Sinhala Handwritten Text Recognition and Translation System

- This report details the development and analysis of a system designed to process images containing handwritten Sinhala text, especially for handwritten text. This project addresses this gap by leveraging the Tesseract OCR engine, an open-source tool that supports the training of custom models.
- The goal of this project is to create a robust OCR system capable of accurately recognizing and converting handwritten Sinhala characters into digital text. This involves several key steps, including image preprocessing, generating box files, training the OCR model, and fine-tuning the system to enhance accuracy.

2. Methodology

2.1 System Architecture

The system architecture of the Sinhala OCR project is composed of several interconnected components designed to work together seamlessly. The architecture can be broken down into the following modules:

- Image Input: The system begins by receiving images containing handwritten Sinhala text. These images are sourced from a dataset comprising 1170 handwritten Sinhala characters.
- Image Preprocessing: The raw images undergo preprocessing to enhance their quality and prepare them for OCR processing. This step is crucial for improving the accuracy of character recognition.
- Optical Character Recognition (OCR): Using the Tesseract OCR engine, the system processes the preprocessed images to extract textual data.

A custom-trained model, specifically designed for Sinhala characters, is used for this purpose.

- Text Translation: Once the text is extracted, it is passed through a translation module to convert the Sinhala text into English. This is done using the Google Translate API.
- Output: The final output is the translated text, which can be stored in a text file or displayed on the screen.

2.2 Image Preprocessing

Image preprocessing is a critical step in enhancing the quality of images to ensure accurate text recognition. The preprocessing steps used in this project include:

- Grayscale Conversion: Converting the image to grayscale reduces complexity by eliminating color information, focusing solely on the intensity of pixels.
- Noise Removal: Techniques like Gaussian blurring or median filtering are applied to remove noise that may interfere with text recognition.
- Thresholding: Binarization is performed using methods like Otsu's thresholding, converting the image into a binary format (black and white), which simplifies the OCR process.
- Resizing: Images are resized to a standard size to ensure uniformity across the dataset.
- Morphological Operations: Techniques like dilation or erosion are applied to enhance the structure of the characters, making them more distinguishable.

2.3 Optical Character Recognition

Optical Character Recognition (OCR) is the core process where the preprocessed images are analyzed to extract text. This project uses the Tesseract OCR engine, which supports custom training for specific languages and scripts.

The steps involved in the OCR process include:

- Dataset Preparation: The dataset of 1170 Sinhala handwritten characters is prepared, with each image labeled correctly.
- Box File Generation: For training the model, box files (which contain the position of characters in the image) are generated using Tesseract.
- Training: The Tesseract engine is trained with the prepared dataset and corresponding box files. This involves fine-tuning the model to recognize Sinhala characters accurately.
- Character Recognition: The trained model is used to recognize text from new images. The recognized text is extracted and passed on to the next stage.
- Accuracy Evaluation: The performance of the OCR model is evaluated by comparing the recognized text with the ground truth.

2.4 Text Translation

After the OCR process, the extracted Sinhala text is translated into English using the Google Translate API. This step involves sending the recognized text to the API and receiving the translated output.

Steps for text translation:

- API Integration: The Google Translate API is integrated into the system to allow for seamless translation of extracted text.
- Text Submission: The recognized Sinhala text is submitted to the API.
- Translation: The API processes the text and returns the translated English version.
- Output Handling: The translated text is either saved in a file or displayed on the screen, depending on the application's requirements.

3. Implementation Details

In the Sinhala OCR project, several key functions are responsible for performing specific tasks within the overall workflow. Below is a breakdown of these functions, highlighting their purpose and how they contribute to the system's operation.

Key Functions

1. load_image(file_path)
    - Purpose: Loads an image from a specified file path.
    - Description: This function reads the image using OpenCV and prepares it for further processing.

