Ancient Tamil Palm Leaf character recognition using Transformers

Aswinkumar I, Sangeetha Sivanesan, S Jaya Nirmala

Proceedings of the 39th Pacific Asia Conference on
Language, Information and Computation (PACLIC 39)

Emmanuele Chersoni, Jong-Bok Kim (eds.)

2025

© 2025. Aswinkumar I, Sangeetha Sivanesan, S Jaya Nirmala. Ancient Tamil Palm Leaf character recognition using Transformers. In Emmanuele Chersoni, Jong-Bok Kim (eds.), *Proceedings of the 39th Pacific Asia Conference on Language, Information and Computation* (PACLIC 39), 662-670. Institute for the Study of Language and Information, Kyung Hee University. This work is licensed under the Creative Commons Attribution 4.0 International License.

Ancient Tamil Palm Leaf character recognition using Transformers

Aswinkumar I[1], Sangeetha Sivanesan[2], S Jaya Nirmala[2]
[1]Thiagarajar College of Engineering Madurai, Tamil Nadu, India,
[2]National Institute of Technology Tiruchirappalli, Tamil Nadu, India
tceaswin@gmail.com, sangeetha@nitt.edu, sjaya@nitt.edu

Abstract

Optical character recognition (OCR) for ancient Tamil scripts is highly challenging because of the variability, degradation and differences in styles of historical handwritten documents. This research was aimed at developing an effective OCR system for the Vattaeluthu script, which is a historical form of Tamil text, using transformer-based architecture. We created an updated labelled training dataset, which consisted of 15,423 unique glyph samples from 75 different categories. The dataset contains a combined manual annotation from palm-leaf glyphs as well as a significant amount of synthetic data augmented using an AI image augmentation technique. The experimental results demonstrate the potential of deep learning within this context. The CNN reached a classification accuracy of 90.63%, and the Transformer based model, which we optimized, achieved a classification accuracy of 96.18%. We ultimately combined the final models using a Streamlit-based user interface, enabling efficient character recognition, visualizations, and easy reconstruction of output. This research provides a solid foundation for digitally preserving ancient Tamil texts, as well as serving as a benchmark for future heritage optical character recognition research.

Keywords: Vattaeluthu, Tamil OCR, Deep Learning, TrOCR, CNN, Palm-leaf Manuscripts, Heritage Digitization, Glyph Recognition.

1 Introduction

South India's linguistic and cultural legacy is exemplified by the Vattaeluthu script, an ancient style of Tamil writing. It is a vital tool for comprehending historical writings, inscriptions, and classical Tamil literature. Even with its scholarly importance, mechanically recognizing and digitizing Vattaeluthu characters still presents difficulties for document interpretation and historical text preservation. The use of contemporary machine learning methods is restricted by the absence of sizable, annotated datasets. Recent developments in deep learning, particularly Transformers, have opened up new avenues for the recognition of ancient scripts in order to address these problems. These models can handle both stylistic variations and the physical damage often present in ancient manuscripts. A comprehensive OCR pipeline created especially for Vattaeluthu character recognition is shown in this study. The system had created by training CNN and finetuned the TrOCR models separately, and using a sizable synthetic dataset increases its efficacy. This improvement is essential for increasing the model's ability to generalize across various document circumstances and writing styles. By using this approach, we hope to promote a more comprehensive conservation of historical culture by improving the digital preservation and scholarly accessibility to Tamil cultural literature.

2 Literature Survey

The Optical Character Recognition for historical and ancient scripts has been indeed undergone a significant transformation, evolving from foundational classical methods to more sophisticated deep learning solutions, particularly in the context of ancient Tamil inscriptions. Earlier efforts in character recognition for ancient scripts frequently involved classical image processing techniques and machine learning algorithms. These approaches often served as a groundwork for understanding and digitizing historical texts.

Features like bends, loops, and curls from segmented characters, Scale Invariant Feature Transform (SIFT), corner detection and area property were extracted and Support Vector Machines (SVM), Artificial Neural Networks (ANN), and K-Nearest Neighbours (KNN) were were applied by

the researchers in 2019. (Merline Magrina and Santhi, 2019).

Current methods use automation and end-to-end learning processes to try to overcome the constraints of earlier approaches. Enhancement of images is now directly performed using Fully Convolutional Networks (FCNs) (Sivashanth et al., 2024), employing ResNet-50 backbones and DeepLabV3+ architectures as models. Dilated convolutions are used to extract contextual characteristics at various scales.

