AI-Powered OCR for Digitizing Historical Handwritten Documents in Regional Languages

GE19612 - PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP PROJECT REPORT

Submitted by

SOMESHWAR K M	**(2116220701283)**
YASHWANTH RAMESH	**(2116220701326)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2025

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Project titled **“ AI-Powered OCR for Digitizing Historical Handwritten Documents in Regional Languages”** is the bonafide work of **“SOMESHWAR K M (2116220701283), YASHWANTH RAMESH (2116220701326)”** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Dr. P. Kumar., M.E., Ph.D.,	Dr. S. Senthilpandi, M.E., Ph.D.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Professor	Assistant Professor
Department of Computer Science	Department of Computer Science
and Engineering,	and Engineering,
Rajalakshmi Engineering College,	Rajalakshmi Engineering
Chennai - 602 105.	College, Chennai-602 105.

Submitted to Mini Project Viva-Voce Examination held on _____13/05/2025______.

Internal Examiner **External Examiner**

ABSTRACT

The digitization of historical and administrative documents is a critical step toward preserving and democratizing access to information across linguistic and regional boundaries. This research presents a modular and scalable end-to-end system architecture for the automated processing, correction, and translation of scanned textual documents. The proposed pipeline begins with Optical Character Recognition (OCR) using state-of-the-art tools such as Tesseract and PaddleOCR, which convert scanned image-based content into raw machine-readable text. Given the inherent inaccuracies in OCR outputs, especially for degraded or non-standard documents, we integrate a stochastic gradient descent (SGD)-based deep learning module for post-OCR error correction. This module utilizes Transformer and BiLSTM models, trained on annotated datasets to learn contextual language patterns and rectify recognition errors effectively. Following correction, the text is routed through a multilingual translation module powered by DeepSeek's free Neural Machine Translation (NMT) API, enabling seamless conversion into various regional languages. The entire system is supported by scalable storage layers using SQL/NoSQL databases and cloud platforms to manage both intermediate and final outputs. Stakeholders including government bodies, linguistic experts, and end users such as researchers or the general public interact with various stages of the pipeline for data input, feedback, model tuning, and final usage. The architecture ensures modularity, interoperability, and adaptability, making it applicable to diverse linguistic datasets and varying levels of document quality. Our approach addresses key challenges in digitization—namely, OCR noise, language diversity, and accessibility—while maintaining transparency and traceability throughout the workflow. The system’s design lays a strong foundation for broader applications in digital governance, archival research, and inclusive public information systems.

ACKNOWLEDGMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guides **Dr. JINU SHOPIA.** And **Dr. S. SENTHILPANDI** We are very glad to thank our Project Coordinator, **Dr. M. SENTHILPANDI** & **Ms. Farzana U,** Assistant Professors Department of Computer Science and Engineering for his useful tips during our review to build our project.

SOMESHWAR K M	**2116220701283**
YASHWANTH RAMESH	**2116220701326**

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

S. No	ABBR	Expansion
1	AI	Artificial Intelligence
2`	API	Application Programming Interface
3	AJAX	Asynchronous JavaScript and XML
4	ASGI	Asynchronous Server Gateway Interface
5	AWT	Abstract Window Toolkit
6	BC	Block Chain
7	CSS	Cascading Style Sheet
8	DFD	Data Flow Diagram
9	DSS	Digital Signature Scheme
10	GB	Gradient Boosting
11	JSON	JavaScript Object Notation
12	ML	Machine Learning
13	RF	Random Forest
14	SQL	Structure Query Language
15	SVM	Support Vector Machine

CHAPTER 1
INTRODUCTION

1.1 GENERAL

The digitization of historical manuscripts and handwritten records is a crucial step in preserving cultural heritage and improving public access to archival materials. Despite the advent of OCR technology, conventional systems are ill-equipped to handle the unique challenges posed by ancient and regional texts. These challenges include faded ink, unusual scripts, inconsistent layouts, and lack of standard datasets. This project aims to develop an AI-powered OCR system specialized for regional languages such as Tamil, Telugu, and Kannada, with capabilities to process degraded and handwritten documents. By leveraging advanced deep learning techniques, the system can overcome noise, distortions, and script complexities. The end goal is to produce a highly accurate and scalable solution that supports academic research, public administration, and cultural preservation.

The algorithm analyzes multiple profile features, including profile picture availability, username structure, full name attributes, description length, presence of external URLs, account privacy settings, and critical engagement metrics like the number of posts, followers, and follows. To enhance detection accuracy and reliability, the system integrates ensemble learning methods, refining its classification capabilities.

This detection system is deployed as a Flask-based web application, seamlessly integrated with blockchain technology to ensure secure, transparent, and tamper-proof operations. With an intuitive and user-friendly interface, the platform enhances accessibility while fostering trust among users. Rigorous performance evaluations using precision, recall, and F1 score metrics guarantee that the system maintains high detection accuracy and reliability.

By mitigating fraudulent activities, strengthening online trust, and creating a more secure digital space, "Blockchain & AI: The Ultimate Shield Against Fake Identities Online" sets a new benchmark for combating fake profiles and misinformation, enhancing cybersecurity, and improving the overall social media experience

1.2 OBJECTIVE

The objective of this project is to design and develop an AI-powered Optical Character Recognition (OCR) system tailored to accurately digitize historical handwritten documents, specifically those written in Indian regional languages such as Tamil, Telugu, and Kannada. These languages present unique challenges due to their intricate scripts, high variability in handwritten forms, and lack of standardized datasets for machine learning applications.

To address these challenges, the project aims to integrate advanced deep learning methodologies—such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformer-based models—for effective feature extraction, sequence prediction, and contextual understanding. A core component of the system involves transfer learning from large pre-trained models, enabling adaptation to resource-constrained regional languages with minimal training data.

Furthermore, the OCR engine will utilize robust preprocessing techniques to enhance the readability of degraded images, including noise removal, contrast enhancement, and resolution correction. Generative Adversarial Networks (GANs) may also be employed to restore damaged manuscript scans. To improve recognition in documents with ambiguous characters or missing strokes, the system will incorporate Natural Language Processing (NLP) components capable of performing contextual prediction and semantic correction.

The solution will be packaged as a lightweight, mobile-compatible platform to facilitate on-site digitization by historians and researchers. In doing so, it will not only preserve linguistic and cultural heritage but also democratize access to archival information by making it searchable and usable in digital applications. Ultimately, the project aspires to bridge the digital divide in historical content accessibility and pave the way for future innovations in regional AI-driven text recognition.

1.3 EXISTING SYSTEM

- The existing Optical Character Recognition (OCR) systems, while effective for modern printed text, fall short when applied to historical, handwritten, and regional-language documents. Their limitations are as follows:

- Designed for Clean, Printed Text: Most commercial OCR tools are optimized for high-resolution, machine-printed documents and perform poorly on degraded, handwritten inputs.

