Techno-Science Research Journal V13 (2) (2025) P 33-40

Content list available at ITC

Techno-Science Research Journal

Journal Homepage: http://techno-srj.itc.edu.kh/

Techno-Science Research Journal

Word Spotting on Khmer Printed Documents

Hengly Em[1*], Dona Valy[1,2], Bernard Gosselin[3], Phutphalla Kong[1]

[1] *Department of Information and Communication Engineering,, Institute of Technology of Cambodia, Russian Federation Blvd., P.O. Box 86, Phnom Penh, Cambodia*

[2] *Research and Innovation Center, Institute of Technology of Cambodia, Russian Federation Blvd., P.O. Box 86, Phnom Penh, Cambodia*

[3] *Laboratory of Information, Signals and Artificial Intelligence, Faculty of Engineering, University of Mons, 31, Boulevard Dolez 7000 Mons, Belgium*

Received:08 August 2024; Revised: 10 September 2024; Accepted: 17 September 2024; Available online: 30 August 2025

Abstract: *Word spotting in Khmer printed documents presents a unique challenge due to the complexities of the Khmer script and the vast array of font styles employed. The scarcity of large, publicly available datasets further complicates this task. This work proposes a two-module approach for achieving accurate and efficient word spotting in Khmer documents. Separate datasets are utilized for text detection and recognition. The first module employs the state-of-the-art YOLOv8 model on a dataset of 10,050 text samples. The model's performance is evaluated using the F1 score, a metric that balances precision and recall in locating text. The second module leverages the fine-tuned Transformer-based TrOCR model for recognition, trained on 22,567 labeled words, with recognition accuracy measured by the Character Error Rate (CER). The first module achieves an impressive F1 score of 0.987 in locating Khmer words within documents. The second module's TrOCR model results in a CER of 8.41%. By overcoming script and font challenges through focused datasets and advanced models, this approach demonstrates potential for improving document processing and information retrieval for the Khmer language.*

Keywords: Khmer printed documents, Word spotting, YOLOv8, TrOCR, Information retrieval

1. INTRODUCTION

The preservation and analysis of historical and contemporary documents form the backbone of various disciplines. These documents, often written in diverse languages, offer invaluable insights into history, culture, and societal evolution. However, for languages with unique writing systems and a vast array of fonts, extracting information from scanned documents remains a significant hurdle. This challenge is particularly acute for the Khmer language, spoken by over 16 million people primarily in Cambodia.

The task of word spotting which involves automatically locating and recognizing individuals words within digital images of documents, presents a critical barrier to unlocking the wealth of information contained within Khmer language materials. This challenge stems from several factors. The Khmer script itself possesses intricate visual characteristics, with characters often composed of stacked and connected components. Furthermore, unlike many other languages, Khmer written text lacks spaces between words, making traditional segmentation methods ineffective for locating individual words. This, along with the font variations, significantly complicates the automated recognition process. The scarcity of large, publicly available datasets specifically designed for Khmer word spotting adds another layer of difficulty, hindering the development of robust and accurate systems.

Several studies have explored word spotting in document images across various linguistic contexts, with deep learning models proving particularly effective for this task. For instance, Busta et al. [2] proposed a method for scene text localization and recognition, which achieved state-of-the-art accuracy on the ICDAR 2013 and 2015 datasets and was faster than competing methods. The International Conference on Document Analysis and Recognition (ICDAR) provides benchmark datasets for various text recognition challenges. The ICDAR 2013 and ICDAR 2015 datasets are widely used in the field of scene text

[*] Corresponding author: Em Hengly
E-mail:emhengly@gmail.com; Tel: +855-78 476 343

detection and recognition, containing images of text inatural scenes with annotations for both text regions and corresponding transcripts, offering diverse challenges in terms of fonts, orientations, and complex backgrounds. Haifeng & Siqi [4] introduced deep learning object detection networks and YOLOv2 for natural scene text detection, transforming multi-object detection problems into two classification problems and achieving a detection speed of 0.105 seconds per image. Alghyaline [1] proposed a Printed Arabic Optical Character Recognition approach (PAOCR) using a You Only Look Once object detector, along with four techniques and a Hunspell library. This system achieved a 95.7% Character Recognition Rate compared to existing systems. Yoshihashi et al. [7] presented a Context-Free TextSpotter model, a lightweight, low-computing text spotting method for scene-text spotting on mobile platforms. It achieved real-time text spotting on a GPU with only three million parameters, with acceptable transcription quality degradation. Fang et al. [3] proposed an autonomous, bidirectional, and iterative ABINet++ for scene text spotting, addressing limitations in language models due to implicit modeling, unidirectional feature representation, and noise input. The method achieved superior performance in low-quality images and experiments in English and Chinese, improving accuracy and speed compared to attention-based recognizers. Li et al. [5] proposed an end-to-end text recognition approach, TrOCR, using pre-trained image and text Transformer models, which outperformed current state-of-the-art models in printed, handwritten, and scene text recognition tasks.