One notable system is DR-LIFT: Detection, Recognition, and Labeling Text Interpreter Framework, a customized deep learning framework that integrates several models across three primary stages to address inscriptions dating back to the third century BCE. The tasks include alphabet detection, Text detection with arbitrary shapes , Prediction of text shapes. It also adopts context-aware recognition Finally, for labeling and sentence recognition, a Neural Graph Machine (NGM) is adopted (Murugan and Visalakshi, 2024)

The old Vattaeluthu script has visually similar glyphs, and the extreme lack of digitized, annotated datasets for training contemporary machine learning models are the main causes of this difficulty. Additionally, previous efforts frequently relied on very small datasets, which limited the models' capacity to generalize across variations in handwriting, fading, and manuscript deterioration.

Our research addresses these issues in a substantially more thoroughly. We have curated a significantly larger and more diverse dataset by combining manually annotated palm-leaf manuscript samples with synthetically generated data to balance character class distribution. This augmentation process ensures that even the least frequently occurring glyphs are adequately represented in training, improving recognition consistency across the full character set.

Another key differentiator of this work is the introduction of Transformer-based architectures specifically the TrOCR model into the domain of ancient Tamil palm-leaf manuscript recognition. To the best of our knowledge, no prior work has been successfully applied Transformer models for Vattaeluthu OCR.

Using our enhanced dataset to refine the TrOCR model, this study establishes a new standard for historical Tamil OCR.

3 Proposed Work

The proposed work outlines a comprehensive system for the recognition of Tamil Vattaeluthu characters, encompassing meticulous dataset construction, a multi-stage processing pipeline, and the training and evaluation of advanced deep learning models.

3.1 Dataset Description

The foundation of this research relies on a robust and diverse dataset, critical for training high-performing deep learning models. The initial base for our dataset was derived from the Mendeley repository, specifically the "Tamil palmleaf Vatteluttu Dataset"(Sasikaladevi, 2024). This foundational collection originally comprised 7,100 glyph samples covering 71 distinct Vattaeluthu characters.

To ensure high-quality dataset generation and improve the diversity of character samples, a meticulous manual annotation process was carried out. We had been using Label Studio, an open-source data annotation tool, to annotate characters from raw, binarized palm-leaf manuscript images. These binarized inputs were derived from historically significant Tamil texts, including *Naladiyar* (27 original images, resulting in 26 usable ground truth images), *Tholkappiyam* (221 original images, yielding 163 usable samples), and *Thirikadugam* (14 original images, contributing 10 final samples) (Jailingeswari and Gopinathan, 2024).

The annotation process was performed entirely by hand. Each character within the palm-leaf manuscript was manually identified and enclosed using rectangular bounding boxes. This was done within Label Studio's tool as shown in Figure 1. [1]

Each character was cropped from the original binarized manuscript image using the bounding box coordinates. To maintain consistency with the original background conditions, padding was added around each character to simulate the black background used in earlier datasets. This step helped standardize the image dimensions and background contrast across all samples.

The final annotated dataset combined these newly processed samples with previously curated Mendeley data, ultimately producing a total of 2,180 manually labelled character images across 59 distinct ancient Tamil characters. This refined dataset forms a robust foundation for training deep

[1] Label Studio

Dataset	# Characters	# Images
Mendeley dataset (Sasikaladevi, 2024)	71	7,100
Manually labeled instances	59	2,180
AI generated instances	68	6,143
Combined Dataset	**75**	**15,423**

Table 1: Dataset composition from Mendeley repository, manual annotation, and AI-generated augmentation.

learning models in ancient Tamil OCR tasks.

Recognizing the potential for class imbalance and the need for a larger volume of training data, we implemented a sophisticated augmentation strategy using Gemini AI. A custom Python script was developed to interact with the Gemini multimodal generative model.

The script was configured to take an existing glyph image and generate diverse handwritten-style variants. The core prompt provided to Gemini AI instructed it to:
[2]

> "You are generating synthetic training data for a historical Tamil Vatteluttu OCR model. Given a single example glyph image, generate diverse handwritten-style variants that represent the same character. Apply variations in stroke thickness, stroke curvature, ink flow, brush/pen pressure, natural writing slant, and aging effects such as mild fading or bleed. Ensure each generated image contains only one glyph, centered, with no noise, borders, artifacts, or text outside the character. Output each image as a clean black glyph on a transparent or pale off-white background, in a centered square format (e.g., 224×224). Maintain legibility while simulating realistic handwriting diversity."

