Punolikhon: A Deep Learning Approach for Handwritten Text Recognition of Bengali and English Documents Using A Lightweight Model

by

SANJANA ZAMAN

21201051

MD. KAWSER ALAM NOMAN

22101761

KHANDOKER SAKIB UR RAIYAN

22201597

MD. SULAIMAN HOSSAIN

20301010

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
BRAC University
February 2026.

© 2026. BRAC University
All rights reserved.

Declaration

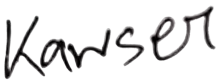
It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Sanjana Zaman

21201051

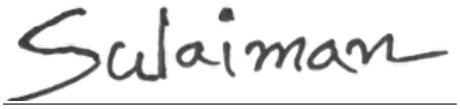


Md. Kawser Alam Noman

22101761

Khandoker Sakib Raiyan

22201597



Md. Sulaiman Hossain

20301010

Approval

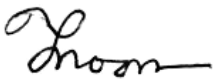
The thesis titled "Punolikhon: A Deep Learning Approach for Handwritten Text Recognition of Bengali and English Documents Using A Lightweight Model " submitted by

1. SANJANA ZAMAN (21201051)
2. MD. KAWSER ALAM NOMAN (22101761)
3. KHANDOKER SAKIB UR RAIYAN (22201597)
4. MD. SULAIMAN HOSSAIN (20301010)

of Spring, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science in Fall, 2025.

Examining Committee:

Supervisor:
(Member)



Dr. Jannatun Noor Mukta

Associate Professor, CSE
Director, BSDS
United International University

Co-Supervisor:
(Member)

Md. Moynul Asik Moni

Lecturer
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi
Associate Professor; Chairperson
Department of Computer Science and Engineering
Brac University

Abstract

It is very difficult for computers to read and extract data from the handwritten form of documents in bilingual models like Bengali and English. The condition becomes much more complex for Bengali scripts as it has numerous complex scripts, integrated letters, numerous styles, which makes the recognition process way more complex. This research focuses on a deep learning oriented model named Punolikon, a lightweight bilingual Optical Character Recognition Model which helps to identify both the Bengali and English handwritten form of documents efficiently and effectively. The model applies a novel hybrid deep learning model which includes Spatial Transformer Networks (STN) for input normalization, ResNet-style bottleneck blocks with skip connections for channel reduction, MobileNetV3-Small backbone for effective feature extraction and Transformer encoders with multi-head self-attention for robust model sequence. The recognition module handles conjunct consonants, vowel signs and Unicode normalization problems by using a grapheme-based tokenization technique created especially for the complex Bengali letter. For ensuring clean and clear text portion this model also relies on CLAHE along with graysclale at preprocessing period. This hybrid model achieves 84.3% word accuracy and 9.87% Character Error Rate on overall text recognition using a test set of 18,953 word samples. Fixing Unicode Normalization improved accuracy by 11%. The model shows strong performance with 88.4% character accuracy and 92.08% average prediction confidence. The detection part uses YOLOv8n for word localization and for the recognition pipeline applies a hybrid STN-CNN-Transformer structure developed with CTC loss to handle word crops. Finally, this integrated architecture helps the model to gain satisfactory rate of accuracy in case of recognizing both Bengali (85.3%) and English (83.3%) handwriting by keeping the system robust and effective. Also this provides a comprehensive solution for digitising handwritten and printed historical Bengali-English records, offers administration digitisation and academical resources preservations.

Keywords: Handwritten Text Recognition, Bengali-English OCR, Deep Learning, Multi-Head Attention, MobileNetV3, CTC Loss, Spatial Transformer Network, ResNet Bottleneck, Transformer Encoder, YOLOv8n

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Handwritten documents are vital sources of information, preserving history and culture and knowledge carried across generations. Many countries like Bangladesh, they store important documents in handwritten forms which includes official documents, academic notes and historical manuscripts. We know that Optical Character Recognition (OCR) has been widely applied for translating printed text into a machine readable text but it struggles with handwritten documents. Specially, handwritten text in Bengali language which has a complicated script provides additional obstacles. Bengali scripts have conjunct consonants, vowels and punctuation that make it more difficult and challenging for typical OCR algorithms to reliably recognize. Additionally, in bilingual documents such as those containing both Bengali and English texts, the problem becomes more obvious. Therefore, we need an efficient and reliable OCR system that can such challenges quite easily. Our thesis introduces a bilingual lightweight OCR model which we have named as Punolikhon that aims to overcome these challenges. We have implemented a deep learning based hybrid architecture, Punolikhon which can recognize both Bengali and English handwritten text. This can help us to preserve and digitize valuable historical and educational documents for future generations.

1.2 Rational of the Study or Motivation

The motivation for this study comes from the limitations of existing OCR algorithms when it comes to accuracy in the recognition of handwritten texts by lightweight architectural model. In other countries including Bangladesh, handwritten papers are crucial. While there have been breakthroughs in OCR models for printed texts, handwritten texts especially in Bengali combined with English that too in a lightweight model remains a challenge. Existing OCR models often work well with clear, printed text but struggle when applied to handwritten documents particularly when the handwriting is irregular or text is faded or deteriorating. Futhermore, many of these systems are built to handle only one language and do not allow bilingual text recognition. This research is inspired by the need for a system which can efficiently process handwritten documents in both Bengali and English with a lightweight model. Additionally, there is a void in the avail-

ability of OCR models that can handle both the complexity of Bengali script and the multilingual character of many ancient texts. This research intends to address this void by constructing a bilingual lightweight OCR system that can recognize old handwritten texts in both languages.

1.3 Problem Statement

Traditional OCR models which are lightweight built encounter substantial restrictions when it is applied to handwritten documents. These models often fail to precisely recognize texts when the quality of the document is low or when the handwriting is complex. This challenge is amplified when dealing with multilingual documents because the OCR model must simultaneously process two different scripts Bengali and English. In the case of Bengali text, the complexity of the text with it's conjunct consonants and many diacritics that makes it exceptionally tough for OCR models to reliably recognize letters and phrases. Existing OCR models are often trained on clear and printed texts. Therefore, we badly need an OCR model that can handle these challenges easily. This research focuses on creating a bilingual Bengali-English OCR model (Punolikhon) using a hybrid lightweight model that can efficiently recognize and digitize handwritten texts. This model must also be able to handle both Bengali and English text from the same document making it versatile for utilize in various real world applications.

1.4 Objective

The ultimate purpose of this research is to construct a reliable and adaptive bilingual lightweight OCR engine model that can easily read, detect and recognize both the Bengali and English handwritten text.

The fundamental objective of this model are :

1. To create and develop a hybrid deep learning oriented model which combines YOLOv8n for detecting text and a HybridHTR_STN_Transformer model for recognizing the text.

2. To upgrade the quality of the image with some notable preprocessing techniques like CLAHE and gre which removes noise and other distortion.

3. To run and perform the lightweight model accurately and efficiently via data augmentation.

4. To train the model with diverse variations by dataset including Bengali and English Handwritten documents and optimizes the model by using label smoothing, learning rate scheduling and checkpointing for accurate performance.

5. To evaluate and measure the evaluation using notable OCR metrics like CER, WER and Word Accuracy.By using these metrics helps to determine how effectively the model is working.

In short, we aim to construct an effective and adaptive OCR model that can read and recognize old Bengali and English documents by any walk of people using our lightweight model.

1.5 Methodology in Brief

This study adopts a data-based deep learning technique to design and evaluate a two step optical character recognition model that can detect and recognize text from image documents.This research is done using several main stages:

1. **Data acquisition and preprocessing:** Page images are collected from scanned and camera-captured sources. Prior to modeling, handwritten text images are annotated manually and automatically, preprocessing variants are applied to reduce illumination inconsistency and improve stroke visibility, including the original representation, CLAHE-based contrast enhancement, and grayscale normalization[1].

2. **Multimodal architecture development:** A two-stage OCR formulation is adopted, where localization and transcription are addressed as distinct but connected tasks. Word localization is performed using a YOLOv8n-based detector that produces bounding boxes with confidence scores, followed by post-processing to suppress redundant predictions. Word-level transcription is performed using a hybrid STN–CNN–Transformer recognizer, where geometric normalization is first applied, visual features are extracted, sequential context is modeled, and alignment-free transcription is produced via CTC decoding.

3. **Training strategy:** Model training is conducted separately for detection and recognition to support modular optimization. The recognition model is trained on word-level crops derived from annotated bounding boxes, and experimental comparisons are conducted between idealized ground-truth crops and detector-generated crops to characterize the effect of localization quality on transcription performance.

4. **Evaluation framework:** Performance is evaluated in both detection and recognition stages. In the detection part the performance is calculated using standard localization measures and in the recognition section performance is measured by transcription metrics like word accuracy,error rates. Diagnostic experiments are included to quantify performance degradation under detection-generated crops and to identify integration-related limitations.

5. **Implementation:** The pipeline is implemented as a reproducible sequence of steps comprising preprocessing, detection with post-processing, crop generation and normalization, recognition inference, and text post-processing. Outputs are produced as bounding-box–aligned word transcriptions, with optional aggregation for page-level reconstruction and reporting of confidence-oriented statistics.

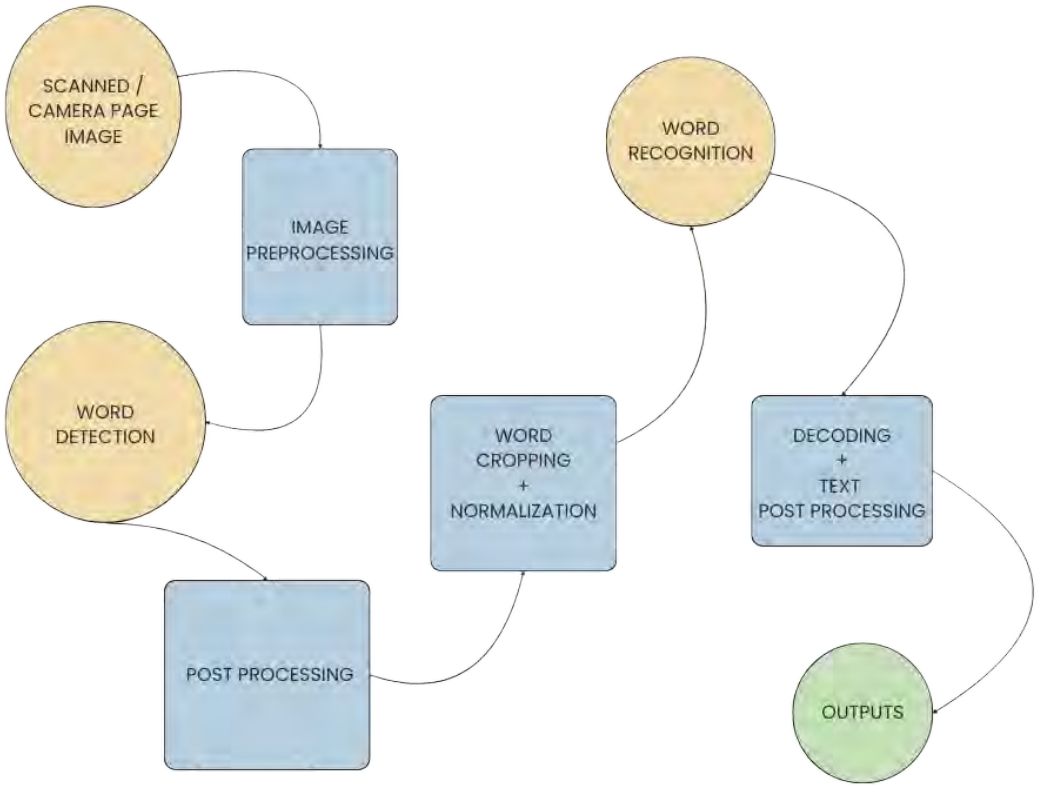


Figure 1.1: Workflow overview of the OCR pipeline

1.6 Scopes and Challenges

This research focuses on **handwritten Bangla OCR at the word level**, where the objective is to localize handwritten word regions from document images and transcribe them into machine-readable text. The scope is motivated by two key gaps in existing work: (i) comparatively limited research and publicly available resources for Bangla handwritten OCR, particularly with **word-by-word bounding-box annotations**, and (ii) the shortage of **lightweight OCR models** that remain reliable under realistic handwriting variability and document noise.

Scopes

The work contributes a novel Bangla handwritten word-level annotated dataset construction pipeline (including annotation and quality control), and designs lightweight, modular deep learning components suitable for practical deployment and systematic evaluation.

The study targets Bangla handwritten document images and addresses both word detection and word recognition within a modular OCR pipeline. The emphasis on word-level annotation and modeling is important because word-level supervision is substantially more difficult to obtain than plain transcriptions, yet it is essential for page-level OCR workflows.

Challenges

Problems with various handwriting : The first and foremost problem arises with the variations of handwriting of individuals for any single scripts[2]. This includes multiple strokes and structural similarities in Chinese characters, diverse sizes, styles, and structural complexity in Tamil, and the presence of'hamzas' and 'dots' complicating Arabic, particularly Quranic literature with its prevalent diacritics [3]. Amharic and Bangla scripts are also challenging due to small shape changes and diverse writing styles [4], [5]. Even hand-drawn chemical structures are troublesome due to their many styles and poor image quality[6]. Given the wide range of variables,achieving high accuracy is generally difficult [7].

Data scarcity and annotation limitations (Bangla-specific): A major problem is that there are not many large,publicly available datasets for languages, especially Bengali. There has been very little work done on handwritten Bengali scripts especially in word level annotations. THz image acquisition for historical documents is expensive and takes a long time, which makes it hard to get the data [8], [9].

Problems with image processing and segmentation: It's hard to split up text lines in old documents because the fonts, styles, and degradation are all different [10], [11]. This often results in overlapping characters or touching bounding polygons that merge lines and make it hard to recognize them correctly. THz images specifically suffer from specklenoise and high-intensity artifacts.

Performance and Generalization of Deep Learning Models: Deep learning models are often big and need a lot of computational power, which makes them hard to use on devices with limited resources. They often aren't strong enough and can't be used on new documents or compressed images. They also tend to overfit when the data is small or not evenly spread out [3].

Evaluation and Compatibility : As usual, the regular way of pixel oriented architecture suffers in case of extracting features, as a result, it brings incomplete results. As the results are corrupted and incomplete, it is not possible to compare the system's performance for further approaches [4].

Chapter 2

Literature Review

2.1 Preliminaries

Handwritten Text Recognition (HTR) is the machine-readable digital text produced from hand-written texts. From digitalizing historical records, form processing, and real-time transcription to throughout many academic and professional spheres, this field is crucial. Still challenging especially when working on intricate scripts like Bengali and Chinese because of their convoluted character frameworks, multiple strokes, and character similarities. Variations in individual handwriting styles, unequal spacing, broken strokes, and noise or damage in scanned documents provide further problems that can seriously lower recognition accuracy. Usually running two modes online which logs dynamic information from pen movements and offline, which runs stationary scanned pictures HTR systems.

In the last few years, deep learning (DL) techniques have shown remarkable improvements over traditional machine learning techniques in HTR. As with the conventional approaches it relies on manually designed features extraction, but deep learning models can easily and effectively extract high-level representations automatically from any types of complex data.

Convolutional Feature Extraction

Convolutional Neural Networks (CNNs) are always a core component for the purpose of feature extraction in HTR. Their convolutional layers are much capable of learning and understanding visual patterns. It also can detect character architecture directly from input data images, by minimizing the synthetic datasets. Most importantly, CNNs have shown efficiency along with effectiveness for numerous scripts which include Bangla, Tamil, Hindi, English and Ethiopic [4], [5].

Spatial Transformer Networks (STN)

To handle any sorts of geometric degradation and variations in handwritten format, a Spatial Transformer Network (STN) is implemented into the recognition pipeline. STNs provide a module which is learnable and can transform input images, which helps to align

characters for the performance of better recognition. It reduces the effect of deformed text and distortion, which allows subsequent layers to pay concentration on perfect feature extraction [12].

MobileNetV3 for Lightweight Feature Representation

On the other hand, to ensure lightweight feature extraction, MobileNetV3 is implemented for feature selection and extraction because of its lightweight design and computational adaptability. It also uses depthwise separable convolutions along with the optimized network blocks, making it usable and suitable for real-time HTR applications by maintaining high accuracy [13].

Grapheme Tokenization and Unicode Normalization

To recognize any complex symbols like the Bengali languages has, it definitely needs a Grapheme tokenization method model to get better performance. That's why the grapheme level is used instead of the character level for recognizing those complex alphabets. Grapheme tokenization first segments the whole text into some meaningful parts, which adopts combined forms of consonants, vowels, and diacritics of Bengali languages [14], [15].

Moreover, Unicode NFC normalization gives stable encoding. It reduces discrepancies between visually identical but differently encoded sequences and improves model reliability [16].

Sequence Modeling and Text Decoding

After doing feature extraction, either features are decoded directly with the help of sequence modelling layers together with CTC (Connectionist Temporal Classification) loss or recurrent layers like LSTM or GRU are used. Without strict alignment between the input image and its matching text, CTC allows training with variable-length sequences. As a result, handwritten words can be reliably recognised even if there are missing or distorted letters [17].

2.2 Review of Existing Research

Handwritten character recognition (HCR) is an important area of research, particularly for its applications in medical records, ancient manuscripts, and general document processing. Recent advancements in machine learning and deep learning have significantly improved the performance of HCR systems. This review discusses some recent papers that focus on HCR in various domains. The papers employ different deep learning architectures, including Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, [18] and federated learning. Here we are going to address the challenges posed by noisy, messy, and complex handwritten characters.

In 2022 Tabassum presents a machine learning-based approach to recognize hand written medical terms. In developing countries it is crucial to ensure accurate prescriptions in healthcare settings [19] .But in Bangladesh most doctors still rely on handwritten prescriptions. The authors highlight the significance of illegible hand writing that leads to errors in prescriptions. This paper gives a novel data augmentation technique Shifting, Rotation and Stretching (RSS) which is applied to increase the sample size and also improve model stability. With RSS augmentation, that model shows an average accuracy of 93.0 % which shows a notable improvement of 19.6 % over models that did not use data augmentation [19]. The authors use a Bidirectional Long Short-Term Memory (BiLSTM) network to process the data sequence of medical terms. This choice is motivated by the ability of LSTMs to effectively handle sequential data and capture the temporal dependencies inherent in handwriting. The smartpen proposed in this work could potentially reduce medical errors and save costs by digitizing prescriptions in real-time. This work illustrates the potential of deep learning in assisting doctors in improving their workflow and minimizing the impact of poor handwriting in medical prescriptions.

