IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 October 2023, accepted 7 November 2023, date of publication 13 November 2023,
date of current version 20 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3332361

RESEARCH ARTICLE

Toward a Low-Resource Non-Latin-Complete Baseline: An Exploration of Khmer Optical Character Recognition

RINA BUOY[1], (Graduate Student Member, IEEE), MASAKAZU IWAMURA[1], (Member, IEEE), SOVILA SRUN[2], AND KOICHI KISE[1]

[1]Department of Core Informatics, Graduate School of Informatics, Osaka Metropolitan University, Sakai, Osaka 599-8531, Japan
[2]Department of Information Technology Engineering, Faculty of Engineering, Royal University of Phnom Penh, Phnom Penh 12156, Cambodia

Corresponding author: Rina Buoy (sp22676n@st.omu.ac.jp)

This work was financially supported by Japan Society for the Promotion of Science (JSPS) Kakenhi Grant 22H0054 and RUPP-OMU Partnership under the Higher Education Improvement Project (HEIP).

ABSTRACT Many existing text recognition methods rely on the structure of Latin characters and words. Such methods may not be able to deal with non-Latin scripts that have highly complex features, such as character stacking, diacritics, ligatures, non-uniform character widths, and writing without explicit word boundaries. In addition, from a natural language processing (NLP) perspective, most non-Latin languages are considered low-resource due to the scarcity of large-scale data. This paper presents a convolutional Transformer-based text recognition method for low-resource non-Latin scripts, which uses local two-dimensional (2D) feature maps. The proposed method can handle images of arbitrarily long textlines, which may occur with non-Latin writing without explicit word boundaries, without resizing them to a fixed size by using an improved image chunking and merging strategy. It has a low time complexity in self-attention layers and allows efficient training. The Khmer script is used as the representative of non-Latin scripts because it shares many features with other non-Latin scripts, which makes the construction of an optical character recognition (OCR) method for Khmer as hard as that for other non-Latin scripts. Thus, by analogy with the AI-complete concept, a Khmer OCR method can be considered as one of the non-Latin-complete methods and can be used as a low-resource non-Latin baseline method. The proposed 2D method was trained on synthetic datasets and outperformed the baseline models on both synthetic and real datasets. Fine-tuning experiments using Khmer handwritten palm leaf manuscripts and other non-Latin scripts demonstrated the feasibility of transfer learning from the Khmer OCR method. To contribute to the low-resource language community, the training and evaluation datasets will be made publicly available.

INDEX TERMS Khmer script, non-Latin scripts, character stacking, no explicit word boundaries, text recognition, image chunking.

I. INTRODUCTION

Optical character recognition (OCR) is the process of recognizing text from images, and has a wide range of practical applications. The overall OCR pipeline consists of two subtasks, namely text detection and text recognition, which can be performed separately or simultaneously [1]. A text detector locates text regions that are then transcribed by a text recognizer. Depending on the image modality, the recognition task can be further categorized as handwritten recognition (HWR), document OCR, or scene text recognition (STR).

However, the development of OCR for different languages has been unequal because many existing approaches have been proposed whose accuracy is maximized on word-level Latin datasets. This is partially because many text detection approaches, driven by Latin word-level annotation datasets [2], [3], are based on words [4]. With word-level datasets, input images can be reasonably resized to a fixed

The associate editor coordinating the review of this manuscript and approving it for publication was Wenbing Zhao.

