A Two-Stage Approach to Khmer Scene Text Recognition using Faster R-CNN and TrOCR

Sethisak San **Mitona Chan** **Eklim Sek**
School of Digital Technology, American University of Phnom Penh, Phnom Penh, Cambodia
2023471san@aupp.edu.kh

Abstract

Khmer scene text recognition presents significant challenges due to the complex structure of Khmer's script, which includes stack consonants, special diacritics, and a lack of consistent word spacing. In this work, we propose a two-stage approach to detect and recognize Khmer scene text. We first utilize Faster R-CNN, a deep learning-based object detection model, to identify text region coordinates in various images. Subsequently, the detected regioned are processed by TrOCR, a Transformer-based model, for character recognition. Our model is trained and evaluated on a combination of real-world and synthetic data, including the KhmerST dataset, the 62k Khmer Printed Dataset, and the Khmer Annotation dataset. Our approach achieves strong detection performance, with a high recall rate of 91.4% and a precision of 70.1%, indicating robust text localization. The recognition stage yields a Character Error Rate (CER) of 18.3%. On a manually curated real-world test set, the detection model achieves a precision of 62.2%, recall of 55.5%, demonstrating reasonable generalization. Our analysis shows that while the detector effectively localize the text, recognition errors are primarily linked to the script's inherent complexity and inconsistencies in text-line segmentation.

Keywords: *Khmer text recognition; scene text detection; OCR; Faster R-CNN; TrOCR; deep learning; computer vision*

1 Introduction

Text detection from natural images is not an easy task, particularly for complex scripts such as Khmer. Text may appear in different forms, including printed text, handwritten text, and scene text. Printed text is easier to detect because it is often shown in clear space and font size. On the other hand, handwritten style is more difficult due to the styles, sizes, and alignment of

Figure 1. Printed text display in clean background with consistent space and font use.

the word. Regarding the text of the scene, it may appear in various sizes, fonts, orientations, and backgrounds in real-world scenarios, as illustrated in Figure 1-3. Furthermore, the characteristics of the Khmer language, stacked consonants, diacritical marks, and the absence of obvious word spaces make it even more difficult to detect [1] . Therefore, Khmer scripts present unique challenges that make scene text recognition far more difficult than widely studied languages like English or Chinese [2]. Moreover, unlike Latin scripts, Khmer is not space-separated between words consistently, which complicates segmentation. Such language features combined with real-world issues such as low-quality images, non-uniform lighting, noisy environments, and inconsistent fonts pose very challenging tasks to effective text detection and recognition [1, 3].

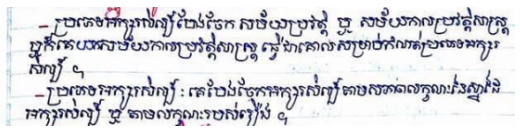

Figure 2. Handwritten shows in different writing styles, sizes, spacing, and stroke.

Early OCR engines, like Tesseract [4], have not been very successful when applied to Khmer, while the TRBA model [2], in which ResNet [5] is combined for feature extraction, BiLSTM [6] for sequence modeling, and Transformer-based attention [7], achieves significantly better performance [2]. Most of the existing work has been done on printed text or synthetic data, whereas

a. lighting variation b. shadow and reflection c. complex background

Figure 3. Scene text affect by lighting variation, shadows, reflections, and complex background.

natural scene images include variations such as lighting, blur, and background noise, in which can significantly impact the OCR performance.

To tackle these difficulties, our project comes up with a two-stage approach. In the first stage, the model learns intermediate concepts, which then serve as the feature space for the second stage [8]. Specifically, we first employ an object detection method named Faster R-CNN (Region-based Convolutional Neural Network) [9] for detecting and localizing text areas in real images, in which make use of FPN (Feature Pyramid Network) [10]. In the second stage, we utilize TrOCR, a Transformer-based OCR model, to detect and convert recognized text areas into readable digital text that can be processed by computers.