While these studies showcase significant progress in word spotting for various languaes, the complex nature of the Khmer script necessitates a tailored approach.

Our approach tackles this challenge with a novel two-module deep learning system. The first module leverages a cutting-edge object detection model, trained on a comprehensive dataset of Khmer text samples. This module effectively pinpoints individual words within documents. Following successful detection, the second module utilizes a powerful Transformer-based recognition model fine-tuned for the Khmer script. This fine-tuning enables the model to accurately recognize the identified words.

The following sections will detail our methodology (Section 2), present the experimental results (Section 3), discuss our findings (Section 4), and conclude the study (Section 5).

2. METHODOLOGY

This section details the development of our two-module deep learning system for word spotting in Khmer documents. We begin by describing the creation and characteristics of our dataset (Section 2.1), followed by the specific configurations of each module (Section 2.2 and 2.3).

2.1 Datasets

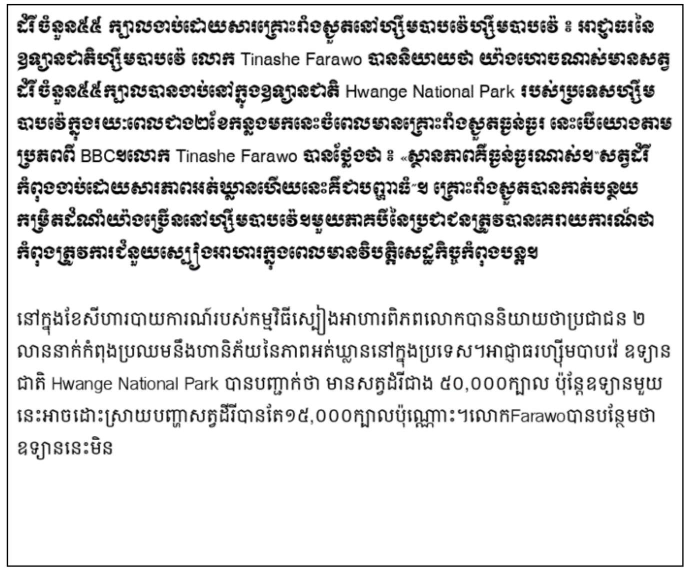


Fig. 1. Text Detection Dataset Sample

2.1.1 Text Detection Dataset

For our text detection task, we opted to create a custom dataset. We sourced Khmer text files from a publicly available repository on GitHub [12] . This dataset comprises 10,050 samples, each synthetically generated as an image containing Khmer text with a font size of 20 as shown in **Fig. 1**. To ensure a comprehensive representation of Khmer script variations and enhance the model's ability to generalize to unseen data, the text within these images varies across ten different font types.

The dataset creation process involved transforming it into image files accompanied by corresponding annotation files in XML format. These annotations delineate the boundaries of individual words within the images. The diversity in font types and sizes ensures a comprehensive representation of Khmer script variations, making the dataset suitable for training and evaluating a word spotting model.

2.1.2 Tex Recognition Dataset

To train the text recognition model, we constructed a dedicated dataset by leveraging the output from the text detection module. We utilized the annotation files generated during text detection dataset creation (Section 2.1.1), which typically contain bounding box coordinates for each word within the image (See **Fig. 1**). Based on these coordinates, we cropped individual word images from the original text detection dataset images. This process essentially extracts each word as a separate image for training the recognition model.

The resulting text recognition dataset comprises a total of 28,208 word images. To simplify the training process and data management, we opted to store both the image name and the corresponding text label (the actual Khmer word) within a single text file (See **Fig. 2**). Each line in this file follows a specific

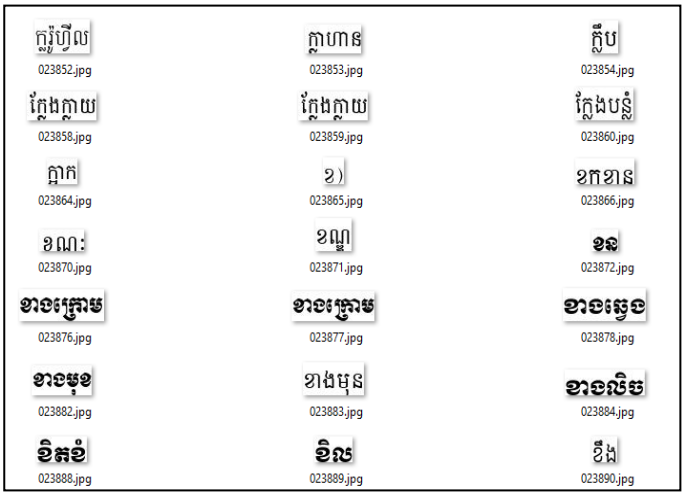


Fig. 2. Text Recognition Dataset Sample

format, typically separating the image filename and word label with a space.