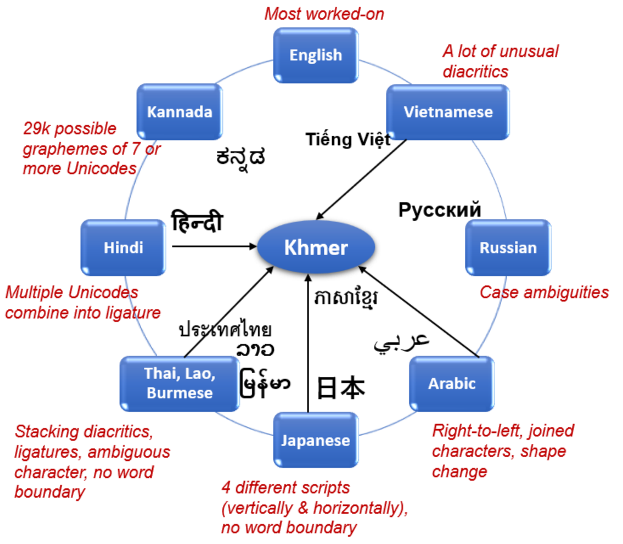

Most worked-on
English
A lot of unusual diacritics
Kannada
Vietnamese
29k possible graphemes of 7 or more Unicodes
ಕನ್ನಡ
Tiếng Việt
Русский
हिन्दी
Hindi
Khmer
Russian
Multiple Unicodes combine into ligature
ភាសាខ្មែរ
Case ambiguities
ประเทศไทย
ລາວ
عربي
Thai, Lao, Burmese
မြန်မာ
日本
Arabic
Japanese
Stacking diacritics, ligatures, ambiguous character, no word boundary
Right-to-left, joined characters, shape change
4 different scripts (vertically & horizontally), no word boundary

FIGURE 1. Convex hull of languages [5] (redrawn with modifications), showing relations between the Khmer script and other non-Latin scripts. Each arrow indicates the existence of one or more shared features with the Khmer script.

size (e.g., 32 × 100 pixels) for efficient mini-batch training in which images of the same size are processed in parallel. However, word-level approaches and practices may not be optimal for textline images of non-Latin scripts [4], which can be arbitrarily longer.

As shown in Fig. 1, non-Latin scripts often possess special features, such as diacritics, character stacking, implicit word boundaries, connected characters, shape changes, non-uniform character sizes, and ligatures [5]. As shown in Fig. 2, some scripts, such as Thai, Khmer, and Vietnamese, allow characters to be stacked on top of each other; such stacking makes the information in the height dimension as rich as that in the width dimension. Other scripts, such as Thai, Khmer, and Japanese, do not have explicit word boundaries, and therefore textline images may be arbitrarily long in comparison to word-level Latin text images. Due to the complex structure, accurate recognition of all characters requires spatially rich visual features. These spatially rich visual features are represented by two-dimensional (2D) feature maps that capture information in both height and width dimensions. However, extracting 2D features imposes an additional computational burden on subsequent processes and the majority of existing methods [5], [6], [7], [8], [9], [10], [11], [12], [13] for low-resource non-Latin scripts are still based on 1D feature maps.

Capturing spatial feature dependencies is a crucial step in the text recognition pipeline. A self-attention mechanism [14] is an effective architecture for modeling 2D feature dependencies [4]. However, the self-attention mechanism exhibits quadratic time and memory complexity in relation to the width and height of 2D feature maps. As mentioned earlier, in the case of non-Latin scripts where explicit word boundaries are absent, a textline image can have an arbitrary length. This, combined with 2D feature modeling, creates a significant computational bottleneck. For instance, in Fig. 3(a), the self-attention complexity is $O(C)$ where $C = (H'W')^2$ for an input width of W and H' and W' are the height and width of 2D feature maps. If we examine Fig. 3(b) and consider an input five times wider (represented as $5W$), the self-attention complexity escalates to $O(5^2C)$. This computational challenge is not particularly serious for most word-level (i.e., cropped) Latin scene text recognition methods [15], [16], [17], [18], [19], [20], as they typically resize the input image to a fixed size. However, the inference accuracy degrades significantly for long input images [4], [19]. For non-Latin scripts, resizing long textline images to a significantly smaller width can potentially impact the legibility of small-sized characters, such as subscripts and diacritics. Conversely, resizing images to an excessively large size is computationally inefficient and impractical for mini-batch training. This is because shorter textline images require padding to match the width of other images in the batch, which adds unnecessary computational overhead during processing. Therefore, directly applying these methods to non-Latin scripts without considering their specifications leads to suboptimal results and inefficiencies.

