An Efficient OCR Pipeline for Semi-Structured Khmer Documents Using Layout Analysis and Text-Type Classification

Sopanha Lay **Vichet Kao** **Seavchhing Kong** **Mengchhuong Ang** **Hongly Va**
Cambodia Academy of Digital Technology (CADT)
Phnom Penh, Cambodia
hongly.va@cadt.edu.kh

Abstract

The digitization of documents presents a significant challenge, particularly for complex, low-resource scripts like Khmer language. Standard Optical Character Recognition (OCR) systems often fail when faced with this mixed-media content, hindered by the complexity and scarcity of Khmer training data. This paper introduces a modular, resource-efficient pipeline designed to address these limitations through a classification-first approach that identifies text type prior to recognition, enabling the application of specialized OCR engines accordingly. At the core of our framework is a lightweight text-type classifier based on the MobileNetV3 architecture, integrated within a pipeline that leverages pre-trained models for layout analysis. Trained on a custom Khmer dataset, our experiments establish a performance benchmark for this difficult task: the classifier achieves an accuracy of **74.6%** on a held-out validation set. Crucially, its performance remains stable at **74.5%** within the full end-to-end pipeline, indicating the model is robust to noise from automated segmentation. Our work validates the classification-first strategy as essential for mixed-media OCR and provides a realistic benchmark for this low-resource task.

Keywords: *Optical Character Recognition (OCR), Khmer Language, Deep Learning, Text Type Classification, Document Analysis*

1 Introduction

As Cambodia accelerates its national digital transformation, digitizing its vast archive of official documents is a critical priority. Key documents, such as birth certificates, serve as foundational records for civil registration but exist primarily in paper format. This creates significant bottlenecks in data management, retrieval, and analysis, processes that remain manual, slow, and prone to error.

The primary challenge in automating the digitization of these documents is their **mixed-media content structure**: a combination of standardized, machine-printed template text and highly variable handwritten entries. Standard Optical Character Recognition (OCR) engines, which are typically optimized for only one text type, struggle to process this mixed content accurately [1].

This core problem is exacerbated by two further constraints inherent to the Khmer language context. First, the **Khmer script is inherently complex**, featuring a large character set and diacritics that challenge many OCR architectures. Second, there is a significant **scarcity of large-scale, annotated datasets** for Khmer document OCR [2],[3]. This data scarcity, in particular, makes it computationally infeasible to train a single, large, state-of-the-art model to handle all challenges simultaneously. Therefore, our work focuses specifically on solving the mixed-media problem through an efficient, modular approach practical within these low-resource constraints.

Initial research for this project considered a monolithic, end-to-end pipeline leveraging state-of-the-art Transformer-based models, such as TrOCR [4], which builds upon prior CNN-RNN architectures [5]. However, iterative analysis revealed that such an approach was impractical due to the aforementioned data and resource constraints. Consequently, a strategic pivot was made toward a more modular and efficient pipeline. This paper presents a novel hybrid approach that intelligently separates document analysis from targeted text recognition. The primary contribution is the development of a lightweight, highly accurate text-type classifier, built on a MobileNetV3 architecture [6]. This component enables the system to first identify the document's structure using robust tools like PaddleOCR [7], classify each text segment as either printed or handwritten, and then apply the most suitable recognition engine for each type. The objectives of this work are to:

- **Establish a robust baseline for document analysis** by leveraging a pre-trained model (PP-DocLayout & RT-DETR-L) to perform layout segmentation on complex, semi-structured Khmer documents.
- **Design, train, and evaluate a lightweight text-type classifier** (MobileNetV3) to accurately distinguish between printed and handwritten Khmer text snippets, quantifying its performance in both ideal and real-world conditions.
- **Demonstrate the value of a classification-first OCR pipeline** by integrating the classifier and using quantitative metrics (CER) to prove its superiority over a naive, single-engine approach for mixed-media documents.

2 Related Work

Research into Khmer OCR has evolved significantly, moving from traditional machine learning methods targeting isolated characters to end-to-end deep learning systems capable of handling complex documents.

2.1 Evolution from Traditional Methods

Early research focused on recognizing isolated, printed Khmer characters using traditional machine learning. Techniques such as Support Vector Machines (SVMs) were successfully applied, achieving high accuracy on clean, well-segmented characters [8]. However, these methods were often highly dependent on the quality of pre-processing and character segmentation, making them less robust for noisy, real-world documents with varied layouts.