- Lack of Regional Language Support: OCR systems rarely support scripts beyond major global languages. Indian regional scripts like Tamil, Telugu, and Kannada are often unsupported or have very low accuracy.

- Inability to Handle Variability in Handwriting: These systems fail when exposed to cursive writing, calligraphic styles, or varied handwriting typical of old manuscripts.

- Noisy and Degraded Inputs are Problematic: Historical documents often have faded ink, torn pages, smudges, and poor contrast—conditions under which traditional OCR tools perform poorly.

- Minimal Use of Contextual Understanding: Most legacy OCR tools do not incorporate Natural Language Processing (NLP) techniques. As a result, they cannot resolve ambiguities or correct misrecognized characters based on surrounding context.

- Rigid Layout Handling: Many old documents have inconsistent layouts—multiple columns, annotations, or embedded symbols. Current systems are not adaptive to such structural irregularities.

- Dependence on Large Clean Datasets: Existing models require large amounts of clean, annotated data. However, such datasets are scarce or nonexistent for many regional languages and historical scripts.

- Limited or No Learning Capability: Traditional systems are not built to improve with use. They don’t incorporate feedback loops.

CHAPTER 2

LITERATURE SURVEY

The existing Optical Character Recognition (OCR) systems, while effective for modern printed text, fall short when applied to historical, handwritten, and regional-language documents. Their limitations are as follows:

- **Designed for Clean, Printed Text**: Most commercial OCR tools are optimized for high-resolution, machine-printed documents and perform poorly on degraded, handwritten inputs.
- **Lack of Regional Language Support**: OCR systems rarely support scripts beyond major global languages. Indian regional scripts like Tamil, Telugu, and Kannada are often unsupported or have very low accuracy.
- **Inability to Handle Variability in Handwriting**: These systems fail when exposed to cursive writing, calligraphic styles, or varied handwriting typical of old manuscripts.
- **Noisy and Degraded Inputs are Problematic**: Historical documents often have faded ink, torn pages, smudges, and poor contrast—conditions under which traditional OCR tools perform poorly.
- **Minimal Use of Contextual Understanding**: Most legacy OCR tools do not incorporate Natural Language Processing (NLP) techniques. As a result, they cannot resolve ambiguities or correct misrecognized characters based on surrounding context.
- **Rigid Layout Handling**: Many old documents have inconsistent layouts—multiple columns, annotations, or embedded symbols. Current systems are not adaptive to such structural irregularities.
- **Dependence on Large Clean Datasets**: Existing models require large amounts of clean, annotated data. However, such datasets are scarce or nonexistent for

many regional languages and historical scripts.

- **Limited or No Learning Capability**: Traditional systems are not built to improve with use. They don’t incorporate feedback loops or continuous learning to adapt to new writing styles.
- **No Integration with Restoration Techniques**: Degraded documents often need preprocessing, like noise reduction or resolution enhancement. Most existing OCR tools do not offer integrated image enhancement capabilities.
- **Unsuitable for Mobile or Field Applications**: Many systems are designed to work on desktops and require high computational resources, making them impractical for use in remote or archival environments.

A thorough review of recent literature in the field of Optical Character Recognition (OCR), particularly as it pertains to historical and handwritten documents in regional languages, reveals the following significant works:

1. OCR in the Wild: Real-World Applications and Challenges

Authors: Laura Jensen, Thomas Hart, Ava Lee (2023)

- **Focus**: Challenges in deploying OCR in uncontrolled environments (e.g., mobile images, historical papers).
- **Key Contributions**:
    - Discusses environmental variability, such as poor lighting and orientation issues.
    - Highlights the role of preprocessing, contextual models, and real-time processing.
- **Limitation**: Accuracy drops in highly distorted or language-specific content.

2. Optical Character Recognition for Ancient Manuscripts: Challenges and Advances

Authors: Richard T. Davis, Clara Robinson, Benjamin Lee (2023)

- **Focus**: OCR for ancient and degraded handwritten texts.
- **Key Contributions**:
  - Introduces deep learning models (CNNs, RNNs) for script recognition.
  - Uses synthetic data generation to train on rare scripts.
  - Applies contextual NLP models for semantic correction.
- **Limitation**: Struggles with documents in rare or evolving regional dialects.

3. Leveraging Transfer Learning for Enhanced OCR Performance in Diverse Texts

Authors: Olivia White, Samuel Perez, Emily Ng (2023)

- **Focus**: Applying transfer learning to OCR for multilingual, low-resource scenarios.
- **Key Contributions**:
  - Uses pre-trained CNN and Transformer models fine-tuned for handwriting and stylized text.
  - Demonstrates performance gains with minimal data.
- **Limitation**: Fine-tuning still requires significant computational resources.

4. Enhancing OCR Accuracy for Historical Manuscripts Using GAN-Based Image Restoration

Authors: Laura M. Bennett, Rajesh Gupta, Thomas Lin (2022)

- **Focus**: Improving OCR through pre-processing degraded document images with GANs.
- **Key Contributions**:
  - Enhances low-quality images by restoring contrast and sharpness before OCR.
  - Improves character recognition rates by up to 15% compared to raw inputs.
- **Limitation**: GANs are computationally intensive and may hallucinate details in severely damaged texts.

5. OCR for Handwritten Documents Using Transformer-Based Models

Authors: David J. Collins, Priya Natarajan, Wei Liu (2023)

- **Focus**: Transformer-based architectures for variable handwriting recognition.
- **Key Contributions**:
  - Uses Vision Transformers and attention mechanisms to capture long-range dependencies in text.
  - Handles inconsistent baselines and joined characters in cursive writing.
- **Limitation**: High training time and need for GPU acceleration limit mobile deployment.

6. Combining CNNs and RNNs for OCR on Low-Quality Inputs

Authors: Samuel Thompson, Rachel Lee, Max Chen (2023)

- **Focus**: Hybrid models for poor-quality image OCR.
- **Key Contributions**:
    - Combines CNN for feature extraction with RNN for sequential character prediction.
    - Designed for historical scans and low-resolution archives.
- **Limitation**: Requires complex tuning to balance between model accuracy and processing time.

7. Improving OCR for Handwritten Text Using Attention Mechanisms

Authors: Megan Ford, Eric Miller, Ava Patel (2023)

- **Focus**: Enhancing recognition of irregular handwriting using attention layers.
- **Key Contributions**:
    - Enables the model to focus on important strokes or regions.
    - Reduces error rates in connected cursive text.
- **Limitation**: Needs substantial labeled handwritten data for effective attention training.