Krithiga in 2023 tells about the challenges to recognise ancient Tamil characters. Specifically, the Vattezhuthu script, which changes significantly over time. Because those languages were written on materials like rock or palm leaves, the process of digitizing those ancient manuscripts is very challenging because of distorted text conditions. The authors significantly discuss numerous preprocessing ideas like feature extraction, and denoising to deal with the degradation of ancient characters [11]. The paper reviews CNN and ResNet architectures which shows significant results in categorizing ancient Tamil characters. It also analyses the significance of proper dataset creation and provides us the scarcity of designated ancient text data. Researchers should synthesize new datasets or increase existing ones to address the limited availability of training data. The models discussed in the paper are focused on the segmentation of ancient text and turns them into recognizable characters. They use methods like horizontal projection analysis and generative adversarial networks (GANs) to improve segmentation accuracy. This paper shows the complexity of dealing with overlapping text and the challenges of various writing styles and materials [11]. This comprehensive analysis not only shows the technical challenges but also the future scope of developing a more robust model which can handle variations in scripts. Deep learning helps to preserve historical records for the future by digitizing them [11].

Kim introduces a novel deep learning-based model to improve the performance of Chinese character recognition. They address the problem of messy handwriting, which is common in Chinese handwriting and it drastically reduces the performance of traditional Optical Character Recognition (OCR) [19]. Their method classifies characters as either neat or messy. It uses a federated learning approach to handle data privacy concerns. By training models on local devices and averaging the model weights they show that federated learning can improve the recognition performance while protecting user data [3]. The paper compares their federated learning approach with global and local learning models.They show that their method outperforms existing approaches including FedAVG with a 301.2% improvement in accuracy and a 241.54% increase in the area under the curve (AUC) score. The authors specifically use EfficientNetB0 for the classification task which allows the model to achieve high performance even with limited data. The paper also employs various deep learning models such as ResNet50V2 and Swin Transformer in order to compare their performance in classifying messy handwritten Chinese characters.

The mixture of federated learning is remarkable as it helps to improve the model without requiring data. It needs to be uploaded to a central server which addresses privacy concerns. This federated approach is a vital contribution of the paper showing how dispensed learning can be. It is used to improve handwriting recognition systems especially in environments where data privacy is prioritized. Murali in 2025 proposes a Deep Progressively Learning Model (HCR-DPLM) for noisy handwritten character classification specifically for offline handwritten text recognition. The model focuses on overcoming challenges in noise removal and character segmentation. They are key obstacles in offline handwritten recognition. The authors propose a transfer learning strategy which leverages pre-trained networks for feature extraction that helps in reducing the need for large datasets. This approach improves the model's core ability by allowing it to work efficiently with smaller datasets [20]. The authors highlight that the pixel-level classifier employed in their method. It helps to remove noise effectively from handwritten character images. They show their method significantly improves its performance particularly while dealing with noisy data.

Dutta uses deep learning to enhance terahertz (THz) imaging, particularly its use in non-destructive historical documents analysis [8]. THz imaging is valuable to examine damaged documents without damaging them. It often suffers from noise, especially in high-speed imaging scenarios. To address this issue, the authors apply an unsupervised CycleGAN model to generate paired noisy and clean THz images, followed by a supervised Pix2pixGAN model for image denoising. Their results show that, after the denoising process 99% of characters on Xuan paper are clearly recognizable and 61% on standard paper are sufficiently clear. These findings highlight the potential of deep learning to improve the quality of THz images, enabling more effective analysis of historical documents without causing harm to the artifacts [8].

Ouyang in 2024 had a thesis showing ChemReco, a deep learning-based tool designed for recognizing hand-drawn chemical molecular structures, particularly those involving carbon, hydrogen, and oxygen atoms. Their model addresses the need for efficient conversion of hand-drawn chemical structures into machine-readable formats like SMILES codes which are crucial for computational analysis in chemistry. To take care of the lack of real training data, they generate and depend on the synthetic images of hand-drawn figures. Finally, their model uses a special encoder-decoder design consisting of EfficientNet and Transformer networks, which helps to achieve a high accuracy of 96.9%. For that, Chem Reco is very useful for the chemical research field and also for educational purposes. It can also accomplish tasks like automatically checking and grading hand-drawn diagrams of chemistry of the students [6].

Zaman et al. in 2022 proposes a CNN-based model for Bangla handwritten word recognition to evaluate different CNN architectures such as 2-Layer CNN, Pure CNN and 3-Layer CNN this shows 97% accuracy. This research uses a custom dataset that uses 100 frequently used Bengali words [21]. Jubaer et al. introduces the BN-DRISHTI model in 2023. This model is a combination of the YOLO framework with Hough and Affine transformations. This model is trained on an extended version of the BN-HTRd dataset (v4.0). It has achieved 99.97% F-score in line segmentation and 98% in word segmentation [22]. Pal et al. shows a two-stage Single Shot Detector (SSD) system for Bengali handwritten word recognition in 2024. This model used MobileNetV2 as backbone structure for character detection and followed by fine-tuning to recognize words. Its F1-score is 82.86%, showcasing the model is efficient to recognize Bengali words [23].

Zulkarnain et al. develop a bbOCR pipeline to recognise Bengali words in 2023.It is a combination of YOLO and image preprocessing.This model uses Bangla text documents for experiment and shows that the model has the capability to handle noisy and real-world handwritten data [24].Basher et al. proposes BnGraphemizer which is a grapheme-based tokenizer in 2023. It splits Bengali words into graphemic units for effective segmentation.This model used CRNN as their main structure and shows that BnGraphemizer-based model has outstanding performance in character level [25].Roy et al.tells grapheme based methods in 2023 to handle complex Bengali characters. These novel methods, Vowel Diacritics Separation (VDS) and All Diacritics Separation (ADS),show better performance in comparison to traditional OCR [26].Rahman et al. proposes a Deep Learning-based Bangla Handwritten Text Recognition in 2023 combining CNN and RNN architecture.This model achieves promising results specially in recognition complex Bengali Characters [27].

Mridha et al. in 2021 proposed a dataset containing Bengali handwritten text.It contains isolated characters words.This research main goal was to improve BHTR model.It shows CNN models efficiently recognize character recognition.Though some challenges remain due to different handwritten styles [28].Hossain et al.proposes a hybrid CNN-CTC architecture for Bengali handwritten character recognition in 2022. Their model was trained on the BN-HTRd dataset.It achieved 92% accuracy in character segmentation [29]. Azad et al. introduces a new model namely BornoNet in 2018.It is a CNN-based model that can recognize handwritten Bengali characters.This model has three distinct dataset and has achieved 95.7%-98% accuracy in character recognition [30].Taufique et al. shows a model Inception CNNs in 2018 to recognize handwritten Bengali characters.It has achieved 96.7% accuracy in character recognition.It shows the efficiency of CNN architectures for handwriting Bengali tasks [31].

Parthiban et al.(2020) proposes an RNN-based OCR framework.That uses TensorFlow.It achieves approximately 90% accuracy and highlights scope for further architectural improvements in future research. Although it is limited for its dataset size and has lack of word level recognition [32]. Ingle et al. (2019) addresses the challenges to integrate offline handwritten text recognition into large-scale OCR systems by leveraging online handwriting data and efficient non-recurrent neural network models.The approach achieves accuracy comparable to LSTM-based methods while improving scalability and enabling seamless integration of handwritten and printed text recognition.Memon et al. review 176 research papers in 2020 that were published between 2000 and 2019 on handwritten OCR that highlight major systems, datasets and advances driven by machine learning and AI [33]. The study summarizes state-of-the-art methods while identifying research gaps and future directions in handwritten character recognition [34]. Moreover,Yuan et al. (2012) presents a CNN-based approach for offline handwritten English character recognition using a modified LeNet-5 architecture with error-correcting codes. Evaluated on the UNIPEN dataset.This model achieves 93.7% accuracy for uppercase and 90.2% for lowercase characters [35]. This paper shows a comprehensive analysis of Handwritten Character Recognition (HCR) which gives an idea of human handwriting variations and its digitization advantages. It summarizes existing methodologies, their advantages, limitations and accuracy.It shows the ongoing need to improve efficiency and recognition performance [36].

Mishra et al. demonstrates a CNN-based OCR system trained on the NIST dataset of over 100,000 images in 2023. It recognizes both handwritten and printed characters.

The model achieves 90.54% accuracy by learning image features and predicting character classes. It shows the effectiveness of deep learning in Intelligent Character Recognition [37]. Ansari et al. (2022) shows an CNN-based institutional OCR model that implements numeric recognition by using greyscale. It shows 99% accuracy [38]. But this paper is restricted to digit only [38]. Mehta et al. (2016) shows a character recognition system for scanned images using a three-layer ANN combined with a Nearest Neighbour approach. The model classifies segmented characters into Unicode, addressing variations in fonts and styles, though accuracy metrics are not reported [39]. Afroge et al. in 2016 implements artificial neural networks and gets 99% accuracy for numeric digits, 97% accuracy in capital letters and 96% accuracy for small letters. But this research is limited to digits or characters only [40]. Katoch et al. show a CNN-based approach in 2022 on the A–Z Handwritten dataset. It shows the efficiency of deep learning in character recognition. It achieves 100% accuracy on the test data [41].

Above all studies mentioned show the importance of deep learning methods across various fields. These studies show the importance of deep learning for solving complex problems including handwriting text recognition in different languages,analysis of historical documents and chemical structure recognition. To overcome problems like data scarcity,writing variations they use data augmentation techniques,synthetic datasets and advanced neural networks architectures. Methods using GRU layers also CRNN show good accuracy as reported by Hossain et al. (2023) and Basher et al. (2023) [25], [27]. But maximum papers did not give notable metrics like CER, WER for Bengali OCR. It makes it difficult to show the comparison.Some papers create their own custom make datasets like BanglaWriting, BanglaWord Dataset, BN-HTRd while other papers use synthetic datasets. From basic architectures of CNN to advanced process like CTC loss as well as YOLO-based segmentation methodologies shifted. However,these processes have limitations with highly complex handwriting styles like overlapped characters or cursive writings.

2.3 Summary of Key Findings

Recent advancements in HCR and HTR have enormous impacts on areas of medical document preservation,ancient manuscripts identifying difficult styles of handwriting. Deep Learning models such as CNNs and BiLSTMs play an important role by adjusting with noisy datasets and improving model robustness.

Using CNN and ResNet architectures deep learning helps to recognize fragmented and damaged texts that helps in preservation of ancient Tamil manuscripts [11].Similarly, terahertz (THz) imaging shows the importance of deep learning in order to improve document readability without damaging fragile artifacts [8].

Deep learning improves handwriting recognition across multiple domains, including chemical structure recognition from hand-drawn diagrams, which highlights the broader impact of Transformer-based models over natural-language OCR [6].

Moreover, noticeable improvements in the field of handwriting recognition have been ensured by the use of deep learning models like CNNs and sequence models [42]. These improvements demonstrate that deep learning can help in document preservation,chemical structure identification, healthcare and bilingual or multilingual writing recognition.

Author	**Year**	**Model Architecture**	**Dataset**	**Metric**	**Limitation**
Parthiban et al.	2020	Basic RNN architecture	Handwritten English	Accuracy 90%	Needs large datasets; no word-level recognition
Hossain et al.	2024	DenseNet121 with GRU layer, CTC loss	Handwritten Bengali	CER 0.091, WER 0.273, Accuracy 95.643%	System's tokenization depends heavily on the dataset accuracy and method
Basher et al.	2023	BnGraphemizer, CRNN with BiLSTM	Bangla (Synthetic Dataset)	–	Limitations in handling writing styles
Roy et al.	2024	CNN with Vowel Diacritics Separation (VDS), All Diacritics Separation (ADS)	Bangla (Synthetic Dataset)	–	Language limitations; model struggled with real-time application
Jubaer et al.	2023	YOLO with Hough and Affine transformation for skew correction (BN-DRISHTI system)	Bangla (BN-HTRd v4.0)	99.97% for line segmentation and 98% for word segmentation	Limited number of handwritten dataset (only 786 images); skew correction needed to enhance line segmentation
Mishra et al.	2023	CNN	NIST dataset	90.54%	Only single characters; not word- or line-level recognition

Table 2.2: Summary of selected literature, model architectures, datasets, metrics, and limitations.

Chapter 3

Requirements, Impacts, and Constraints

3.1 Final Specifications and Requirements

This section shows both the operational requirements along with the technical requirements for the purpose of implementing and deploying Punolikhon, a bilingual OCR system that can easily recognize both the Bengali and the English handwritten form text accurately and effectively. These configurations guarantee the system can perform well on standard hardware and software, keeping it also suitable for real life deployment. Punolikhon is implemented using a lightweight architecture at the inference phase, which allows the users to function on everyday platforms like mobile devices. In contrast, the training phase remains a development-time process that may require higher compute resources; therefore, training requirements and user-facing deployment requirements are conceptually distinct. The system is thus deployable without requiring end-users to access any high-end GPUs, because model development and model consumption represent different operational contexts.

From the perspective of hardware requirements, Punolikhon targets mobile and edge-oriented inference, where the primary objective is to execute recognition and detection with minimal latency and limited memory. For user-side inference on modern mobile devices, the system is expected to operate on an Android or iOS phone with at least 6 GB RAM, a recent multi-core CPU, and an embedded mobile GPU or neural acceleration unit (e.g., Adreno GPU, Apple Neural Engine, or equivalent). In this deployment mode, the system performs forward inference only and does not require the backpropagation-intensive computations associated with training. As a result, the practical deployment requirement is not a discrete desktop GPU such as RTX-class hardware, but rather a device capable of running an optimized inference runtime and handling image preprocessing reliably. Storage requirements for a mobile deployment are primarily determined by the model binary and application assets; therefore, 1–3 GB of free storage is typically sufficient for the application, cached outputs, and logs, although higher storage availability improves usability for batch scenarios. For smoother user experience and reduced latency during on-device processing, a recommended configuration includes 8 GB RAM, adequate thermal headroom, and a modern SoC that supports mixed-precision arithmetic or dedicated inference acceleration.

At the same time, Punolikhon also supports server-based deployment for institutional use cases where throughput is prioritized. In such a deployment setting, a modest GPU can

accelerate inference for multiple concurrent users; however, the user remains disconnected from the underlying compute hardware because access is mediated through an application interface or API. This separation is essential to interpret the phrase "lightweight and deployable" correctly: the lightweight claim is primarily grounded in the model's architectural efficiency at inference time and its suitability for resource-constrained environments, whereas large-scale compute is optional and primarily associated with training, fine-tuning, or high-throughput centralized serving. For development and reproduction of training results, a stronger GPU and higher RAM remain beneficial, and a workstation-grade environment accelerates experimentation. A recommended development configuration includes an NVIDIA RTX 3060 (12 GB) or higher, 16 GB RAM or more, and SSD storage beyond 50 GB for datasets, checkpoints, and logs. The mentioned configurations help to train the model efficiently and effectively, but it does not provide any information about what users need to install or what to use to completely run the OCR model. For a full-functionable operation, Punolikhon needs a stable software stack. It is fully compatible on Windows 10 or 11, Ubuntu 20.04 or more, and macOS 11 or more for the purpose of research and development along with the task of evaluation. For mobile handsets, the system should be compatible in case of runtime for export or wrapped like TorchScript, ONNX Runtime Mobile and many more for the final application. The core architecture of the model is implemented using Python. Furthermore, Python 3.9 or its higher version is necessary for the development purposes. PyTorch 2.0 or more is generally used for training along with the model development. GPU acceleration can be accessible when CUDA 11.8 or more is supported. Some notable key libraries are used like OpenCV for the purpose of image processing, NumPy for any numerical calculations, Albumentations at the time of training for advanced detailed augmentations, TorchVision for transforming any standardized image. For data labeling and annotating, Label Studio is used to figure out the handwriting regions from the data samples by drawing bounding boxes around them which helps to easily manage and trace the labels along the dataset. This configuration makes the OCR system consistent from training phase to evaluation phase and can be easily set for the deployment.

Finally, for Punolikhon's overall performance, it has some notable functional requirements. For the purpose of text detection, the system should figure out text portions in handwritten form documents where at least 70% recall is maintained along with an IoU threshold of only 0.5. Again, for text recognition purposes on bilingual handwriting formats, the model should have at least 80% accuracy, which helps to make the predicted word reliable for further usage. The system can easily process both the Bengali and the English text without any different methods for preprocessing individually. Moreover, it uses a single unified method to process bilingual texts in a single workflow. It accepts any images of standardized inputs of different sizes and ratios, to control various document layouts. The output is generated via standardized Unicode where NFC normalization is integrated for character formatting consistently. Lastly, for any recognized words, Punolikhon gives scores of confidence, so users can easily compare reliability. It also supports human checks where needed. It also cancels any uncertain prediction for both the mobile along with the real life document digitization process.

3.2 Societal Impact

The implementation of a proper OCR model for both the Bengali and English handwritten text has some significant positive impacts in numerous sectors.
In case of cultural preservation, this OCR model plays a significant role. The ability of this OCR to digitize any forms of Bengali handwritten text helps to preserve important manuscripts for future generations for further research and other activities. Most importantly, various writers of Bengali literature wrote their poems, novels, and other works in handwritten forms. So, digitization of these helps to share them worldwide and keep them away from any kind of deterioration. Digitizing any forms of handwritten text helps to make them searchable and easily accessible for any kind of work whether it is research or any public works. Thus it is promoting the concept of cultural and knowledge preservation.
This OCR system outperformed the educational benefits. Numerous handwritten materials like notes, lecture sheets and other usable stuff can easily be converted to digital forms very quickly and can be shared easily to a large number of audiences. This form of digitization helps researchers to work with and to analyse with for any further necessities. Furthermore, a vast set of digitized handwritten Bengali documents can play a crucial role in teaching and learning the Bengali language efficiently and effectively for future generations.
Also in case of administrative impact, this OCR plays a vital role here. This OCR model can easily be used to digitize any forms of government and non-government documents like birth certificates, legal documents, land records and many more, which helps to make them easily accessible. Also in the health care or medical sector, prescribed medicines of doctors and their records,reports and other stuff can easily be converted to digital form to improve their record keeping capacity by minimizing the chances of errors.