2.2 Text Detection Module

In this study, we utilized the You Only Look Once v8 (YOLOv8) deep learning model, which excels in real-time object detection and has been trained on numerous applications. YOLOv8 represents the advancement in the YOLO family [13] of detectors. Although YOLOv8 [11] was introduced by Ultralytics in 2023, we have adapted this algorithm to train our model for Khmer text detection. Our model is structured based on the YOLOv8 algorithm, with its architecture depicted in Fig. 3.

The detail of model architecture consists of backbone, neck, and head we followed from [10] (See **Fig. 3**). In the following subsections, we introduce the design concepts of each part of the modelarchitecture, and the module of different parts.

Backbone: The YOLOv8 model begins with the CSPDarknet53 backbone, which is responsible for extracting features from the input image of Khmer text. CSPDarknet53 utilizes the Cross Stage Partial (CSP) [8] architecture, which divides the feature map into two sections. One section undergoes convolution operations, while the other bypasses these operations and is later concatenated with the processed section. This design improves information flow and gradient propagation during training, making it both efficient and effective. The backbone consists of a series of convolutional layers with residual connections, batch normalization, and Leaky ReLU activation functions. These layers progressively extract hierarchical features, starting with basic shapes and textures and advancing to more complex patterns and structures found in the Khmer text.

Neck: The neck acts as a bridge between the backbone and the head. The neck of the YOLOv8 model employs the Path Aggregation Network (PANet) [9] to bridge the backbone and the head. PANet enhances the feature hierarchy by introducing a bottom-up path augmentation that complements the top-down pathway typically used in Feature Pyramid Networks (FPN). This dual-path approach is crucial for preserving spatial information, which is essential for accurately detecting small and intricate details such as individual Khmer words. PANet uses adaptive feature pooling to aggregate features from different stages of the backbone and employs feature fusion techniques to combine low-level and high-level features. This results in a richer and more detailed feature representation, which significantly improves the model's ability to accurately detect and localize Khmer words in the text image.

Head: The head of the YOLOv8 model is responsible for making predictions, specifically the bounding boxes of each Khmer word in the text image. The input image is divided into a grid, with each cell predicting multiple bounding boxes. For each bounding box, the model outputs coordinates, objectness scores, and class probabilities. In the context of detecting Khmer words, these predictions include the location and size of each word's bounding box. Non-Maximum Suppression (NMS) is applied to eliminate redundant bounding boxes, retaining only the most confident detections This ensures precise and accurate localization of each word.

Loss: Loss: The loss calculation process consists of two parts: the sample assignment strategy and the loss calculation. Both the sample assignment strategy and the loss calculation followed the formula from [10].

YOLOv8 adopts a sample assignment strategy similar to TOOD's TaskAlignedAssigner as shown in Eq.1. This method determines which predicted boxes correspond to actual objects (positive samples). Instead of relying solely on IoU overlap, it considers a combined score based on both the predicted classification confidence and the predicted box's quality (measured by regression metrics like CIoU). Boxes with higher combined scores are more likely to be selected as positive samples.

$$t = s^{\alpha} + u^{\beta} \quad \text{(Eq.1)}$$

where s is the predicted score corresponding to the ground truth, and u is the IoU of the predicted bounding box and the ground truth bounding box. α and β are hyperparameters that control the relative importance of the classification confidence s and the box quality u.

The loss calculations of YOLOv8 has classification and regression branches, where the classification branch uses Binary Cross-Entropy (BCE) Loss, and the equation is shown as below:

$$l_n = -w_n [y_n \log x_n + (1 - y_n) \log(1 - x_n)] \quad \text{(Eq.2)}$$

where w_n is the weight, y_n is the groud truth value, and x_n is the predicted value.