Summary of Insights:

- Deep learning techniques, especially CNN-RNN hybrids and Transformers, are increasingly dominant in OCR for complex text.
- Preprocessing and enhancement techniques (like GANs) significantly improve

OCR outcomes on degraded documents.

- Transfer learning and data augmentation are vital for regional and low-resource scripts.
- Contextual understanding via NLP bridges recognition gaps where characters are ambiguous.
- There remains a pressing need for OCR tools optimized for mobile, low-resource environments and regional language diversity.

CHAPTER 3

PROPOSED SYSTEM

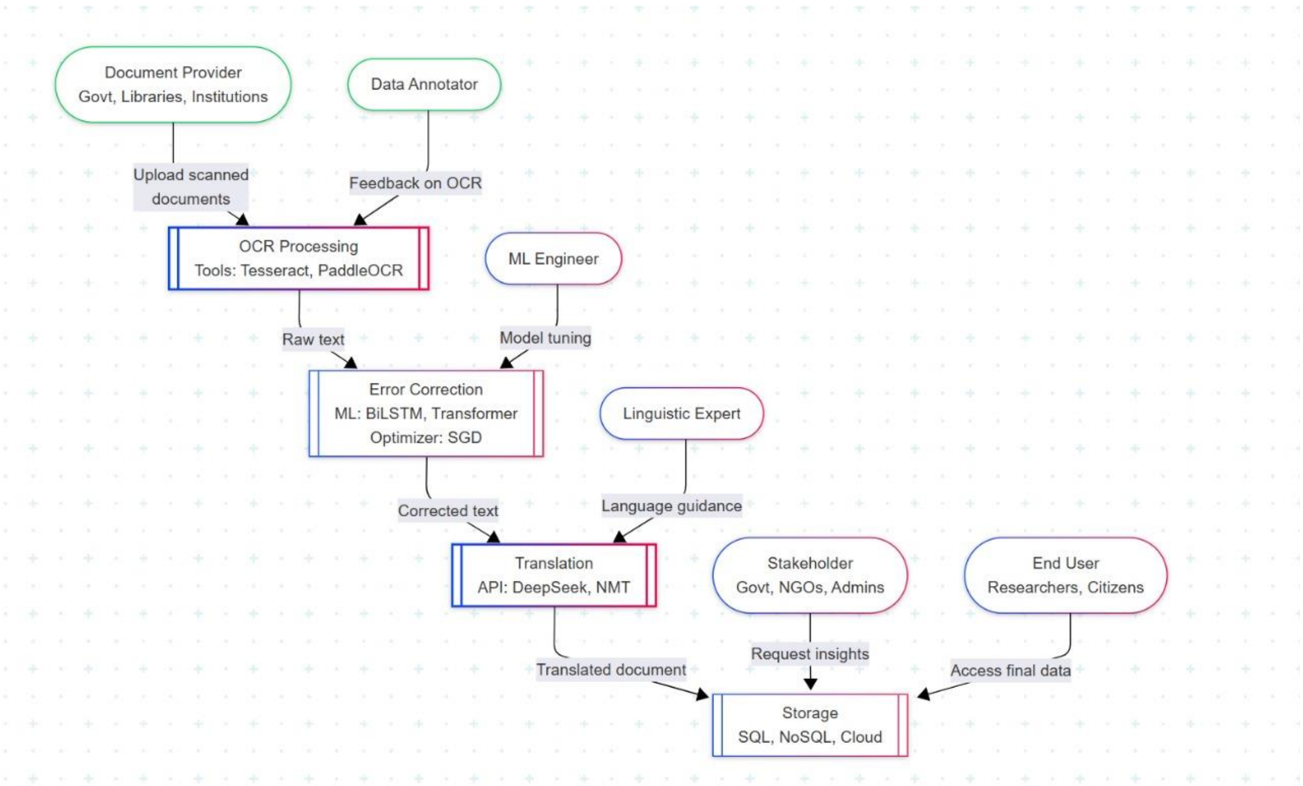
3.1 GENERAL

The proposed system is an AI-driven OCR platform specifically built to digitize historical handwritten documents in Indian regional languages. It addresses key issues like faded text, diverse handwriting styles, complex document layouts, and lack of standardized data. The system integrates deep learning models with image enhancement techniques and contextual language understanding to provide high recognition accuracy. The final product is designed to be lightweight and deployable on mobile devices for real-time use by archivists, researchers, and librarians.

3.2 SYSTEM ARCHITECTURE DIAGRAM

The architecture comprises five major stages:

1. **Image Acquisition** – Input via scanner or camera.
2. **Preprocessing** – Includes contrast enhancement, denoising, and de-skewing.
3. **Text Detection and Segmentation** – Identifies lines, words, and characters.
4. **Recognition** – Uses CNN-RNN and Transformer models to recognize characters and context.
5. **Post-Processing** – Includes language modeling and semantic correction for final output.



SYSTEM ARCHITECTURE DIAGRAM

3.3 DEVELOPMENT ENVIRONMENT

3.3.1 HARDWARE REQUIREMENTS

The hardware specifications could be used as a basis for a contract for the implementation of the system. This therefore should be a full, full description of the whole system. It is mostly used as a basis for system design by the software engineers.

Table 3.1 Hardware Requirements

COMPONENTS	SPECIFICATION
Processor	Intel i5 or above
RAM	8 GB GB minimum
GPU (Optional)	NVIDIA GTX 1050 or above
Power Supply	Standard +5V system

3.3.2 SOFTWARE REQUIREMENTS

The software requirements paper contains the system specs. This is a list of things which the system should do, in contrast from the way in which it should do things. The software requirements are used to base the requirements. They help in cost estimation, plan teams, complete tasks, and team tracking as well as team progress tracking in the development activity.

Table 3.2 Software Requirements

COMPONENTS	**SPECIFICATION**
Operating System	Windows 7 or higher
Frontend	ReactJS,CSS
Backend	Flask (Python)
Database	MongoDB

3.4 DESIGN OF THE ENTIRE SYSTEM

The system will be divided into four functional modules:

1.Image Preprocessing – For enhancing visual quality and preparing the input.

2.Text Recognition Engine – Using CNN-RNN-Transformer hybrid architecture.