3.3 Environmental Impact

The impact on the environment for this OCR project focuses on the training phase along with the stage of deployment. In the training phase, the model needs quite a good amount of GPU hours for training purposes, as it needs to extract features from a vast amount of dataset of both Bengali and English languages. So it demands significant energy usage. To tackle this condition of significant energy usage, notable efficient architecture can be used. For that reason, MobileNetV3 is taken into consideration which can significantly reduce the energy consumption at the time of training and can lower the overall computational cost. Because of this, the model can easily generate a carbon footprint, though its impact is not that significant, but it is obvious to know that there exists a computational cost at the time of training deep learning models. However, it can be optimized or can be mitigated via any model architecture which is not that heavy. At the time of deployment, when the model is completely trained, it needs to be efficient in interference too (when new images are used to recognize text). For fastest processing along with minimal computational cost, lightweight design is implemented for the model. The model is also designed for consumer hardware like personal computers or handheld devices, which significantly reduces the environmental impact by minimizing the reliance on any kind of cloud computing. This OCR system also supports batch processing system, as a result, it can easily manage vast amounts of data at once, which brings significant efficiency

by neglecting repeated pre-processing. For long term sustainability, with the help of digitizing documents of handwritten forms, this model can easily reduce the usage of paper for long term, which contributes significantly to environment sustainability. Usage of efficient architecture brings minimalist computational cost for both the training and deployment. Finally, deploying the system as open source encourages development in any community for preserving and handling handwritten text documents by lowering research efforts, which ultimately contributes to long term sustainability.

3.4 Ethical Issues

Ethical issues are one of the main concerns during any system development. So, at the time of building the OCR system and at the time of deployment, numerous ethical considerations are kept in the process to ensure that the system will be used responsibly. First of all, to avoid the biasness issues in the system, a large variety of handwritten documents of different patterns and styles are collected to ensure a diverse dataset. As a result, the system can perform accurately on any forms of handwritten documents which ensures the OCR model is completely fair for any kind of documents. Furthermore, it includes bilingual features, as the dataset relies on both the Bengali and English handwritten text documents, which clarifies the model is fair for both the languages.
In case of data privacy, all the data that are used to train the OCR system is collected and sourced from open source, and taken the permissions where necessary. It illustrates that the rights of any individual are respected. However, the system may function with any sorts of documents like personal information or sensitive information. Users can easily take necessary actions to maintain the data protection laws along with the privacy regulations. But with the mentioned OCR model, there is no chance of transmitting or storing any processed data. It ensures the personal information that is provided by the users are totally private and fully secured.
To get rid of misuse, the system is solely designed only for the purpose of detection and recognition and there is no way that it can generate any sorts of fake handwritten documents of its own. As a result, it prevents the misuse of creating fake handwritten documents. Also, the users can verify the results obtained from the OCR for any official and legal purposes, to ensure and compare the reliability and accuracy of the model.

3.5 Standards

To ensure that the OCR model is abiding by all the rules and regulations along with the industry standards, various standard keys need to be maintained. So, the first mentioned standard is Unicode Standards. For handling and maintaining Bengali text formats across all platforms, the system completely relies on Unicode 15.0 standards, which ensures that the text format is recognized and encoded at a consistent level in any known known formats. Also, to ensure proper representation along with consistency, NFC (Normalization Form C) is used to normalize the output text version.
As this project solely relies on images, the image standard is maintained strictly. As the project deals with a vast number of images, the commonly used standard image formats like JPEG, PNG, JFIF, TIFF and BMP are supported by the system. For maintaining the purpose of color space in images, grayscale conversion is managed internally and the images are further processed by the system using RGB color spacing. Finally, the system

outperforms any images whose standard DPI is kept at least 200 for handwritten text and 150 for any printed text.
Moreover, the core model is also built keeping the standard which is supported globally. The model is kept in the standardized PyTorch (.pt) format, which enables the system to work efficiently. It also ensures the deployment process across all notable formats. The architecture is also supported by export to Open Neural Network Exchange (OONX) format for the purpose of cross platform release.

3.6 Project Management Plan

The project management plan shows the actual roadmap of how the only idea will be converted to an OCR system to recognize and detect any forms of handwritten texts and how the OCR will be developed and maintained. This project will be segmented into a number of different stages to ensure clearer vision and goals and proper time management. The first stage will rely on conduction of literature review to know and understand the underlying working procedure of OCR systems and find the issues that are completely relevant with the handwritten Bengali and English text. The second stage starts with the data collection process, where it needs to make sure that diverse handwritten texts of both Bengali and English text are arranged, gathered, and finally annotated. In the next phase, core model architecture will be implemented by integrating both the recognition and detection architecture. After the development phase, the next phase will be for the purpose of training and model optimization, where various optimization techniques are implemented to ensure a good accuracy of the system. After completing the training stage, the overall performance of the system will be evaluated using some notable metrics. Finally, in the last stage, there will be the preparation for the final documentation along with report writing, where detailed analysis along with comparison, results, and conclusions will be included. Lastly, the time schedule will be planned carefully for each of the stages to ensure efficient project progression.

3.7 Risk Management

Risk management is one of the most important tasks of this project as it helps to figure out any probable problems during any steps and clarifies how to solve them. Among various risks, one notable risk is insufficient power supply to the computer, which heavily affects the training procedure of models for the purpose of deep learning. As a solution, it suggests implementing MobileNetV3 which is known for its efficiency, as it does not require that much power for any training purposes. It also suggests implementing any other cloud based resources if necessary. Another notable risk factor is overfitting. It occurs when the model cannot show its captiveness on new data points because somehow the model becomes biased towards some specific data points during training or becomes much more specific towards some data points. To get rid of these issues, various effective techniques like scaling, rotating, and addition of noise are implemented. As a result, datasets are expanded by ensuring variations which finally helps to reduce the issues of overfitting. As this model recognizes Bengali and English handwritten text documents, there always exists a risk of poor accuracy rate due to various handwritten patterns. To fix this, grapheme oriented tokenization is taken into consideration along with optimum tuning for the model. Moreover, the model may struggle while managing Unicode for the

Bengali text, as most of the characters might not be accessed correctly. To tackle this, UFC normalization is applied to improve the accuracy rate for the Bengali text. Finally, there exists a chance that the model may train too slow or the model might not be able to give any notable performances, in that case it clarifies to use methods like gradient control, learning rate adjustment to normalize and standardize the training and learning process for effective and efficient result.

3.8 Economic Analysis

In this economic analysis section, it basically evaluates the overall costs for developing the OCR (Optical Character Recognition) and also figures out the beneficial sides of it compared to the costs. Most of the costs goes for buying and managing hardwares like GPUs, which are needed to train the model efficiently and effectively. It also includes the process of usage of cloud GPU. On the other hand, software tools which are used for various purposes are open source and most of them are completely free of cost. For the purpose of collecting data and annotating them for further training and analysis, the whole team needs to spend quite a good amount of time. So, it illustrates that the system is completely open source, which means there is no scope to spend a single penny for any kind of licenses, unlike any other commercially used OCR systems. So, it saves money. But this system can create a scope for making money by providing various relevant services like document digitization. As a result, organizations can easily and effectively process a vast amount of data at a lower cost. Also, transcription via OCR can be done much faster compared to traditional transcription methods, which normally require a lot of time and effort. So, it is evident that the system can easily and efficiently manage any sort of documents within minimalist costs, making it a cost effective way to process data for educational institutions, government offices, hospitals which need to digitize a large volume of documents. Finally, this following open source model helps to digitize documents of any handwritten forms of both the Bengali and English languages through a cheaper price compared to any commercial OCR model which basically charges per page. So, this system is much more affordable to build and help to make and save money by providing document digitization services in a cheaper and faster way.

Chapter 4

Proposed Methodology

4.1 Design Process and Methodology Overview

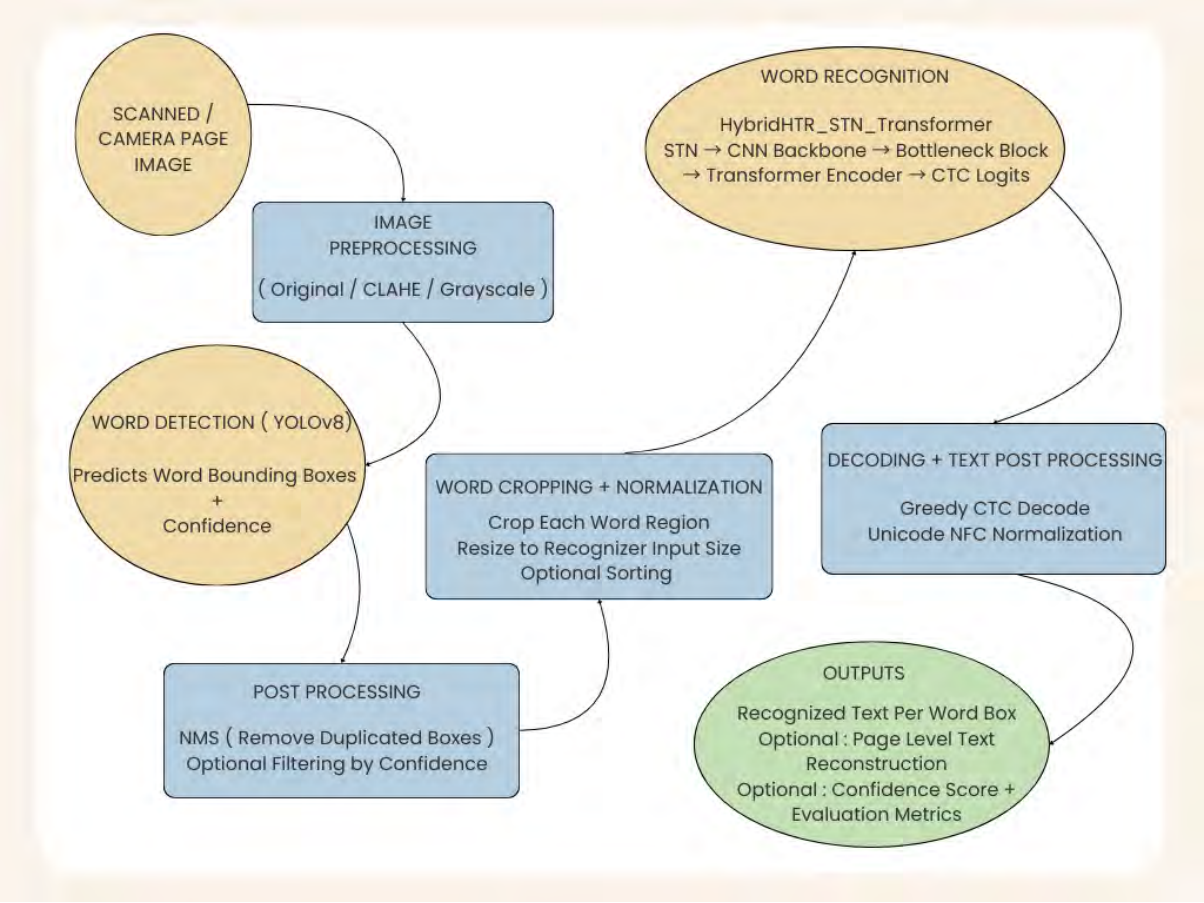


Figure 4.1: Methodology overview

The proposed methodology in the Figure 4.1 follows a two-stage OCR pipeline that converts scanned or camera-captured document images into machine-readable text. In the first stage, input page images are enhanced using preprocessing techniques (original, CLAHE, grayscale) to improve text visibility under different lighting conditions and document quality.The grayscale technique's idea is taken from dataset named Bangla-MNIST[43]. Word-level text regions are then detected using a YOLOv8n-based model,

which generates bounding boxes along with confidence scores[44]. Redundant or low-quality detections are removed through post-processing, primarily using non-maximum suppression.
In the second stage, the detected word regions are cropped from the page and normalized by resizing and padding to meet the fixed-height input requirement of the recognition model. When needed, the word regions are ordered to support consistent text reconstruction. Each normalized word image is then recognized using the Hybrid-HTR_STN_Transformer model, which combines spatial transformer-based normalization, convolutional feature extraction, Transformer-based sequence modeling, and CTC-based prediction[45]. The final text outputs are obtained using greedy CTC decoding and Unicode NFC normalization to ensure consistent Bengali text representation[46]. The system produces word-level transcriptions aligned with their corresponding bounding boxes, with optional aggregation for full-page text reconstruction and confidence-based evaluation.

4.1.1 Objective Formulation

The main goal of this research is to build an OCR system that can convert images into machine-readable text under different lighting conditions and document quality.To achieve this goal, the OCR task is divided into a two-stage process that consists (i) word-level text detection and (ii) word-level text recognition.
The detection task focus on accurately identifying word-level bounding boxes from a image while removing redundant and overlapping predictions through postprocessing.It aims to transcribe each detected word image into its corresponding Unicode text sequence using an alignment-free decoding method. To maintain consistency and correct representation in textual output, Unicode normalization is applied with particular attention.
A secondary objective of this research is to analyze how detection accuracy affects recognition performance.It is achieved by comparing recognition results got from ideal word crops derived from ground-truth annotations with those detector-generated crops. This analysis helps to identify performance and clarifies whether overall OCR accuracy is primarily limited by localization errors or transcription errors.

4.1.2 Data Acquisition and Preprocessing

Data acquisition and annotation

Word-level OCR requires page images with corresponding word bounding boxes and word transcriptions. For Bangla handwriting, publicly available datasets with word-level annotations are limited. To address this limitation, the dataset used in this study is constructed from two sources. Approximately 20% of the labeled samples are taken from the BanglaWriting dataset[28], while the remaining 80% is created specifically for this research to capture greater variation in handwriting style, document quality, and word-level annotations.
To support efficient dataset creation, a dedicated annotation and data-preparation software tool is developed to generate word bounding boxes and associated transcriptions from page images. The design and workflow of this tool are described in Section 4.2.3.

Dataset conversion and organization

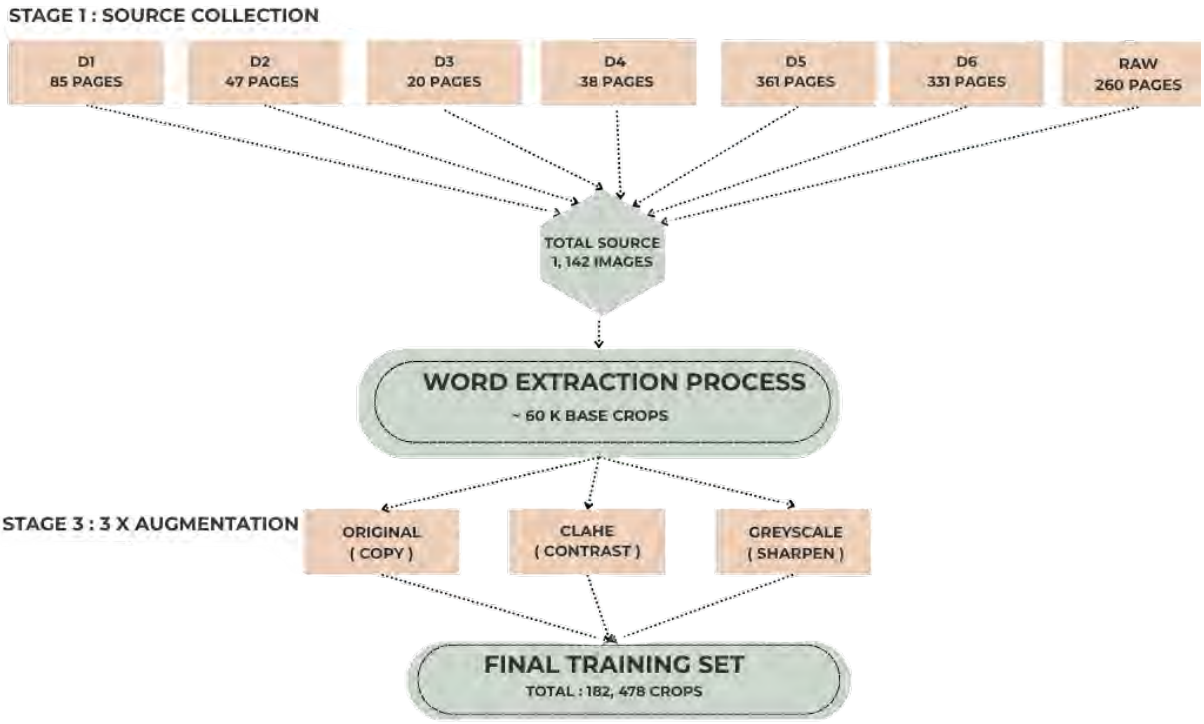


Figure 4.2: Dataset expansion

After data acquisition, annotations are converted into two aligned training targets. Detection targets consist of page-level records containing image paths and corresponding word bounding boxes. Recognition targets consist of word-level image crops paired with their ground-truth transcriptions. All page images are standardized to a common format, such as JPG, and stored in a structured directory layout. Annotation files are preserved in a dedicated directory to ensure reproducibility.

Split strategy (train, validation, test)

To avoid data leakage, dataset splitting is performed at the page-image level rather than the crop level. This ensures that word crops from the same page do not appear in multiple splits. Once the page-level split is defined, both detection and recognition samples are assigned to training, validation, and test sets based on their source page.

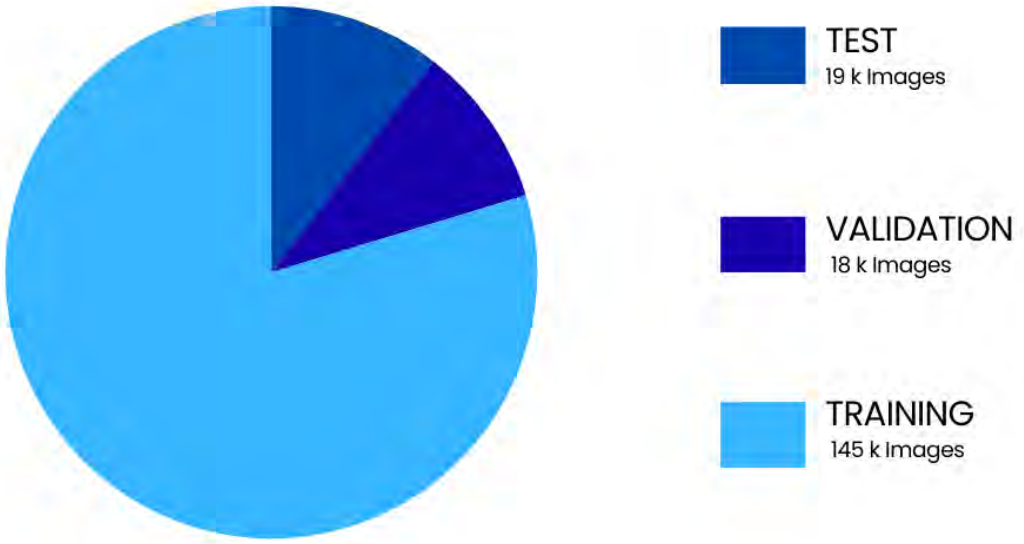


Figure 4.3: Data splitting

Recognition crop generation

In recognition part, word crops are generated by applying annotated bounding boxes to the corresponding input images. Each crop is treated as an individual image and a recognition manifest is created containing the crop path, the associated transcription and an optional language tag when applicable. This crop-based technique allows recognition models to be trained independently of detection while preserving a link to the original page images.