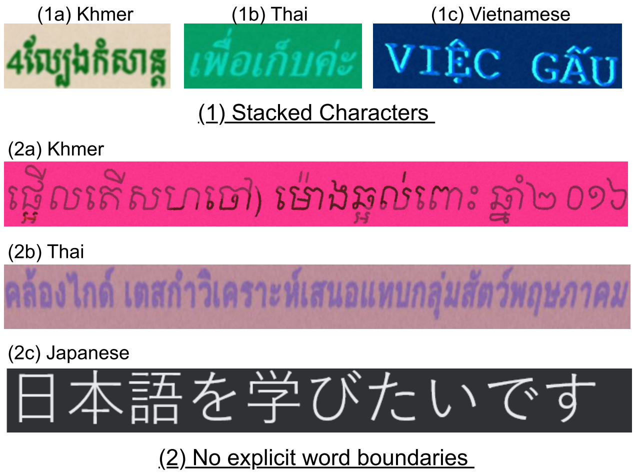


FIGURE 2. Illustrations of character stacking and no explicit boundaries. (1) Scripts with character stacking: Khmer, Thai, and Vietnamese. (2) Scripts without explicit word boundaries: Khmer, Thai, and Japanese.

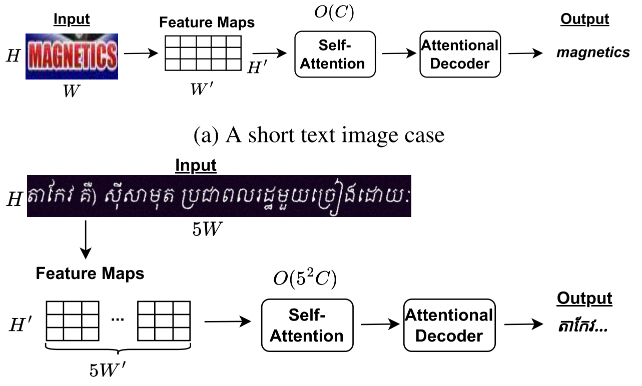
In this paper, we propose a convolutional Transformer-based text recognition method for low-resource non-Latin scripts that have character stacking, diacritics, ligatures, and implicit word boundaries. This is because Transformer-based text recognition architectures [17], [18] have been widely adopted for Latin scene text recognition, thanks to their flexibility in handling 2D feature maps. However, in contrast to most Latin scene text recognition methods, the proposed method uses chunk-level 2D spatial feature maps, where feature dependencies are modeled only within the local regions instead of the entire input image. This is achieved by using a modified image chunking and merging technique. The proposed technique reduces the training complexity caused by arbitrarily long input images from a quadratic to a linear relation. This is accomplished by dividing a long textline image into multiple consecutive overlapping chunks and processing the features of each chunk independently. The features are then merged together for character decoding using a Transformer decoder, which relies on all previous outputs from all chunks to predict the next character. To ensure input continuity when a character is located at the boundary between two chunks, overlapping margins or regions are introduced to each chunk. With the image chunking and merging technique, the input images are not required to be resized to a fixed size and the resolution of the small-sized characters, such as subscripts and diacritics, is not undesirably affected.

The proposed method is applied to the Khmer script because it is representative of low-resource non-Latin scripts. The Khmer script shares many features with other non-Latin scripts, such as ligatures, diacritics, stacking, implicit word boundaries, highly ambiguous characters, and non-uniform character sizes, as shown in Fig. 1. Therefore, the construction of an OCR method for Khmer is at least as hard as that for other non-Latin scripts. Thus, by analogy with the AI-complete concept, a Khmer OCR method can be considered as one of the "non-Latin-complete" methods and can be used as a low-resource non-Latin baseline method.

Approximately 2.8 million images of synthetic document and scene text of various Khmer fonts were generated for training. We compared our proposed approach with the representative Latin baseline methods trained on the Khmer script by using the Khmer ID card dataset, the KHOB dataset, and the historical handwritten palm leaf dataset [21]. In all cases, our proposed method achieved lower character error rates (CERs), compared to the baseline methods. By transfer learning on small-scale datasets, the proposed approach was further fine-tuned on other low-resource non-Latin scripts, including Thai, Laos, Burmese, Vietnamese, and Hindi. The fine-tuning results demonstrated the feasibility of transfer learning from the Khmer script to other non-Latin scripts with similar features.