2.2 Modern Deep Learning Approaches

The advent of deep learning brought more powerful and flexible solutions [9]. Key advancements in Khmer OCR include:

- **Efficient Architectures:** Annanurov and Noor demonstrated that compact CNN models could effectively recognize handwritten consonants, proving the feasibility of efficient models for resource-constrained environments [10]. This insight is central to our work's focus on lightweight models.
- **Historical Documents:** For the unique challenges of historical texts, researchers developed specialized datasets like SleukRith for palm-leaf manuscripts [11] and applied sophisticated encoder-decoder models to handle stylistic variations and document degradation [12].
- **State-of-the-Art Models:** Recent work by Buoy et al. has advanced the field with Transformer-based models specifically designed to handle the long, unbroken text lines common in Khmer [3]. This work, along with the introduction of the KhmerST dataset by Nom et al. [2], has highlighted that even state-of-the-art architectures still face significant challenges with the Khmer script, underscoring the need for domain-specific solutions.

2.3 Research Gap

Despite these advances, a specific research gap remains in the processing of semi-structured official documents that contain a **mixture of printed and handwritten text**. Prior work has largely focused on either purely printed or purely handwritten text, or on different document types (e.g., historical manuscripts, scene text). The challenge of efficiently and accurately handling mixed-media content in a single pipeline for official documents like birth certificates is largely unaddressed. This project directly targets this gap by proposing a modular pipeline designed to classify and process this mixed content.

3 Methodology

To address the challenges of digitizing mixed-media Khmer documents, we designed a modular, multi-stage pipeline that prioritizes resource efficiency and accuracy. Our approach combines powerful pre-trained models for general tasks (like layout analysis) with a lightweight, custom-trained classifier for the specialized sub-task of differentiating text types. This section details the overall architecture and each of its core components, from initial image preprocessing to the final targeted text recognition.

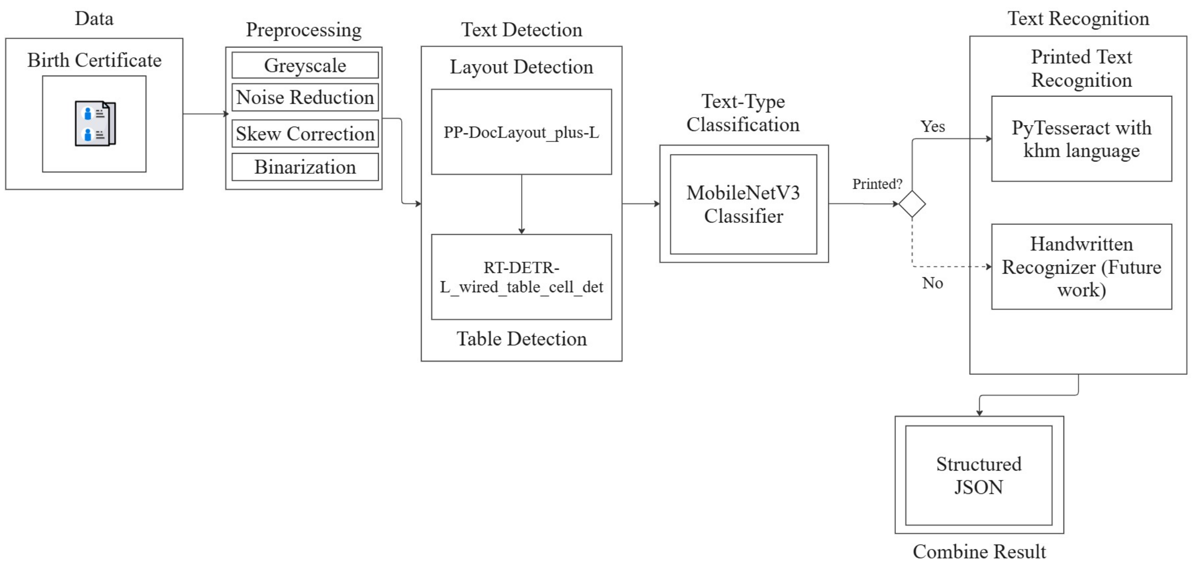


Figure 1: The high-level architecture of the proposed modular OCR pipeline. The path for handwritten text recognition is shown as a placeholder for future integration, for which a model like TrOCR could be used. This component was not implemented in the current work.