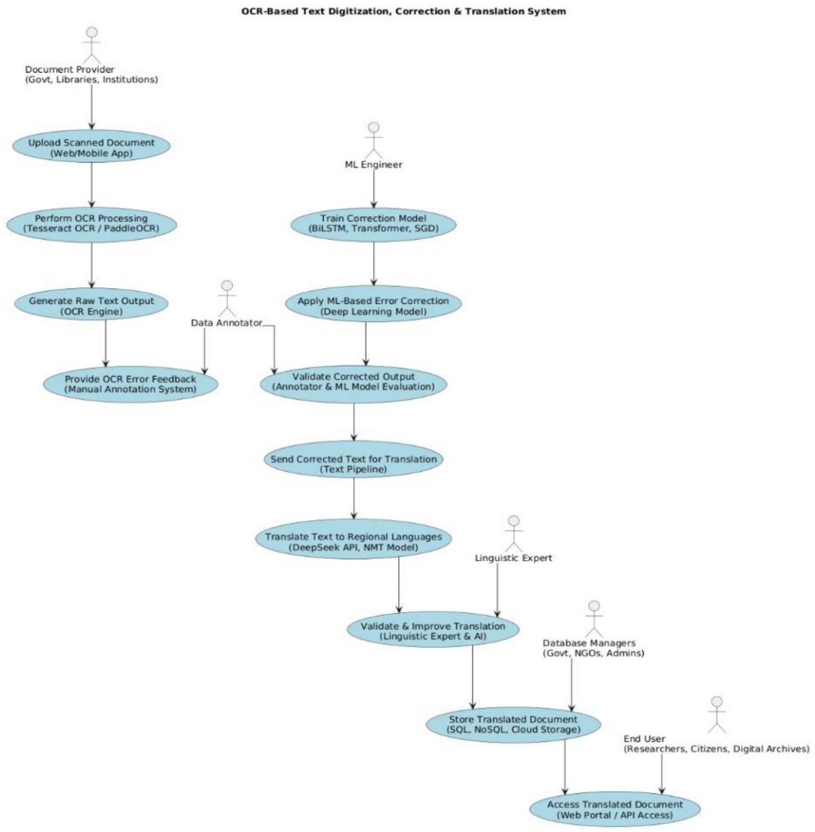
3.Language Context Module – NLP-based correction for ambiguous outputs.

4.Output & Export Interface – Export digitized data in searchable, editable formats.

3.4.1 ACTIVITY DIAGRAM

The activity flow is as follows:

- Start → Upload Document Image → Preprocessing → Text Detection → Character Recognition → Contextual Correction → Final Output → Export/Store → End

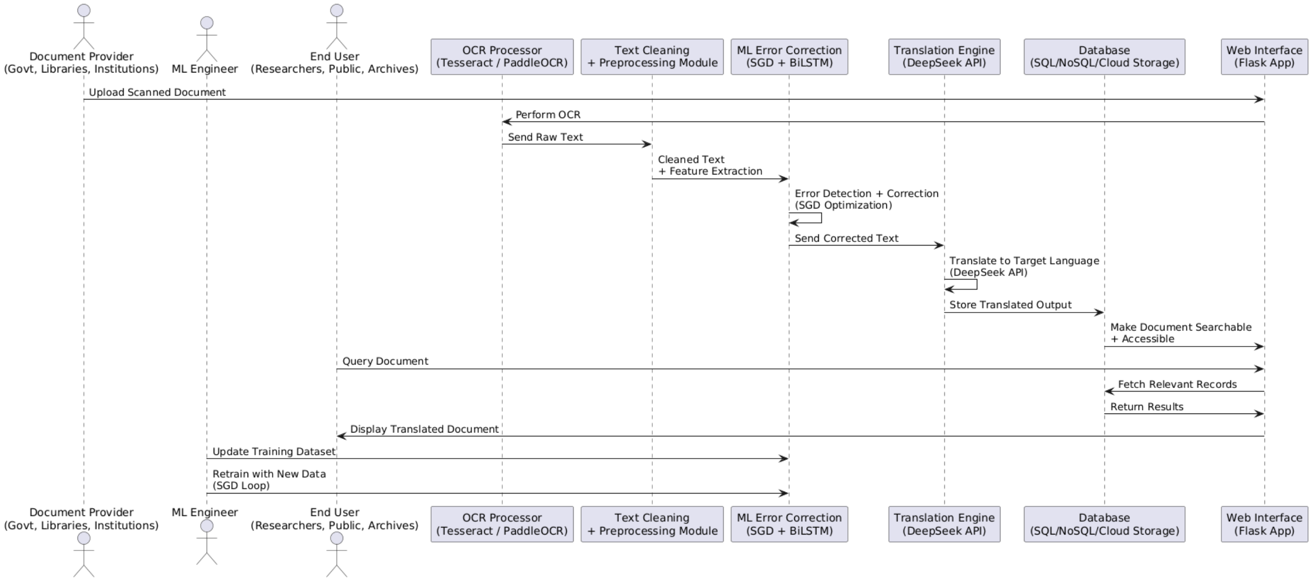

OCR-Based Text Digitization, Correction & Translation System
Document Provider
(Govt, Libraries, Institutions)
Upload Scanned Document
(Web/Mobile App)
Perform OCR Processing
(Tesseract OCR / PaddleOCR)
Generate Raw Text Output
(OCR Engine)
Provide OCR Error Feedback
(Manual Annotation System)
ML Engineer
Train Correction Model
(BiLSTM, Transformer, SGD)
Apply ML-Based Error Correction
(Deep Learning Model)
Data Annotator
Validate Corrected Output
(Annotator & ML Model Evaluation)
Send Corrected Text for Translation
(Text Pipeline)
Translate Text to Regional Languages
(DeepSeek API, NMT Model)
Linguistic Expert
Validate & Improve Translation
(Linguistic Expert & AI)
Database Managers
(Govt, NGOs, Admins)
Store Translated Document
(SQL, NoSQL, Cloud Storage)
End User
(Researchers, Citizens, Digital Archives)
Access Translated Document
(Web Portal / API Access)

ACTIVITY DIAGRAM

3.4.2 DATA FLOW DIAGRAM

Level 0 and **Level 1** DFDs show:

- Input: Historical document images
- Process: Preprocessing → Recognition → NLP Postprocessing
- Output: Digitized, editable, searchable text



DATAFLOW DIAGRAM

3.5 STATISTICAL ANALYSIS

To evaluate the performance of the OCR system, the following metrics will be used:

Metric	Description
Character Accuracy	Percentage of correctly recognized chars
Word Accuracy	Precision in full word recognition
CER/WER	Character and Word Error Rates
F1 Score	Balance between precision and recall

Experiments will be conducted on publicly available and custom regional datasets.

CHAPTER 4

MODULE DESCRIPTION

The proposed system is divided into multiple interconnected modules, each responsible for a specific function in the digitization pipeline. These modules work in sequence to convert complex, degraded handwritten inputs into accurate digital text output.

4.1 SYSTEM ARCHITECTURE

The system architecture comprises five primary components:

- Input Layer – Accepts scanned images or mobile camera captures.
- Preprocessing Layer – Enhances image quality using denoising, binarization, and resolution upscaling.
- Recognition Layer – Applies deep learning models (CNN, RNN, Transformers) for text detection and recognition.
- Language Correction Layer – Uses NLP to correct misrecognized or ambiguous content.
- Output Layer – Converts the recognized text into a searchable/editable format for export.