Our contributions can be summarized as follows:

1) We propose a convolutional Transformer-based text recognition method that uses chunk-level 2D spatial feature maps for low-resource low-attention non-Latin scripts that have complex features, such as character stacking, non-uniform character sizes, implicit word boundaries, and diacritics.
2) We incorporate a modified image chunking and merging technique into the proposed Transformer-based text recognition system to reduce the self-attention complexity caused by arbitrarily long input images from a quadratic to a linear relation and capture 2D spatial dependencies. The input images are not required to be resized to a fixed size and thus, the resolution of the small-sized characters, such as subscripts, vowels, and diacritics, is not undesirably affected.



(a) A short text image case

(b) A long textline image

FIGURE 3. Self-attention complexity as a quadratic function of input width. (a) The self-attention complexity of a short text image is $O(C)$ where $C = (H'W')^2$. (b) The resulting self-attention complexity of a five times longer textline image is $O(5^2C)$. Let H and W be the height and width of the input image and H' and W' be the height and width of the resulting feature maps.

3) We found that our proposed 2D models for the Khmer script achieved superior performance on the real evaluation datasets in comparison to the baseline models.
4) The experimental results demonstrated that OCR methods for other low-resource non-Latin scripts, including Thai, Laos, Burmese, Vietnamese, and Hindi, can be efficiently trained by transfer learning from the Khmer OCR method.

II. KHMER SCRIPT AS A REPRESENTATIVE OF NON-LATIN SCRIPTS

Khmer is the official language of Cambodia and is spoken by approximately 17 million speakers. The Khmer script is an abugida system in which each consonant is attached to an inherent invisible vowel [22]. In the Khmer writing system, there are 33 consonants, 14 independent vowels, 23 dependent vowels, and eight diacritics. The Unicode Standard code points from U+1780 to U+17FF are assigned to these symbols [23]. According to the Guinness World Records, the Khmer script has the largest alphabet [24], and the Khmer language is considered to be one of the most complex writing systems [21], [23]. Fig. 4 presents the Khmer alphabet inventory and the corresponding Unicode points.

Depending on the fonts used, some pairs of characters in Figs. 5(1) and (3) are highly ambiguous. The only distinction between these sets of characters is a single stroke. In some extreme cases, some subscript forms are almost identical, as shown in Fig. 5(2). Although some characters have one connected glyph, some are composed of multiple disconnected glyphs, each of which is a separate character, as shown in Fig. 5(3c). Thus, a sophisticated contextual 2D recognition system is required for recognizing and distinguishing these characters [24].

Khmer text is written from left to right with optional spaces for readability purposes. The text is composed of orthographic syllables or character clusters. Each cluster comprises a base consonant or an independent vowel, up to two consonant subscripts, a dependent vowel, and a diacritic [24]. Depending on their roles in a cluster, consonants can take one of two different shapes, namely the base and subscript forms. Unlike Latin script, the sizes of characters in a cluster vary significantly as the sizes of subscripts, diacritics, and some vowels are considerably smaller, compared with a base consonant. Some examples of character clusters are illustrated in Fig. 6(1) showing disproportionate sizes of the contributing characters in the clusters. The character resolution is significantly sensitive to image resizing, as shown in Fig. 7 showing diminishing resolution of the small-sized characters in red boxes with increasing downsizing factors. In addition, Figs. 6(1a), (1b), and (1c) also show several possible sequences of the same word, which make it difficult to compare two identically-looking words without knowing the underlying sequences. The character normalization scheme [25] can normalize these sequences to a canonical sequence. This is achieved by decomposing each character cluster into smaller units (e.g., subscripts and vowels), applying rule-based corrections at the unit level, and recombining the corrected units into a normalized cluster.

In addition, some consonants and subscripts can form ligatures with dependent vowels, diacritics, or both. Some ligatures are recognizable, whereas others are nontrivial. Ligatures are also font-dependent, and those caused by the popular calligraphic Khmer Muol font are hard to recognize. Fig. 6(2) shows some examples of ligatures. Because of the complex structure of the Khmer script as described above, capturing rich 2D spatial dependencies is crucial for recognizing individual characters correctly [26].