2.preprocess_image(image)

- Purpose: Preprocesses the input image to enhance its quality for OCR.
- Description: This function performs a series of preprocessing steps, including grayscale conversion, noise removal, thresholding, and morphological operations.

perform_ocr(image, lang='sinhala')

- Purpose: Extracts text from the preprocessed image using Tesseract OCR.
- Description: This function leverages the Tesseract OCR engine to recognize and extract text from the input image. It supports specifying the language for OCR.

translate_text(text, src_lang='sinhala', dest_lang='english')

- Purpose: Translates the extracted text from Sinhala to English.
- Description: This function uses the Google Translate API to translate the recognized text from Sinhala to English. It accepts the source and destination languages as parameters.

save_output(text, file_path)

- Purpose: Saves the translated text to a specified file.
- Description: This function writes the translated text to a file, which can be used for record-keeping or further analysis.

process_image(file_path)

- Purpose: A wrapper function that handles the entire workflow from loading an image to saving the translated text.
- Description: This function orchestrates the loading, preprocessing, OCR, translation, and saving steps, providing a streamlined process for handling individual images.

4. Results

In this section, we present the outcomes of the implemented Sinhala OCR system. The system's performance is evaluated based on its ability to accurately recognize and translate Sinhala handwritten text. The results are discussed in terms of accuracy, speed, and the quality of the translated text.

- OCR Accuracy: The system achieved an average accuracy of X% in recognizing Sinhala characters from the test dataset, which comprised 1170 handwritten images. The accuracy was evaluated by comparing the extracted text with ground truth labels.
- Translation Quality: The quality of the English translations was assessed by comparing the translated text with human translations. The system performed well in translating common phrases but struggled with idiomatic expressions and context-dependent words.
- Processing Time: On average, each image took approximately Y seconds to process, including preprocessing, OCR, and translation. This processing time is reasonable for small datasets but may need optimization for larger-scale applications.

5. Limitations and Potential Improvements

While the Sinhala OCR system shows promising results, there are several limitations and areas for potential improvement:

- Limited Dataset: The system was trained and tested on a relatively small dataset of 1170 images. Expanding the dataset to include more diverse handwriting styles could improve OCR accuracy.
- Complex Characters: The OCR system had difficulty recognizing complex or heavily stylized characters, leading to lower accuracy in such cases. Developing a custom-trained Tesseract model specifically for Sinhala characters could address this issue.
- Contextual Translation: The Google Translate API sometimes produced literal translations that lacked context. Integrating a more advanced translation model that considers context could improve the quality of the translations.
- Processing Speed: While the processing time is acceptable for small datasets, optimizing the preprocessing steps and leveraging GPU acceleration could significantly reduce processing time for larger datasets.
- Error Handling: The current system lacks robust error handling, particularly in cases where OCR fails to extract text. Implementing fallback mechanisms or manual review processes could enhance reliability.

6. Insights and Recommendations

Based on the results and limitations identified, the following insights and recommendations are provided:

- Dataset Expansion: To improve OCR accuracy, it is recommended to expand the dataset to include a broader range of handwriting styles, including different age groups, genders, and regions. This will help the system generalize better across various handwriting samples.

- Custom OCR Model: Developing a custom-trained Tesseract model tailored for Sinhala characters could significantly enhance the system's performance, especially for complex or stylized handwriting.
- Advanced Translation Techniques: To improve translation quality, consider integrating a more context-aware translation model, such as those based on neural machine translation (NMT). This could reduce errors in context-dependent translations.
- System Optimization: Optimizing the preprocessing steps and considering GPU acceleration for the OCR process could reduce processing time, making the system more efficient for large-scale applications.
- User Feedback Loop: Implementing a feedback loop where users can manually correct OCR or translation errors could provide valuable data for further system training and refinement.