The regression branch uses Distribute Focal Loss (DFL) and Complete IoU (CIoU) Loss, where DFL is used to expend the

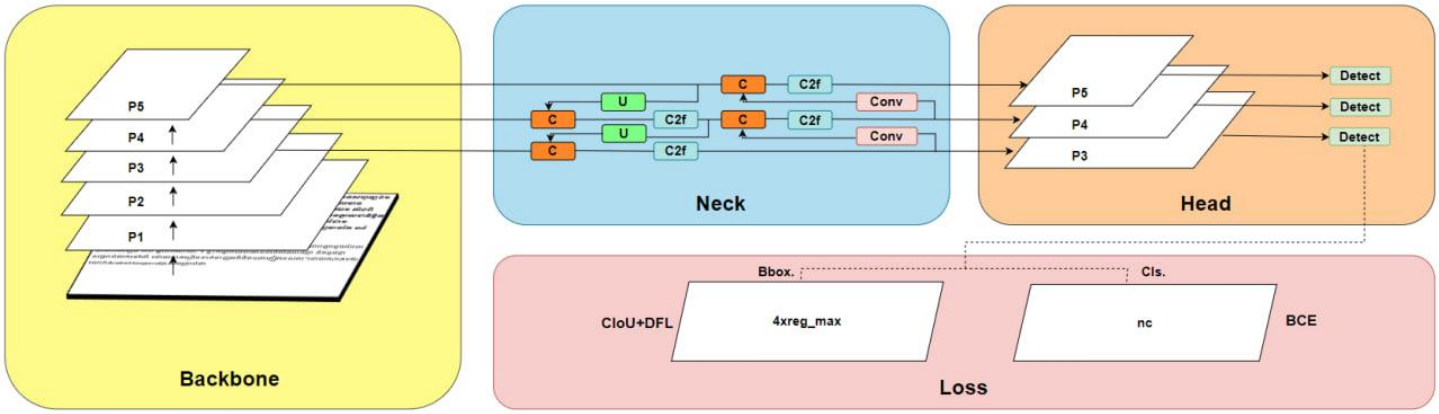


Fig. 3. YOLOv8 Model Architecture [10]

probability of the value around the object *y*. Its equation is shown as follows:

$$DFL(\mathcal{S}_n, \mathcal{S}_{n+1}) = -((y_{n+1} - y)\log(\mathcal{S}_n) + (y - y_n)\log(\mathcal{S}_{n+1})) \quad \text{(Eq.3)}$$

where the equation of $\mathcal{S}_n$, $\mathcal{S}_{n+1}$ are shown below:

$$\mathcal{S}_n = \frac{y_{n+1} - y}{y_{n+1} - y_n}, \mathcal{S}_{n+1} = \frac{y - y_n}{y_{n+1} - y_n} \quad \text{(Eq.4)}$$

CIoU Loss adds an influence factor to Distance IoU (DIoU) Loss by considering the aspect ratio of the prediction and the ground truth bounding box. The equation is shown as below:

$$CIoU_{Loss} = 1 - IoU + \frac{Distance_2^2}{Distance_c^2} + \frac{v^2}{(1 - IoU) + v} \quad \text{(Eq.5)}$$

where *v* is the parameter that measures the consistency of the aspect ratio, defined as below:

$$v = \frac{4}{\pi^2}\left(\tan^{-1}\frac{w^{gt}}{h^{gt}} - \tan^{-1}\frac{w^P}{h^P}\right)^2 \quad \text{(Eq.6)}$$

Where *w* is the weight of the bounding box, *h* is the height of the bounding box.

2.2 Text Recognition Module

For the recognition module, we fine-tuned the TrOCR (Transformers for Optical Character Recognition) [5] model to recognize Khmer text from images. TrOCR is state-of-the-art OCR model based on the transformer architecture, pre-trained on large datasets of text images and fine-tuned for specific OCR tasks.

TrOCR combines the strengths of both Vision Transformer (ViT) and traditional transformer models to achieve high accuracy in text recognition tasks. The model architecture consists of two main components (see **Fig. 4**):

Vision Encoder: The Vision Transformer (ViT) serves as the encoder in the TrOCR model. The standard Transformer take *1D*-dimensional sequence of token embeddings as its input. To adapt this for *2D*-dimensional images, we transform the image $x \in \mathbb{R}^{H \times W \times C}$ into a sequence of flattened 2D patches $x_P \in \mathbb{R}^{N \times (P^2 \cdot C)}$. Here, (H, W) represents the dimension of the original image, C denotes the number of channels, (P, P) is the size of each image patch, and $N = HW/P^2$ is the resulting number of patches, which also determines the effective input sequence length for the Transformer. Subsequently, the patches are flattened into vectors and linearly projected to *D*-dimentional vectors through all of its layers.

The special token "[CLS]" is retain utilized in image classification tasks. This "[CLS]" token aggregates information from all the patch embeddings to represent the entire image. Additionalluy, the distillation token is including in the input sequence when employing DeiT [5] pre-trained models for encoder initialization. This enables the model to learn from the teacher model. The 1D position embeddings are added to the patch embeddings to retain positional information.