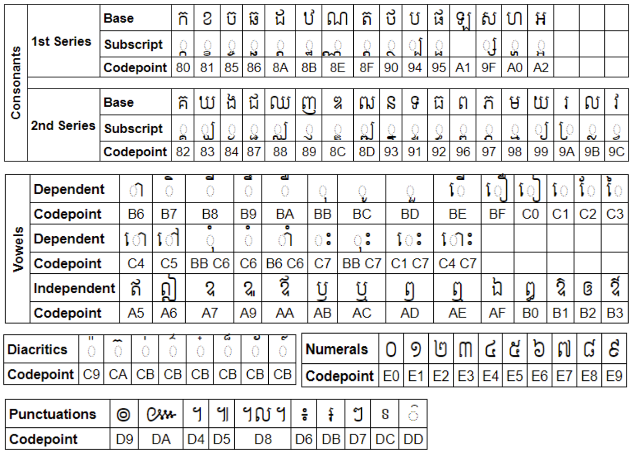
The Khmer script is closely related to the scripts of Thai, Laos, and Burmese in many respects. The Khmer and Hindi scripts are both abugida but the latter uses spaces as a word delimiter. Conversely, the Vietnamese script is based on Latin and, like the Khmer script, uses additional diacritics for functional and tonal purposes. Therefore, the Khmer script was used as the representative of low-resource non-Latin scripts in this study.

III. RELATED WORK

A. WORK ON LATIN TEXT RECOGNITION

In this section, we present a brief review of the most recent seminal deep learning-based work on Latin text recognition. Text recognition methods can, broadly, fit into a unified segmentation-free framework, consisting of rectification, visual feature extraction, sequence modeling, and transcription [4], [10], [15], [16], [27], [28], [29]. Some of these methods can handle textlines of arbitrary lengths, while others require resizing to a fixed size. However, resizing a long textline input to a fixed size can negatively impact the resolution of small-sized characters as discussed earlier. Similarly, segmenting a long textline input into smaller independent units using rudimentary image processing techniques is particularly challenging for complex backgrounds and can also result in discontinuity in linguistic context during character decoding.

For regular text images, rectification is optional. Shi et al. [30] proposed a convolutional recurrent neural network (CRNN) architecture for extracting visual-temporal features, and used a connectionist temporal classification (CTC) decoder for transcription. Other variants of the CRNN architecture include GRCNN [31], which uses the gated recurrent convolution layer (GRCL) for context modulation, and Rosetta [32]. Conversely, for irregular text images, the input image is first transformed by using the spatial Transformer network (STN) [33] to correct distorted text geometries for subsequent downstream stages. Shi et al. [19] proposed an attention-based sequence-to-sequence (seq2seq) recognition network (ASTER) that uses an STN rectification layer. ASTER also uses a CRNN



Consonants																				
	1st Series	Base	ក	ខ	ច	ឆ	ដ	ឋ	ណ	ត	ថ	ប	ផ	ឡ	ស	ហ	អ			
		Subscript	◌្ក	◌្ខ	◌្ច	◌្ឆ	◌្ដ	◌្ឋ	◌្ណ	◌្ត	◌្ថ	◌្ប	◌្ផ		◌្ស	◌្ហ	◌្អ			
		Codepoint	80	81	85	86	8A	8B	8E	8F	90	94	95	A1	9F	A0	A2			
	2nd Series	Base	គ	ឃ	ង	ជ	ឈ	ញ	ឌ	ឍ	ន	ទ	ធ	ព	ភ	ម	យ	រ	ល	វ
		Subscript	◌្គ	◌្ឃ	◌្ង	◌្ជ	◌្ឈ	◌្ញ	◌្ឌ	◌្ឍ	◌្ន	◌្ទ	◌្ធ	◌្ព	◌្ភ	◌្ម	◌្យ	◌្រ	◌្ល	◌្វ
		Codepoint	82	83	84	87	88	89	8C	8D	93	91	92	96	97	98	99	9A	9B	9C