The process of creating the dataset was to balance the 200 photos per class over all 75 characters. The script used `gemini-2.5-flash` as the model and asked for 5 image variants per API call. This AI-driven augmentation significantly expanded the dataset, bringing its final size to 15,423 images as shown in Table 1. For consistent model input, every image in the final dataset was normalized and reduced to 448×448 pixels.

Finally, a Python script was used to cross-validate the similarity between the real images and the images generated by Gemini AI. This process measured the Cosine Similarity for each class of Tamil Vattaeluthu characters to ensure the AI generated images were a close match to the real ones. The results of this comparison are shown in Table 6.

3.2 Methodology

The proposed OCR pipeline for Tamil Vattaeluthu character recognition follows a systematic multi-stage approach, designed to process raw image inputs into recognizable text outputs. The system architecture facilitates both preprocessing and segmentation of glyphs before feeding them into deep learning classifiers for recognition. As shown in Figure 2, each phase of this pipeline is detailed in the following subsections.

3.2.1 Pre-Processing

The first phase in processing any palm leaf manuscript image is to prepare it for accurate character recognition by enhancing its visual clarity and structural consistency. To begin, the image is converted into shades of grey, reducing it to essential visual data by removing color distractions. This simplification lays the foundation for further refinement. Then, a method called Contrast Limited Adaptive Histogram Equalization (CLAHE) is applied that enhances contrast in limited areas and then it is used to make low-contrast or faded characters more noticeable. This guarantees that even the manuscript's faintest markings are visible. A soft-focus filter is used to softly smooth the image once the contrast has been improved. This keeps each character's edges crisp and undamaged while reducing undesired graininess and background noise. The image is then converted to black-and-white using adaptive thresholding that adjusts to changes in illumination in various manuscript sections. Because uneven exposure and fading are typical in historical manuscripts.

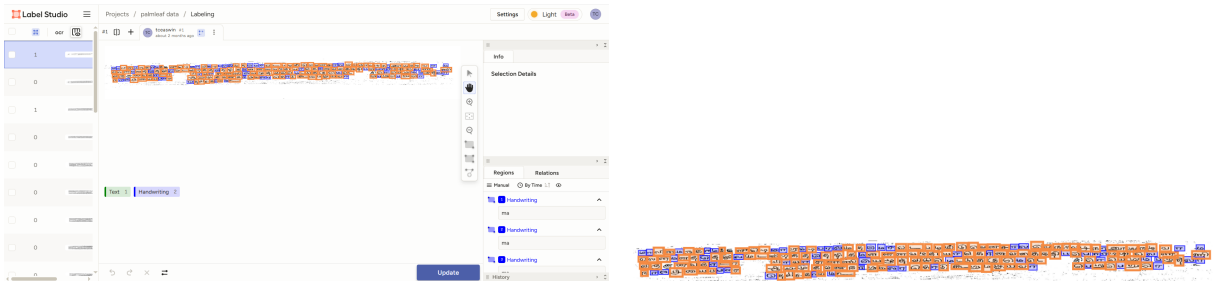
The outcome of this stage is a cleaner, sharper image where the individual characters are well-defined and ready for precise segmentation. [3]

3.2.2 Segmentation

The next stage is to meticulously dissect the manuscript image into its component characters after it has been improved and transformed into a binary format. This is accomplished by identifying each character's outer bounds within the

[2] Gemini app

[3] Open Cv



(a) Manual mapping of letters in Label Studio. (b) Binarized palm leaf with marked letters.

Figure 1: Label Studio annotation process: (a) bounding box mapping of characters, (b) annotated binarized palm leaf manuscript.

image. Following the identification of possible character shapes, a rigorous filtering procedure is used to guarantee that only significant and properly formed glyphs are retained for additional examination. Every shape that is detected is assessed using a set of real-world criteria:

First, the size of the image is assessed. Essentially, specks or tiny dots that might be noise are eliminated since only those that take up a sizable portion of the visual field are kept.

The ratio of the shape's height to width is evaluated. A shape is removed if it looks too stretched in one direction, because likely a mistake or unwanted mark and not a real character.

The valid segmented glyphs are first scaled and padded to uniform proportions before being given into the classifier. The extracted ROIs are specifically prepared to meet the input dimensions needed by the classification model (e.g., 128×128 for CNN, or 384×384 for TrOCR) by the resize and pad function. Glyphs are sorted line-wise after segmentation, first by grouping according to their Y-axis position and then by sorting along the X-axis inside each line.

3.2.3 Vattaeluthu Script Recognition Models

Feature extraction is the core mechanism by which the deep learning models learn meaningful representations from the pre-processed glyph images.