Text Decoder: The text decoder block in TrOCR is designed to generate recognized tex sequences from visual features by the Vision Encoder. The text decoder uniquely integrates an "encoder-decoder attention" mechanism between the multi-head self-attention and the feed-forward network. This module allocates attention differently by utilizing the keys (K) and values (V) from the encoder's output, while the queries (Q) are derived from the decoder's hidden states. The attention mechanism computes how well the queries align with the encoder's keys, and uses these alignments to weigh and aggregate the values, as described by the formula in (Eq.7).The self-attention (Eq.8) mechanism in the decoder utilizes attention masking to ensure that, during traing, it does not access more information than it would during prediction. Given that the decoder's output is shifted one position relative to its input, the attention mask ensures that the output at position *i* only attends

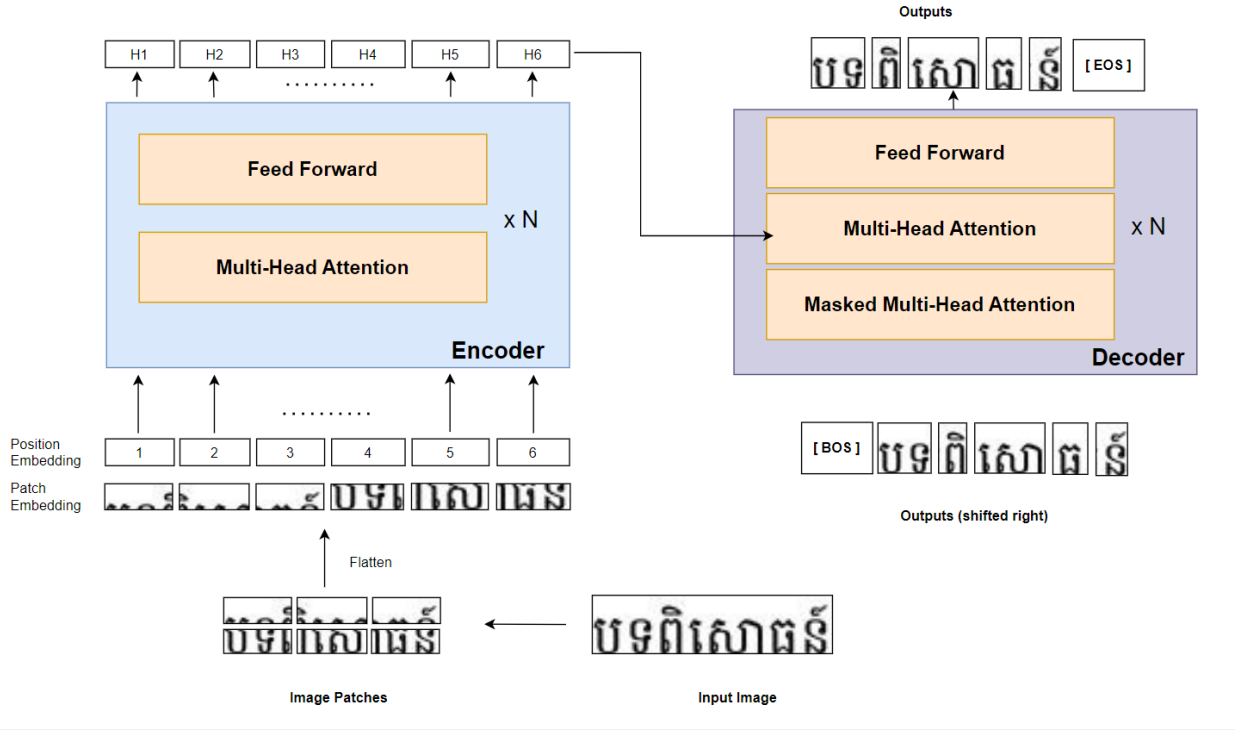


Fig. 4. Khmer TroOCR Model Architecture [5]

to the predicting outputs, which the inputs at positions less than *i*. The hidden states (Eq. 9) from the decoder are projected via a linear layer, converting them from the model's dimensionality to the vocabulary size V. These projections are then used to compute probabilities for each token in the vocabulary using the softmax function (Eq.10). Finally, beam search is employed to determine the most probable sequence of tokens for the output.

$$Q = h^{l-1}W^Q,\ K = h^{l-1}W^K,\ V = h^{l-1}W^V \quad \text{(Eq.7)}$$

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}} + M)V \quad \text{(Eq.8)}$$

where h^{l-1} is the input to the *l*-th decoder layer (hidden states from the previous layer). W^Q, W^K, W^V are the learned weight matrices for queries, keys, and value, respectively. d_k is the dimension of the keys.

$$h_i = Proj(Emb(token_i)) \quad \text{(Eq.9)}$$

$$\sigma(h_{ij}) = \frac{e^{h_{ij}}}{\sum_{K=1}^{V} e^{h_{ik}}} \ for\ j = 1,2,3,\dots,V \quad \text{(Eq.10)}$$

where h_{ij} is the logit corresponding to token *j* at position *i*. *V* is the size of the vocabulary.

