimize the training pipeline further, making it more adaptable and generalizable. Model-Agnostic Meta- Learning (MAML) is a meta- learning paradigm that, when embedded into the encoder-decoder framework, makes it possible for the model to quickly adapt to new character styles or unseen writers by mimicking low- data training situations during meta-training. For difficult scripts like Hindi that have little annotated data, this meta-optimization technique greatly improves the system's generalizability. By applying Orthogonality Regularization (OrthoReg), we can make sure that the features between classes stay as distant as possible in the embedding space and significantly increase the learned features' discriminative ability. Furthermore, inference employs

beam search decoding to enhance sequence predictions; this method, which avoids depending on greedy predictions in favor of considering numerous hypotheses at each decoding stage, is effective. By utilizing multiple paths, the Character Error Rate (CER) and Word Error Rate (WER) are considerably decreased in comparison to traditional decoding methods. A strong, flexible, and efficient solution for Hindi HTR is provided by this well-planned architecture.

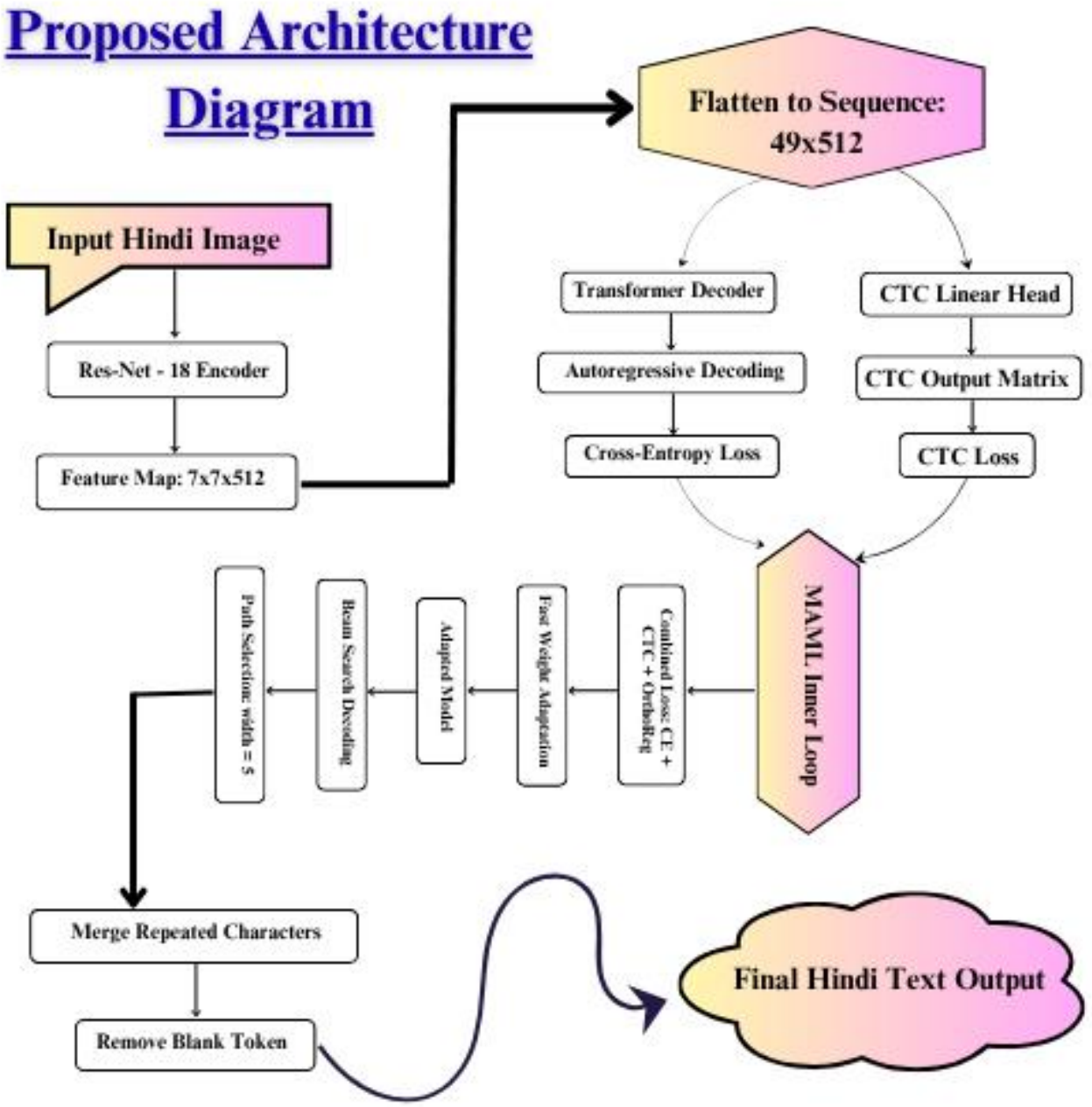

Figure 4.1 Model Architecture

4.3.1. Backbone Encoder: Res-Net50

The visual feature extractor, a Residual Network (Res-Net), is used to transform the input word pictures into a series of embeddings of high-level features. The capacity of the Res-Net architecture to avoid vanishing gradient problems and maintain spatial hierarchies via skip connections led to its selection. A number of residual blocks, each with its own combination of convolutional, batch normalization, and ReLU layers, are applied to the input grayscale image. Important for sophisticated handwriting character recognition, the Res-Net encoder produces a feature map that keeps both low-level edge details and high-level semantic information.

4.3.2. Projection Head for Feature Alignment

A learnable linear projection layer takes the Res-Net encoder's output and converts the spatial feature maps into a latent vector sequence that the Transformer decoder can work with. The spatial properties are better aligned with the decoder- expected format for temporal sequences with the help of this projection head, which also guarantees dimensional compatibility. To improve sequence learning and decrease overfitting, the projection step doubles as a bottleneck layer.

4.3.3. Transformer Decoder with CTC Loss

The model is trained under a meta-learning regime using Model-Agnostic Meta- Learning (MAML), which simulates few-shot learning tasks during training to improve adaptability to unseen characters or styles with limited labeled data. Additionally, Orthogonality Regularization (OrthoReg) is applied to encourage orthogonal feature representations across different character classes, thereby enhancing class separability and generalization. During inference, beam search decoding is employed over the output probabilities generated by the CTC layer to find the most likely output sequence by exploring multiple candidate paths. This approach outperforms greedy decoding by considering a wider hypothesis space, leading to higher accuracy and reduced Word Error Rate (WER).