TrOCR Model To accurately recognize ancient script characters, a highly capable deep learning model known as TrOCR is employed(Li et al., 2021). This model is based on an advanced Transformer architecture pre-trained on a large variety of handwritten text, providing a strong foundation for understanding intricate visual patterns.

Similar to neural machine translation models, TrOCR follows a two-stage encoder–decoder design as shown in Figure 3.

Encoder: In our implementation, the encoder is adapted to work directly with convolution-based feature extraction. This modification aligns with our specific dataset characteristics and the challenges of ancient palm-leaf manuscripts. The encoder processes the input glyph image to extract high-level visual features while preserving spatial relationships essential for accurate character recognition.

Decoder: The extracted feature representations are passed to a Transformer-based language decoder, which interprets these visual patterns and generates the corresponding character sequence. The decoder handles sequential dependencies, making it capable of performing both individual glyph classification and continuous text recognition.

Feature Extraction in the Encoder The encoder processes the grayscale glyph image through multiple convolutional layers to identify edges, curves, and finer script patterns. Batch normalization and activation functions ensure stable learning. The processed feature maps are then transformed into a sequence of feature tokens understandable by the Transformer decoder.

Training Objective Loss Function: Typically, Cross-Entropy Loss is used during training to compare predicted sequences with the ground-truth labels.

The focus here is the model is fine-tuned using individual characters, with the focus on character-level training rather than the conventional full-text transcription. The image of a single glyph is first processed by the model's encoder, which extracts deep and meaningful visual features. These features are then passed through an additional processing layer that helps convert this visual understanding into a specific character prediction.

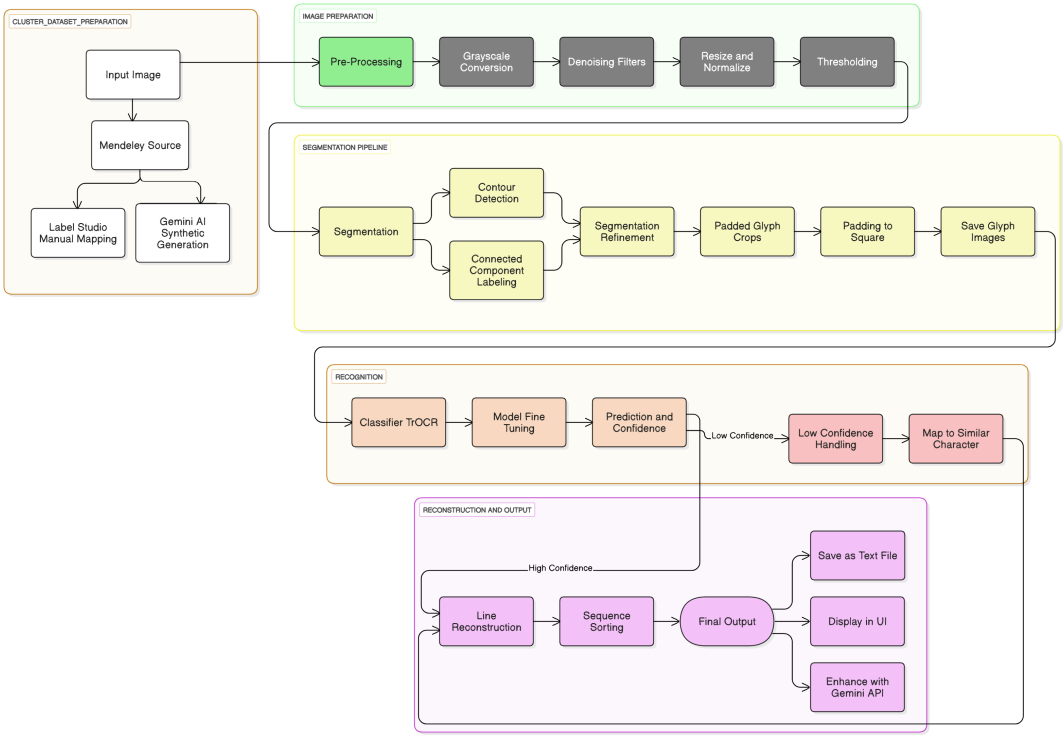


Figure 2: System Architecture Diagram.

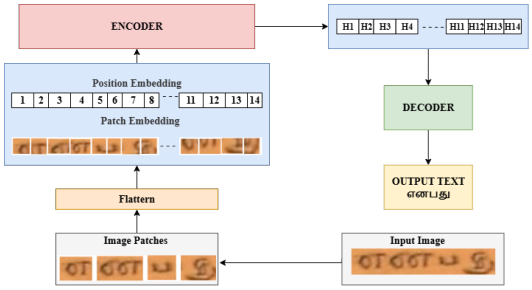


Figure 3: TrOCR architecture diagram.