2 Related Work

2.1 Seq2Seq with Attention on Khmer OCR

Khmer Optical Character Recognition (OCR) conducted by [11] introduced one of the pioneering end-to-end sequence-to-sequence (Seq2Seq) frameworks employing attention mechanisms for recognizing printed text. The encoder architecture comprised convolutional residual blocks in conjunction with Gated Recurrent Units (GRUs) [12], whereas the decoder utilized an attentive GRU to produce character sequences directly from complete text-line images, eliminating the need for segmentation or manually engineered feature extraction. In order to train the model, the researchers created approximately 92,000 synthetic images sourced from the Asian Language Treebank (ALT) [13] by employing seven widely used Khmer fonts, alongside further augmentations including noise, dilation, and rotation. Evaluation on test dataset with 3000 images shows the model achieved a Character Error Rate (CER) of only 1% and a Word Error Rate (WER) of 9%, in which surpassed the Tesseract OCR that's reported to achieved a CER of 3% and a WER of 26% [11].

Building on this foundation, Buoy et al. [14] further improved this by increased the dataset to over three million synthetic text-line images in various Khmer fonts. Unlike the earlier research, this study applied with augmentation by adding noise, blur, and complex background during training in order to make the model more generalized in real life situations. The result with the dataset of 6400 augmentation validation images achieved a CER of 0.7% while Tesseract only achieved with CER of 35.9%. Even with de-augmented data, the new model still have better accuracy with 0.24% vs 1.6% for Tesseract. The main difference between these two studies is in scale and robustness; with the 2021 study showing that Seq2Seq with attention could work for Khmer OCR, while the 2022 study improved it by enlarging the dataset, using more complex augmentation, and achieving top performance in noisy and font-variable conditions [14].

2.2 Khmer STDR

Previous studies on Khmer Optical Character Recognition (OCR) mostly focused on printed text with sequence-to-sequence (Seq2Seq) models with attention. However, these systems only worked well for printed text in most situations. They did not address recognizing scene text, which is when text is found in natural settings with distortions, blur, and background noise. Nom et al. [1] introduced KhmerST, the first dataset for Khmer scene text. This dataset contained 1,544 images that the researchers have annotated; 997 indoors and 547 outdoors. Unlike fake printed datasets, KhmerST shows real-world differences, including raised or flat text, poor lighting, far or partly hidden signs, and complex backgrounds. Each image is annotated with polygonal bounding boxes at text-line level, producing 3,463 cropped regions for recognition tasks. The authors also did experiment on both detection and recognition task with the YOLO family and the Transformer model. Several YOLO architectures were fine-tuned; among them, YOLOv8 achieved the best result for the text detection task, with a recall of 0.832 and mAP50 of 0.899. For text recognition, the researchers investigated the performance of two transformer models; TrOCR and Tesseract. TrOCR achieved a Character Error Rate (CER)

of 1.01% and Word Error Rate (WER) of 2.24%, while Tesseract fall behind with 1.30% CER and 4.75% WER.

2.3 Cross-Lingual Learning for Khmer STDR

In high-resource languages like English and Chinese, have shown great performance in both text detection and text recognition, however low-resource language like Khmer remain a problem in the field. To overcome these limitations, Nom et al., [15] proposed CLL framework by using high resource language fine-tune on Khmer languages . The authors uses YOLOv11 [16] and TrOCR on the WildKhmerST [17]for training and KhmerST [1] for evaluation. In this study, the authors compares both training from scratch and fine-tuning models pretrained on high resources language. For text detection, they used YOLOv11 and compared a model that built from scratch to the improved one that using weights pretrained on the COCO dataset [18]. The improved YOLOv11 performed better (precision 0.840, recall 0.809, mAP50 = 0.889) and needed much less computing power than the model built from scratch (precision 0.807, recall 0.801, mAP50 = 0.852). For recognition, they used TrOCR and found that they could significantly reduce the error rate if they upgraded the pretrained Latin script models with the Khmer data. They achieved CER of 0.17–0.18 with WER of 0.33–0.35 compare to the CER 0.40–0.44 and WER 0.55–0.59 if they were trained from scratch [15].

3 Methodology

Scene text detection and recognition have been widely studied, but Khmer presents unique challenges due to the complexity of its script [1]. To obtain desirable outcomes, a two-stage approach is preferred over end-to-end models, with Stage 1 focusing on precise text detection and Stage 2 on recognition. This separation enhances detection accuracy in complex backgrounds and allows independent optimization, making it better suited for low-resource environments [19].

Among detection models, EAST [20] provides real-time performance through direct regression of text boxes without complex post-processing. However, its coarse outputs often miss fine details critical for Khmer script. CRAFT [21] achieves high accuracy by predicting pixel-level character regions and affinity links, enabling detection of curved or irregular text, but it requires character-level annotations, which are scarce for Khmer datasets.