Text normalization and label handling

Before training and evaluation Unicode NFC normalization is applied for consistent labeling especially for Bangla text.This helps to reduce inconsistencies of same text's multiple valid Unicode representations.Invalid or empty transcriptions are handled during dataset processing in order to prevent ambiguous supervision.

Image preprocessing variants

Dataset images have different lighting variation,different contrast and background noise. To make the model more robust multiple preprocessing are applied that includes original image, CLAHE-based contrast enhancement and grayscale-based normalization. These

techniques are generated consistently across dataset splits and stored separately to enable controlled experimentation.

Crop normalization for recognition input

Each word crop is normalized to match the model input requirements before recognition. Each crops are resized to a fixed height such as 128 pixels, while preserving aspect ratio.Padding is also applied when stable batching and CTC-based decoding is necessary to maintain.Through this process a unified dataset is produced that supports both detection and recognition with page-level annotation for text area localization and crop-level annotation for transcription tasks.

4.1.3 Model Architecture

Detection Model

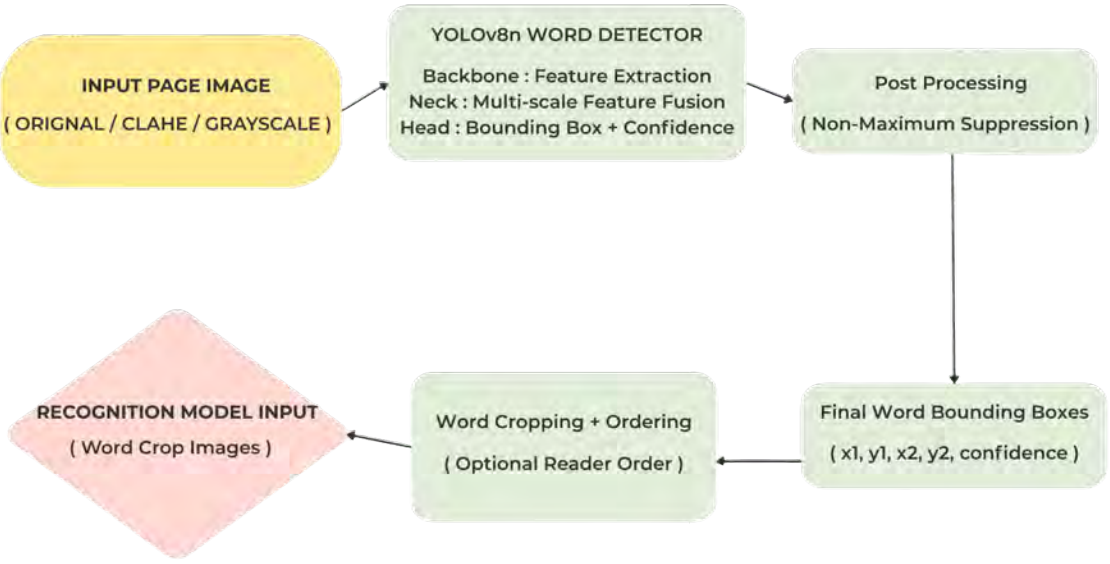


Figure 4.4: Detection Model Architecture

The detection part uses a segmentation-based word localization module that calculates a dense probability map of text regions and converts the resulting segmentation into word-level bounding boxes through a deterministic post-processing pipeline. Rather than direct bounding box regression this method is used,as it maintains consistency and allows detection sensitivity to be controlled explicitly through fixed thresholds and geometric constraints.

Input and preprocessing

Given a document image $I \in \mathbb{R}^{H \times W \times 3}$, the image was converted to RGB and resized to a fixed detector input resolution:

$$I_r = \text{Resize}(I,\ 768 \times 1024).$$

The network input tensor was then constructed as:

$$\mathbf{x} \in \mathbb{R}^{1 \times 3 \times 768 \times 1024}, \qquad \mathbf{x} = \frac{1}{255}\,\text{ToTensor}(I_r).$$

Hybrid backbone feature extraction

The detector backbone employed MobileNetV3-Small feature extraction, followed by an explicit refinement stage combining residual learning and global self-attention.
Let $\mathcal{B}(\cdot)$ denote the MobileNetV3-Small feature trunk. The extracted feature tensor is:

$$\mathbf{F}_0 = \mathcal{B}(\mathbf{x}), \qquad \mathbf{F}_0 \in \mathbb{R}^{B \times 576 \times h \times w}.$$

Residual refinement via lightweight bottlenecks

Three lightweight residual bottleneck blocks were applied to compress and refine the representation:

$$576 \to 256 \to 256 \to 128.$$

Each bottleneck block followed a ResNet-style structure incorporating Batch Normalization and ReLU activations:

$$\mathbf{y} = \text{ReLU}\Big(\text{BN}(\text{Conv}_{1\times1}) \to \text{BN}(\text{Conv}_{3\times3}) \to \text{BN}(\text{Conv}_{1\times1}) + \text{shortcut}(\mathbf{x})\Big),$$

where the shortcut path was implemented as a 1×1 projection when the input and output channel dimensions differed. After this stage, the refined feature tensor became:

$$\mathbf{F}_r \in \mathbb{R}^{B \times 128 \times h \times w}.$$

Global context via multi-head self-attention

To model long-range dependencies across the spatial feature map, a Multi-Head Self-Attention layer was applied with embedding dimension $d = 128$ and number of heads $n_h = 4$.
The feature map was flattened into a sequence:

$$\mathbf{S} \in \mathbb{R}^{(hw) \times B \times 128}, \qquad \mathbf{S} = \text{Flatten}(\mathbf{F}_r),$$

followed by:

$$\text{Attn}(\mathbf{S}) = \text{MHA}(\mathbf{S}, \mathbf{S}, \mathbf{S}), \qquad \mathbf{F} = \mathbf{F}_r + \text{Reshape}(\text{Attn}(\mathbf{S})).$$

This design preserved local stroke-level cues learned by convolution while enabling non-local interactions that improve segmentation continuity under broken strokes and degraded backgrounds.

Decoder and segmentation output

A lightweight decoder transformed the refined features into a dense 1-channel logit map using alternating convolution and bilinear upsampling operations:

$$128 \xrightarrow{\text{Conv 3×3, ReLU}} 256 \xrightarrow{\text{Upsample×2}} 128 \xrightarrow{\text{Conv 3×3, ReLU}} \xrightarrow{\text{Upsample×2}} 64 \xrightarrow{\text{Conv 3×3, ReLU}} \xrightarrow{\text{Upsample×2}} 32 \xrightarrow{\text{Conv 3×3, ReLU}} \xrightarrow{\text{Upsample×2}} 1 \xrightarrow{\text{Conv 1×1}}.$$

The network output was:

$$\mathbf{Z} = f_\theta(\mathbf{x}), \qquad \mathbf{Z} \in \mathbb{R}^{1\times 1\times H'\times W'}.$$

Pixel-wise probabilities were obtained using the sigmoid function:

$$\mathbf{P} = \sigma(\mathbf{Z}), \qquad \mathbf{P} \in [0,1]^{H'\times W'}.$$

Training Specification

Supervision mask construction

Detection supervision was formulated as binary segmentation. For each training sample, the ground-truth word boxes (in normalized coordinates) were rasterized into a binary mask $Y \in \{0,1\}^{H_0\times W_0}$. To reduce boundary ambiguity and to emulate DB-style shrinkage, each bounding box $[x_1, y_1, x_2, y_2]$ was shrunk around its center using a fixed shrink ratio:

$$r = 0.8.$$

Let $c_x = \frac{x_1+x_2}{2}$, $c_y = \frac{y_1+y_2}{2}$, $h_w = \frac{x_2-x_1}{2}r$, and $h_h = \frac{y_2-y_1}{2}r$. The shrunken box was defined as:

$$[x_1', y_1', x_2', y_2'] = [c_x - h_w,\ c_y - h_h,\ c_x + h_w,\ c_y + h_h],$$

and pixels inside the shrunken region were assigned foreground value. The image–mask pair was subsequently resized jointly to 768×1024 to match the detector input.

Optimization objective

Let $\mathbf{Z}$ denote the predicted logits and Y denote the target mask (resized to $\mathbf{Z}$'s spatial resolution when required). Training minimized a combined loss:

$$\mathcal{L} = \mathcal{L}_{BCE}(\mathbf{Z}, Y) + \mathcal{L}_{Dice}(\sigma(\mathbf{Z}), Y),$$

where $\mathcal{L}_{BCE}$ was implemented as Binary Cross Entropy with logits to provide stable pixel-wise supervision, and $\mathcal{L}_{Dice}$ was included to mitigate foreground–background imbalance by directly optimizing region overlap.

Training hyperparameters

The detector was trained using AdamW with learning rate 10^{-3} and weight decay 10^{-4}. The batch size was 4. ReduceLROnPlateau was applied as a validation-loss-driven scheduler with factor 0.5, patience 5, and minimum learning rate 10^{-6}. The maximum epoch budget was 100 with early stopping patience of 15 epochs without validation-loss improvement. Checkpointing saved the best model on validation-loss improvement, with periodic checkpoints saved every 5 epochs.
Data augmentation was applied during training through controlled geometric and photometric perturbations, including small rotations ($\pm 3^\circ$), brightness/contrast variation, and Gaussian noise, while ensuring consistency through joint image–mask transformation.

Mask-to-box conversion

At inference time, the predicted probability map was resized to the original image size:

$$\mathbf{P}_{orig} = \text{Resize}(\mathbf{P},\ H \times W).$$

Binary segmentation was performed using a fixed threshold:

$$\tau = 0.3, \qquad \mathbf{M}(u, v) = \mathbb{I}\left[\mathbf{P}_{orig}(u, v) \geq \tau\right].$$

External connected components were extracted from $\mathbf{M}$, and each component was converted into an axis-aligned rectangle:

$$(x, y, w, h) = \text{BoundingRect}(\text{component}).$$

Very small detections were discarded using explicit minimum-size constraints:

$$w \geq 5, \qquad h \geq 5.$$

Each box was then normalized to $[0, 1]$ coordinates:

$$b = \left[\frac{x}{W}, \frac{y}{H}, \frac{x+w}{W}, \frac{y+h}{H}\right],$$

producing the final set of predicted word boxes $\{b_i\}_{i=1}^{N}$ used for crop extraction and recognition.

Recognition Model

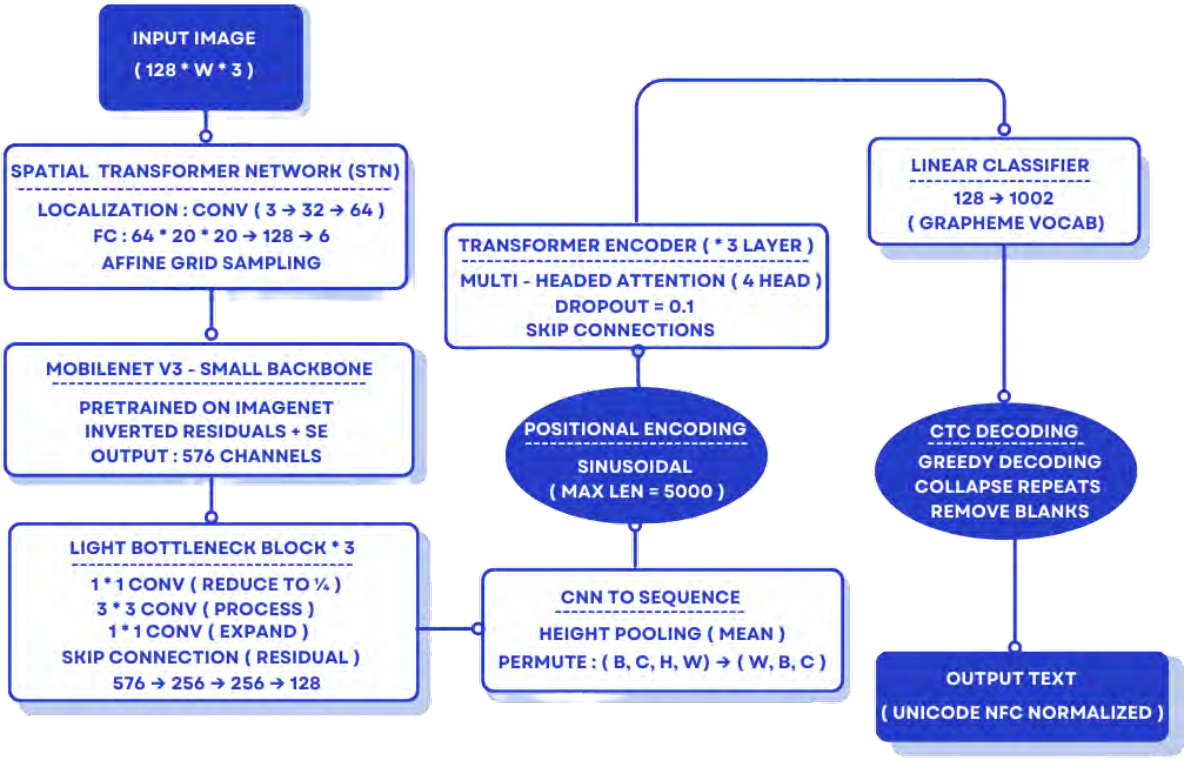


Figure 4.5: Hybrid Recognition Model

The recognition stage was formulated as a CTC-based sequence recognition problem operating on word-level cropped images. A hybrid architecture was adopted to address complementary sources of variability in handwritten Bangla word images, namely geometric distortions, fine-grained stroke patterns, and long-range contextual dependencies across the word width. Accordingly, the final recognizer integrated (i) a Spatial Transformer Network (STN) for geometric normalization, (ii) a lightweight MobileNetV3-Small convolutional backbone for local feature extraction, (iii) residual bottleneck refinement blocks to strengthen discriminative stroke representations, and (iv) a Transformer encoder to model global dependencies over the derived feature sequence. The model produced a per-time-step distribution over a grapheme vocabulary and was decoded using the CTC collapse rule.

Input representation and normalization

Let $I \in \mathbb{R}^{H \times W \times 3}$ denote an RGB word crop. Each crop was resized to a fixed height ($H_{in} = 128$) while preserving aspect ratio:

$$W_{in} = \left\lfloor W \cdot \frac{128}{H} \right\rceil, \qquad I_r = \text{Resize}(I,\ 128 \times W_{in}).$$

The network input tensor was then constructed as:

$$\mathbf{x} \in \mathbb{R}^{B\times 3\times 128\times W_{in}}, \qquad \mathbf{x} = \frac{1}{255}\,\mathrm{ToTensor}(I_r).$$

This variable-width representation allowed the effective sequence length to scale with the word width, which is required for CTC alignment.

Grapheme vocabulary and label space

A grapheme tokenizer (BengaliGraphemeTokenizer) was employed. Let $|\mathcal{V}|$ denote the vocabulary size loaded from `vocab_grapheme.json`. The classifier output dimension was:

$$V = |\mathcal{V}|.$$

The CTC blank symbol was defined as index blank = 0. A ground-truth transcription was mapped to an integer target sequence:

$$\mathbf{y} = [y_1, \ldots, y_U], \qquad y_u \in \{1, \ldots, V-1\}.$$

Spatial Transformer Network (STN)

Geometric normalization was performed using an STN placed before the feature extractor. The localization sub-network consisted of:

$$\text{Conv2D } (3 \to 32),\ 7\times 7;\ \text{MaxPool } 2\times 2;\ \text{ReLU}$$

$$\text{Conv2D } (32 \to 64),\ 5\times 5;\ \text{MaxPool } 2\times 2;\ \text{ReLU}.$$

To ensure a fixed-dimensional regression input under variable crop widths, adaptive pooling was applied:

$$\text{AdaptiveAvgPool2D}(20\times 20).$$

The affine transformation parameters were regressed using:

$$\text{Linear}(64\cdot 20\cdot 20 \to 128) \to \text{ReLU} \to \text{Linear}(128 \to 6),$$

yielding $\boldsymbol{\theta} \in \mathbb{R}^{2\times 3}$. The regressor was initialized to the identity transform, and the input was warped via:

$$\mathbf{x}' = \mathrm{GridSample}\big(\mathbf{x},\ \mathrm{AffineGrid}(\boldsymbol{\theta})\big).$$

This component reduced the burden on subsequent stages by explicitly compensating for skew, slant, and scale variation.

Hybrid CNN feature extractor: MobileNetV3-Small + residual bottlenecks

The CNN stage extracted local stroke-level descriptors from the normalized image $\mathbf{x}'$. First, MobileNetV3-Small feature extraction was applied:

$$\mathbf{F}_0 = \mathcal{B}(\mathbf{x}'), \qquad \mathbf{F}_0 \in \mathbb{R}^{B\times 576\times H_f\times W_f}.$$

To increase representational capacity for handwriting-specific structures at low computational cost, residual bottleneck refinement was appended with the explicit channel progression:

$$576 \to 256 \to 256 \to 128.$$

Each bottleneck block used a ResNet-style shortcut connection with Batch Normalization and ReLU activations. The refined feature map was:

$$\mathbf{F} \in \mathbb{R}^{B\times 128\times H_f\times W_f}.$$

CNN-to-sequence conversion (CTC-ready representation)

The 2D feature map was converted into a 1D sequence by collapsing the height dimension using mean pooling:

$$\mathbf{G}(b,c,t) = \frac{1}{H_f}\sum_{h=1}^{H_f}\mathbf{F}(b,c,h,t), \qquad \mathbf{G} \in \mathbb{R}^{B\times 128\times W_f}.$$

The width axis was then treated as the temporal dimension, yielding a time-major sequence:

$$\mathbf{S} \in \mathbb{R}^{T\times B\times 128}, \qquad T = W_f.$$

Transformer encoder for context modeling

A Transformer encoder modeled global dependencies across the feature sequence $\mathbf{S}$. A sinusoidal positional encoding was added:

$$\mathbf{S} \leftarrow \mathbf{S} + \mathrm{PE}(\mathbf{S}).$$

The encoder was configured with:

$$d_{model} = 128, \quad n_h = 4, \quad L = 3, \quad d_{ff} = 512, \quad \text{dropout} = 0.1.$$

The contextualized sequence representation was:

$$\mathbf{H} = \mathrm{TransformerEncoder}(\mathbf{S}), \qquad \mathbf{H} \in \mathbb{R}^{T\times B\times 128}.$$

Classification head

A linear classifier produced per-time-step logits over the grapheme vocabulary:

$$\mathbf{Z}_t = \mathbf{W}\mathbf{H}_t + \mathbf{b}, \qquad \mathbf{W} \in \mathbb{R}^{V\times 128},$$

resulting in:

$$\mathbf{Z} \in \mathbb{R}^{T\times B\times V}.$$

CTC objective

The recognizer was trained using CTC loss, enabling alignment between an input-dependent sequence length T and a target length U without requiring character-level segmentation. For a path $\pi \in \{0, \ldots, V-1\}^T$ and collapse operator $\mathcal{B}(\cdot)$, the conditional likelihood is:

$$p(\mathbf{y} \mid \mathbf{x}) = \sum_{\pi:\ \mathcal{B}(\pi)=\mathbf{y}} \prod_{t=1}^{T} p(\pi_t \mid t, \mathbf{x}),$$

and the optimized loss is:

$$\mathcal{L}_{CTC} = -\log p(\mathbf{y} \mid \mathbf{x}).$$

Optimization configuration

In the baseline grapheme training configuration, AdamW optimizer with learning rate 2×10^{-4} and weight decay 10^{-4} was used (optionally disabled during fine-tuning). ReduceLROnPlateau was applied with factor 0.5, patience 5, and minimum learning rate 10^{-6}. Gradient clipping used a maximum norm of 5.0, with an epoch budget configurable up to 200 and early-stopping behavior governed by the trainer configuration. Data augmentation was applied at the crop level using controlled geometric and photometric transformations (rotation, shift/scale/rotate, elastic and grid distortion, brightness/contrast variation, noise, blur, motion blur, and sharpening) to improve robustness to realistic handwriting variability.