3. EXPERIMENTS and RESULTS

3.1 Dataset and Evaluation Metric

The datasets utilized for both text detection (Section 2.1.1) and text recognition (Section 2.1.2) modules were meticulously constructed. To facilitate model training and evaluation, these datasets were divided into training, validation, and testing subsets following a stratified random sampling approach which shown in **Table 1** and **Table 2**.

For text detection module, we utilized F1-score (Eq.11) as the evaluation metric. The F1-score equation is as follows:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad \text{(Eq.11)}$$

The equations of *Precision* and *Recall* are shown below:

$$Recall = \frac{T_p}{T_p + F_N}, Precision = \frac{T_p}{T_p + F_P} \quad \text{(Eq.12)}$$

where *Recall* measures the proportion of actual text regions in an image that are correctly detected by the model, *Precision* measures the proportion of detected text regions that are actually correct. T_p is a correctly detected text region, F_N is a text region that exists in the image but was not detected by the system, and

F_P is a region that was incorrectly identified as text by the system.

For text recognition module, we employed Character Error Rate (CER) (see Eq.13) to evaluate the performance of a Khmer text recognition. It measure the accuracy of character-level recognition by comparing the recognized text to the ground truth. CER is calculated as follows:

$$CER = \frac{Substitutions+Deletions+Insertions}{Total\ Characters} \times 100\% \qquad \text{(Eq.13)}$$

where *Substitutions* is an incorrect characters in the recognized text compared to the ground truth, *Deletions* a character missing from the recognized text, *Insertions* is an extra characters present in the recognized text and *Total Characters* is a total number of characters in the ground truth text.

3.2 Experiments Setup

All experiments in this paper were conducted on a single computer with the following hardware configuration: NVIDIA RTX 3090, PyTorch deep learning frame-work, Python 3.10 programming language, and CUDA 11.8 GPU accelerator. The model parameter settings for both modules are shown in **Table 1** and **Table 2**, respectively. Fine-tuning was applied to parameters such as learning rate, batch size, number of epochs, and the architecture-specific layers to improve the detection and recognition accuracy of the models for Khmer text.

Table 1. Text detection model parameters

Parameter types	Parameter settings
Image size	640x640
Batch	24
Epochs	100
Learning rate	0.0001
Optimizer	Auto
Training set	75%
Testing set	15%
Validation set	15%

Table 2. Text Recognition model parameters

Parameter types	Parameter settings
Image size	384x384
Batch	24
Epochs	30
Learning rate	0.005
Optimizer	Adam
Training set	80%
Testing set	20%

Table 3. Comparative Performance Metrics of YOLO Models for Text Detection

Model	Precision	Recall	F1-score
YOLOv1	0.956	0.775	0.856
YOLOv5l	0.986	0.982	0.984
YOLOv8l	0.989	0.986	0.987

3.3 Experiments Results

3.3.1 Text Dectection Results

Table 3 provides a detailed comparison of the performance metrics for three YOLO models—YOLOv1, YOLOv5l, and YOLOv8l—used in text detection. In experiments conducted, YOLOv1, after adjustments inspired by the original YOLO model, achieved a precision of 0.956, recall of 0.775, and an F1-score of 0.856. While the high precision indicates effective identification of text instances, the lower recall suggests that many text instances were missed, resulting in a moderate F1-score. YOLOv5l demonstrated significant improvements, with a precision of 0.986, recall of 0.982, and an F1-score of 0.984, indicating a better balance between identifying and correctly detecting text instances. The YOLOv8l model exhibited the highest performance, with a precision of 0.989, recall of 0.986, and an F1-score of 0.987, making it the most effective model for text detection among the three, due to its superior balance of precision and recall. The results for YOLOv8l are depicted in **Fig. 5**, illustrating its effectiveness in text detection tasks.

3.3.2 Text Recognition Results

The TrOCR-small-printed model's performance was assessed with various batch sizes, learning rates, and epochs, as detailed in **Table 4**. With a batch size of 16, a learning rate of $5x10^{-5}$, and 30 epochs, the CER was high at 45.13%, indicating suboptimal performance. Increasing the learning rate to $5x10^{-3}$ while maintaining the same batch size and number of epochs significantly improved the CER to 9.39%. Further refinement, using a larger batch size of 24 and extending the epochs to 40, resulted in the lowest CER of 8.41%. These findings highlight that both a higher learning rate and a larger batch size with more training epochs substantially enhance the model's accuracy, emphasizing the critical role of hyperparameter tuning in optimizing performance. The output from the model is illustrated in **Table 5**.