The model has been adapted to classify characters from a set of 75 distinct symbols, each representing an ancient glyph. By doing this, the system uses the Transformer's ability to detect even the smallest visual features to convert visual patterns into precise character labels.

All glyph pictures are downsized to a square format with equal height and width, each measuring 384 pixels, to provide uniformity and compatibility with the model's training settings. Regardless of the scale or shape of the original image, this resizing aids to the improvement of model's accuracy.

Fine-Tuning TrOCR Model for Ancient Tamil Script Recognition

The Transformer-based OCR model, optimization also used the Adam algorithm with a moderate learning rate suitable for fine-tuning. To avoid overtraining and reduce unnecessary computation, early stopping was enabled. If validation accuracy failed to improve over five consecutive rounds, training would automatically stop, and the best-performing version of the model would be retained. The full training process was limited to a maximum of fifty rounds to strike a balance between thorough learning and computational efficiency as shown in table 8.

This made it possible for the system to begin with a solid grasp of common handwriting patterns and then modify it to fit the particulars of the old script. The training procedure concentrated on individual glyphs, each coupled with its appropriate descriptor, rather than overall sentence-level input for training.

The model's final classification layer was modified to identify 75 different characters used in the script in order to conform to the particular task. To do this, the output stage of the model had

to be resized to fit the target character set.

The entire dataset was meticulously divided into three parts to provide a fair evaluation as shown in table 7. The remaining sample was utilized just to assess the model's accuracy on previously unseen data, while a smaller portion was used to validate the model's progress during training and the rest was used to train the model.

CNN Model A bespoke Convolutional Neural Network (CNN) was trained utilizing an adaptable deep learning architecture in order to compare the effectiveness of Transformer Model with the standard CNN. The model received a constant stream of image samples during training.

The training was carried out using the Adam optimizer algorithm with a carefully chosen low learning rate to ensure stable convergence. The augmentation methods including comprised slight brightness shifts, moderate contrast alterations, and random horizontal flips. These augmentation methods improved the model's capacity to generalize to unseen glyph samples by making it more resistant to changes in illumination.

To balance performance and efficiency, two key strategies were used during training: one to automatically adjust the learning rate when learning rate is reduced, and another to stop training early if no improvement was observed when compared to previous iterations. This will prevent overfitting, even though the training process was permitted to run for up to 100 complete passes through the dataset.

3.3 Results and Discussion

As per the evaluations, the TrOCR-based classifier and CNN-based classifier are both capable of identifying Tamil Vattaeluthu characters. A comparison investigation demonstrates how much better the Transformer-based architecture performs.

On all important criteria, however, the improved TrOCR model performed noticeably better than the CNN. A remarkable 96.18% test accuracy was attained. Specifically, TrOCR's macro-averaged precision, recall, and F1-score were 0.9614, 0.9634, and 0.9616 as shown in Table 2.

Character Error Rate (CER), Word Error Rate (WER), and Levenshtein Distance(LD) were among the other text-based evaluation measures that were calculated to give a more thorough picture of the models' performance. With a CER of 0.0208, the TrOCR model demonstrated an extremely low percentage of wrong characters in comparison to all anticipated characters. The partial nature of glyph-based predictions, where accuracy at the character level affects complete word reconstruction, was reflected in the Word Error Rate (WER), which was assessed at 0.5000. The model's capacity to provide predictions with little textual deviance was further supported by the discovery that the Average Levenshtein Distance between the predicted and ground truth sequences was 0.50 as shown in Table 3. These extra measures highlight the model's proficiency in producing character sequences that are almost accurate in addition to its classification accuracy, which is crucial for developing trustworthy OCR systems for old scripts.[4]

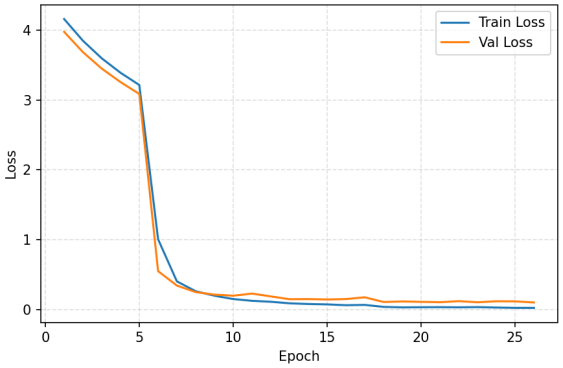


Figure 4: TrOCR fine-tuning: Train Loss vs Validation Loss.