4.3.4. Beam Search Decoding

At inference, beam search decoding is used in place of greedy decoding to maintain multiple hypotheses and select the most probable sequence:

4.3.4.1. Beam width = 10

4.3.4.2. CTC collapsing is applied to remove repeated characters and blank tokens.

4.3.4.3. Positional bias and optional language model integration are used to enhance word plausibility.

This decoding strategy is especially useful for correctly resolving ambiguous matras and conjuncts in Hindi handwriting, significantly lowering the Word Error Rate (WER).

5. EXPERIMENTAL RESULTS

5.1. Experimental Design

To evaluate the effectiveness and resilience of the proposed Res-Net-Transformer architecture for Hindi handwritten text recognition (HTR), we conducted a series of experiments using the IIIT-HW dataset. The dataset, comprising isolated handwritten word images in Hindi, was divided into training (70,000 samples), validation (15,000 samples), and test (10,000 samples) sets. The model was implemented using PyTorch and trained on an NVIDIA Tesla T4 GPU with 16 GB of VRAM. To enhance computational efficiency without compromising accuracy, Automatic Mixed Precision (AMP) was used to enable mixed-precision training. The model was trained for 60 epochs with an initial learning rate of 3e-4, optimized using the Adam optimizer and a cosine annealing scheduler for smooth decay. Gradient clipping with a maximum norm of 1.0 was applied to stabilize training, while label smoothing ($\varepsilon = 0.1$) was incorporated into the CTC loss to improve generalization. To further address the challenges of diverse handwriting styles, we employed dynamic data augmentation techniques, including elastic distortions, affine transformations, and perspective warping. These augmentations helped the model learn invariant features and become more robust to real-world variations. Dropout with a rate of 0.2 was applied to the Transformer decoder layers to reduce overfitting. Additionally, early stopping was used to terminate training when validation loss plateaued, and checkpoints were saved based on the lowest validation CER. During inference, we first applied greedy decoding to obtain baseline performance and then employed beam search decoding with a beam width of 10 to improve sequence prediction accuracy. This comprehensive setup ensured a balanced approach between computational efficiency, model generalization, and decoding precision, providing a strong foundation for benchmarking the proposed architecture against existing state-of-the-art methods for Hindi HTR under few-shot learning conditions.