To address these challenges, Faster R-CNN, proposed by [9], offers a robust two-stage detection framework. It first generates region proposals, followed by refined classification and bounding box regression. When paired with a Feature Pyramid Network (FPN) backbone [10], the detector gains powerful multi-scale feature extraction capabilities. This is particularly advantageous for detecting small and intricate elements, such as Khmer diacritics.

The Faster R-CNN + FPN architecture strikes an effective balance between accuracy and annotation simplicity. Unlike CRAFT, it only requires word-level bounding boxes, reducing the burden of intensive annotation. The FPN further enhances detection of small glyphs across multiple resolutions, making this combination highly suitable for Khmer scene text detection. When integrated with a strong recognition network, this approach provides a practical and scalable solution for Khmer scene text recognition in real-world settings.

After selecting a detector, the next step is to choose a recognizer capable of accurately handling the complexities of Khmer text. Traditional OCR engines like Tesseract rely on image preprocessing and LSTM-based sequence modeling [22], which work well for clean printed text but struggle with noisy, irregular scene text. Deep learning-based recognizers such as CRNN [23], RARE [24], ASTER [25], and SAR [26] have advanced this field by integrating CNNs, recurrent layers, spatial transformers, and attention mechanisms. While these models achieve strong performance on regular text, they face challenges with highly complex scripts like Khmer, which contain stacked glyphs and intricate diacritics.

The Transformer-based OCR model, TrOCR, proposed by Li et al. [27], introduces a fully transformer-based encoder-decoder architecture. The encoder is a vision transformer (ViT) that processes image patches, while the decoder is a pre-trained language transformer similar to RoBERTa. TrOCR leverages large-scale pre-training on synthetic and real-world image-text pairs, significantly improving performance after fine-tuning on target datasets. According to the KhmerST benchmark, TrOCR substantially out-

performs Tesseract for Khmer scene text recognition [1]. This demonstrates the effectiveness of modern transformer-based architectures for low-resource languages such as Khmer.

By comparing all of these different models, we find TrOCR to be the most suitable one as its global self-attention makes it excellent at modeling long-range dependencies between base consonants and diacritics, which is crucial for Khmer language.

3.1 Pipeline and Detailed Architecture

Given the previously mentioned challenges of Khmer script recognition [1], we propose a two-stage architecture that combines Faster R-CNN with a Feature Pyramid Network (FPN) for text detection and TrOCR (Transformer-based OCR) for text recognition. This hybrid approach leverages the strengths of advanced object detection and sequence-to-sequence modeling, enabling precise localization and robust recognition of Khmer text in complex, real-world scenes. The system operates as a two-step pipeline (1) Text Detection: Faster R-CNN with FPN identifies and localizes text regions within the input image, producing bounding boxes that tightly enclose words or lines of text. (2) Text Recognition: Each detected region is cropped and passed to the TrOCR model, which encodes the visual features and decodes them into a sequence of Khmer characters as can be observed from Figure 4.

3.1.1 Text Detection with Faster R-CNN and FPN

Standard Faster R-CNN can struggle with small-scale features, such as the fine details and diacritical marks that are common in Khmer text, often resulting in incomplete detection. To address this limitation, we integrate a Feature Pyramid Network (FPN) into the Faster R-CNN framework. The FPN enhances multi-scale feature representation by combining high-resolution, low-level features with high-level semantic features. This design improves the model's ability to detect both large and small text instances in diverse scene settings. The detector outputs a set of bounding boxes, each representing a localized text region. These bounding boxes are then cropped and resized to a consistent size before being forwarded to the recognition stage.

3.1.2 Text Recognition with TrOCR

For the recognition step, we employ TrOCR, a Transformer-based, sequence-to-sequence OCR model. TrOCR reframes text recognition as a translation task, mapping visual features from cropped text regions directly to a sequence of text tokens. This eliminates the need for explicit character segmentation, which is particularly difficult for Khmer due to its connected characters and complex positional rules.

TrOCR is composed of two main components: (1) A Vision Transformer (ViT)-based encoder for visual feature extraction. (2) A Transformer-based decoder for autoregressive text generation.