3.1 Pipeline Architecture Overview

The system ingests a scanned document image and processes it through a sequential workflow, as illustrated in **Figure 1**. The pipeline consists of four main stages: (1) automated image preprocessing to normalize input; (2) layout analysis to detect all text regions; (3) text-type classification to label each region as printed or handwritten; and (4) targeted text recognition where the appropriate OCR engine is applied based on the classification.

3.2 Document Preprocessing

To standardize input and improve the performance of subsequent modules, a series of automated image processing steps are applied using the OpenCV library. This crucial stage transforms raw, often-degraded scans into clean, machine-readable images. The visual progression of this workflow is detailed in **Figure 2**. The algorithmic process is as follows:

1. **Grayscale Conversion:** The input 3-channel RGB image is first converted to a single-channel grayscale format.

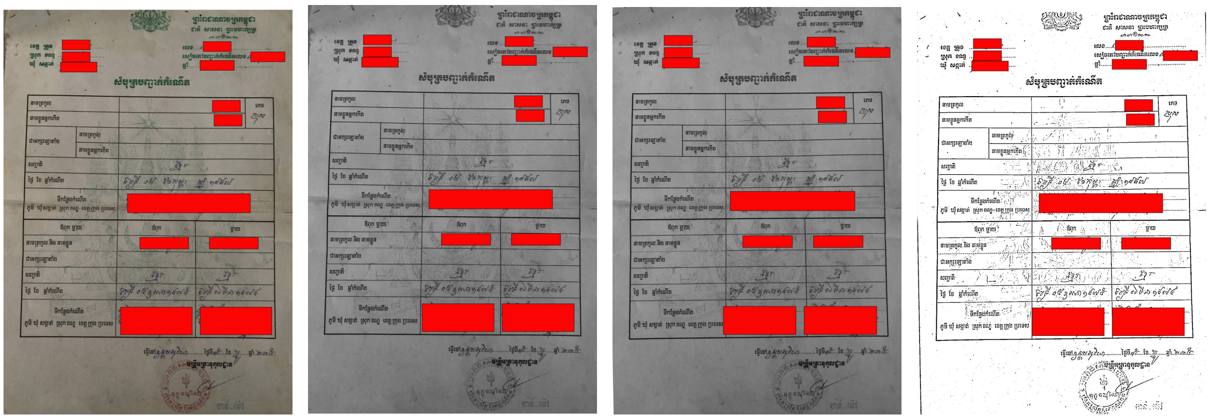
2. **Noise Reduction:** A Gaussian blur with a 5x5 kernel is applied to mitigate high-frequency noise from the scanning process without overly blurring character edges.

3. **Skew Correction:** The dominant angle of the text is detected and corrected. This is achieved by first applying adaptive thresholding to a temporary copy of the image, finding text contours, and calculating the orientation of the minimum area rectangle enclosing these contours. The original grayscale image is then rotated by this calculated angle to ensure all text lines are horizontal.

4. **Final Binarization:** A second adaptive thresholding is applied to the deskewed grayscale image. Unlike global thresholding, this method calculates a localized threshold for different regions of the image, making it highly robust to variations in lighting and background noise, resulting in a clean final output.

3.3 Layout and Text-Type Analysis

This phase performs a two-step analysis of the document's content. First, a pre-trained layout detector identifies all logical text regions. Each detected snippet is then passed to our novel, custom-trained text classifier, which labels it as either 'printed' or 'handwritten'. This critical step prepares each snippet for the appropriate recognition engine in the final stage. The models used are summarized in **Table 1**.



(a) Original Image **(b)** Grayscale Conversion **(c)** Skew Correction **(d)** Final Binarized Output

Figure 2: The automated preprocessing workflow from (a) Raw input, to (b) Grayscale, to (c) Deskewed, and finally (d) the Final binarized output.

Table 1: Core Models and Their Roles in the Pipeline

Component	**Model Used**	**Role**
Layout Analysis	PP-DocLayout	Detects logical regions (text, tables).
Table Cell Detection	RT-DETR-L	Detects cells in table.
Text type Classifier	MobileNetV3	Classifies regions as ‘printed’ or ‘handwritten’.
Printed OCR	Tesseract v5	Transcribes text from ‘printed’ regions.

3.3.1 Layout Analysis with PaddleOCR

Our pipeline employs a two-stage approach for document structure detection, leveraging pre-trained models from the PaddleOCR framework [7]. This strategy allows us to analyze both the high-level document layout and the fine-grained table structures without needing to train custom detectors.