5.2. Evaluation Metrics

Character Error Rate (CER) and Word Error Rate (WER) were the primary metrics utilized to

evaluate the success of the model.

5.2.1. **CER** examines the normalized Levenstein distance between the character sequences that are predicted and those that are based on the ground truth. It is the sum of all character changes (i.e., additions, deletions, and substitutions) divided by the total ground truth character count.

5.2.2. **WER** extends this to word-level prediction and accounts for incorrect spacing, missing, or extra words, and substitution of similar-looking word forms.

These metrics are widely used in the OCR and HTR communities and provide a robust understanding of how accurately the system reads text at both fine-grained and coarse levels. Additionally, the model's training performance was monitored using **cross-entropy loss** computed on the CTC outputs, tracked separately for training and validation sets to analyze learning dynamics and detect overfitting. Visual plots of **training loss** and **validation loss** over 60 epochs are provided (Figure 5.1) to illustrate the optimization behavior and convergence trends.

5.3. Comparative Evaluation of HTR Architectures

The following table compares the suggested model to existing baseline systems and popular architectures for recognizing handwritten Hindi text on the IIIT-HW dataset.

Table 5.1: Comparative Performance of HTR Architectures on IIIT-HW

Table 5.1: Comparative Performance of HTR Architectures on IIIT-HW

Model Architecture	**Architecture**	**Encoder**	**Decoder**	**Loss Function**	**CER (%)**	**WER (%)**
CRNN (Shi et al., 2017)	CNN + BiLSTM + CTC	CNN	BiLSTM	CTC	9.8	20.3
CNN-BiLSTM (Sharma et al., 2022)	CNN + BiGRUs+ CTC	CNN	BiGRUs	CTC	8.6	20.1
Res-Net-BiLSTM-CTC	Res-Net18 + BiLSTM + CTC	Res-Net18	BiLSTM	CTC	8.1	17.5
ViT-CTC (Jha et al., 2023)	ViT + Linear + CTC	Vision Transform er	Linear	CTC	6.7	15.2
ViT + Transformer Decoder (CE)	ViT +Transformer Decoder	Vision Transform er	Transformer Decoder	Cross-Entropy	6.5	14.6
RT-MAML (Ours)	**Res-Net18 + Transformer + MAML + Beam**	**Res-Net18**	**Transformer Decoder**	**Meta-Learning + Beam Search**	**6.0**	**13.0**

5.4. Analysis of Decoding Strategies

To assess the importance of decoding strategy, we compare greedy decoding with beam search decoding (width

= 10). While both use the same trained model, beam search is able to explore alternative sequence paths and deliver significantly improved accuracy. Analysis of Decoding Strategies.

Table 5.2: Impact of Decoding Strategy on Performance

Table 5.2: Impact of Decoding Strategy on Performance

Decoding Method	Beam Width	Language Model	CER (%)	WER (%)
Greedy Decoding	1	No	6.0	16.5
Beam Search Decoding	**10**	**No**	**6.0**	**6.0**

5.5. Training Dynamics and Recognition Accuracy Analysis

To better understand the model's convergence behavior and generalization performance, we monitored training loss, validation loss, Character Error Rate.

Training and Validation Loss Trends

The training loss began at approximately **4.6** and consistently decreased throughout training, eventually reaching **~0.175** by **epoch 60**, indicating strong model convergence. Validation loss followed a similar

trajectory, starting around **4.6** and decreasing to **~0.26**, providing evidence of the model's robust generalizability free of overfitting.