Greedy CTC decoding

During inference, greedy decoding was applied:

$$\hat{\pi}_t = \arg\max_v \mathbf{Z}_{t,v},$$

followed by CTC collapse(removal of blanks and consecutive repeats):

$$\hat{\mathbf{y}} = \mathcal{B}(\hat{\pi}),$$

and the decoded token sequence was mapped back to Unicode text using the grapheme tokenizer.

4.1.4 Model Training Strategy

The training methodology was designed separately for the detection and recognition modules because their learning objectives are fundamentally different. The detection model was trained as a **binary segmentation** task with strong foreground–background imbalance, whereas the recognition model was trained as a **variable-length sequence prediction** task optimized through CTC alignment. Accordingly, different loss formulations, scheduling strategies, monitoring criteria, and checkpointing rules were employed.

Training Strategy and Optimization — Detection Module

Objective and supervision design

The detector was trained to predict a single-channel logit map $\mathbf{Z}$ representing text-region likelihood. Supervision was provided in the form of a binary target mask Y generated by

rasterizing word-level ground-truth bounding boxes. To reduce ambiguity near boundaries and to follow a DB-like shrink principle, each box was contracted around its center using a fixed shrink ratio:

$$r = 0.8,$$

before rasterization. The image–mask pair was then jointly resized to the fixed detector input resolution 768×1024 to ensure consistent spatial learning.

Loss function (foreground–background imbalance handling)

To mitigate severe class imbalance between background pixels and foreground text pixels, optimization was performed using a combined objective:

$$\mathcal{L} = \mathcal{L}_{BCE}(\mathbf{Z}, Y) + \mathcal{L}_{Dice}(\sigma(\mathbf{Z}), Y),$$

where BCEWithLogits ensured stable pixel-wise gradients and Dice loss promoted region overlap for comparatively small word regions.

Optimization and learning-rate scheduling

The detector was optimized using **AdamW** with:

$$\text{lr} = 10^{-3}, \qquad \text{weight decay} = 10^{-4}.$$

A ReduceLROnPlateau scheduler was applied with factor 0.5, patience 5, and minimum learning rate 10^{-6}, driven by validation loss.

Regularization and augmentation

To improve robustness to document variability, augmentation was applied jointly to the image and its supervision mask, including small rotations ($\pm 3^\circ$), brightness/contrast variation, and Gaussian noise, while preserving pixel-level correspondence.

Early stopping and checkpointing

Training was run for a maximum of 100 epochs with early stopping patience of 15 epochs without validation-loss improvement. The best-performing model was saved based on validation loss, with periodic checkpoint snapshots saved every 5 epochs.

Training Strategy and Optimization — Recognition Module (Enhanced HybridHTR_STN_Transformer)

Objective (CTC-based sequence learning)

The recognizer was trained using CTC loss to align the variable-length model output sequence with the target grapheme sequence without requiring character-level segmentation. For a path $\pi \in \{0, \ldots, V-1\}^T$ and CTC collapse operator $\mathcal{B}(\cdot)$, the likelihood is:

$$p(\mathbf{y} \mid \mathbf{x}) = \sum_{\pi:\ \mathcal{B}(\pi)=\mathbf{y}} \prod_{t=1}^{T} p(\pi_t \mid t, \mathbf{x}), \qquad \mathcal{L}_{CTC} = -\log p(\mathbf{y} \mid \mathbf{x}).$$

In the enhanced configuration, the training script supported two loss modes: standard CTC loss and Focal CTC loss (when available), which emphasized hard-to-recognize samples using $\gamma = 2.0$ and $\alpha = 0.25$.

Batch strategy and computational efficiency

Training was executed with batch size = 16 and gradient accumulation steps = 2, resulting in an effective batch size of:

$$B_{\text{eff}} = 16 \times 2 = 32.$$

Automatic Mixed Precision (AMP) was enabled to improve throughput and reduce GPU memory pressure. Gradient clipping with max norm 5.0 was applied before optimizer steps to prevent unstable updates.

Optimization algorithm

The recognizer was optimized using **AdamW** with:

$$\text{lr} = 3 \times 10^{-4}, \qquad \text{weight decay} = 10^{-4},$$

and standard AdamW momentum parameters $(\beta_1, \beta_2) = (0.9, 0.999)$.

Learning-rate scheduling

A cosine restart policy was used to improve exploration and convergence stability using CosineAnnealingWarmRestarts with:

$$T_0 = 10, \qquad T_{mult} = 2, \qquad \eta_{min} = 10^{-6}.$$

Addressing distribution imbalance (when mixed-script applies)

When mixed-script imbalance existed, WeightedRandomSampler was used to upsample English samples by a factor of:

$$u = 6.0,$$

with replacement sampling. This increased the probability of selecting English-labeled samples during mini-batch formation without altering the stored dataset distribution.

Regularization and augmentation

To improve generalization to real-world degradations, strong crop-level augmentation was applied, including geometric transforms (rotation up to 10°, shift/scale/rotate, perspective transform, elastic transform, grid distortion), photometric transforms (brightness/contrast, color jitter, Gaussian noise), blur-based degradations (Gaussian blur and motion blur) and sharpening, and robustness transforms (morphological erosion/dilation to simulate stroke-width variation and coarse dropout to simulate occlusions).

Monitoring, checkpointing, and model selection

Training was executed for 50 epochs. Validation loss was computed using standard CTC loss, while recognition quality was monitored using validation CER and word accuracy under greedy CTC decoding. Checkpoints were saved each epoch, and the best-performing recognizer was selected using the minimum validation CER, with the best weights saved as `model_best_cer.pt` and the corresponding full state saved as `checkpoint_best.pt`. To limit storage, only the last three epoch checkpoints were retained in addition to the best checkpoint.

4.1.5 Preliminary Design (Model) Specification

During the preliminary phase, multiple candidate configurations were explored to determine a practical and accurate OCR pipeline for handwritten Bangla document images. The initial baseline employed a generic pre-trained text detector and a conventional CNN-based recognizer; however, preliminary trials indicated that detection quality degraded in low-contrast and noisy handwritten pages, and recognition performance was sensitive to geometric distortions and complex grapheme compositions.
Subsequently, a segmentation-based detection design was investigated, where the detector predicts a dense text-region mask and bounding boxes are obtained through deterministic post-processing. This approach provided more stable localization under fragmented strokes and background artifacts compared to the baseline box-based detector, motivating the adoption of a custom detector trained on the target data distribution. For recognition, alternatives beyond a standard CNN–RNN baseline were evaluated, and a hybrid sequence model was selected to better capture long-range dependencies across variable-width word crops. In addition, explicit geometric normalization via an STN module was incorporated to reduce sensitivity to skew and slant in cropped word images.
Based on these preliminary investigations, the final specification was fixed as a two-stage OCR pipeline comprising a hybrid segmentation-based detector and a hybrid STN–CNN–Transformer recognizer, as this combination provided the most reliable convergence and generalization in preliminary experiments while remaining computationally feasible.

4.2 Data Collection

A word-level handwritten OCR dataset is constructed to support both detection and recognition. Due to the limited availability of Bangla datasets with word-level bounding box annotations, the dataset was created by combining a small portion of an existing benchmark with a larger self-collected corpus. Approximately 20% of the annotated samples were taken from the BanglaWriting dataset[28], while the remaining samples were obtained through additional data collection and manual annotation. After quality control, a total of 1,142 document images were retained for further processing. Today, this collection has contributed to make bengali word level annotated data available 4 times than previously.

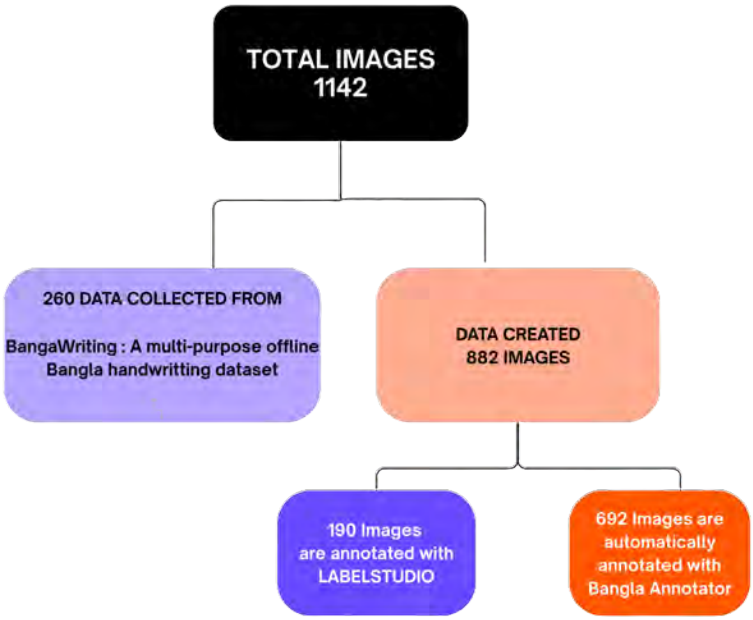


Figure 4.6: Data details

Word instances were extracted from these document images using annotated bounding boxes to generate recognition-ready crops. This initial conversion produced approximately 59,766 word crops prior to augmentation. A three-fold crop-level augmentation procedure was then applied, resulting in a final set of 179,297 recognition crops. The augmented dataset was subsequently organized into fixed splits, consisting of approximately 145,000 training samples, 18,953 validation samples, and 18,953 test samples.

4.2.1 Data Cleaning

Data cleaning was performed using explicit validation steps at the file, annotation, and crop levels to reduce noise and ensure stable training. At the file level, document images were verified by attempting to load and process them through the preprocessing pipeline, and corrupted or invalid files were removed. At the annotation level, bounding boxes were validated for geometric consistency, with boxes containing invalid coordinates or zero area being filtered.
At the crop level, extracted word images were screened based on measurable quality criteria. Crops that were excessively small, near-blank, or severely degraded were excluded. Label-level cleaning was also applied by removing empty transcriptions, trimming unintended whitespace, and discarding malformed or inconsistent text strings.

4.2.2 Data Transformation

To standardize learning conditions and improve robustness, data transformation was applied at both the document-image and crop levels. Multiple preprocessing variants were generated for each document image to handle contrast and illumination variation, including original images, CLAHE-enhanced images, and grayscale-based variants.
For recognition, word crops were resized to a fixed height while preserving aspect ratio, with padding applied where necessary to ensure batch compatibility. Pixel intensities

were normalized to a consistent range, and text labels were normalized using Unicode NFC to ensure consistent textual representation and reduce label redundancy. The three-fold crop-level augmentation was applied in a label-preserving manner, and augmented samples were restricted to their designated dataset split to avoid data leakage.

4.2.3 Data Integration

Only 260 images of word level bounding boxes are taken from BanglaWriting: A multi-purpose offline Bangla handwriting dataset which is a rare dataset of bengali handwritten text done on word level annotations[28]. So, out of 1142 images 882 image's annotations are self-done. And even among the 882 images 190 images are annotated at word level by using Label Studio, an annotation tool. Then the other 692 images are also self-annotated but using a different tool and will be discussed below. It is a self annotation tool developed to annotate Bengali handwritten texts. Each sample included document references, word bounding boxes, and corresponding tran- Scriptions. A manifest-based indexing approach was used to store crop paths and labels, enabling consistent use across training and evaluation pipelines while maintaining trace- ability across data sources and preprocessing variants.

Bangla Annotator

(Software to develop bengali word level annotations automatically) :

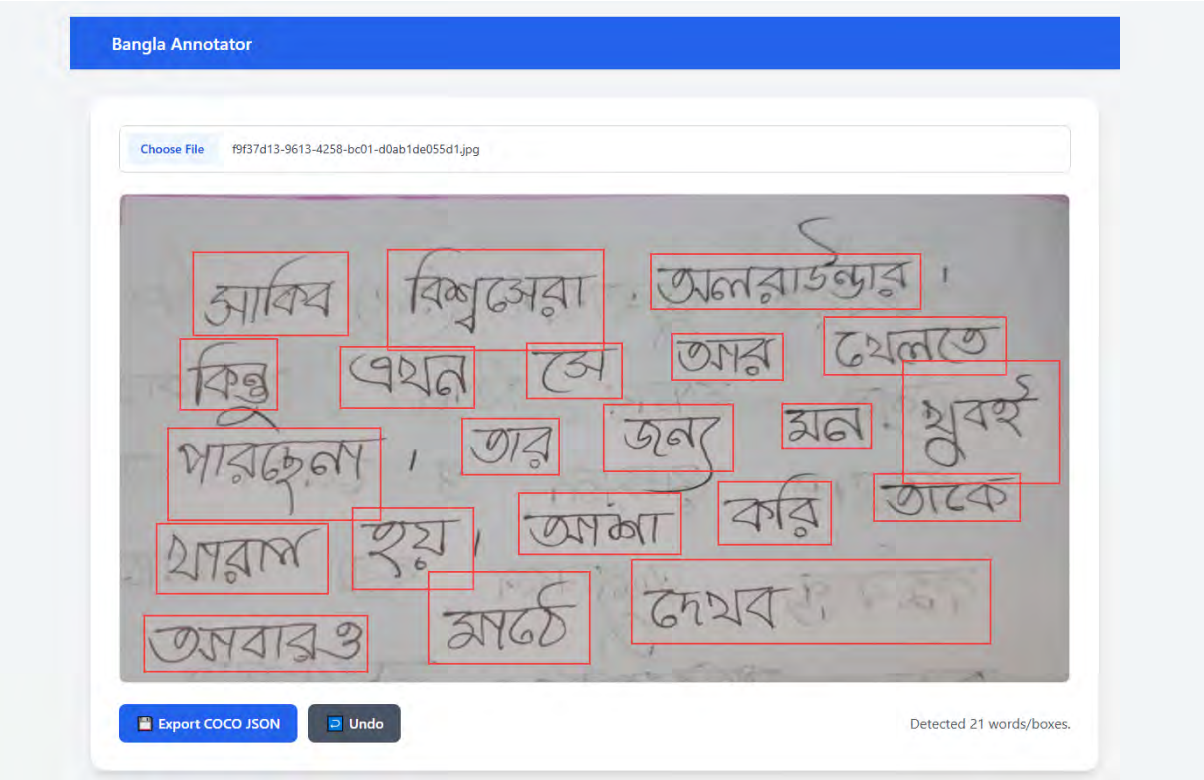


Figure 4.7: Automatic Detected Words with Bounding Boxes

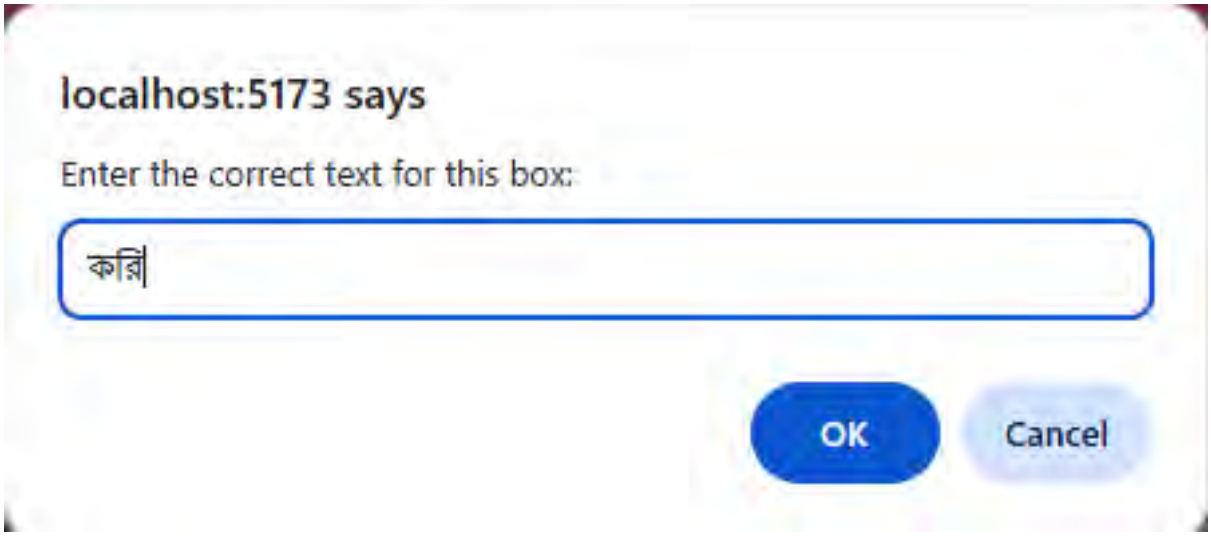


Figure 4.8: Pop-UP Text Edit Box

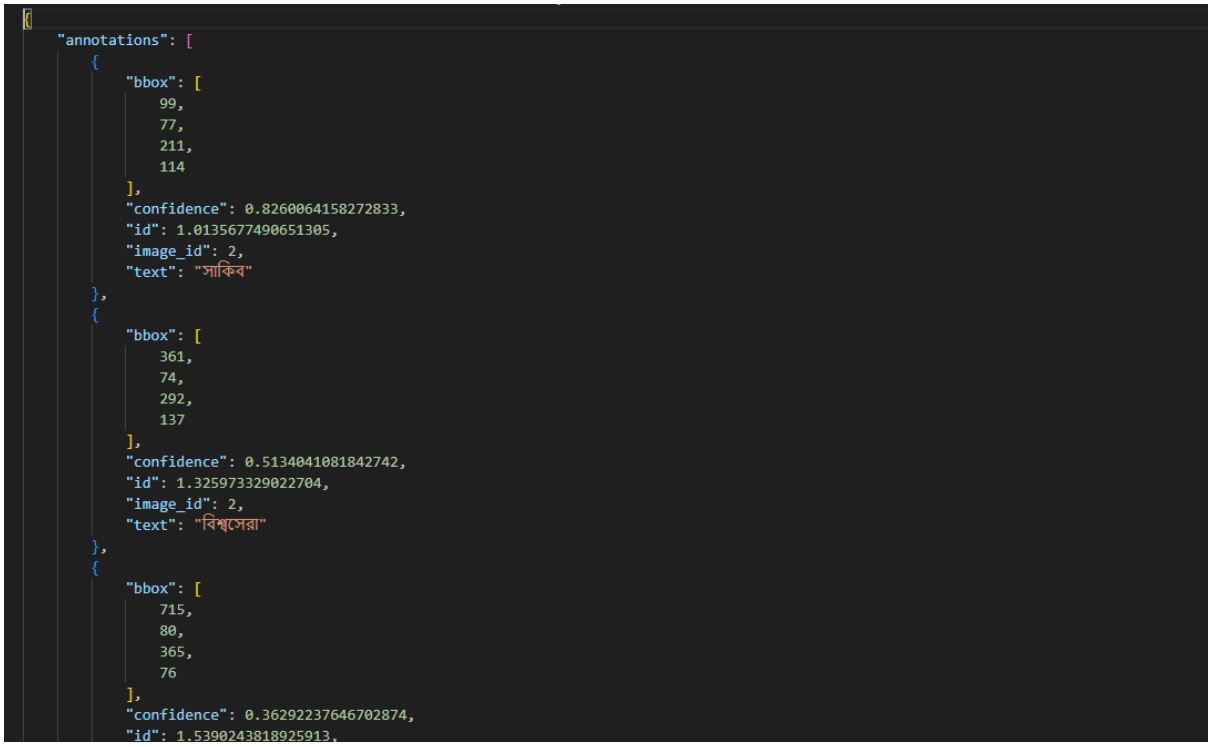


Figure 4.9: Generated COCO-JSON

A web-based system for automatic text annotation with bounding boxes from images was developed and named Bangla Annotator. The system generates bounding boxes on an uploaded image and detects each word automatically after a few seconds, after which the corresponding COCO JSON can be downloaded. The interface also provides an UNDO control to revert recent annotation edits. On the backend, EasyOCR was used to process images. When an image is uploaded, it is sent to the server, where EasyOCR detects handwritten text regions [47]. The backend then generates bounding box coordinates and associated text from these detections, and stores the annotations in COCO JSON format for later use in OCR training and document analysis. On the