This feature enables TrOCR to more accurately decipher the intricate and frequently deteriorated characteristics of Vattaeluthu glyphs, resulting in more reliable recognition, especially on difficult and deteriorated samples.

An overall accuracy of 90.63% was attained by the CNN classifier. Additionally, precision, recall, and F1-score were used to assess its performance; the macro-averaged results were 0.9086, 0.9058, and 0.9056, respectively. A thorough classification report showed that different character classes performed differently. Some classes performed worse than others, such Glyphs like '൹', which had a precision of 0.41 and recall of 0.38, but several classes had high scores (e.g., Glyphs '◌ി': 1.00 precision, 0.95 recall; Glyphs '൪': 1.00 accuracy, 1.00 recall; Glyphs '൭': 1.00 precision, 1.00 recall). This draws attention to certain difficulties the CNN had with particular glyph variations.

[4] Github Link of the code

The final fine-tuned model of TrOCR, have been successfully deployed with a Streamlit-based user interface.

The effect of dataset frequency on recognition accuracy is a significant finding from the assessment procedure. In the manually gathered dataset, a number of glyphs were extremely rare, resulting in an imbalance in the data that can jeopardize the generalization of the model. In order to guarantee a minimum of 200 samples per class, synthetic augmentation was carried out on low-occurrence characters. The training process was greatly improved by this endeavor, particularly for characters with limited representation.

Interestingly, the classification results indicate that both rare and frequently occurring characters yielded very high accuracy, suggesting that the model learned to generalize well even from initially imbalanced data thanks to synthetic augmentation. For example:

Glyphs like 'சீ' and 'கீ', which originally had 0 samples in manual annotation, achieved precision and recall near or equal to 1.00 after augmentation.

Characters with extremely low original frequency like 'ணி' (1), 'ளி' (0), 'தீ' (0), and 'கூ' (4) in manual annotation were also recognized with high precision and recall.

Conversely, high-frequency glyphs, such as 'ாற' (300), 'ம' (143), and 'க' (150), also achieved consistently excellent recognition metrics, reinforcing the robustness of the augmentation strategy.

This highlights the efficacy of the data synthesis pipeline driven by Gemini AI and shows the model's ability to function consistently across different character distributions. In addition to improving classification metrics, the balanced dataset made the system more applicable to historical documents with irregular glyph distributions in the actual world.

Model	Accuracy	Precision	Recall	F1 Score
TrOCR	96.18%	0.9614	0.9634	0.9616
CNN	90.63%	0.9086	0.9058	0.9056

Table 2: Model Performance Metrics.

Model	CER	WER	Avg.LD
TrOCR	0.0208	0.5000	0.50
CNN	0.2292	1.0000	5.50

Table 3: Character Error Rate, Word Error Rate, and Average Levenshtein Distance for both models.

4 Conclusion

This research presents a system designed for ancient Tamil Vattaeluthu scripts. It achieves high accuracy by Transformer-based TrOCR. Our dual-model approach compares a custom CNN with a fine-tuned Transformer-based TrOCR architecture. This comparison shows the advantages of Transformer-based models in classifying ancient glyphs. The TrOCR model effectively learns complex patterns and long-range dependencies.

This work offers several contributions. We created a large, carefully curated dataset through manual annotation and innovative AI-driven synthetic data augmentation. We also developed a complete image processing and segmentation pipeline and successfully deployed the recognition models. This system provides a useful resource and sets a new standard for the digital preservation and accessibility of Tamil heritage texts.

Acknowledgments

We sincerely thank all the dataset contributors and the active open-source community for their efforts in improving this field including Mendeley Data.

References

I Jailingeswari and S Gopinathan. 2024. Tamil handwritten palm leaf manuscript dataset (thplmd). *Data in Brief*, 53:110100.

Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. 2021. Trocr: transformer-based optical character recognition with pre-trained models (2021). *arXiv preprint arXiv:2109.10282*.

M Merline Magrina and M Santhi. 2019. Ancient tamil character recognition from epigraphical inscriptions using image processing techniques. matjournals 4 (2): 40–48.

Balasubramanian Murugan and P Visalakshi. 2024. Ancient tamil inscription recognition using detect, recognize and labelling, interpreter framework of text method. *Heritage Science*, 12(1):1–21.

Sasikaladevi. 2024. Tamil palmleaf vatteluttu dataset.