First, for overall document structure, we use the **PP-DocLayout_plus-L** model. This identifies high-level layout elements, such as text paragraphs and full table regions.

once a table region is identified, we use a specialized **RT-DETR-L_wired_table_cell_det** model trained for table cell detection. This model processes the cropped table area to precisely localize individual cells, preparing them for subsequent analysis.

3.3.2 Text-Type Classifier: Architecture, Training, and Inference

A key contribution of this work is the development of a lightweight and accurate classifier capable of distinguishing printed from handwritten Khmer text snippets. This section details its architecture, the dataset it was trained on, and its role during the pipeline’s inference process.

Architecture and Dataset The classifier uses a **MobileNetV3-Large** architecture [6], pre-trained on ImageNet. Input snippets (224×224) pass through Inverted Residual Blocks and a Global Average Pooling layer, producing a feature vector fed to a fully-connected layer that outputs logits for printed and handwritten classes (Figure 3). Training used a custom dataset of 1,677 text regions from Cambodian birth certificates (956 printed, 721 handwritten), manually cropped and labeled with PPOCRLabelV3, a tool from the PaddleOCR suite. The dataset was stratified into 1,342 training and 335 validation samples (**Table 2**), with this fixed split used for all training and evaluation instead of k-fold cross-validation.

Table 2: Dataset Split for Text-Type Classifier

Class	Training	Validation	Total
Printed	765	191	956
Handwritten	577	144	721
Total	1,342	335	1,677

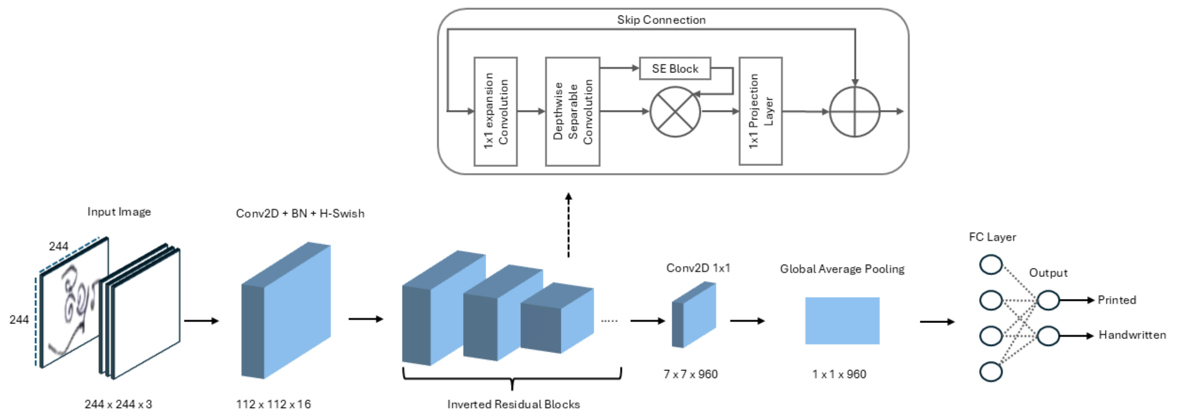


Figure 3: The proposed MobileNetV3-Large architecture for Khmer text-type classification. The main data flow passes through a convolutional backbone of Inverted Residual Blocks. A detailed view of the Inverted Residual Block, with its internal expansion, depthwise convolution, SE module, projection layer, and residual (skip) connection, is shown in the callout box.

Table 3: Text-Type Classifier Hyperparameters

Parameter	**Value**
Architecture	
Model Architecture	MobileNetV3-Large (x1.0)
Pre-trained Weights	ImageNet
Input Size	224 x 224 x 3
Training Hyperparameters	
Optimizer	Adam
Learning Rate	Piecewise (0.001 - 0.0001)
LR Scheduler	Step Decay (at epoch 15)
Batch Size	32
Epochs	25
Loss Function	Cross-Entropy Loss (CELoss)
Data Augmentation	RandCrop, RandFlip, AutoAugment

Training Training is performed using the Adam optimizer to minimize categorical cross-entropy loss. For this binary task, the model's final layer outputs two logits corresponding to each class, which are passed through a Softmax activation function. The loss is then calculated using one-hot encoded labels with the following formula:

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=0}^{1} y_{i,c}\log(p_{i,c}) \qquad (1)$$

where $y_{i,c}$ denotes the one-hot encoded true label and $p_{i,c}$ the predicted probability for sample i and class c. Key training hyperparameters are detailed in Table 3.