frontend, Vite, Node.js and Tailwind CSS were used. Vite provides a fast de- velopment environment, Node.js supports the build process and API connectivity, and Tailwind CSS was used to build a responsive user interface. The frontend displays the uploaded image with automatically generated bounding boxes layered on top to support visual review of OCR results. When a bounding box is incorrect or mismatched, the interface supports manual cor- rection. Users can draw, click, delete, or redraw bounding boxes directly on the image, and manually edit text labels when OCR output is inaccurate. When a user redraws a bounding box, a prompt requests the corrected word, and after adjustments the modified annotations are sent back to the backend and stored. The system also reports the number of detected words based on bounding boxes, supporting efficient word-level annotation with minimal manual effort.

4.2.4 Data Reduction

Rather than applying classical dimensionality reduction techniques, data reduction was achieved through controlled filtering based on quality and reliability constraints. Crops with insufficient visual content, unreliable geometry, or ambiguous labels were removed. These steps reduced noise in the effective label distribution and improved training stability.

4.2.5 Summary of Preprocessed Data

After cleaning, transformation, integration, and reduction, the finalized dataset consisted of 1,142 document images and 179,297 recognition crops generated through three-fold augmentation. The training, validation, and test splits were fixed at approximately 145,000, 18,953, and 18,953 samples, respectively, and were used consistently across all experiments to ensure fair and comparable evaluation.

4.3 Implementation of Selected Design

The selected OCR system was implemented as a two-stage pipeline consisting of a custom word detector and a word recognizer, with supporting utilities for dataset preparation, training, evaluation, and end-to-end inference. The implementation was organized to ensure reproducibility by enforcing consistent preprocessing, fixed data splits, and explicit checkpointing of the best-performing models.

4.3.1 System Organization and Module Separation

The implementation was divided into four functional components. Data preparation utilities handled generation of manifests, crop extraction, and split management for detection and recognition training. The detection module was the Hybrid MNV3Seg segmentation-based detector, trained with mask supervision and deployed to generate word bounding boxes. The recognition module was the HybridHTR_STN_Transformer recognizer (`hybrid_2_me`), trained using CTC-based sequence learning on word crops. End-to-end inference and evaluation were provided through a unified pipeline that executes detection, crop extraction, recognition, and optional visualization or metric reporting.
This separation ensured that detection and recognition could be trained, replaced, and evaluated independently, while still supporting an end-to-end OCR workflow.

4.3.2 Implementation of the Detection Stage

The detection stage was implemented as a segmentation predictor that maps an input document image to a single-channel logit map, which is converted to a probability mask using a sigmoid activation. During inference, the detector operated at a fixed input resolution of 768×1024 to standardize computation across documents. The predicted probability map was resized back to the original image resolution and binarized using a fixed threshold ($\tau = 0.3$). Connected components were extracted from the binary mask, and each component was converted into an axis-aligned bounding rectangle. Very small components were filtered using explicit minimum size constraints ($w \geq 5$) and ($h \geq 5$), resulting in normalized word bounding boxes used for cropping.
For training, binary supervision masks were generated from ground-truth bounding boxes using a DB-like shrink operation with shrink ratio ($r = 0.8$), and the detector was optimized using the combined BCE + Dice loss formulation. Model weights were saved based on validation-loss improvement to ensure the retained checkpoint reflected the best generalization behavior.

4.3.3 Implementation of the Recognition Stage

The recognition stage is implemented in the dedicated module as a hybrid sequence recognizer combining an STN, a CNN feature extractor, and a Transformer encoder. Each word crop was resized to a fixed height 128 while preserving aspect ratio, producing variable-width inputs and therefore variable-length feature sequences. This design avoided distorting handwriting morphology and allowed the temporal resolution T to adapt to word width.
The recognizer employs a grapheme-level tokenizer to transform ground-truth transcriptions into integer sequences, enabling training with CTC alignment without character-level segmentation. During forward propagation, the CNN feature map was converted into a width-major sequence by collapsing the height dimension, after which the Transformer encoder modeled contextual dependencies across the sequence. The final linear classifier produced logits of shape (T, B, V), directly compatible with CTCLoss. During inference, greedy CTC decoding (argmax followed by collapse and blank removal) produced the final transcription.
The enhanced training implementation incorporated engineering measures to improve scalability and stability on large crop corpora, including mixed precision (AMP), gradient accumulation to increase effective batch size, cosine warm restarts scheduling, strong augmentation, gradient clipping, CER-based best-model selection, and controlled checkpoint retention.

4.3.4 End-to-End Pipeline Integration

After training, the detector and recognizer were integrated into a single end-to-end OCR pipeline. The pipeline accepts a document image, performs word detection to obtain normalized bounding boxes, extracts word crops using the predicted boxes, and applies the recognizer to each crop. The implementation also supports visualizations that display predicted bounding boxes and recognized text on the images and save the results in a structured format for analysis and evaluation.The two stages are kept separated.The detector gives word-level bounding boxes in normalized coordinates $[x_1, y_1, x_2, y_2]$, The implementation also supports visualizations that display predicted bounding boxes and

recognized text on the images and save the results in a structured format for analysis and evaluation.The two stages are kept separated.The detector gives word-level bounding boxes in normalized coordinates

4.3.5 Reproducibility and Model Artifact Management

To ensure reproducibility and traceability, the implementation maintained fixed dataset partitions for training/validation/testing and consistent preprocessing parameters (e.g., detector input size, mask threshold, crop height). It also maintained checkpoint-based model persistence with best-performing weights saved according to validation criteria, along with training logs and metric exports to compare runs and configurations. As a result, the final system implementation supports both controlled experimentation (ablation and tuning) and end-to-end deployment for document OCR inference under consistent settings.

Chapter 5

Result Analysis

5.1 Performance Evaluation

In the performance section, the overall performance of the Punolikhon model at every phase of its pipeline, which includes the task of preprocessing along with the purpose of word detection and recognition are evaluated. It also includes gtbox word cropping as well as detection, which finally helped to evaluate recognition performance of OCR. This evaluation is completed via some notable quantitative metrics along with some controlled test techniques. The name of the techniques are mentioned below :

Precision

It measures how many of the detected objects are actually correct.
where:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

TP (True Positives): Correctly predicted defects. FP (False Positives): Incorrectly predicted defects.
The higher the precision is, the fewer false positives there are, which makes the model more reliable.

F1-Score

Keeps stability between precision and recall. F1-score balances precision and recall.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

To measure the model accuracy F1 score is used.With the help of this score we can tell how frequently the model makes the right prediction.The better F1 score, the better model as a classifier.

Mean Average Precision (mAP)

It identifies the model accuracy to detect objects by countingIoU thresholds, precision across multi classes and precision-recall curves

where:

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i$$

N = Number of object classes
AP$_i$ = Average Precision for class i
By measuring mean precision among several classes and precision-recall curves it calculates the model's recognition performance.

Average Intersection over Union (Average IoU)

Tells the overlap between truth and predicted bounding boxes. It gives an idea of how precisely the model limits the text regions.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

A higher IoU value denoting the predicted box closely matches the truth which indicates letter performance.The average IoU indicates the mean among all correct predictions.

False Positive Rate (FPR)

By False Positive Rate it is measured how frequently the model predicts incorrect word regions where nothing exists.

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

FP (False Positives): Incorrect detections made by the model.
TN (True Negatives): Correctly identified non-text regions. A lower FPR indicates that the detector avoids unnecessary or incorrect detections, which is crucial for precise localization of handwritten text.

Character Error Rate (CER)

CER evaluates the recognition performance by comparing the predicted character sequence with the ground truth.

$$\text{CER} = \frac{S + D + I}{N}$$

S: Substitutions (incorrect characters)
D: Deletions (missing characters)
I: Insertions (extra characters)
N: Total characters in the ground truth Lower CER indicates the model is recognizing characters more accurately, even for complex or degraded handwriting.

Word Error Rate (WER)

Word Error Rate measures the recognition accuracy at the word level, capturing substitutions, insertions, and deletions of words.

$$\text{WER} = \frac{S + D + I}{N}$$

S: Incorrectly predicted words
D: Words missed by the model
I: Extra words generated
N: Total words in the ground truth A lower WER reflects better recognition capability and sentence-level coherence.

Character and Word Accuracy

These metrics directly measure the proportion of correctly predicted characters or words.

$$\text{Character Accuracy} = 1 - \text{CER}$$

$$\text{Word Accuracy} = \frac{\text{Correct Words}}{\text{Total Words}}$$

Higher accuracy values mean the model is more consistent in recognizing both individual characters and complete words, which is critical for precise OCR performance.

Average Confidence

Average confidence measures the model's self-assessed certainty for its predictions. It is computed by averaging the confidence scores assigned to all recognized words or characters. A higher average confidence reflects stronger model certainty, indicating reliable recognition performance.

5.2 Analysis of Design Solutions (Detection and Recognition)

5.2.1 Detection Model: Quantitative Performance

The final detector (hybrid backbone, mask-threshold 0.35) achieved strong localization quality at the standard IoU threshold and remained precision-dominant, indicating a low false-positive tendency. On the evaluation subset of 183 images, the detector achieved the following results.

Metric	**Value**
mAP@0.5	0.7438
Precision@0.5	0.8966
Recall@0.5	0.7955
F1@0.5	0.8431
Average matched IoU@0.5	0.6940
mAP@0.75	0.0234
F1@0.75	0.1608

Table 5.1: Detection performance summary on the evaluation subset (183 images)

At the stricter overlap threshold, performance reduced substantially. This difference between IoU thresholds indicates that word regions were typically localized in the correct vicinity, but the predicted boundaries were not consistently tight enough to satisfy

strict overlap constraints. This behavior is consistent with segmentation-derived detection pipelines, where boundary tightness may be affected by broken strokes, touching words, and contour-to-box conversion sensitivity. Practically, the strong F1@0.5 suggests that the detector is suitable for generating candidate crops for recognition, while strict boundary refinement remains a limiting factor for high-IoU evaluation.

5.2.2 Recognition Model (Hybrid_HTR_STN): Quantitative Performance

The final recognition model is evaluated on 18,953 test samples using greedy decoding. The model achieved an overall word accuracy of 0.8427 (84.27%), indicating that more than four out of five test words were transcribed with an exact match. The character-level performance remained higher, with overall character accuracy of 0.8889 (88.89%).
In terms of error rates, the overall CER mean is 0.0987 and the overall WER mean is 0.0985, reflecting stable character and word-level performance at approximately the 9–10% error range. The average prediction confidence is 0.9210, indicating that the model's output distribution remained relatively sharp and that most predictions were not produced under high uncertainty.

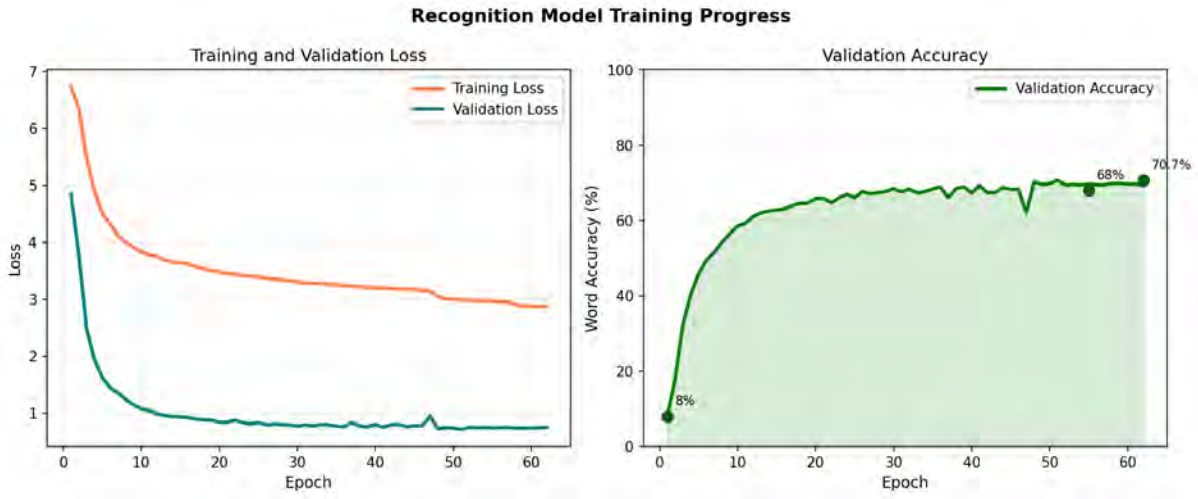


Figure 5.1: Recognition model training curves using tokenizer

Metric	Value
Overall word accuracy	0.8427
Overall character accuracy	0.8889
Overall CER mean	0.0987
Overall WER mean	0.0985
Average confidence	0.9210

Table 5.2: Overall recognition performance of Hybrid_HTR_STN on 18,953 test samples (greedy decoding)

A language-wise breakdown further highlighted the differential difficulty across scripts.

Metric	Bangla	English
Word accuracy	0.8572	0.8333
Character accuracy	0.8960	0.8339
CER	0.0882	0.1816
WER	0.0879	0.1816

Table 5.3: Language-wise recognition performance of Hybrid_HTR_STN (greedy decoding)

These results indicate that the Bangla subset achieved stronger character-level stability than English, while English has showed comparatively higher error rates under the same decoding strategy. This difference is consistent with the practical challenges of mixed-script recognition, where token frequency and representation imbalance can affect the stability of sequence modeling and decoding. Overall, the results support that the hybrid design (STN-based normalization + lightweight CNN features + attention-based sequence modeling) produces robust recognition performance under realistic word-crop conditions.

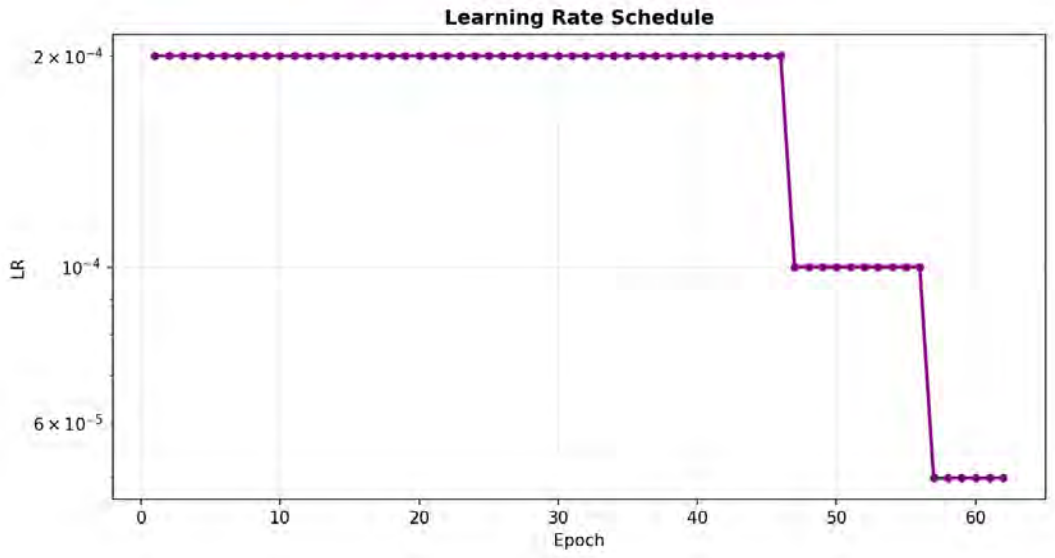


Figure 5.2: Recognition learning-rate schedule across epochs

5.3 Final Design Adjustments (based on empirical findings)

Several adjustments are introduced after diagnostic evaluation to improve measurement correctness and model reliability. First, Unicode NFC normalization is applied during evaluation so that visually identical Bangla grapheme clusters are not treated as different strings due to alternative Unicode compositions. Second, a grapheme-based tokenization strategy is adopted to better represent conjuncts, diacritics, and orthographic dependencies common in Bangla handwriting. Third, the inclusion of the STN module was retained because it provides geometric normalization that reduces the burden on the sequence model when handwriting is skewed, stretched, or locally distorted. Finally,

training refinements such as stronger augmentation and stability-focused schedules are used to reduce overfitting and improve generalization under real crop variability.