For Vision Transformer (ViT) Encoder, The cropped text region is first divided into non-overlapping patches of size 16×16 pixels. Each patch is linearly embedded into a vector of dimension 768 (hidden_size). Positional embeddings are added to preserve the spatial relationships among patches. The resulting sequence of patch embeddings is processed through 12 Transformer layers, each equipped with 12 self-attention heads, to generate a rich representation of the visual input.

For the Transformer Decoder, the decoder operates autoregressively, generating one Khmer token at a time. At each step, it leverages cross-attention mechanisms to align the encoded visual features from the ViT encoder with the partially generated output sequence. The decoder also consists of 12 layers but uses 16 attention heads and a hidden dimensionality of 1024 (d_model), providing greater capacity for handling complex text sequences.

To effectively recognize Khmer text, TrOCR is configured with several important adjustments: Vocabulary Size: A tokenizer is designed with a vocabulary of 50,265 tokens, covering all Khmer characters, numbers, and punctuation marks. Input Size: Cropped text regions are resized to 384×384 pixels before being processed by the encoder, ensuring consistent input dimensions. Output Generation: The decoder continues generating tokens until it outputs an end-of-sequence (<EOS>) token, which signifies completion of the transcription. A Two-Stage Approach workflow process proceeds as follows: (1) Input Scene Image: A full-scene image containing Khmer text is provided to the system. (2) Text Detection: Faster R-CNN with FPN gener-

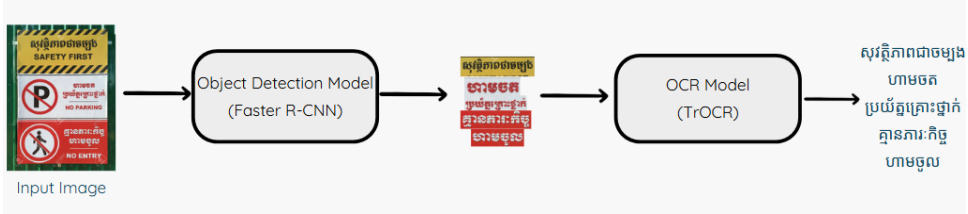

Figure 4. Two-Stage Approach OCR pipeline integrating detection and recognition

ates bounding boxes for text regions at multiple scales. (3) Cropping: Each bounding box is used to extract and normalize individual text regions. (4) Recognition: TrOCR converts each cropped region into a sequence of Khmer characters.

3.2 Dataset

3.2.1 Training and Validation Datasets

To test and train our Khmer scene text recognition system, we gathered and combined data from three public sources: KhmerST [1], the 62k Khmer Printed Dataset [28] (synthetic images), and the Khmer Annotation Dataset [29]. The datasets provide a good balance of Khmer text samples — ranging from clean synthetic text lines to actual scene text in natural scenes. By combining these datasets, we could train our model to recognize both the visual pattern of Khmer script and learn to deal with the issues of scene variability, font differences, and image noise.

• KhmerST Dataset (GitLab) [1]: This dataset contains 1,544 real-world Khmer text images from diverse settings, including signs, banners, and advertisements. It provides line-level polygon annotations converted to bounding boxes for detection and cropped regions for OCR training, featuring varied lighting, angles, and backgrounds to simulate real-world scenarios.

• 62K Khmer Printed Dataset (Hugging Face) [28]: This dataset consists of 62,300 synthetically generated images of printed Khmer text. The images were generated with different font sizes and color variations.

• Khmer Annotation Dataset (Kaggle) [29]: This Khmer Annotation data is a collection of images of Khmer text with XML-style annotations consisting of 3,376 image of printed Khmer text. It is a record with an image file and relevant width and height, word-level annotations with both textual content written in Khmer script and coordinates (x1, x2, y1, y2) of a bounding box localizing each word in the image.

3.2.2 The KST-Wild Test Set

Figure 5. Sample images from our KST-Wild test set. The images showcase the diversity of challenges, including varied fonts, complex backgrounds, reflections, and non-horizontal text orientations.

For the final, rigorous evaluation of our model's performance, we prepared a separate test set, which we refer to as the Khmer Scene Text In-the-Wild (KST-Wild) set. This dataset comprises of 310 real-scene images collected specifically to simulate challenging, real-world use cases and was not used during training or validation. The collection process was to include the following: (1) Real-world text from street signs, billboards, shops, posters, vehicles, and labels. (2) A wide range of font styles, background textures, and image conditions (e.g., tilted text, low lighting, occlusion). (3) Only real-scene images (no synthetic text).