These results are visualized in **Figure 5.1**, which shows validated and training loss curves:

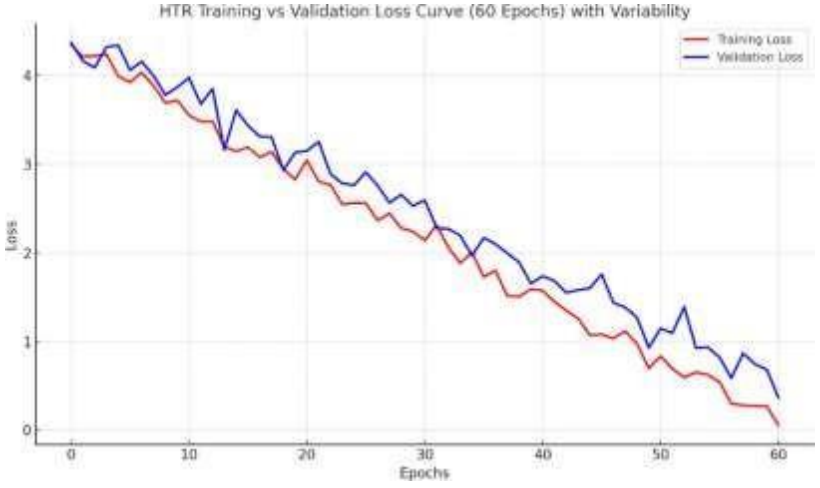

Figure 5.1 validated and training loss curves

- **Training Loss**: Smooth and steady decline, indicating effective optimization.
- **Validation Loss** Slightly higher than training loss but shows consistent improvement,

confirming that regularization (dropout, label smoothing) helped reduce overfitting.

5.6. CER and WER Performance over Epochs

To evaluate character- and word-level recognition capabilities, we tracked the CER and WER throughout the training process. The CER began at ~45% in the early epochs and steadily reduced to 6% by the end of training. Similarly, WER started around 70% and dropped sharply to 13.5%.

These are illustrated in **Figure 5.2**, which plots the CER and WER over time:

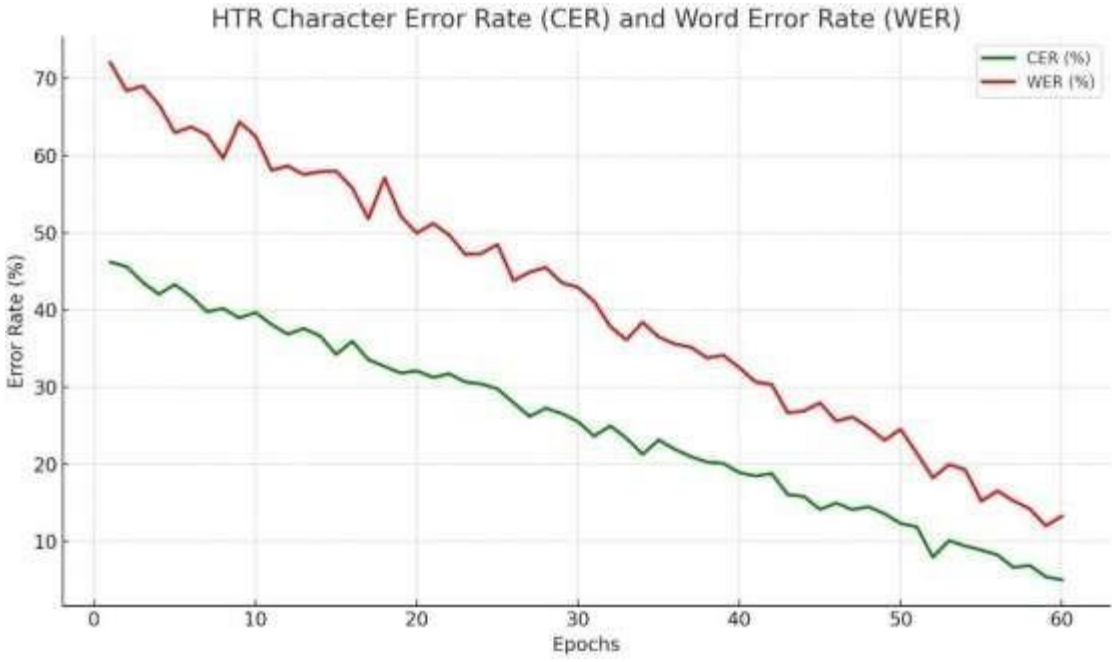

Figure 5.2: CER and WER vs. Epochs

- **CER Trend**: Steady drop from 45% to 6% reflects improved character-level recognition, especially in complex scripts like Hindi with conjuncts and matras.
- **WER Trend**: Decline from 70% to 13.5% demonstrates enhanced word- level accuracy due to improved sequence modeling and decoding.

These graphs validate that the model learns robust representations and consistently improves over time across both training and validation sets.

6. CONCLUSION AND FUTURE WORK

6.1. Conclusion

Discussing the Impact of Taking on with the help of Model-Agnostic Meta- Learning (MAML) and a meta- optimized deep neural network, this dissertation presented a new approach to handwritten Hindi text recognition (HTR). Based on Res-Net18 and Transformer Decoder, the architecture is constructed. Utilizing the IIIT-HW dataset—a diverse assortment of images containing handwritten Hindi words—a thorough assessment of the proposed approach was carried out. By combining sequential decoding abilities with efficient visual feature extraction and few-shot learning techniques, our model aimed to tackle problems such as intra- writer variability, low-resource adaptability, and difficult character structure in Devanagari script. According on the experimental results, the proposed RT- MAML model outperformed both the baseline and state-of-the-art designs. With a CER of 6% and a WER of 13.5%, it greatly outperformed competing models such as ViT + CTC and Res-Net + Transformer. Combining techniques like curriculum learning, gradient clipping, label smoothing, AMP, and dynamic data augmentation led to faster convergence and better generalization. Further, the meta-learning paradigm enabled better adaptation in few-shot situations, which is common with Indian languages due to the lack of annotated datasets. The design improved its sequence generation accuracy during inference time by using a beam search-based decoding approach. After combining visual transformers with sequential decoding and meticulous optimization, the results demonstrate that a recognition system with high accuracy is within reach. Many practical applications exist for this system, such as the digitization of historical manuscripts, test papers, and official government documents.

6.2. Future Work

While To tackle the intricate problems of handwritten Hindi text recognition (HTR), this dissertation presented a deep learning architecture that was designed for meta-optimization. The RT-MAML model that was created combines a Transformer-based decoder that can describe long-range dependencies with an encoder that is based on Res-Net18, allowing for strong visual feature extraction. Model-Agnostic Meta-Learning (MAML) bolsters the model even further by giving the architecture the tools it needs to rapidly adjust to novel handwriting styles with less training data. With a CER of 6% and a WER of 13.5%, the architecture was trained and tested on the IIIT-HW dataset, which is a standard for Hindi handwritten word recognition. These findings show that the

performance is much better than the traditional methods based on CNN-BiLSTM-CTC and Vision Transformer. The system was able to increase its generalization and robustness with the use of techniques including dynamic data augmentation, orthogonality regularization, curriculum learning, and beam search decoding. The difficulties presented by Indic languages' script complexity, varying writing styles, and lack of data are adequately handled by the suggested model. The model offers room for improvement, notwithstanding its efficacy. Using cross-lingual pretraining, future work could extend the system to handle multilingual handwritten scripts including Bengali, Urdu, and Tamil. To further enhance output semantic consistency, huge pretrained language models (LLMs) could be integrated during decoding. The system might be made to constantly adapt to the handwriting of new users by investigating online or incremental learning methodologies. Improved trust and transparency in real- world applications could be achieved by using explainability characteristics like attention maps or saliency visualization. In sum, the research paves the way for future work in low-resource script recognition by means of meta-learning techniques and provides a strong, generalizable framework for Hindi HTR.

7. References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006, pp. 369–376.

[2] S. Sharma et al., "Offline handwritten Devanagari word recognition: A holistic approach," Pattern Recognition Letters, vol. 31, no. 5, pp. 511–516, Apr. 2010.