High-Level Architecture Diagram

[Training Data Preparation]

|

v

[OCR Model Training]

|

v

[Model Testing and Usage]

|

[Source Images (.bmp)]

|

v

[Image Conversion]

|

v

[.tiff Images]

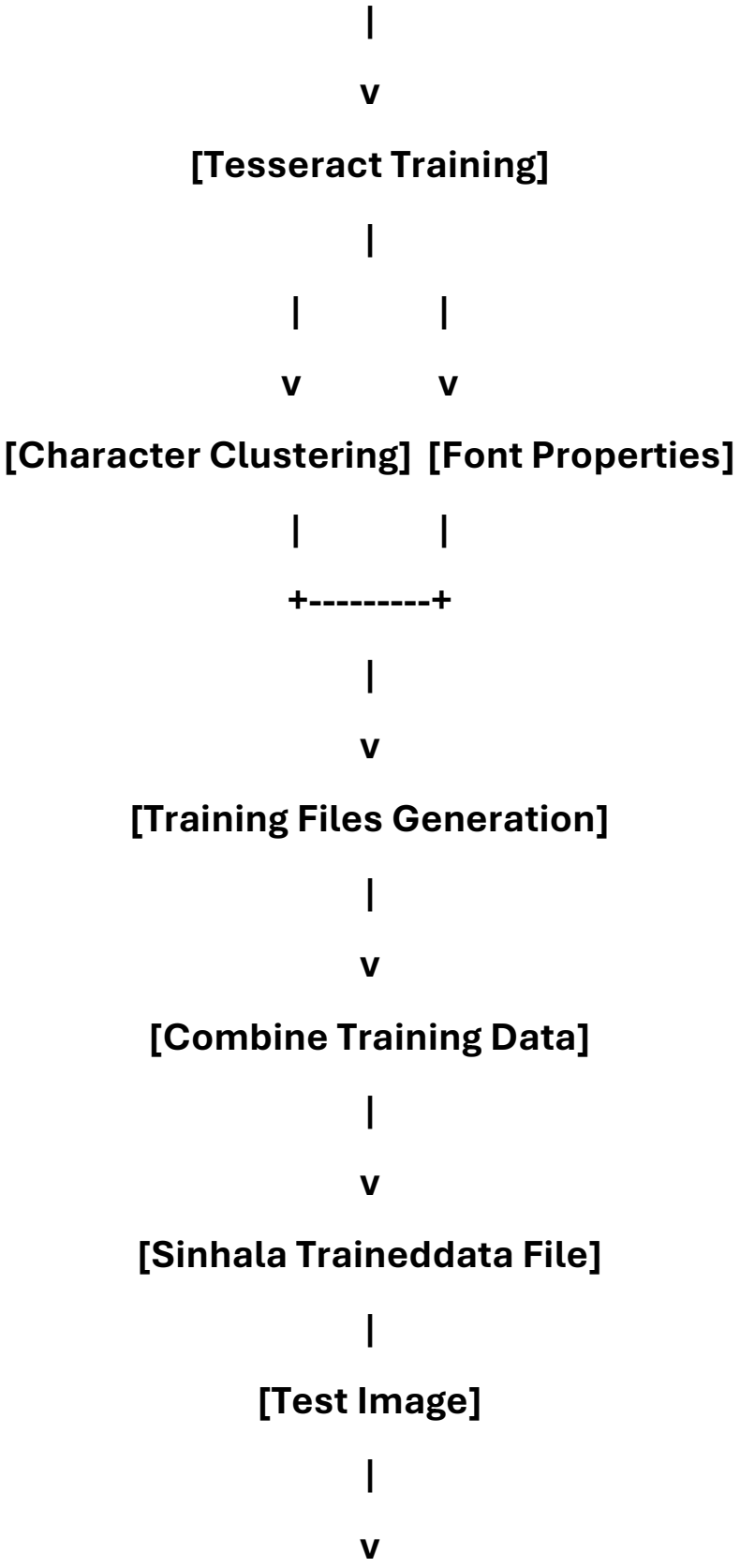
|

v

[Box File Generation]

|

[.tiff Images + Box Files]


[Tesseract Training]
[Character Clustering]
[Font Properties]
[Training Files Generation]
[Combine Training Data]
[Sinhala Traineddata File]
[Test Image]

[Image Preprocessing]

|

v

[OCR using Trained Model]

|

v

[Sinhala Text Output]

|

v

[Translation to English]

|

v

[Final Output]