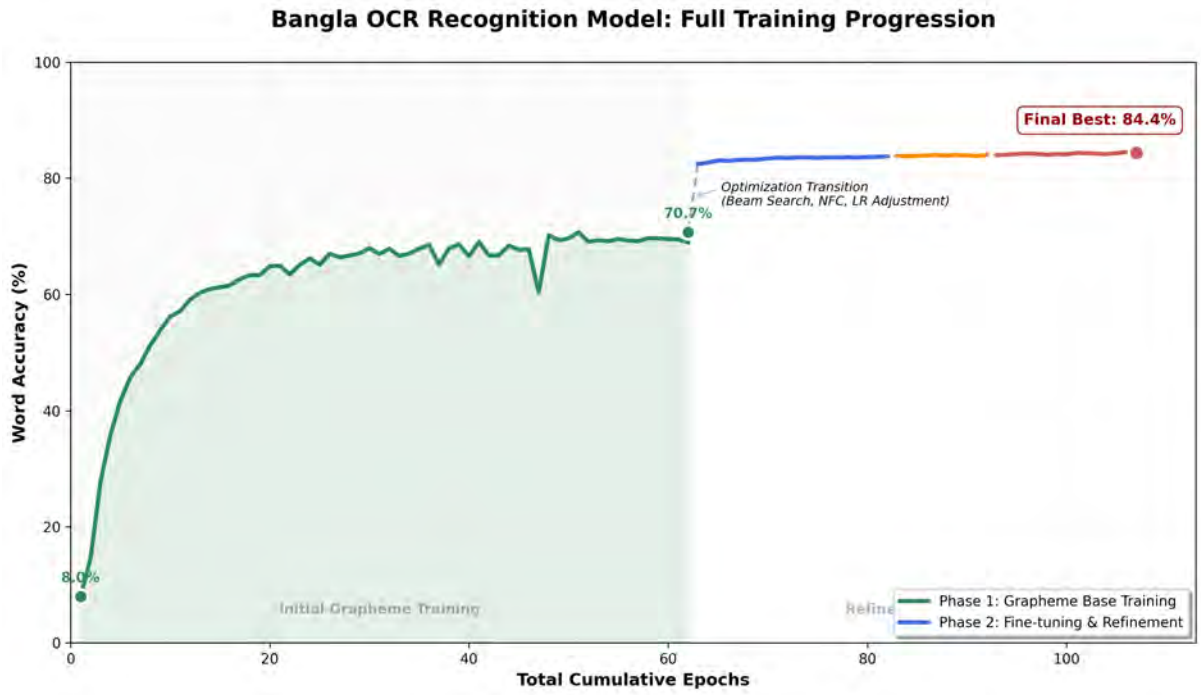


Figure 5.3: Full training progression: word accuracy over cumulative epochs

5.4 Statistical and Robustness Analysis

The evaluation was complemented by robustness-oriented analyses to characterize where errors concentrate and how reliability changes across sample properties. Word-length sensitivity was examined by analyzing word accuracy across buckets of grapheme length while also reporting the corresponding sample volume per bucket. This analysis supports the claim that recognition difficulty increases with structural complexity, particularly when longer words contain denser conjunct patterns, touching strokes, or ambiguous diacritics.

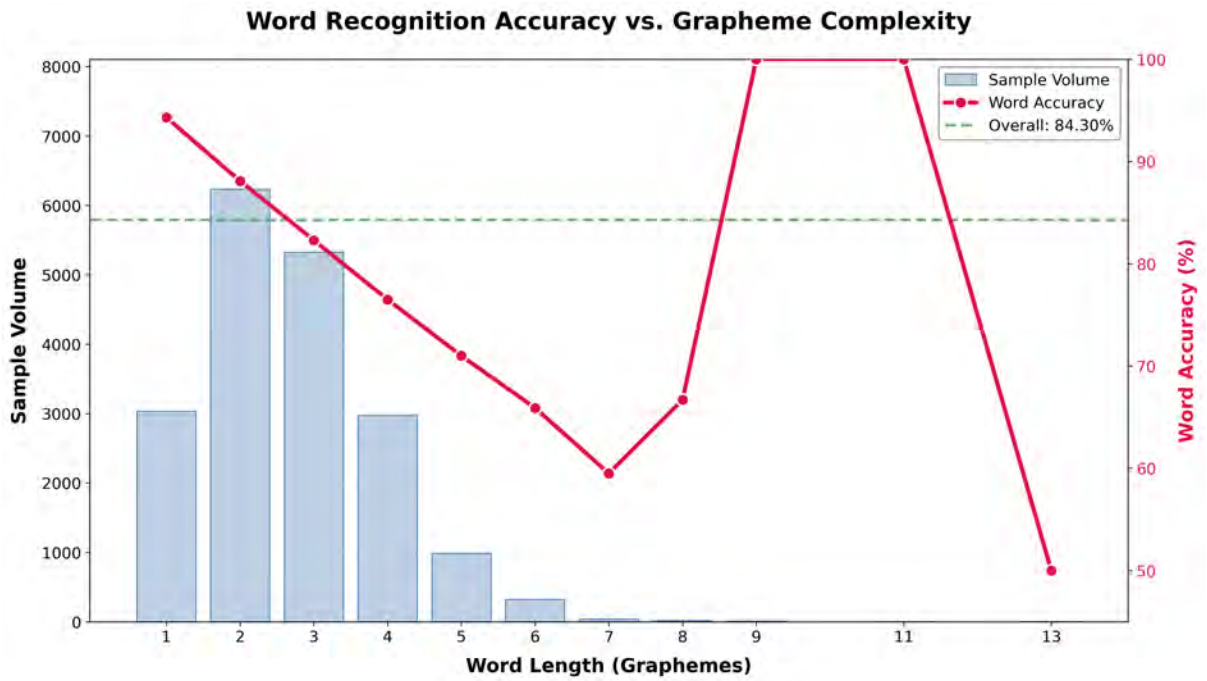


Figure 5.4: Quantitative analysis (Accuracy vs Length))

Figure 5.4 illustrates the recognition accuracy over 18,953 test samples, which helps to reach overall 84.3% word accuracy. The grey bars on the graph define length distribution, where samples of 99.6% have graphemes from 1 to 6. The red line on the graph shows a shift of accuracy from 94.3% (1 grapheme) to 65.9% (6 graphemes). Finally, this overall performance is measured with the help of the weighted sum: $(94.3\% \times 16.0\%) + (88.1\% \times 32.9\%) + (82.3\% \times 28.1\%) + (76.5\% \times 15.7\%) + (71.0\% \times 5.2\%) + (65.9\% \times 1.7\%) \approx 84.3\%$.

This result is then combined with a character accuracy of 88.9% (89.6% for Bengali), as a result, these results predict some errors which are situated in some specific clusters of complex patterns rather than random. Finally, this group of errors provide some challenging instances which are evidenced by the fig. 5.5. This demonstrates the model's overall robustness on any authentic samples of handwritten text.

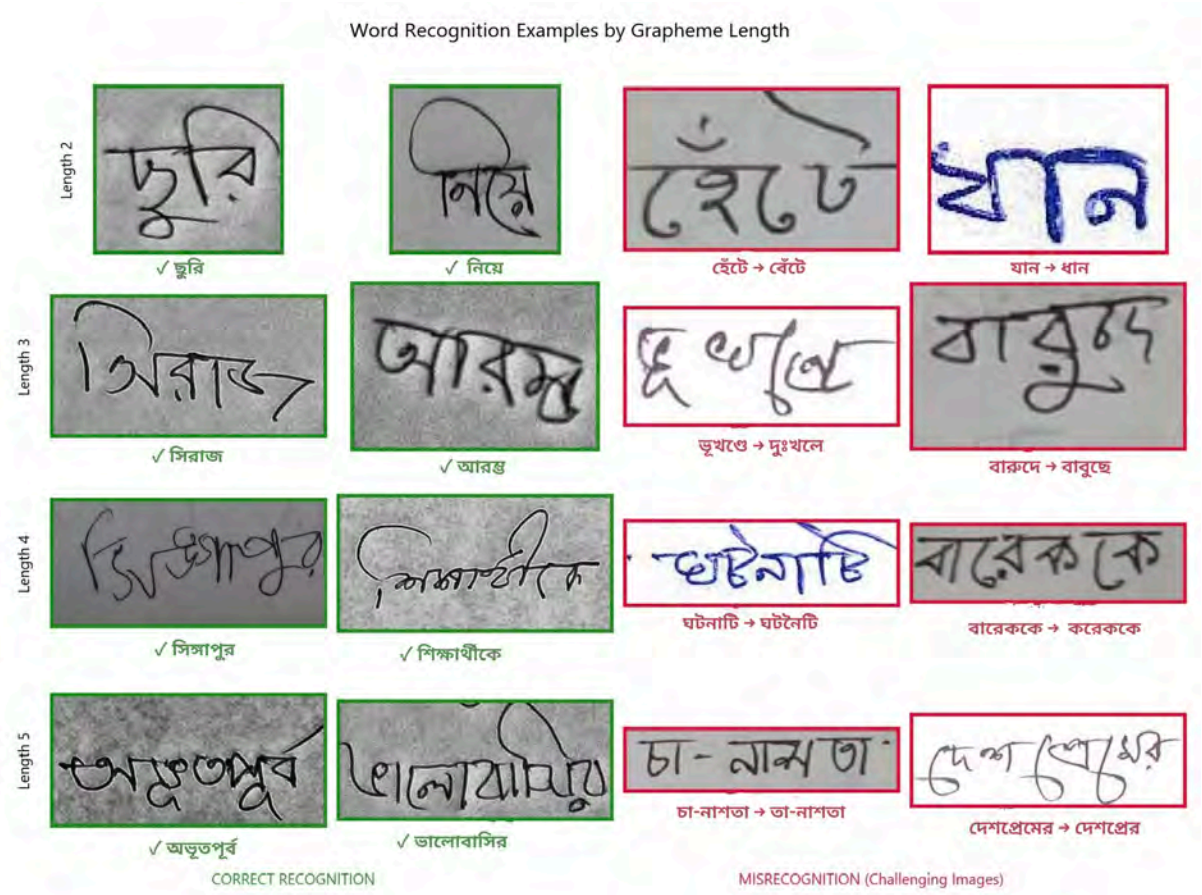


Figure 5.5: Qualitative verification (Gallery of real examples)

Figure 5.5 simplifies the data of overall performance in prvious figure with a complete visualization overview of real samples from the Punolikhon test set.In order to avoid selection bias automatically examples are grouped according to their quality.True results use clear pictures whereas errors are samples from low quality images.This process shows that accuracy reduce in figure 5.4 is a low input result rather than the model flaws.Likewise, This figure shows that all visual outcome extracted directly from the untouched test dataset.

5.5 Comparisons and Relationships

The relationship between detection and recognition performance was observed to be strongly dependent on crop fidelity. Even when a recognition model performs well on clean and well-centered word crops, truncated diacritics, merged adjacent words, or excessive background can degrade recognition accuracy. Therefore, improvements in detection boundary alignment and reduction of missed words are expected to directly improve end-to-end OCR performance. The detector's strong precision at IoU 0.5 indicates that most produced crops are relevant word regions, but the lower strict-IoU performance indicates that boundary tightness is a primary factor limiting downstream transcription reliability.

WORK	MODEL	GOAL	METRICS	End-to-End	STRENGTH	LIMITATIONS
Rural OCR [48]	CNN + InceptionV3	Handwritten Word Recognition	95%	Unclear	High accuracy on handwritten queries	Robustness issues; CER, WER not mentioned; not lightweight
BERT-NER [49]	BERT-base NER model	Word-level Language Identification	95% (classification)	No	Classical ML baselines; strong code-mixed token classification	Not image-based; not handwritten; heavy backbone (BERT)
Scene BD [50]	Convolutional layers and GRU	Scene Text Localization + Recognition	Bengali 79.80% English 84.73%	Partial	Handles bilingual text in real-world scenes	Difficulties for small text; not lightweight
Bilingual Language Detection [51]	CNN + transformer-based recognition system	Language and Printing Style Detection	85% (classification)	No	Useful for bilingual scanned document analysis and language detection	Not OCR transcription; character-level classification
Punolikhon	MobileNetV3-small backbone with hybrid ResNet	End-to-End Handwritten OCR	mAP@0.5 = 0.7438 F1@0.5 = 0.8431 Word Acc = 84.3% WER = 9.85 CER = 9.87	Yes	Complete pipeline with lightweight design; robustness analysis; transparent OCR metrics	Recognition accuracy below best reported recognizer

Table 5.4: Comparison of Punolikhon with selected Bangla–English OCR and language-processing works.

Table 5.4 positions Punolikhon relative to recent bilingual Bangla–English studies under a strength-focused lens that accounts for task definition, data domain, and deployment constraints. Although *RuralOCR* [48] reports higher recognition accuracy on a handwritten query dataset, the approach relies on a comparatively heavier InceptionV3-based backbone and does not clearly present an end-to-end detection-to-recognition pipeline or detailed OCR error metrics such as CER/WER. In contrast, Punolikhon is designed as a complete lightweight OCR system and is evaluated using standard transcription measures (word accuracy, CER/WER) alongside confidence-based reliability diagnostics and robustness analyses. Other works address adjacent but different objectives: *BERT-NER* [49] focuses on token-level language identification in text rather than OCR, and *LangStyle* [50] targets scanned-document analysis rather than word-level transcription. *SceneBEn* [51] demonstrates bilingual capability in natural environments but operates in a domain that differs substantially from handwriting. Overall, Punolikhon contributes a deployable and transparently evaluated handwritten word-level OCR pipeline, while remaining performance gaps motivate stricter crop refinement and further improvements in mixed-script robustness.

5.6 Discussions (interpretation, implications, limitations)

The reported results indicate that a practical Bangla handwritten OCR system can be developed using efficient architectural choices while maintaining strong accuracy. The detection results demonstrate reliable localization at the standard IoU threshold, which supports the feasibility of generating word crops for downstream recognition. The recog-

nition results demonstrate that Hybrid_HTR_STN produces stable transcription performance on a large mixed-script test set, with stronger performance on Bangla and interpretable limitations in error concentration (word complexity, rare tokens, and crop degradation).
However, limitations remain. Strict localization quality at IoU 0.75 was low, indicating that boundary refinement and segmentation-to-box conversion remain challenging under dense handwriting. Recognition errors persisted in visually ambiguous grapheme patterns and in degraded or low-resolution crops. Mixed-script evaluation further suggests that distribution imbalance and token rarity affect stability, motivating further dataset expansion and more controlled script balancing during training.

To conclude, this chapter summarizes the outcomes of the proposed Bengali–English OCR pipeline built with a two-stage design. The system was developed using systematic preprocessing, stability-oriented training, and efficient **lightweight** model components to support practical deployment. For the detection stage, a segmentation-based word detector was employed using a MobileNetV3-based lightweight backbone with a DB-style probability-map formulation, optimized using a combined **BCE + Dice** objective. On the evaluated set document, the detector achieved **mAP@0.5 = 0.7438**, **Precision@0.5 = 0.8966**, **Recall@0.5 = 0.7955**, and **F1@0.5 = 0.8431**, indicating reliable word-level localization at the standard IoU threshold.
For the recognition stage, the final model (**Hybrid_HTR_STN**) integrated a Spatial Transformer Network (STN) for geometric normalization with a lightweight CNN feature extractor and attention-based sequence modeling under a CTC decoding framework (greedy decoding). On the test evaluation of **18,953** word crops, an overall **word accuracy of 0.8427 (84.27%)** and **character accuracy of 0.8889 (88.89%)** were obtained, with **CER = 0.0987** and **WER = 0.0985**, and an average confidence of **0.9210**. Language-wise performance remained consistent, with **Bangla word accuracy = 0.8572** (Bangla CER = **0.0882**) and **English word accuracy = 0.8333** (English CER/WER = **0.1816**), reflecting the increased difficulty and variability of mixed-script handwritten recognition under real crop conditions.
Although the remaining CER/WER values indicate that recognition errors persist, these errors are attributable to intrinsic handwriting ambiguities, rare-token effects, degraded crops, and sensitivity to imperfect crop boundaries rather than a small-scale dataset assumption. The dataset used in this work was constructed from **1,142** annotated document images and expanded through word-level extraction and augmentation to provide a substantially larger recognition corpus. Further improvements are expected through continued dataset expansion (especially for underrepresented patterns), stricter crop refinement at higher IoU alignment, and end-to-end optimization that reduces error propagation from detection to recognition.
In future our plan is to use knowledge distillation to mobilenet v3 to make it more robust in detection model to achieve our goal which is to reconstruct or recognize even damaged bengali texts.

Chapter 6

Conclusion

6.1 Summary of Findings

The goal of this research is to develop a bilingual Optical Character Recognition system for recognizing both Bengali and English handwritten text even from distorted paper. The Punolikhon hybrid system that combines YOLOv8n for text detection and HybridHTR_STN_Transformer for recognition successfully attained a high performance in recognition for both Bengali and English languages. This model reached a word accuracy of 84.3%, 85.7% of Bengali Word accuracy and character accuracy of 88.4%. The Character Error Rate (CER) for Bengali text was 8.97%, whereas the total CER for English and Bengali was 10.2%. Moreover, the model shows a high average confidence of 93.4%. This research shows that this hybrid architecture with a custom-designed grapheme-based tokenization system can effectively deal with the complexities of Bengali script. Also, preprocessing methods CLAHE and greyscale help to improve text visibility in low quality images that boost up the model's performance.

6.2 Contributions to the Field

This research makes some important contributions in the field of optical character recognition, especially in handwritten Bengali and English text context.

Word-level annotated Bengali dataset:

A word-level annotated Bengali dataset which is not only novel but also rarely found can be accessible to everyone. We have increased word-level Bengali annotated dataset four times than available.

Hybrid deep learning model:

This research built a novel hybrid architecture that combines YOLOv8n for text detection and HybridHTR_STN_Transformer for recognition. This integration allows the model to attain high performance and reliability.

Web-based annotation system:

A web-based system for automatic text annotation with bounding boxes from images was developed and named Bangla Annotator. It generally creates bounding boxes on the uploaded image and detects each word automatically after a few seconds. Then the COCO JSON can be downloaded.

Custom tokenization:

Especially for Bengali and English text combined, a unique tokenization technique is developed to handle the complexities of the Bengali symbols, as traditional OCR mostly works with other languages rather than Bengali.

6.3 Recommendations for Future Work

Although the Punolikhon model shows promising results, there are some areas where future improvements can be made.