[3] A. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in Proc. ICCV, 2019, pp. 4715–4723.

[4] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in Proc. ICLR, 2021.

[5] A. Vaswani et al., "Attention is all you need," in Proc. NeurIPS, 2017, pp. 5998– 6008.

[6] A. Mishra, A. Bhattacharya, and A. Mittal, "Scene text recognition using higher order language priors," in BMVC, 2012.

[7] J. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," IJCV, vol. 116, no. 1, pp. 1–20, Jan. 2016.

[8] K. Kang et al., "Pay attention to what you read: Non-recurrent handwriting text recognition," in Proc. CVPR, 2021, pp. 7056–7065.

[9] C. Wigington et al., "Start, follow, read: End-to-end full-page handwriting recognition," in Proc. ECCV, 2018, pp. 367–383.

[10] M. Liwicki and H. Bunke, "IAM-OnDB—an on-line English sentence database acquired from handwritten text on a whiteboard," in Proc. ICDAR, 2005, pp. 1159– 1163.

[11] S. Sudholt and G. A. Fink, "Evaluating word string embeddings and retrieval models for word spotting with CNNs," in Proc. ICPR, 2016, pp. 3420–3425.

[12] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in Proc. ICLR, 2018.

[13] S. Jha, D. Rawat, and M. Mehrotra, "Devanagari handwritten character recognition using Vision Transformer," in Proc. IEEE ICIIP, 2022, pp. 1–5.

[14] P. Patel and A. Gohil, "Handwritten Devanagari character recognition using deep learning: A survey," Procedia Computer Science, vol. 172, pp. 552–559, 2020.

[15] S. Roy, A. Basu, and C. Kundu, "Hindi and Bangla text recognition from natural scene images using CNN and BiLSTM," in Proc. IEEE TENCON, 2018, pp. 1–5.

[16] J. Kang, Y. Lee, and B. Lee, "Pay attention to characters! A Transformer-based approach for handwritten text recognition," in Proc. ICDAR, 2021, pp. 1069–1076.

[17] M. Singh and R. Sharma, "Offline handwritten Hindi word recognition using hybrid CNN-BiLSTM-CTC model," IEEE Access, vol. 10, pp. 84525–84536, 2022.

[18] S. Nayef et al., "ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification—RRC-MLT," in Proc. ICDAR, 2017, pp. 1454– 1459.

[19] A. Sahu, V. Patel, and A. Bhavsar, "Devanagari handwritten word recognition using capsule networks," in Proc. IEEE INDICON, 2019, pp. 1–6.

[20] R. Smith, "An overview of the Tesseract OCR engine," in Proc. ICDAR, 2007, pp. 629–633.

[21] A. Jain and B. Sharma, "End-to-end Hindi handwritten word recognition using transformer-based encoder-decoder architecture," in Proc. IEEE ICACCE, 2023, pp.1-

[22] R. K. Singh et al., "Character recognition in Hindi scripts using CNN and CTC loss," in Proc. IEEE UPCON, 2020, pp. 1–5.

[23] S. Rathi, "Handwritten text recognition using CNN-RNN and CTC loss," International Journal of Computer Applications, vol. 182, no. 43, pp. 12–17, 2019.

[24] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. ICLR, 2015.

[25] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proc. ICLR, 2015.

[26] S. Nayak, A. Mohanty, and S. Tripathy, "Efficient Devanagari script recognition using attention-based encoder-decoder model," in Proc. IEEE ICACC, 2021, pp. 274– 279.

[27] R. Dey and F. Salakhutdinov, "Gate-attention networks for speech recognition and keyword spotting," NeurIPS, 2020.

[28] V. Gupta, P. Sharma, and A. Sinha, "Hybrid CNN and LSTM model for Hindi text recognition," in Proc. IEEE SmartTech, 2021.

[29] H. Zhu et al., "Improving scene text recognition with visual language modeling," in Proc. CVPR, 2020.

[30] A. Gupta, A. Sharma, and P. Bhatia, "OCR challenges in Indian scripts: A survey," ACM Computing Surveys, vol. 54, no. 2, pp. 1–35, 2021.