Inference Workflow During inference, each text snippet detected by the layout analysis module is resized to 224x224 and passed through the trained MobileNetV3 classifier. Based on the predicted class, snippets are routed as follows:

- **Printed text:** Sent to Tesseract OCR, configured with the Khmer language pack, for transcription.
- **Handwritten text:** Bounding box and class label are preserved for future processing by a specialized handwritten recognizer.

This workflow allows the pipeline to efficiently direct each snippet to the most suitable recognition engine. The modular design also enables straightforward future integration of a dedicated handwritten OCR module, maintaining the pipeline's efficiency and adaptability.

3.3.3 Targeted Text Recognition

After the classifier sorts the text snippets, they are routed to specialized recognition engines. This targeted approach is the final and crucial stage of the pipeline.

Printed Text Recognition Snippets classified as ‘printed’ are sent to the Tesseract OCR engine for transcription. We use the standard Tesseract model configured with the official Khmer language pack, as detailed in **Table 4**. This engine forms the baseline for printed text recognition within our pipeline.

Table 4: Configuration for the Printed OCR Engine

Parameter	Value
OCR Engine	Tesseract v5.3.3
Language Pack	Khmer (‘khm‘)
OCR Engine Mode (OEM)	3 (Default)
Page Seg. Mode (PSM)	3 (Default)

Handwritten Text Recognition Snippets classified as ‘handwritten’ are preserved with their bounding boxes and class labels but are not transcribed. Although developing a handwritten recognition model was beyond the current scope, the pipeline’s modular design allows for the seamless integration of a specialized model such as a fine-tuned TrOCR in future work.

3.4 Evaluation Metrics

The performance of our custom text-type classifier is evaluated using four standard metrics derived from the confusion matrix components: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). The formulas for Accuracy, Precision, Recall, and F1-Score are standard and defined as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4)$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5)$$

These metrics provide a comprehensive assessment of the classifier’s performance, which is critical for the overall success of the pipeline.

4 Experiments and Results

To evaluate our proposed pipeline, we conducted experiments on both the isolated text-type classifier and the end-to-end system. The results highlight both the potential of our approach and the challenges of bridging the gap between controlled validation and real-world application.

4.1 Experimental Setup

The classifier was trained in a cloud-based environment on a single NVIDIA T4 GPU, using PyTorch Lightning and Weights & Biases for experiment tracking. The pipeline was implemented in Python 3.9, leveraging PaddlePaddle, OpenCV, and Pytesseract.

4.2 Classifier Performance on Curated Snippets

The classifier’s baseline performance was first established by evaluating it on the held-out validation set, which consists of 335 manually cropped text snippets. These snippets were sourced from real documents and intentionally represent a range of visual qualities, thereby providing a realistic benchmark for the classification task itself, isolated from any pipeline-induced artifacts.

The overall results of this evaluation are presented in **Table 5**, with a detailed breakdown of the predictions shown in the confusion matrix in **Figure 4**. The model achieved an overall accuracy of **74.6%**, a figure that highlights the inherent difficulty of reliably distinguishing between printed and handwritten Khmer text even under these controlled conditions.

A closer analysis of the errors in **Figure 4** reveals a key characteristic of the baseline model. The primary source of error is the misclassification of 74 handwritten snippets as printed, which is also reflected in the low Recall for the handwritten class (48.6%) in **Table 5**. This indicates a baseline bias in the model towards the ‘printed’ class. This behavior is likely because many instances of neat or simple handwriting in the dataset are visually similar to printed fonts, making them difficult for the model to differentiate without further contextual features. This baseline performance provides a crucial point of comparison for the end-to-end pipeline evaluation.

Table 5: Performance Metrics on Curated Validation Set

Class	Precision	Recall	F1-Score
Printed	70.9%	94.2%	80.9%
Handwritten	86.4%	48.6%	62.2%
Macro Avg.	78.6%	71.4%	71.6%
Accuracy		**74.6%**	

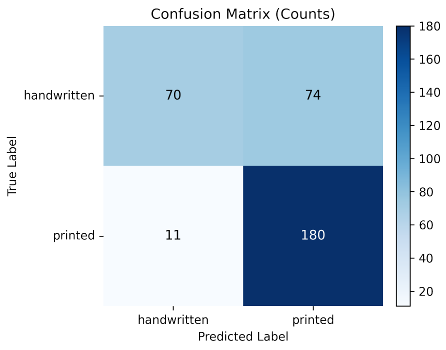


Figure 4: Confusion Matrix on the Curated Validation Set (335 Samples).