Suthakar Sivashanth, Eugene Yugarajah Andrew Charles, and Siyamalan Manivannan. 2024. A fully automated approach for enhancing ancient tamil inscriptions using deep learning. In *2024 8th SLAAI International Conference on Artificial Intelligence (SLAAI-ICAI)*, pages 1–5. IEEE.

Appendix

Class	Prec.	Rec.	F1	Class	Prec.	Rec.	F1	Class	Prec.	Rec.	F1
மூ	1.00	0.92	0.96	ட	1.00	0.95	0.98	ப	1.00	0.95	0.97
ா	0.98	0.98	0.98	ம	0.97	1.00	0.99	ெ	0.93	0.81	0.86
ய	0.96	0.92	0.94	ல	1.00	1.00	1.00	ன	0.95	0.91	0.93
ற	1.00	1.00	1.00	க	0.91	0.95	0.93	வ	0.91	1.00	0.95
ண	0.95	0.90	0.92	த	0.95	1.00	0.98	ச	1.00	0.93	0.97
ங	1.00	1.00	1.00	ள	0.94	1.00	0.97	தி	1.00	1.00	1.00
வி	1.00	0.90	0.95	றி	1.00	1.00	1.00	லி	0.92	1.00	0.96
னி	1.00	1.00	1.00	யி	1.00	0.90	0.95	ழி	0.95	1.00	0.98
பி	1.00	1.00	1.00	ரி	1.00	1.00	1.00	சி	0.96	1.00	0.98
ணி	1.00	1.00	1.00	மி	0.96	0.96	0.96	கி	0.88	1.00	0.94
டு	0.80	0.94	0.86	கு	1.00	1.00	1.00	ளு	0.52	0.59	0.55
லு	1.00	0.87	0.93	மு‌	0.86	0.92	0.89	ணு	0.94	0.94	0.94
னு	0.95	0.95	0.95	ஞு	1.00	1.00	1.00	பு	0.96	1.00	0.98
று	1.00	1.00	1.00	ருு	1.00	1.00	1.00	சு	1.00	1.00	1.00
து	1.00	1.00	1.00	வு	0.89	0.89	0.89	யு	0.94	1.00	0.97
ழு	1.00	1.00	1.00	டி	1.00	1.00	1.00	ளி	0.96	1.00	0.98
எ	1.00	1.00	1.00	ழு	1.00	1.00	1.00	கீ	1.00	0.93	0.96
றூ	1.00	1.00	1.00	மீ	1.00	1.00	1.00	வீ	1.00	1.00	1.00
நூ	1.00	1.00	1.00	றீ	1.00	1.00	1.00	தீ	1.00	1.00	1.00
கூ	1.00	1.00	1.00	தூ	1.00	1.00	1.00	சூ	0.91	1.00	0.95
யீ	1.00	1.00	1.00	லூ	0.96	1.00	0.98	உ	1.00	1.00	1.00
அ	1.00	1.00	1.00	ழீ	1.00	1.00	1.00	யூ	1.00	1.00	1.00
சீ	1.00	1.00	1.00	ணீ	1.00	0.95	0.97	ஆ	1.00	1.00	1.00
இ	0.94	1.00	0.97	ை	0.88	0.94	0.91	ர	0.93	0.93	0.93
ந	0.97	1.00	0.98	ஒ	0.96	0.96	0.96	ளூ	1.00	1.00	1.00
Accuracy: (0.96), **Macro Avg:** (0.96, 0.96, 0.96), **Weighted Avg:** (0.96, 0.96, 0.96))											

Table 4: Detailed classification report for TROCR model on Tamil character recognition.