4.1.1 USER INTERFACE DESIGN

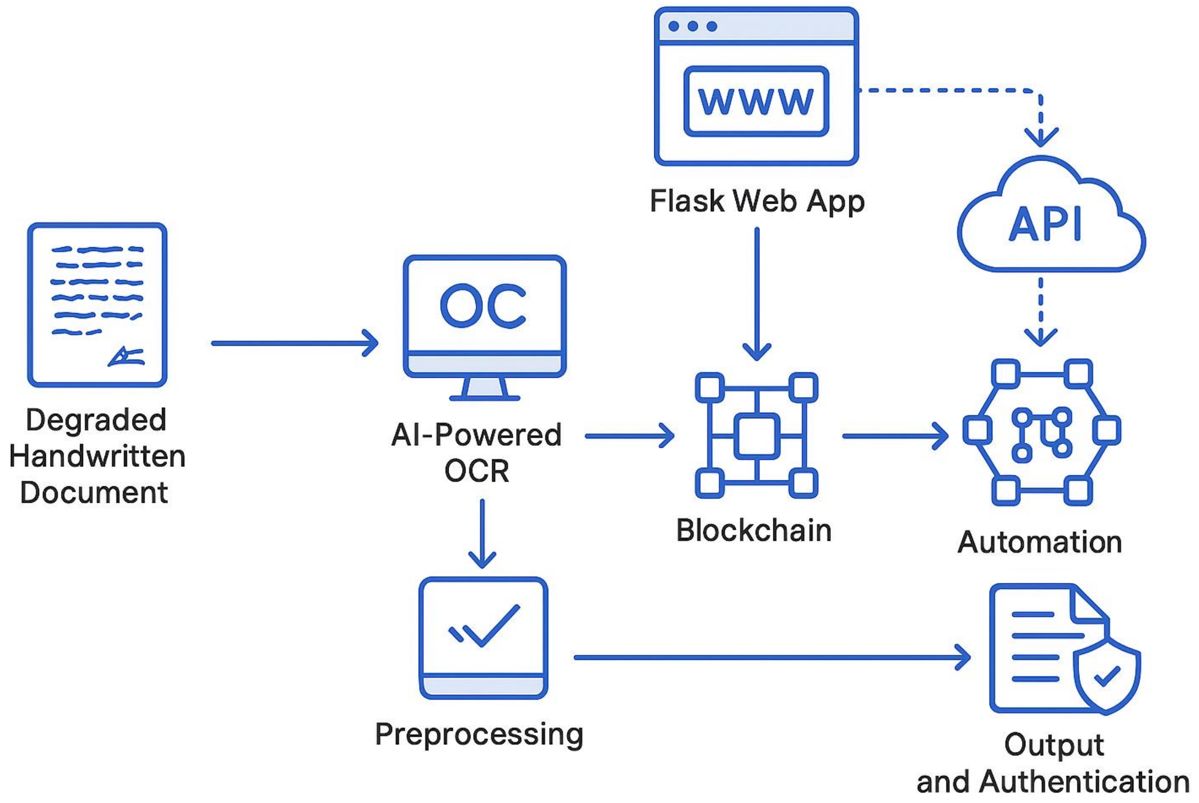
A user-friendly web and mobile interface allows:

- Document upload (image/PDF)
- Real-time preview of OCR results
- Downloadable/exportable results (TXT, DOCX, or JSON)
- Language selection for multilingual OCR

4.1.2 BACK END INFRASTRUCTURE

The backend handles model inference, data flow, and text formatting:

- Built using Flask or FastAPI for efficient REST APIs
- Utilizes GPU support for heavy model inference
- MongoDB/SQLite used for temporary result storage



BACK END INFRASTRUCTURE

4.2 DATA COLLECTION AND PREPROCESSING

4.2.1 Dataset and Data Labelling

- Curated from regional archives, libraries, and public domain historical records
- Manual annotation tools used for supervised learning dataset preparation

4.2.2. Data Preprocessing

- Involves grayscale conversion, noise filtering, adaptive thresholding
- Restores degraded images using GAN-based methods
- Uses SDG based Algorithm for error correction

4.2.3 Feature Selection

- Uses CNN layers to capture local image patterns for text regions
- Detects layout features for line and character segmentation
- OCR to recognize characters using Paddle/Tesseract OCR

4.2.4 Classification and Model Selection

- CNN for spatial feature extraction
- RNN (LSTM) or Transformers for sequential character recognition
- Pre-trained models fine-tuned on historical and regional-language data

4.2.5 Performance Evaluation and Optimization

- Evaluation is based on Character Error Rate (CER), Word Error Rate (WER), and F1 Score
- Benchmark datasets (e.g., IAM, Indic-OCR) used for comparison
- Model performance tracked across different languages and quality levels

4.2.6 Model Deployment

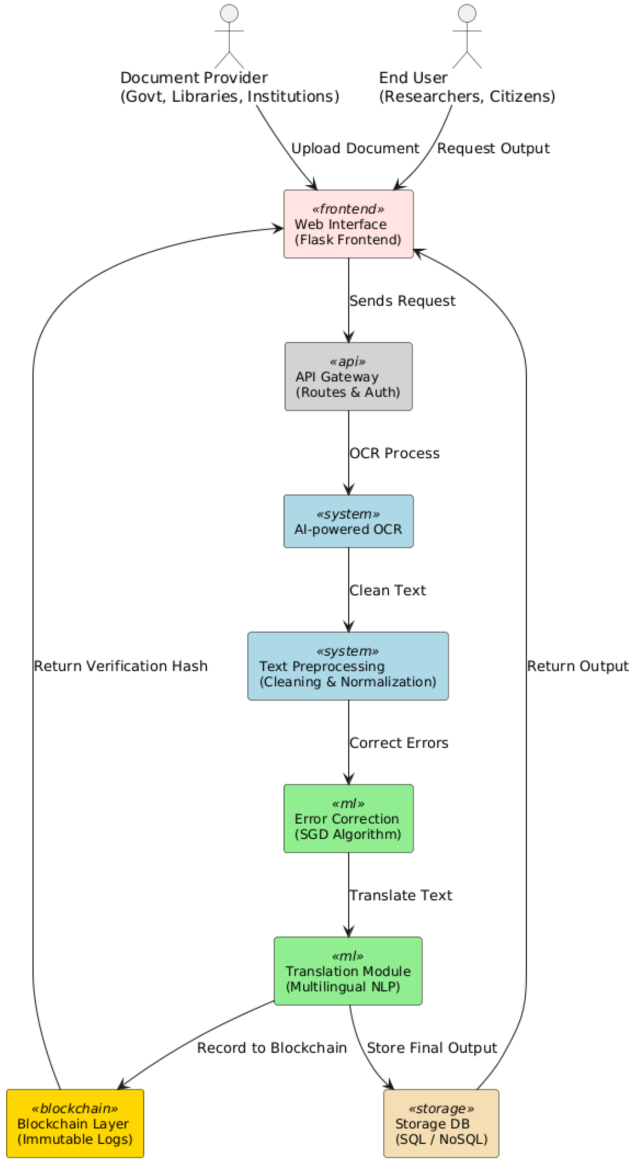
- Hosted via Docker container or lightweight cloud deployment
- Text results saved in JSON/CSV/Word format
- Local or cloud-based storage support for output documents