To standardize training data, we normalized text areas and manually annotated images using the VGG Image Annotator (VIA), a lightweight web-based tool for creating boxes and bounding shapes [30]. This dataset serves as the benchmark for our final reported metrics.

Table 1. Summary of Datasets Used

Source	Images	Type	Use
Merged for Training/Validation Pool			
KhmerST [1]	1,544	Scene	Train/Val
62K Printed [28]	62,300	Synthetic	Train/Val
Khmer Annot. [29]	3,376	Printed	Train/Val
Total for Split	**67,220**	**Mixed**	**80/10/10**
Held-out Final Test Set			
KST-Wild (Ours)	**310**	**Wild**	**Final Eval.**

4 Experiments and Results

4.1 Experimental Setup

TrOCR Training: We utilized Amazon SageMaker Studio on Amazon Web Services (AWS) to train the TrOCR model on two datasets: a 62K-sample Khmer Printed Text dataset and the Khmer Scene Text (KhmerST) dataset. Training was conducted on an `ml.g5.4xlarge` instance, equipped with one NVIDIA A100 Tensor Core GPU, 64 vCPUs, and 256 GB of RAM. The process required approximately 8.3 hours to complete.

Object Detection Training: For the text detection component, we trained a Faster R-CNN model on the KhmerST dataset using an `ml.g4dn.8xlarge` instance, which provides one NVIDIA Tesla T4 GPU and 128 GB of RAM. The training process completed in approximately 30 minutes.

Pipeline Integration: Since text detection and recognition are distinct tasks, we integrated both models into a unified pipeline for practical use. The detection model first analyzes an input image and outputs bounding box coordinates for regions containing text. These coordinates are then passed to the OCR model, which extracts the text from each detected region. To enhance usability and accessibility, we implemented a simple graphical interface using Gradio, allowing end-users to interact with the pipeline without requiring technical expertise.

4.2 Evaluation Protocols

4.2.1 Object Detection

The object detection task is commonly evaluated using precision, recall, and F1-Score. Precision measures the proportion of correctly predicted bounding boxes among all predicted boxes, while recall measures the proportion of correctly detected objects among all ground-truth objects. while F1-score gives us an average balanced score between recall and precision. Together, these metrics evaluate whether the model is correctly identifying target objects and minimizing missed detections.

Formally, they are defined as:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 - score = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{3}$$

where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives.

These evaluation metrics follow the standard definition used in the PASCAL VOC Challenge [31].

4.2.2 Optical Character Recognition

To evaluate the OCR task, we utilize several complementary metrics.

Character Error Rate (CER) is computed as the Levenshtein edit distance at the character level divided by the total number of characters [32].

Word Error Rate (WER) extends this concept to the word level, capturing insertions, deletions, and substitutions at the word granularity [33, 34].

Together, these metrics provide a robust, multi-level assessment of OCR performance.

4.3 Results and Discussion

4.3.1 Object Detection Result

From the table illustrated, Faster R-CNN have shown impressive results achieving an impressive score of 91.4% recall score indicating that

Table 2. Hyperparameter Settings for the Detection and Recognition Models

Parameter	Faster R-CNN	TrOCR
Model Base	ResNet-50 w/ FPN	microsoft/trocr-base-handwritten
Optimizer	SGD	AdamW
Learning Rate	0.005	1×10^{-5} to 2×10^{-5}
Batch Size	8	16
Epochs	10	5
Weight Decay	0.0005	0.01
Input Size	Variable	384×384 px
IoU Threshold	0.5	–
Max Token Length	–	128 char

Table 3. Performance Comparison of Object Detection and OCR Models

Task	Metric	Score
Object Detection (Faster R-CNN)	Precision	70.1%
	Recall	91.40%
	F1-score	79.40%
Object Detection (Faster R-CNN), KST-Wild	Precision	62.24%
	Recall	55.51%
	F1-score	58.43%
OCR (TrOCR), KST-Wild	CER	18.27%
	WER	54.43%

Figure 6. Sample results of output from Faster R-CNN

the model is able to detect most of the actual relevant objects in the scene text images. it has also shown a relatively good score of 70.1% for the precision score which indicates that the model's output are relatively correct but may output false positive; this gives us a F1-Score of 79.40%. However, on the manually collected dataset, the model have achieved a slightly lower score of F1-score at 58.43%. From these results, we can infer that the model is capable of predicting and outputting bounding box around text areas in scene-text images; however, those bounding box are not localizing the bounding box as accurately as the ground truth in real world images. However, for the task of optical character recognition in scene-text scenarios, this result is acceptable as the model have shown that that it is capable of detecting all of the relevant text region area in the image even if those box may not exactly match the annotated bounding box which could be drawn subjectively.