4.3 End-to-End Pipeline Performance and Challenges

While the classifier's performance on curated snippets is a strong proof-of-concept, its true effectiveness was evaluated by processing 20 complete, unprocessed birth certificates through the end-to-end pipeline. In this scenario, the layout analysis module first segments the document, and these automatically generated "wild" snippets are then fed to the text-type classifier.

The performance of the classifier within this live pipeline dropped significantly. The detailed confusion matrix from this real-world test is presented in **Table 6**, and the corresponding performance metrics are in **Table 7**.

Table 6: Confusion Matrix for Classifier in End-to-End Test

	Predicted Printed	Predicted Handwritten	Total
Actual Printed	241	166	407
Actual Handwritten	11	275	286

The results reveal a surprising and important finding. The classifier's end-to-end accuracy of **74.5%** is nearly identical to its baseline performance on the manually cropped validation set (74.6%). This consistency suggests that our model, having been trained on a realistic dataset of varied quality, is **robust to the additional noise and segmentation artifacts** introduced by the live pipeline. The analysis confirms that the primary bottleneck is not the pipeline's segmentation stage, but the fundamental difficulty of the classification task itself, which our 74.6% baseline accurately reflects.

Table 7: Performance Metrics for Text-type Classifier in End-to-End Pipeline Test

Class	Precision	Recall	F1-Score
Printed	95.6%	59.2%	73.1%
Handwritten	62.4%	96.2%	75.7%
Macro Avg.	79.0%	77.7%	74.4%
Accuracy		**74.5%**	

4.4 End-to-End Pipeline Performance and Error Analysis

When the classifier is integrated into the end-to-end pipeline, its overall accuracy remains stable at **74.5%** (**Table 7**). However, a comparison of the confusion matrices reveals a fascinating and critical shift in the model's error pattern.

While the baseline model's primary error was misclassifying 'handwritten' text as 'printed' (**Figure 4**), this pattern completely reverses in the live pipeline. As shown in **Table 6**, the predominant error becomes the misclassification of 166 'printed' snippets as 'handwritten'. This reversal strongly suggests that artifacts from the automated segmentation stage are the primary cause. When clean printed text is corrupted with noise or table lines, its visual features begin to resemble handwriting, causing the classifier to error. This demonstrates that while the overall accuracy is similar, the nature of the challenge changes significantly between curated and real-world pipeline conditions, providing a deeper insight into the model's behavior.

4.5 Downstream OCR Performance

To quantify the impact of our classification-first approach, we evaluated the performance of a standard OCR engine (Tesseract v5) on both printed and handwritten text snippets from our curated validation set. The primary metric used was the Character Error Rate (CER), a standard for evaluating OCR accuracy. CER is calculated as the Levenshtein distance between the predicted text and the ground truth, normalized by the length of the ground truth text:

$$\text{CER} = \frac{S + D + I}{N} \tag{6}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions, and N is the total number of characters in the ground truth. A lower CER indicates higher accuracy.

Table 8: Tesseract CER on Curated Text Snippets

Text Type	Snippets	CER (%)
Printed	191	43.71%
Handwritten	144	96.35%

The results clearly illustrate the necessity of a hybrid pipeline. While Tesseract is designed for printed text, it still yields a high CER of **43.71%**, underscoring the inherent difficulty of processing scanned, low-quality Khmer documents. As expected, its performance on handwritten text is extremely poor, with a CER of **96.35%**, rendering the output unusable. This confirms that a single OCR engine is insufficient for mixed-media documents and validates our approach of separating text types before recognition.

4.6 Comparative Analysis

Our classification-first pipeline represents a pragmatic compromise compared to other common strategies for mixed-media OCR. Unlike a monolithic **End-to-End Unified Model** (e.g., TrOCR) [4], which requires massive datasets and computational resources often unavailable for low-resource scripts like Khmer, our hybrid method is lightweight and modular. It is also more efficient than **Engine Blending**, which is computationally expensive as it processes each snippet with multiple engines and fails to solve the core problem if no specialized engine for a given text type exists. While the performance of our approach is dependent on the classifier's accuracy, it provides a practical and adaptable solution.