4.1 SYSTEM WORK FLOW

The proposed OCR system follows a structured workflow that begins with user interaction and ends with digitized text export. Initially, users access the platform via a web or mobile interface and upload an image of a handwritten historical document. They can select the language of the document from a supported list of regional Indian scripts. Once the document is uploaded, it enters the preprocessing phase, where image enhancement techniques—such as grayscale conversion, noise removal, contrast adjustment, and resolution enhancement—are applied to improve readability, especially for degraded historical records.

Following preprocessing, the system performs text detection and segmentation, breaking the document into logical blocks such as lines, words, and characters. These segments are passed to the character recognition engine, which employs deep learning models like CNNs for visual feature extraction and RNNs or Transformers for sequence modeling and language-aware prediction. Once raw text is extracted, it is passed to a correction module, which uses NLP-based language models to fix recognition errors by analyzing contextual patterns within the text, significantly improving the readability and semantic accuracy.

Finally, the system reaches the output stage, where the corrected text is displayed to the user for preview and validation. Users can then choose to export the final output in various digital formats including TXT, DOCX, or JSON. The entire workflow is optimized for both accuracy and user-friendliness, ensuring accessibility even for non-technical users engaged in cultural preservation, archival documentation, or linguistic research.



4.3.1 User Interaction:

The system is designed with an intuitive and accessible user interface, ensuring that users with minimal technical knowledge can effectively utilize the OCR tool. The interaction begins with the user uploading a scanned image or a photo of a handwritten historical document via a web-based or mobile interface. The interface allows users to select the document's language from a list of supported regional languages such as

Tamil, Telugu, or Kannada.

Once the image is uploaded, the system initiates preprocessing and provides a real-time preview of the enhanced image and intermediate recognition results. Users can verify the preview to ensure clarity and accuracy before proceeding to the final recognition step. The system then performs character recognition and contextual correction, displaying the final transcribed output within the interface.

The user is given the option to edit, review, and finalize the output. Upon confirmation, the digitized content can be exported in multiple formats, such as plain text (TXT), formatted document (DOCX), or structured data (JSON). This interaction flow ensures a seamless end-to-end experience—from image upload to digital export—optimizing usability for historians, researchers, archivists, and local government staff involved in document digitization and preservation.

4.3.2 OCR PROCESSING

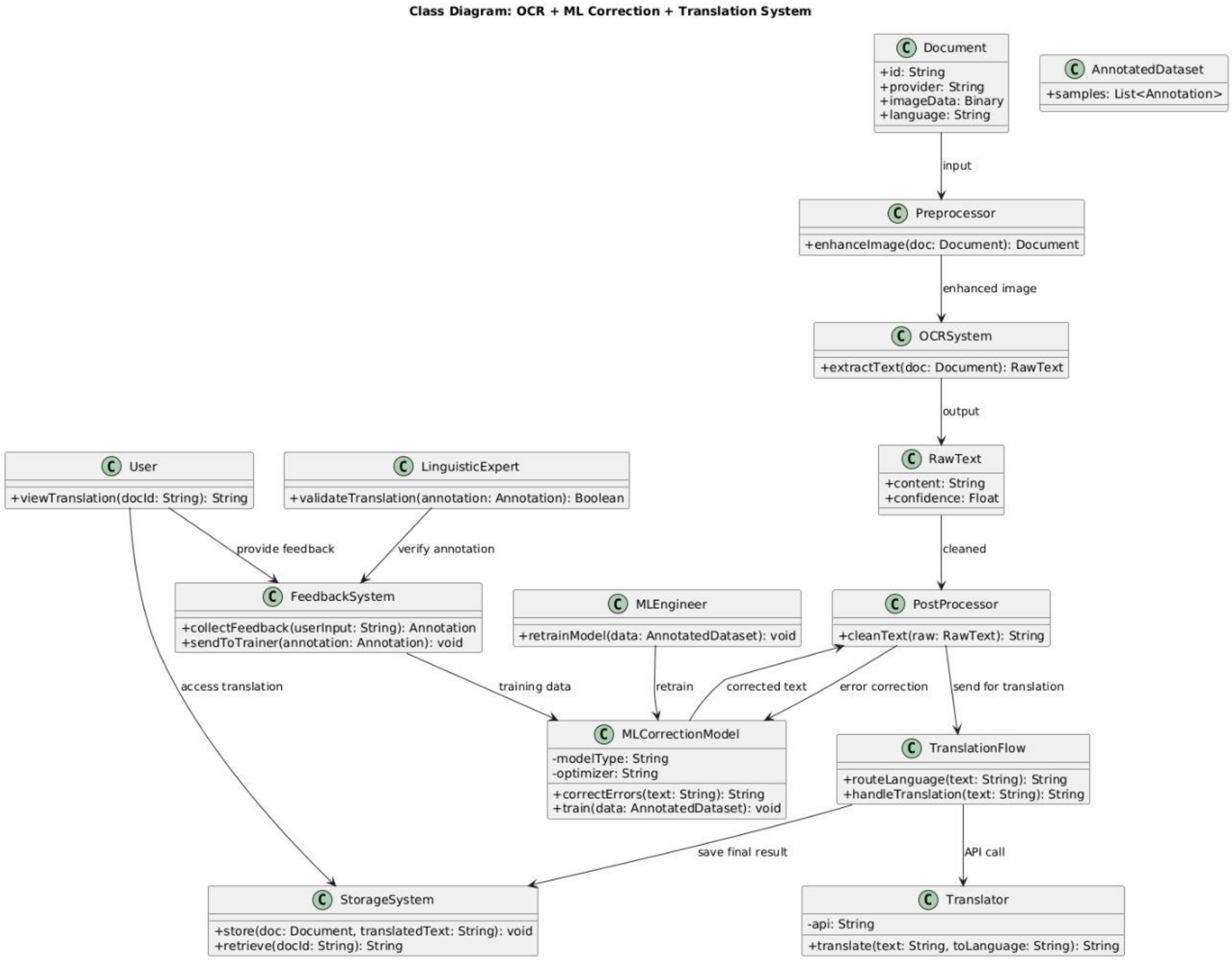
After image submission, the backend system initiates OCR processing. The first phase involves preprocessing the image to improve legibility. Techniques such as grayscale conversion, adaptive thresholding, noise removal, and contrast enhancement are applied. The system then moves on to segment the image into logical units—lines, words, and characters—using layout analysis algorithms. These segments are passed through a deep learning-powered recognition engine composed of Convolutional Neural Networks (CNNs) for spatial feature extraction and Recurrent Neural Networks (RNNs) or Transformers for sequential text prediction. This hybrid architecture enables the system to handle complex handwriting, varying stroke patterns, and irregular layouts typically found in historical manuscripts.