Class	Prec.	Rec.	F1	Class	Prec.	Rec.	F1	Class	Prec.	Rec.	F1
மூ	0.83	0.87	0.85	ா	1.00	0.95	0.97	ட	1.00	0.91	0.95
ம	0.82	0.78	0.80	ப	0.95	0.90	0.93	ெ	0.92	0.90	0.91
ய	1.00	0.76	0.86	ல	0.90	0.90	0.90	ன	0.80	0.86	0.83
ற	1.00	0.90	0.95	க	0.85	0.85	0.85	வ	0.66	0.86	0.75
ண	0.93	0.91	0.92	த	0.89	0.92	0.90	ச	0.91	0.75	0.82
ங	0.96	0.98	0.97	ள	0.85	0.85	0.85	தி	0.89	0.97	0.93
வி	0.94	0.88	0.91	றி	0.95	0.97	0.96	லி	0.91	0.94	0.93
னி	0.89	0.97	0.93	யி	0.92	0.80	0.86	ழி	0.97	0.95	0.96
பி	1.00	0.86	0.93	ரி	0.97	0.97	0.97	சி	0.97	0.84	0.90
ணி	1.00	0.90	0.95	மி	0.84	0.92	0.88	கி	0.82	0.77	0.79
டு	0.86	0.79	0.83	கு	0.76	0.89	0.82	ளு	0.41	0.38	0.39
லு	0.88	0.95	0.92	முு	0.82	0.87	0.85	ணு	0.94	0.81	0.87
னு	0.78	0.88	0.83	ஞு	0.88	0.86	0.87	பு	0.98	1.00	0.99
று	0.86	0.90	0.88	ருு	0.96	0.81	0.88	சு	0.95	0.91	0.93
து	0.90	0.96	0.93	வு	0.93	0.98	0.95	யு	0.98	0.96	0.97
ழு	0.84	0.78	0.81	டி	0.94	0.98	0.96	ளி	0.93	0.97	0.95
எ	0.90	1.00	0.95	ழு	0.92	0.92	0.92	கீ	1.00	0.98	0.99
றூ	0.95	0.97	0.96	மீ	0.89	0.91	0.90	வீ	0.89	0.97	0.93
நூ	1.00	0.98	0.99	றீ	0.97	0.92	0.95	தீ	0.95	0.95	0.95
கூ	0.94	1.00	0.97	தூ	0.95	0.97	0.96	சூ	0.90	0.95	0.93
யீ	0.97	0.88	0.92	லூ	0.95	0.95	0.95	உ	1.00	0.97	0.99
அ	0.94	0.94	0.94	ழீ	0.95	0.95	0.95	யூ	1.00	1.00	1.00
சீ	1.00	0.97	0.99	ணீ	1.00	0.97	0.99	ஆ	0.94	1.00	0.97
ளூ	0.52	0.60	0.56	இ	0.85	0.95	0.90	ை	0.90	0.96	0.93
ர	0.98	0.96	0.97	ந	0.92	1.00	0.96	ஒ	1.00	1.00	1.00
Accuracy: (0.91), **Macro Avg:** (0.91, 0.91, 0.91), **Weighted Avg:** (0.91, 0.91, 0.91)											

Table 5: Detailed classification report for CNN model on Tamil character recognition.

Class	Avg Similarity	Class	Avg Similarity	Class	Avg Similarity
மூ	0.86	ா	0.93	ட	0.88
ம	0.92	ப	0.92	ெ	0.92
ய	0.90	ல	0.93	ன	0.92
ற	0.90	க	0.94	வ	0.87
ண	0.92	த	0.92	ச	0.87
ங	0.92	ள	0.88	தி	0.79
வி	0.93	றி	0.79	லி	0.71
னி	0.87	யி	0.90	ழி	0.89
பி	0.90	ரி	0.89	கி	0.75
ணி	0.78	மி	0.89	கி	0.76
டு	0.91	கு	0.88	ளு	0.78
லு	0.74	மு	0.86	ணு	0.76
னு	0.89	ஞு	0.71	பு	0.91
று	0.88	ரு	0.91	சு	0.71
து	0.89	வு	0.76	யு	0.90
ழு	0.76	டி	0.90	ளி	0.82
எ	0.84	ழ	0.83	கீ	0.88
றூ	0.87	மீ	0.90	வீ	0.80
நூ	0.89	நீ	0.87	தீ	0.89
கூ	0.76	தூ	0.84	சூ	0.75
யீ	0.88	லூ	0.89	உ	0.80
அ	0.78	ழீ	0.89	யூ	0.81
சீ	0.85	ணீ	0.87	ஆ	0.82
ளூ	0.90	இ	0.81	ை	0.90
ர	0.91	நி	0.90	ஒ	0.90
Total Avg Similarity : 0.86					

Table 6: Cosine similarity between Gemini AI generated and real images for each class.

Model	Train	Validation	Test
TrOCR	70%	20%	10%
CNN	80%	0%	20%

Table 7: The split ratio of the dataset.

Hyperparameter	Value/Setting
Learning Rate (LR)	
LR_ENCODER	1e-5
LR_CLASSIFIER	1e-4
Optimizer Settings	Adam optimizer
Encoder params	lr=1e-5
Classifier params	lr=1e-4
Batch Size	16
Loss Function	Cross-Entropy Loss
Early Stopping	PATIENCE = 5
Maximum Epochs	50

Table 8: TrOCR Critical Hyperparameters.