4.3.2 Optical Character Recognition Result

From Table 3, the **KST-Wild** dataset achieved an accuracy of 18.27% in terms of CER and 54.43% in terms of WER. These metrics indicate that, despite the model producing seemingly reasonable predictions in certain cases, the overall transcription accuracy remains relatively low. This discrepancy can be attributed to the behavior of the object detection stage, which crops detected text regions before passing them to the recognition model. In several instances, ground truth annotations contain multi-line text regions, whereas the object detection model separates these into multiple individual image segments. When these segmented regions are processed independently by TrOCR, only partial text—often corresponding to a single line—is recognized, while the remaining lines are omitted. This mismatch between annotation granularity and recog-

Table 4. Recognition Performance Compared with the KhmerST Baseline. Models Were Evaluated on the KhmerST Dataset

Model / Pipeline	**CER (%)**	**WER (%)**
KhmerST Baseline (YOLOv8 + TrOCR) [1]	1.01	2.24
Tesseract (as reported in [1])	1.30	4.75
Our Model (Faster R-CNN + TrOCR)	**5.12**	**12.11**

nition output significantly increases both CER and WER.

In addition, the relatively low performance can be largely attributed to the unique structural characteristics of the Khmer script. Khmer words are commonly formed by combining a primary consonant with one or more subconsonants, resulting in complex compound character structures. The recognition model frequently struggles to correctly identify these composite characters, leading to misrecognitions that substantially degrade overall accuracy. These findings underscore the inherent challenges of OCR for complex scripts such as Khmer, particularly in handwritten or noisy real-world settings.

5 Conclusion

This study presented a two-stage Khmer scene text recognition pipeline that integrates Faster R-CNN with Feature Pyramid Networks for text detection and a Transformer-based OCR model (TrOCR) for recognition. The proposed approach demonstrates that a detection-first architecture is effective for localizing Khmer text in complex real-world scenes, achieving strong recall and robust generalization across varied backgrounds and lighting conditions. While recognition performance on the KST-Wild dataset remains moderate due to challenges such as stacked consonants, inconsistent spacing, and line-level segmentation mismatches, the results highlight the feasibility of applying transformer-based OCR models to low-resource, structurally complex scripts like Khmer. The findings indicate that recognition accuracy is primarily constrained by limited training data, short fine-tuning duration, and imperfect alignment between detection and recognition granularity. Future work will focus on improving text-line grouping strategies, expanding real-world annotated datasets, incorporating language-model–aware decoding, and extending training at scale to fully exploit the capabilities of Transformer-based OCR for Khmer scene text recognition.

Acknowledgment

We extend our heartfelt gratitude to our advisor and professor Dr. Ly Rottana, for his support and guidance on this research from the very beginning. We also would like to extend our appreciation to all the authors and researchers who have worked on similar projects in the past for allowing our research to be built upon their works.

References

[1] V. Nom, S. Bakkali, M. M. Luqman, M. Coustaty, and J.-M. Ogier. Khmerst: A low-resource khmer scene text detection and recognition benchmark. *arXiv:2410.18277*, 2024.

[2] S. Keo, M. Coustaty, S. Bakkali, and M. Rossinyol. State-of-the-art khmer text recognition using deep learning models. In *Proc. Int. Conf. Adv. Eng. Technol. (ACET)*, Phnom Penh, Cambodia, 2024.

[3] D. Valy, M. Verleysen, and S. Chhun. Text recognition on khmer historical documents using glyph class map generation with encoder–decoder model. In *Proc. Int. Conf. Pattern Recognit. Appl. Methods (ICPRAM)*, pages 749–756, 2019.

[4] Wikipedia contributors. Tesseract. Wikipedia, The Free Encyclopedia, n.d. Accessed: Sep. 15, 2025.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv:1512.03385*, 2015.

[6] Z. Huang, W. Xu, and K. Yu. Bidirectional LSTM-CRF models for sequence tagging. *arXiv:1508.01991*, 2015.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez,

L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv:1706.03762*, 2017.