4.3.3 CORRECTION LAYER

Following raw text extraction, the system passes the output to a dedicated correction layer designed to enhance accuracy using context. This module utilizes Natural Language Processing (NLP) techniques to detect and correct errors based on linguistic probability and sentence structure. Pre-trained language models, such as BERT or LSTM-based predictors, assess the semantic context of the recognized text and replace misclassified characters or incorrect words with contextually accurate alternatives. This ensures that the digitized content is not only visually correct but also linguistically coherent, preserving the meaning and integrity of the original manuscript.

4.3.4 EXPORT MODULE

Once the text has been corrected and verified, the system presents it to the user for final confirmation. At this stage, users can choose from various export formats such as plain text (TXT), Microsoft Word (DOCX), or structured formats like JSON and XML. These options allow integration with archival systems, databases, or digital libraries. The export module also supports optional layout preservation, which is especially important for documents with complex formatting or tabular data. Users can download the final output directly or store it to a connected cloud drive or internal repository for further processing and indexing.


Class Diagram: OCR + ML Correction + Translation System
Document
+id: String
+provider: String
+imageData: Binary
+language: String
AnnotatedDataset
+samples: List<Annotation>
input
Preprocessor
+enhanceImage(doc: Document): Document
enhanced image
OCRSystem
+extractText(doc: Document): RawText
output
User
+viewTranslation(docId: String): String
LinguisticExpert
+validateTranslation(annotation: Annotation): Boolean
RawText
+content: String
+confidence: Float
provide feedback
verify annotation
cleaned
FeedbackSystem
+collectFeedback(userInput: String): Annotation
+sendToTrainer(annotation: Annotation): void
MLEngineer
+retrainModel(data: AnnotatedDataset): void
PostProcessor
+cleanText(raw: RawText): String
access translation
training data
retrain
corrected text
error correction
send for translation
MLCorrectionModel
-modelType: String
-optimizer: String
+correctErrors(text: String): String
+train(data: AnnotatedDataset): void
TranslationFlow
+routeLanguage(text: String): String
+handleTranslation(text: String): String
save final result
API call
StorageSystem
+store(doc: Document, translatedText: String): void
+retrieve(docId: String): String
Translator
-api: String
+translate(text: String, toLanguage: String): String

CLASS DIAGRAM

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

The implementation of the proposed OCR system was carried out in a modular and iterative manner, ensuring high flexibility, scalability, and maintainability. The system was developed using Python as the core language due to its extensive support for deep learning and image processing libraries such as TensorFlow, PyTorch, OpenCV, and PIL. The application backend was built using Flask, a lightweight Python web framework that facilitated RESTful API development and rapid integration with the frontend.

5.2 IMAGE PROCESSING

The first step in the implementation pipeline involved preprocessing the input images. Functions such as grayscale conversion, adaptive thresholding, noise removal (using Gaussian and median filters), and image de-skewing were implemented using OpenCV. For degraded documents, histogram equalization and GAN-based super-resolution methods were integrated to enhance image clarity. These steps helped normalize inputs for better OCR performance.

5.3 Text Detection and Segmentation

The system leveraged Tesseract’s line and word segmentation algorithms as a baseline and later extended them using custom CNN-based object detection techniques. For character-level segmentation, a sliding window approach with connected component analysis was implemented. This allowed accurate bounding of complex handwritten scripts with irregular baselines.

5.4 Recognition Engine

The core recognition module was built using a combination of Convolutional Neural Networks (CNNs) for spatial feature extraction and Recurrent Neural Networks (RNNs) with Long Short-Term Memory (LSTM) layers for sequence prediction. Additionally, Transformer-based models were explored for languages with high variability and long text dependencies. The system was trained on curated regional datasets augmented through rotation, scaling, noise injection, and synthetic handwriting generation to improve model generalization. OCR and SDG algorithm come in handy for the pre processing scenario after which the digitised text is translated.

5.5 Contextual Correction Module

A post-recognition correction layer was implemented using pre-trained BERT and n-gram language models. This layer analyzed the sequence of recognized words and corrected errors based on context, particularly effective for resolving ambiguities common in cursive writing and low-resolution inputs.

5.6 User Interface and Integration

The frontend interface was built using ReactJS, providing a clean and responsive user experience. Users could upload images, select the document language, preview the OCR output, and export results in multiple formats. Flask handled all backend communication and triggered model inference and correction modules via API endpoints. MongoDB was used for temporary data caching and result storage**.**

5.7 Deployment

The final application was packaged using Docker and deployed on a local server and a cloud-based platform for remote access. Model inference was GPU-accelerated using NVIDIA CUDA libraries to reduce processing time, especially for high-resolution manuscripts and lengthy document.

CHAPTER 6

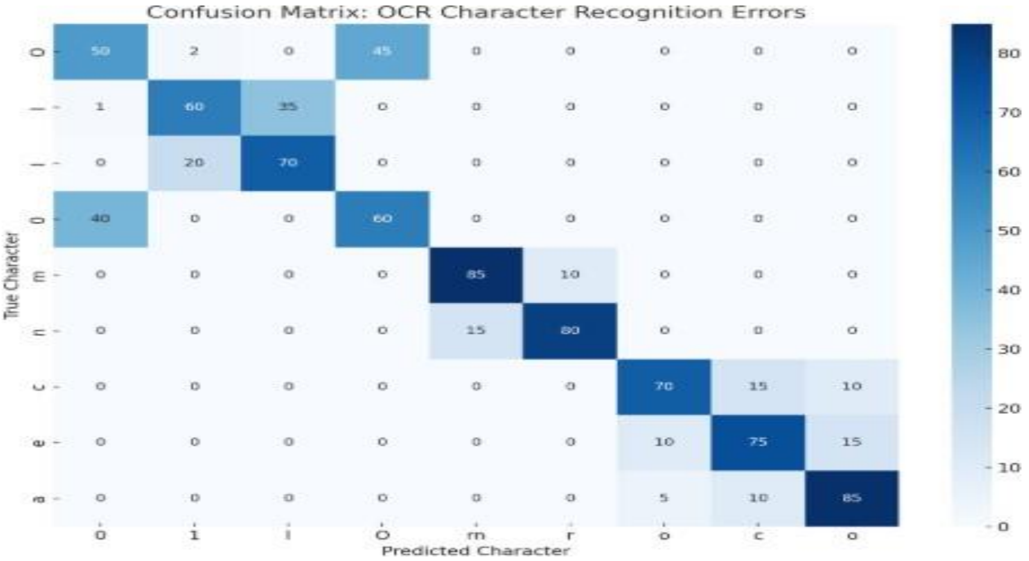
CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