Expansion of dataset:

In order to improve the model accuracy, more sample Bengali dataset can be collected and annotated that will include more variants in handwriting. As there are few publicly available dataset on Bengali with bounding boxes, English dataset can also be increased in order to improve the model's accuracy on English handwritten text.

End-to-end OCR integration:

Although the detection and recognition models were trained and evaluated separately, the complete page-to-text OCR pipeline can be improved further. In end-to-end settings, recognition accuracy may decrease when the recognizer receives detection-generated crops instead of ground-truth crops due to imperfect localization. Future work can focus on improving the reliability of detector-produced crops, such as reducing truncated diacritics or stroke endings, avoiding merged crops containing multiple words, and designing better mask-to-box post-processing. In addition, more robust region-matching and evaluation protocols can be introduced to handle extra or missing detections in a principled way, which will make the end-to-end OCR results more stable and realistic.

Reconstruction model:

A custom built model can be made that can do both recognition and reconstruction at the same time for handwritten documents. There is a gap in traditional OCR as most of them can only work with printed text or only do recognition, not both. A hybrid mode using Knowledge Distillation can be made that can do both tasks. This model would be based on diffusion based image generation which can be tested to recover severely distorted or missing areas of handwritten texts before OCR processing. It will learn to guess and reconstruct faded, torn or obscured words by using semantic context from surrounding texts. Transformer based architecture with contextual embedding can be used so that the model can comprehend long range dependencies between characters and words. It will

allow to build visually consistent and semantically relevant reconstruction. This process can be implemented as a post recognition enhancement stage which will allow the OCR model to produce more complete and clear textual outputs from damaged documents.

Bibliography

[1] H. K. Buddha, J. S. Meka, and P. Choppala, "Ocr image enhancement & implementation by using clahe algorithm," *Mukt Shabd J*, vol. 9, pp. 3595–3599, 2020.

[2] C. Vinotheni and S. L. Pandian, "End-to-end deep-learning-based tamil handwritten document recognition and classification model," *IEEE Access*, vol. 11, pp. 43 195–43 204, 2023. DOI: 10.1109/ACCESS.2023.3270895

[3] M.-S. Kim, C.-H. Son, and S.-H. Choi, "A novel federated learning-based image classification model for improving chinese character recognition performance," *IEEE Access*, vol. 12, pp. 185 971–185 991, 2024. DOI: 10.1109/ACCESS.2024.3514319

[4] M. N. I. Opu, M. E. Hossain, and M. A. Kabir, "Handwritten Bangla character recognition using convolutional neural networks: a comparative study and new lightweight model," *Neural Computing and Applications*, vol. 36, no. 1, pp. 337–348, Nov. 2023. DOI: 10.1007/s00521-023-09008-8 [Online]. Available: https://doi.org/10.1007/s00521-023-09008-8

[5] R. Malhotra and M. T. Addis, "Handwritten amharic word recognition with additive attention mechanism," *IEEE Access*, vol. 12, pp. 114 645–114 657, 2024. DOI: 10.1109/ACCESS.2024.3444897

[6] H. Ouyang et al., "ChemReco: automated recognition of hand-drawn carbon–hydrogen–oxygen structures using deep learning," *Scientific Reports*, vol. 14, no. 1, p. 17 126, Jul. 2024. DOI: 10.1038/s41598-024-67496-7 [Online]. Available: http://dx.doi.org/10.1038/s41598-024-67496-7

[7] R. Malhotra and M. T. Addis, "End-to-end historical handwritten ethiopic text recognition using deep learning," *IEEE Access*, vol. 11, pp. 99 535–99 545, 2023. DOI: 10.1109/ACCESS.2023.3314334

[8] B. Dutta et al., "Deep learning for terahertz image denoising in nondestructive historical document analysis," *Scientific Reports*, vol. 12, no. 1, p. 22 554, Dec. 2022. DOI: 10.1038/s41598-022-26957-7 [Online]. Available: https://doi.org/10.1038/s41598-022-26957-7

[9] M. Boillet, C. Kermorvant, and T. Paquet, "Robust text line detection in historical documents: learning and evaluation methods," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 25, no. 2, pp. 95–114, Mar. 2022. DOI: 10.1007/s10032-022-00395-7 [Online]. Available: https://doi.org/10.1007/s10032-022-00395-7

[10] A. Jindal and R. Ghosh, "Text line segmentation in indian ancient handwritten documents using faster R-CNN," *Multimedia Tools and Applications*, vol. 82, no. 7, pp. 10 703–10 722, Sep. 2022. DOI: 10.1007/s11042-022-13709-y [Online]. Available: https://doi.org/10.1007/s11042-022-13709-y

[11] R. Krithiga, S. R. Varsini, R. G. Joshua, and C. U. Om Kumar, "Ancient character recognition: A comprehensive review," *IEEE Access*, vol. 13, pp. 88 847–88 857, 2025. DOI: 10.1109/ACCESS.2023.3341352

[12] W. Xiao and H. Wu, "Learning features for offline handwritten signature verification using spatial transformer network," *Scientific Reports*, vol. 15, no. 1, p. 9453, Mar. 2025. DOI: 10.1038/s41598-025-92704-3 [Online]. Available: https://doi.org/10.1038/s41598-025-92704-3

[13] H. Sun, J. Sun, H. Yu, B. Guo, M. Liu, and H. Dai, *Enhanced MobileNetV3 for lightweight YI script recognition*. Jan. 2026, pp. 157–166. DOI: 10.1007/978-981-95-4049-5\{_}16 [Online]. Available: https://doi.org/10.1007/978-981-95-4049-5_16

[14] M. M. Hasan, A. N. T. Choudhury, M. Hasan, and M. M. Khan, "GraDeT-HTR: A resource-efficient Bengali handwritten text recognition system utilizing grapheme-based tokenizer and decoder-only transformer," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, I. Habernal, P. Schulam, and J. Tiedemann, Eds., Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 696–706, ISBN: 979-8-89176-334-0. DOI: 10.18653/v1/2025.emnlp-demos.52 [Online]. Available: https://aclanthology.org/2025.emnlp-demos.52/

[15] B. S. Rao, J. N. Rao, B. Vamsi, V. N. Thatha, and K. S. Rao, "A unicode based deep handwritten character recognition model for telugu to english language translation," *International Journal of Computer Science & Network Security*, vol. 24, no. 2, pp. 101–112, 2024.

[16] N. Ansary et al., "Unicode normalization and grapheme parsing of indic languages," in *Proceedings of the 2024 joint international conference on computational linguistics, language resources and evaluation (LREC-COLING 2024)*, 2024, pp. 17 019–17 030.

[17] M. Mohd, F. Qamar, I. Al-Sheikh, and R. Salah, "Quranic optical text recognition using deep learning models," *IEEE Access*, vol. 9, pp. 38 318–38 330, 2021. DOI: 10.1109/ACCESS.2021.3064019

[18] M. Bugeja, A. Dingli, and D. Seychell, "An overview of handwritten character recognition systems for historical documents," *Rediscovering Heritage Through Technology: A Collection of Innovative Research Case Studies That Are Reworking The Way We Experience Heritage*, pp. 3–23, 2020.

[19] S. Tabassum et al., "An online cursive handwritten medical words recognition system for busy doctors in developing countries for ensuring efficient healthcare service delivery," *Scientific Reports*, vol. 12, no. 1, p. 3601, Mar. 2022. DOI: 10.1038/s41598-022-07571-z [Online]. Available: https://doi.org/10.1038/s41598-022-07571-z

[20] M. G., A. M., M. D. N., T. M., V. K. D., and M. V., "Pixel-level reconstruction noisy handwritten characters classification based on deep progressively learning model," in *2025 6th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)*, 2025, pp. 873–877. DOI: 10.1109/ICMCSI64620.2025.10883382

[21] M. T. Hossain, M. W. Hasan, and A. K. Das, "Bangla handwritten word recognition system using convolutional neural network," in *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 2021, pp. 1–8. DOI: 10.1109/IMCOM51814.2021.9377410

[22] S. M. Jubaer, N. Tabassum, M. A. Rahman, and M. K. Islam, *BN-DRISHTI: Bangla Document Recognition through Instance-Level segmentation of handwritten text images*. Jan. 2023, pp. 195–212. DOI: 10.1007/978-3-031-41501-2\{_}14 [Online]. Available: https://doi.org/10.1007/978-3-031-41501-2_14

[23] A. Pal, M. S. Hasan, and S. M. Masudul Ahsan, "Improving character recognition in bangla handwritten words: A two-stage single shot detector approach," in *2024 International Conference on Advances in Computing, Communication, Electrical, and Smart Systems (iCACCESS)*, 2024, pp. 1–6. DOI: 10.1109/iCACCESS61735.2024.10499463

[24] I. M. Zulkarnain et al., "BBOCR: An open-source multi-domain OCR pipeline for Bengali documents," *arXiv (Cornell University)*, Aug. 2023. DOI: 10.48550/arxiv.2308.10647 [Online]. Available: http://arxiv.org/abs/2308.10647

[25] M. J. I. Basher, M. R. Noor, S. Afroze, I. Ahmed, and M. M. Hoque, "Bngraphemizer: A grapheme-based tokenizer for bengali handwritten text recognition," in *2023 IEEE 9th International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE)*, 2023, pp. 183–188. DOI: 10.1109/WIECON-ECE60392.2023.10456463

[26] K. Roy et al., "A multifaceted evaluation of representation of graphemes for practically effective Bangla OCR," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 27, no. 1, pp. 73–95, Aug. 2023. DOI: 10.1007/s10032-023-00446-7 [Online]. Available: http://dx.doi.org/10.1007/s10032-023-00446-7

[27] M. J. Hossain, S. Samiha Zaman, F. R. Akash, M. Rifat Sarker, and M. R. Huq, "Developing a bangla handwritten text recognition framework using deep learning," in *2023 26th International Conference on Computer and Information Technology (ICCIT)*, 2023, pp. 1–6. DOI: 10.1109/ICCIT60459.2023.10441107

[28] D. M. F. Mridha, *BanglaWriting: A multi-purpose offline Bangla handwriting dataset*, Oct. 2020. DOI: 10.17632/r43wkvdk4w.1 Accessed: [Online]. Available: https://easy.dans.knaw.nl/ui/datasets/id/easy-dataset:190255

[29] A. T. Maung, S. Salekin, and M. A. Haque, "A hybrid approach to Bangla handwritten OCR: combining YOLO and an advanced CNN," *Discover Artificial Intelligence*, vol. 5, no. 1, Jun. 2025. DOI: 10.1007/s44163-025-00251-7 [Online]. Available: https://doi.org/10.1007/s44163-025-00251-7

[30] A. S. A. Rabby, S. Haque, S. Islam, S. Abujar, and S. A. Hossain, "BornoNet: Bangla Handwritten Characters Recognition using Convolutional Neural Network," *Procedia Computer Science*, vol. 143, pp. 528–535, Jan. 2018. DOI: 10.1016/j.procs.2018.10.426 [Online]. Available: https://doi.org/10.1016/j.procs.2018.10.426

[31] M. Adnan, F. Rahman, M. Imrul, N. Al, and S. Shabnam, "Handwritten Bangla Character Recognition using Inception Convolutional Neural Network," *International Journal of Computer Applications*, vol. 181, no. 17, pp. 48–59, Sep. 2018. DOI: 10.5120/ijca2018917850 [Online]. Available: http://doi.org/10.5120/ijca2018917850

[32] R. Parthiban, R. Ezhilarasi, and D. Saravanan, "Optical character recognition for english handwritten text using recurrent neural network," in *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, 2020, pp. 1–5. DOI: 10.1109/ICSCAN49426.2020.9262379

[33] R. R. Ingle, Y. Fujii, T. Deselaers, J. Baccash, and A. C. Popat, "A scalable handwritten text recognition system," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, pp. 17–24. DOI: 10.1109/ICDAR.2019.00013

[34] J. Memon, M. Sami, R. A. Khan, and M. Uddin, "Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr)," *IEEE Access*, vol. 8, pp. 142 642–142 668, 2020. DOI: 10.1109/ACCESS.2020.3012542

[35] A. Yuan, G. Bai, L. Jiao, and Y. Liu, "Offline handwritten english character recognition based on convolutional neural network," in *2012 10th IAPR International Workshop on Document Analysis Systems*, 2012, pp. 125–129. DOI: 10.1109/DAS.2012.61

[36] B. M. Vinjit, M. K. Bhojak, S. Kumar, and G. Chalak, "A review on handwritten character recognition methods and techniques," in *2020 International Conference on Communication and Signal Processing (ICCSP)*, 2020, pp. 1224–1228. DOI: 10.1109/ICCSP48568.2020.9182129

[37] A. Mishra, A. S. Ram, and K. C, "Handwritten text recognition using convolutional neural network," *arXiv (Cornell University)*, Jul. 2023. DOI: 10.48550/arxiv.2307.05396 [Online]. Available: http://arxiv.org/abs/2307.05396

[38] A. Ansari, B. Kaur, M. Rakhra, A. Singh, and D. Singh, "Handwritten text recognition using deep learning algorithms," in *2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST)*, 2022, pp. 1–6. DOI: 10.1109/AIST55798.2022.10065348

[39] H. Mehta, S. Singla, and A. Mahajan, "Optical character recognition (ocr) system for roman script english language using artificial neural network (ann) classifier," in *2016 International Conference on Research Advances in Integrated Navigation Systems (RAINS)*, 2016, pp. 1–5. DOI: 10.1109/RAINS.2016.7764379

[40] S. Afroge, B. Ahmed, and F. Mahmud, "Optical character recognition using back propagation neural network," in *2016 2nd International Conference on Electrical, Computer Telecommunication Engineering (ICECTE)*, 2016, pp. 1–4. DOI: 10.1109/ICECTE.2016.7879615

[41] S. Katoch, M. Rakhra, and D. Singh, "Recognition of handwritten english character using convolutional neural network," in *2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST)*, 2022, pp. 1–6. DOI: 10.1109/AIST55798.2022.10064860

[42] A. Serdyuk et al., "Improving online handwriting trajectory reconstruction based on temporal convolutional networks," in *2024 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, 2024, pp. 1–8. DOI: 10.1109/SDF63218.2024.10773805

[43] BengaliAI and R. Ghosh, *Bangla-mnist*, 2021. DOI: 10.34740/KAGGLE/DSV/2029296 Accessed: [Online]. Available: https://www.kaggle.com/dsv/2029296

[44] H. Li, H. Li, L. Shi, et al., "Yolov8-sst lightweight small target fault detection system integrating senet attention mechanism and tr-ocr," *Cluster Computing*, vol. 28, p. 501, 2025. DOI: 10.1007/s10586-025-05182-7 [Online]. Available: https://doi.org/10.1007/s10586-025-05182-7

[45] P. Selvam et al., "A transformer-based framework for scene text recognition," *IEEE Access*, vol. 10, pp. 100 895–100 910, 2022. DOI: 10.1109/ACCESS.2022.3207469

[46] R. Doctor, A. Gutkin, C. Johny, B. Roark, and R. Sproat, "Graphemic normalization of the perso-arabic script," in *Grapholinguistics in the 21st Century, Part I*, ser. G21C 2022, vol. 9, Fluxus Editions, pp. 315–376. DOI: 10.36824/2022-graf-gutk [Online]. Available: http://dx.doi.org/10.36824/2022-graf-gutk

[47] JaidedAI, *GitHub - JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc.* Accessed: [Online]. Available: https://github.com/JaidedAI/EasyOCR

[48] M. Alam, M. J. I. Tutul, M. A. H. Wadud, M. J. Hossen, and M. F. Mridha, "Bilingual bangla ocr for rural empowerment: Detecting handwritten queries and agricultural assistance," *IEEE Open Journal of the Computer Society*, vol. 6, pp. 943–954, 2025. DOI: 10.1109/OJCS.2025.3573317

[49] M. P. Hossain, O. Ohidujjaman, M. S. Uddin, M. N. Huda, and T. Shimamura, "Recognition of bangla and english words in bengali texts using a modified bert-base-ner model," *Iraqi Journal for Computer Science and Mathematics*, vol. 6, no. 4, Article 5, 2025. DOI: 10.52866/2788-7421.1334 [Online]. Available: https://doi.org/10.52866/2788-7421.1334

[50] M. Dutta, A. Mohajon, S. Dev, and D. S. Bappi, "Text recognition of bangla and english scripts in natural scene images," Ph.D. dissertation, Feb. 2024. DOI: 10.13140/RG.2.2.27329.33124

[51] A. K. M. Shahariar Azad Rabby, M. M. Islam, N. Hasan, J. Nahar, and F. Rahman, "A novel deep learning character-level solution to detect language and printing style from a bilingual scanned document," in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 5218–5226. DOI: 10.1109/BigData50022.2020.9378262

[52] Y. Zhu, S. Li, H. Wang, and F. Wei, "FINet: Handwriting trajectory reconstruction of Chinese characters based on the font imitate network," *Pattern Recognition*, vol. 157, p. 110 949, Aug. 2024. DOI: 10.1016/j.patcog.2024.110949 [Online]. Available: https://doi.org/10.1016/j.patcog.2024.110949

[53] N. Khan et al., "Robust arabic and pashto text detection in camera-captured documents using deep learning techniques," *IEEE Access*, vol. 11, pp. 135 788–135 796, 2023. DOI: 10.1109/ACCESS.2023.3336404

[54] R. Chinthaginjala et al., "Enhancing handwritten text recognition accuracy with gated mechanisms," *Scientific Reports*, vol. 14, no. 1, p. 16 800, Jul. 2024. DOI: 10.1038/s41598-024-67738-8 [Online]. Available: https://doi.org/10.1038/s41598-024-67738-8

[55] S. M. M, N. S. Kumar, B. S. C. Reddy, N. V. S. K. Reddy, A. Govardhan, and H. S. Bontha, "Optical character recognition based answer script correction using deep learning and language processing algorithms," in *2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS)*, 2024, pp. 334–338. DOI: 10.1109/ICUIS64676.2024.10866861

[56] I. Rabaev, O. Biller, J. El-Sana, K. Kedem, and I. Dinstein, "Text line detection in corrupted and damaged historical manuscripts," in *2013 12th International Conference on Document Analysis and Recognition*, 2013, pp. 812–816. DOI: 10.1109/ICDAR.2013.166

[57] J. Michael, R. Labahn, T. Grüning, and J. Zöllner, "Evaluating sequence-to-sequence models for handwritten text recognition," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2019, pp. 1286–1293.

[58] T. Ayyavoo and J. John Suseela, "Illumination pre-processing method for face recognition using 2d dwt and clahe," *Iet Biometrics*, vol. 7, no. 4, pp. 380–390, 2018.