[8] S. Mani, W. R. Shankle, M. B. Dick, and M. J. Pazzani. The two-stage machine learning model for guideline development. 1999.

[9] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Adv. Neural Inf. Process. Syst. (NeurIPS)*, volume 28, pages 91–99, 2015.

[10] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 2117–2125, 2017.

[11] R. Buoy, S. Kor, and N. Taing. An end-to-end khmer optical character recognition using sequence-to-sequence with attention. *arXiv:2106.10875*, 2021.

[12] K. Cho, B. van Merrienboer, C. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. *arXiv:1406.1078*, 2014.

[13] H. Riza, M. Purwoadi, Gunarso, T. Uliniansyah, A. A. Ti, S. M. Aljunied, L. C. Mai, V. T. Thang, N. P. Thai, V. Chea, R. Sun, S. Sam, S. Seng, K. M. Soe, K. T. Nwet, M. Utiyama, and C. Ding. Introduction of the asian language treebank. In *Proc. Conf. Oriental Chapter Int. Comm. Coord. Standardization Speech Databases Assess. Tech. (O-COCOSDA)*, pages 1–6, 2016.

[14] R. Buoy, N. Taing, S. Chenda, and S. Kor. Khmer printed character recognition using attention-based seq2seq network. *HCMCOU J. Sci., Eng. Technol.*, 12(1):3–16, 2022.

[15] V. Nom, S. Keo, S. Bakkali, M. M. Luqman, M. Coustaty, and J.-M. Ogier. Cross-lingual learning for low-resource khmer scene text detection and recognition. In *Proc. Int. Conf. Document Anal. Recognit. (ICDAR) Workshops*, Wuhan, China, September 2025.

[16] G. Jocher and J. Qiu. Ultralytics YOLO11, 2024. Version 11.0.0.

[17] V. Nom, S. Keo, S. Bakkali, M. M. Luqman, M. Coustaty, M. Rossinyol, and J.-M. Ogier. Wildkhmerst: A comprehensive dataset and benchmark for khmer scene text detection and recognition in the wild. In *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Wuhan, China, September 2025.

[18] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common objects in context. *arXiv:1405.0312*, 2015.

[19] U. Pal, A. Halder, P. Shivakumara, and M. Blumenstein. A comprehensive review on text detection and recognition in scene images. *Artif. Intell. Appl.*, 2(4):1–5, 2024.

[20] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. EAST: An efficient and accurate scene text detector. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 5551–5560, 2017.

[21] Y. Baek, B. Lee, S. Yun, D. Han, S. Kim, and J. Kim. Character region awareness for the detection of text in the wild. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 9365–9374, 2019.

[22] R. Smith. An overview of the tesseract ocr engine. In *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, pages 629–633, 2007.

[23] 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. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(11):2298–2304, 2017.

[24] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai. Robust scene text recognition with automatic rectification. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 4168–4176, 2016.

[25] B. Shi, X. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai. ASTER: An attentional scene text recognizer with flexible rectification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(9):2035–2048, 2019.

[26] C. Li, X. Tian, C. Shen, C. He, and L. Zhang. Show, attend and read: A simple and strong baseline for irregular text recog-

nition. In *Proc. AAAI Conf. Artif. Intell.*, volume 33, pages 8610–8617, 2019.

[27] M. Li, Y. Lu, F. Zhai, H. Yin, Y. Liu, X. Li, D. Yu, and W. Gao. TrOCR: Transformer-based optical character recognition with pre-trained models. *arXiv:2109.10282*, 2021.

[28] SoyVitou. 62k-images-khmer-printed dataset. Hugging Face, n.d. Dataset.

[29] Keatchakravuth. Khmer annotation. Kaggle, n.d. Dataset.

[30] A. Dutta and A. Zisserman. The VIA annotation software for images, audio and video. In *Proc. ACM Int. Conf. Multimedia*, pages 2276–2279, 2019.

[31] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010.

[32] OCR-D. Ocr-d evaluation specification: ocrd_eval. Online, 2023. Accessed: Sep. 10, 2025.

[33] VISAi. Evaluation metrics for automatic speech and ocr. Online, 2024. Accessed: Sep. 10, 2025.

[34] Cloudraft. A comprehensive guide to ocr. Online, 2024. Accessed: Sep. 10, 2025.