To provide a quantitative analysis of our approach's value, we compare its performance against a naive, single-engine strategy using the CER results from our validation set. A naive approach would apply Tesseract to all snippets, whereas our hybrid approach uses the classifier to route them first, as summarized in **Table 9**.

The data proves the necessity of our classification-first method. The naive approach is severely degraded by Tesseract's 96.35% CER on handwritten text, leading to an unusable overall CER of 65.42%. Our classifier acts as a critical filter, demonstrating that an intelligent, pre-recognition classification step is a mandatory component for achieving viable OCR on mixed-media documents.

4.7 Pipeline Output on Full Documents

To assess the system's real-world capabilities and provide a qualitative example of its operation, a complete birth certificate was processed through the entire pipeline. **Figure 5** provides a visual narrative of this process, illustrating the output at each critical stage and demonstrating a successful end-to-end run.

The process begins with the foundational layout analysis stages. First, the `PP-DocLayout_plus-L` model accurately identifies the document's high-level structure, separating text regions from the main table (**Fig. 5a**). Next, the specialized `RT-DETR-L` model processes the detected table region to precisely localize individual cells (**Fig. 5b**). The final classified output (**Fig. 5c**) is the culmination of the entire workflow and serves as a powerful proof-of-concept for our hybrid approach. This image visually confirms that the custom classifier has successfully labeled the machine-printed template text (blue) and the variable handwritten entries (red) on this document. This successful differentiation is the critical step that enables targeted and accurate text recognition in the final stage.

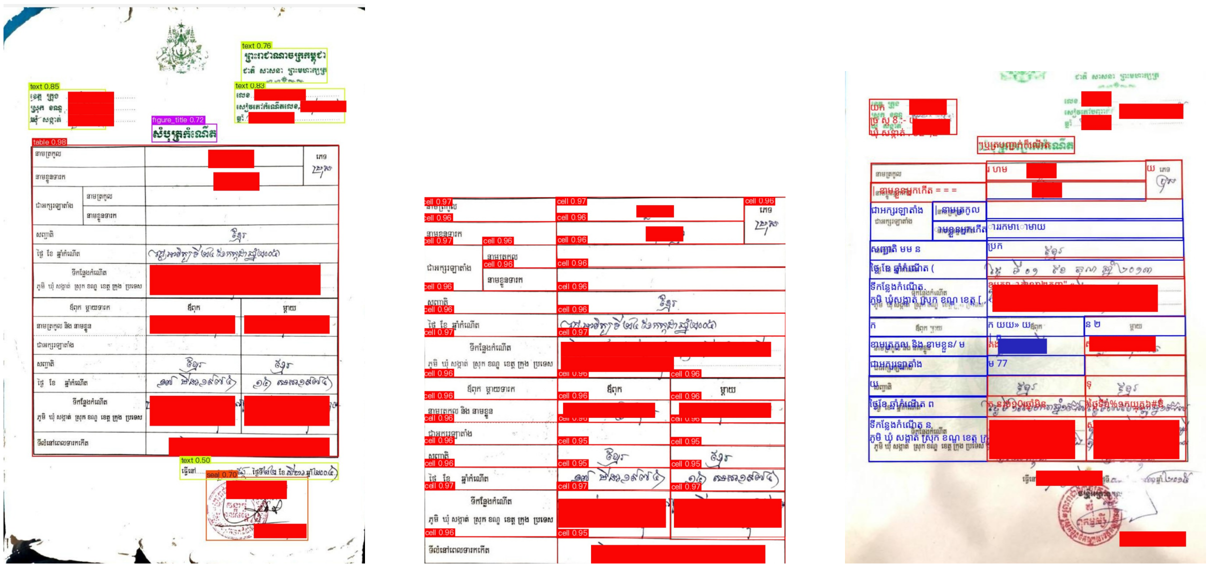
5 Conclusion and Future Work

In this paper, we presented a modular, resource-efficient OCR pipeline for semi-structured Khmer documents, successfully demonstrating the value of a classification-first approach. Our core contribution is the development and rigorous analysis of a lightweight text-type classifier.