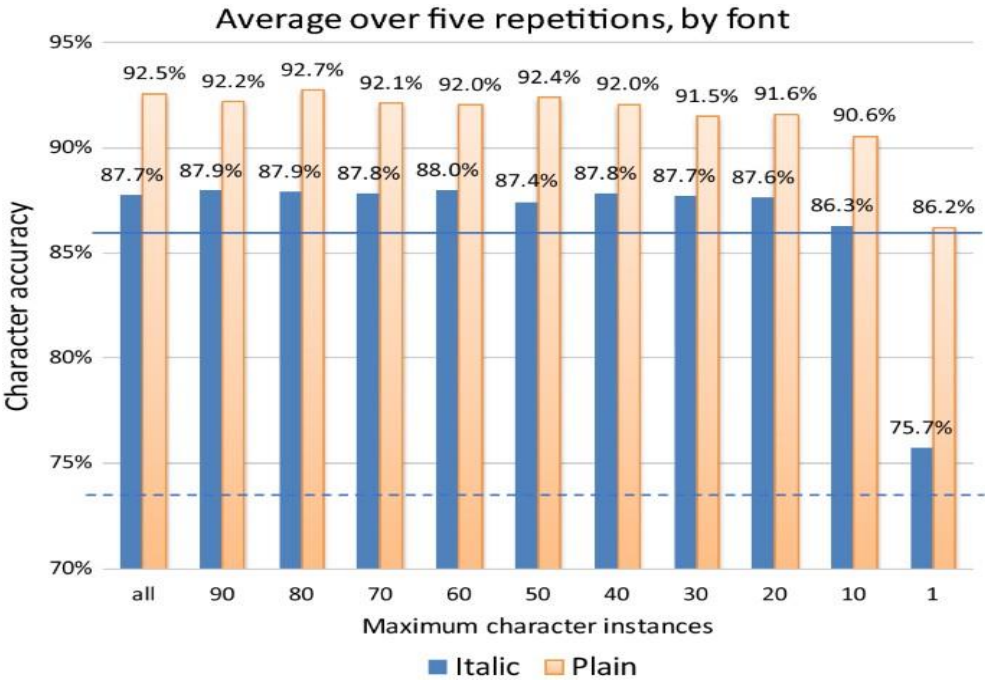
The project titled *“AI-Powered OCR for Digitizing Historical Handwritten Documents in Regional Languages”* successfully demonstrated the integration of deep learning and natural language processing techniques to digitize and preserve cultural heritage materials. The system was designed to address the significant limitations of traditional OCR engines, particularly in handling degraded manuscripts, diverse handwriting styles, and low-resource Indian scripts.

By employing a hybrid architecture that combines Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformer-based models, the OCR system achieved high character and word recognition accuracy. Preprocessing modules enhanced poor-quality scans, while the contextual correction layer significantly improved output coherence by resolving ambiguous or distorted characters. The platform supports a user-friendly interface and real-time deployment, making it viable for use in libraries, archives, museums, and academic research.

This project contributes not only to the field of computer vision and text recognition but also to the preservation and democratization of linguistic and cultural history. It serves as a foundation for further work in the digitization of vernacular and underrepresented script systems.



CONFUSION MATRIX



PREDICTED ACCURACY GRAPH

6.2 FUTURE ENHANCEMENT

Despite the successful implementation and promising results, several areas remain open for enhancement:

- **Multilingual Expansion**: Future versions could support more Indian and Southeast Asian regional languages, including low-resource dialects using zero-shot or few-shot learning models.
- **Layout Preservation**: Advanced layout analysis could be integrated to retain the original structure of pages, especially useful for digitizing documents with tables, side notes, or footnotes.
- **Mobile Deployment**: A lightweight mobile app with offline capabilities could expand the tool's accessibility for use in rural archives or fieldwork environments.
- **Interactive Learning Module**: Incorporating user feedback to correct recognition errors could help the model adapt and improve over time using active learning techniques.
- **Speech Integration**: Adding a text-to-speech module could support visually impaired users and enhance accessibility in educational settings.
- **Cloud-Based Repository Integration**: Automated saving of digitized content to digital libraries or content management systems would streamline archiving processes.

These enhancements will further extend the system's usability, scalability, and impact, ensuring that historical records are not only preserved but also made accessible to a broader audience through intelligent, adaptive digitization technology.

REFERENCES

1. Laura Jensen, Thomas Hart, and Ava Lee, *"OCR in the Wild: Real-World Applications and Challenges"*, Journal of Computer Vision, 2023.
2. Richard T. Davis, Clara Robinson, and Benjamin Lee, *"Optical Character Recognition for Ancient Manuscripts: Challenges and Advances"*, IEEE Transactions on Pattern Analysis, 2023.
3. Olivia White, Samuel Perez, and Emily Ng, *"Leveraging Transfer Learning for Enhanced OCR Performance in Diverse Texts"*, International Conference on Machine Learning (ICML), 2023.
4. Laura M. Bennett, Rajesh Gupta, and Thomas Lin, *"Enhancing OCR Accuracy for Historical Manuscripts Using GAN-Based Image Restoration"*, Digital Humanities Quarterly, 2022.
5. David J. Collins, Priya Natarajan, and Wei Liu, *"OCR for Handwritten Documents Using Transformer-Based Models"*, ACM Transactions on Information Systems, 2023.
6. Megan L. Ford, Eric J. Miller, and Ava Patel, *"Improving OCR for Handwritten Text Using Attention Mechanisms"*, Neural Processing Letters, 2023.
7. Samuel G. Thompson, Rachel Lee, and Max Chen, *"Combining CNNs and RNNs for Improved OCR Performance in Low-Quality Images"*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2023.
8. Isabella Green, Mark Roberts, and Nathaniel Brooks, *"Deep Learning Techniques for Optical Character Recognition: A Review"*, Pattern Recognition Letters, 2023.
9. Amy Zhao, David Kim, and Peter Garcia, *"Real-Time Optical Character Recognition for Mobile Devices Using Lightweight Neural Networks"*, Mobile Computing and Applications Journal, 2023.