Table 4. Comparative Performace Metrics of TrOCR Models for Text Recognition

Model	Batch Size	Learning Rate	Epochs	CER(%)
TrOCR-small-printed	16	$5x10^{-5}$	30	45.13
	16	$5x10^{-3}$	30	9.39
	24	$5x10^{-3}$	40	8.41

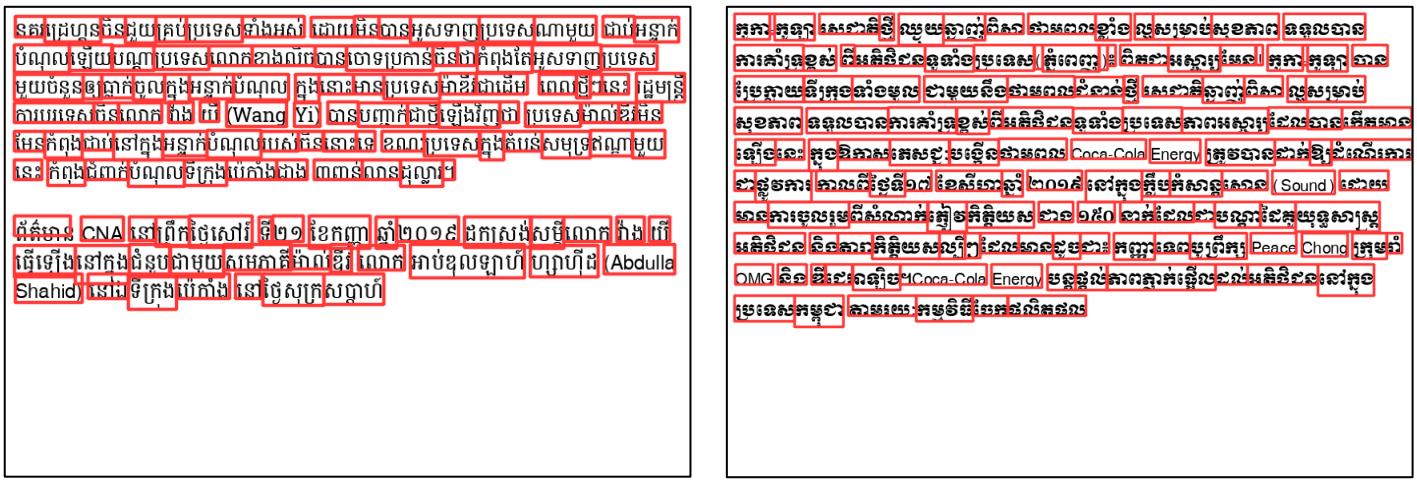
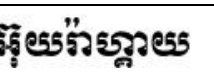
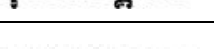
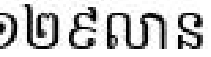
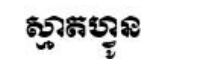
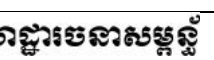


Fig. 5. Predicted results from YOLOv8l model

Table 5. Comparison of Recognized Text with Ground Truth of our Fine-tuned TrOCR model

Images	Recognized Text	Ground Truth Text
	អ្នកវិភាគ	អ្នកវិភាគ
	អ៊ុយរ៉ាហ្គាយ	អ៊ុយរ៉ាហ្គាយ
	១២៩លាន	១២៩លាន
	ស្មាតហ្វូន	ស្មាតហ្វូន
	ហោដ្ឋារចនាសម្ព័ន្ធ	ហោដ្ឋារចនាសម្ព័ន្ធ
	វិធានការព្យាបាល	វិធានការព្យាបាល

3. DISCUSSION

This study effectively combined YOLOv8l for text detection and fine-tuned TrOCR for text recognition to enhance word spotting for Khmer text. YOLOv8l demonstrated high performance with an F1-score of 0.987, accurately locating text regions across various images, as shown in **Fig. 5**. YOLOv8l was chosen for its superior balance of precision and recall, making it highly effective in detecting text regions in diverse anchallenging conditions. Its ability to handle multiple scales and maintain high accuracy even with complex scripts and varied text orientations makes it an ideal choice for text detection tasks, particularly for Khmer text images.

Following detection, the text recognition module, based on a fine-tuned TrOCR model, transcribed the detected text regions. Originally, the TrOCR tokenizer did not support the Khmer language, leading to poor recognition accuracy. To address this, the NllbTokenizer from the "facebook/nllb-200-distilled-600M" model was integrated. This tokenizer was specifically chosen for its advanced multilingual capabilities and its proven effectiveness in handling low-resource languages like Khmer. The NllbTokenizer's comprehensive vocabulary and sophisticated language modeling significantly improved the recognition accuracy for Khmer text, demonstrating its superiority in processing the unique script and linguistic nuances of the Khmer language.