Our key findings are threefold. First, we quantitatively proved the necessity of our approach: applying a standard OCR engine to

Table 9: Quantitative Comparison of Naive vs. Hybrid OCR Approach on the Curated Validation Set

Approach	Description	Resulting CER (%)
Naive Approach	Tesseract is applied to all 335 snippets (printed and handwritten). Performance is degraded by the engine's failure on handwritten text.	**65.42%**
Our Hybrid Approach	The classifier first separates text types. Tesseract is only applied to the 191 printed snippets. The 144 handwritten snippets are routed away, avoiding catastrophic errors.	**43.71%**



(a) Layout Detection (b) Table Cell Detection (c) Final Classified Output

Figure 5: Visual results from the end-to-end pipeline. (a) The layout analysis module correctly identifies high-level text and table regions. (b) A specialized model detects individual cells within the table. (c) The final output demonstrates successful classification, with printed text (blue) and handwritten entries (red) clearly distinguished.

the wrong text type results in a near-total failure (96.35% CER on handwritten text), making a pre-recognition classification step mandatory. Second, we established a realistic performance benchmark for this difficult task, showing our classifier achieves 74.6% accuracy on a validation set of varied-quality snippets. Third, we demonstrated the model's robustness, as its performance remains stable at 74.5% within the noisy, end-to-end pipeline. This indicates the primary challenge is the inherent visual complexity of the text, not segmentation artifacts.

Our work establishes a robust foundation for future development. The clear next steps are:

- **Robustness Enhancement:** While the model showed robustness, future work should still focus on improving the classifier by training it on an even more diverse dataset, including automatically segmented ("wild") snippets.
- **Handwritten Recognition Integration:** The pipeline's modular design allows for the seamless integration of a dedicated handwritten Khmer recognition model (e.g., a fine-tuned TrOCR) to achieve full end-to-end transcription.
- **Post-processing NLP:** The structured JSON output enables further natural language processing, such as developing modules for key-value pairing to link labels (like "Name") to their corresponding

transcribed text.

In conclusion, this project provides a significant step towards a practical solution for low-resource, mixed-media OCR, presenting not only a functional proof-of-concept but also a data-driven validation of the classification-first strategy and a clear, realistic roadmap for future research.

This paper was drafted by the authors. Language editing tools (e.g., AI-assisted writing support) were used only for grammar and phrasing; all content, results, and analyses are original and authored by the listed contributors.

References

[1] S. Patil and et al., "Enhancing optical character recognition on images with mixed text using semantic segmentation," *Journal of Sensor and Actuator Networks*, vol. 11, no. 4, p. 63, 2022.

[2] V. Nom, S. Bakkali, M. M. Luqman, M. Coustaty, and J.-M. Ogier, "Khmerst: A low-resource khmer scene text detection and recognition benchmark," 2024. Accepted at ACCV 2024, arXiv:2410.18277 [cs.CV], `https://doi.org/10.48550/arXiv.2410.18277`.

[3] R. Buoy, M. Iwamura, S. Srun, and K. Kise, "Toward a low-resource non-latin-complete baseline: An exploration of khmer optical character recognition," *IEEE Access*, vol. 11, pp. 128044–128060, 2023.

[4] M. Li, T. Lv, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei, "Trocr: Transformer-based optical character recognition with pre-trained models," *arXiv preprint arXiv:2109.10282*, 2021.

[5] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," 2015. arXiv:1507.05717 [cs.CV], `https://doi.org/10.48550/arXiv.1507.05717`.

[6] A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1314–1324, 2019.

[7] C. Cui, T. Sun, M. Lin, T. Gao, Y. Zhang, and et al., "Paddleocr 3.0 technical report," 2025. arXiv:2507.05595 [cs.CV].

[8] P. Sok and N. Taing, "Support vector machine (svm) based classifier for khmer printed character-set recognition," in *Proceedings of the 2014 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pp. 1–9, 2014.

[9] S. Long, X. He, and C. Yao, "Scene text detection and recognition: The deep learning era," 2018. arXiv:1811.04256 [cs.CV], `https://doi.org/10.48550/arXiv.1811.04256`.

[10] B. Annanurov and N. Noor, "A compact deep learning model for khmer handwritten text recognition," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 10, no. 3, pp. 584–591, 2021.

[11] D. Valy, M. Verleysen, S. Chhun, and J.-C. Burie, "A new khmer palm leaf manuscript dataset for document analysis and recognition: Sleukrith set," in *Proceedings of the 9th International Conference on Advances in Information Technology (IAIT)*, pp. 1–6, Nov. 2017.

[12] S. Born, D. Valy, and P. Kong, "Encoder-decoder language model for khmer handwritten text recognition in historical documents," in *Proceedings of the 14th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, pp. 234–238, 2022.