The text recognition module achieved a CER of 8.41%, as shown in **Table 5**. The majority of target words were accurately recognized, with correct recognitions highlighted in green. Incorrect recognitions, marked in red, occurred primarily in images where the recognized text diverged slightly from the ground truth, reflecting minor errors or incomplete transcriptions. This demonstrates the effectiveness of the text recognition module in accurately handling Khmer text, despite some limitations.

4. CONCLUSIONS

In this study, we tackled the challenge of word spotting in Khmer printed documents, addressing the complexities of the Khmer script and the diverse array of font styles. Our two-module approach, integrating state-of-the-art models for text detection and recognition, demonstrated exceptional accuracy and efficiency.

The first module utilized the YOLOv8 model, applied to a dataset of 10,050 text samples, achieving an impressive F1 score

of 0.987 for accurately locating Khmer words within documents. This high F1 score reflects the model's excellent balance between precision and recall, underscoring its effectiveness in detecting text instances with minimal errors.

The second module leveraged the fune-tuned of advanced Transformer-based TrOCR model for text recognition. Trained on 22,567 labeled words, TrOCR effectively employed the Transformer architecture for both image understanding and wordpiece-level text generation, achieving a CER of 8.41%. This highlights the model's capability in accurately recognizing and transcribing Khmer words, despite the script's complexities and font variations.

Our approach, combining focused datasets with cutting-edge models, successfully overcame the challenges posed by the Khmer script and font diversity. The integration of YOLOv8 for detection and TrOCR for recognition offers a robust solution for improved document processing and information retrieval in the Khmer language. This work provides a strong foundation for future advancements in Khmer text recognition, enhancing the accessibility and usability of Khmer printed materials.

ACKNOWLEDGMENTS

This research study is supported by ARES with AI-ITC R4 Project.

REFERENCES

[1] S. Alghyaline, “A Printed Arabic Optical Character Recognition System using Deep Learning,” Journal of Computer Science, vol. 18, no. 11, pp. 1038–1050, Nov. 2022, doi: 10.3844/jcssp.2022.1038.1050.

[2] M. Bušta, L. Neumann and J. Matas, "Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 2223-2231, doi: 10.1109/ICCV.2017.242.

[3] S. Fang, Z. Mao, H. Xie, Y. Wang, C. Yan, and Y. Zhang, “ABINet++: Autonomous, Bidirectional and Iterative Language Modeling for Scene Text Spotting,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 6, pp. 7123–7141, Jun. 2023, doi: 10.1109/tpami.2022.3223908.

[4] D. Haifeng and H. Siqi, “Natural scene text detection based on YOLO V2 network model,” Journal of Physics Conference Series, vol. 1634, no. 1, p. 012013, Sep. 2020, doi: 10.1088/1742-6596/1634/1/012013.

[5] M. Li et al., “TrOCR: Transformer-Based Optical Character Recognition with Pre-trained Models,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 11, pp. 13094–13102, Jun. 2023, doi: 10.1609/aaai.v37i11.26538.

[6] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[7] R. Yoshihashi, T. Tanaka, K. Doi, T. Fujino, and N. Yamashita, “Context-Free TextSpotter for Real-Time and Mobile End-to-End Text Detection and Recognition,” in Lecture notes in computer science, 2021, pp. 240–257. doi: 10.1007/978-3-030-86331-9_16.

[8] W. Chien-Yao, M. L. Hong-Yuan, W. Yueh-Hua, C. Ping-Yang, H. Jun-Wei, and Y. I-Hau, “CSPNet: A New Backbone that can Enhance Learning Capability of CNN,” IEEE Conference Proceedings, vol. 2020, pp. 1571–1580, Jan. 2020, [Online]. Available: https://jglobal.jst.go.jp/en/detail?JGLOBAL_ID=202002235026591127

[9] Liu, S., Qi, L., Qin, H., Shi, J. & Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, 8759–8768 (2018)

[10] R.-Y. Ju and W. Cai, “Fracture detection in pediatric wrist trauma X-ray images using YOLOv8 algorithm,” Scientific Reports, vol. 13, no. 1, Nov. 2023, doi: 10.1038/s41598-023-47460-7.

[11] Ultralytics, “GitHub - ultralytics/ultralytics: NEW - YOLOv8 🚀 in PyTorch > ONNX > OpenVINO > CoreML > TFLite,” GitHub. https://github.com/ultralytics/ultralytics

[12] Phylypo, “GitHub - phylypo/segmentation-crf-khmer: Word segmentation using Conditional Random Fields (CRF) for Khmer document,” GitHub. https://github.com/phylypo/segmentation-crf-khmer

[13] J. Nelson, “What is YOLO? The Ultimate Guide [2024],” Roboflow Blog, Jul. 22, 2024. https://blog.roboflow.com/guide-to-yolo-models/