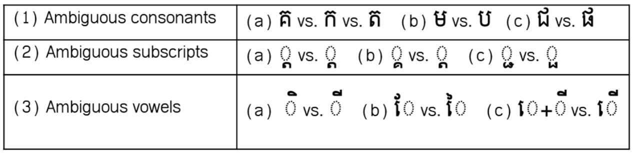
Vowels															
	Dependent	◌ា	◌ិ	◌ី	◌ឹ	◌ឺ	◌ុ	◌ូ	◌ួ	◌ើ	◌ឿ	◌ៀ	◌េ	◌ែ	◌ៃ
	Codepoint	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF	C0	C1	C2	C3
	Dependent	◌ោ	◌ៅ	◌ុំ	◌ំ	◌ាំ	◌ះ	◌ុះ	◌េះ	◌ោះ					
	Codepoint	C4	C5	BB C6	C6	B6 C6	C7	BB C7	C1 C7	C4 C7					
	Independent	ឥ	ឦ	ឧ	ឩ	ឪ	ឫ	ឬ	ឭ	ឮ	ឯ	ឰ	ឱ	ឲ	ឳ
	Codepoint	A5	A6	A7	A9	AA	AB	AC	AD	AE	AF	B0	B1	B2	B3

Diacritics	◌៉	◌៊	◌់	◌៌	◌៍	◌៎	◌៏	◌័
Codepoint	C9	CA	CB	CB	CB	CB	CB	CB

Numerals	០	១	២	៣	៤	៥	៦	៧	៨	៩
Codepoint	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9

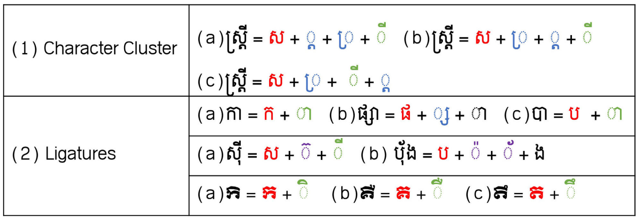
Punctuations	៙	៚	។	៕	។ល។	៖	៛	ៗ	ៜ	◌៝
Codepoint	D9	DA	D4	D5	D8	D6	DB	D7	DC	DD

FIGURE 4. The Khmer alphabet inventory and the Unicode points. U+17 is removed from each Unicode point. The light gray dotted circle indicates characters that must be attached to a base consonant.



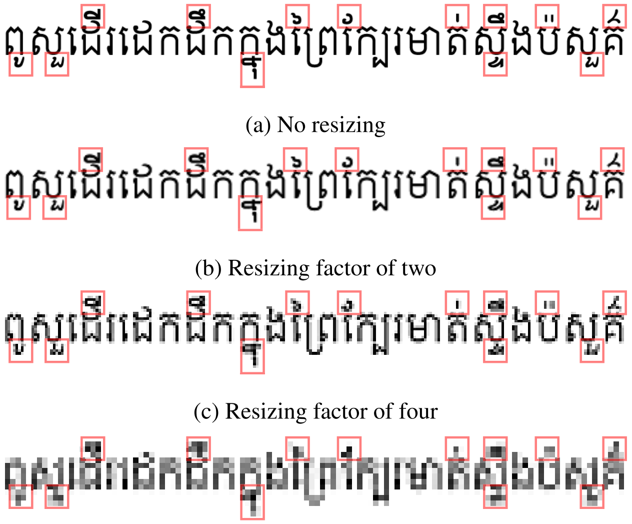
(1) Ambiguous consonants	(a) គ vs. ក vs. ត (b) ម vs. ប (c) ជ vs. ផ
(2) Ambiguous subscripts	(a) ◌្ដ vs. ◌្ត (b) ◌្គ vs. ◌្ក (c) ◌្ឆ vs. ◌្ផ
(3) Ambiguous vowels	(a) ◌ិ vs. ◌ី (b) ◌ែ vs. ◌ៃ (c) េ◌+◌ី vs. ◌ើ

FIGURE 5. Ambiguous characters: (1) consonants, (2) subscripts, and (3) vowels.



(1) Character Cluster	(a)ស្ត្រី = ស + ◌្ត + ◌្រ + ◌ី (b)ស្ត្រី = ស + ◌្រ + ◌្ត + ◌ី (c)ស្ត្រី = ស + ◌្រ + ◌ី + ◌្ត
(2) Ligatures	(a)កា = ក + ◌ា (b)ផ្សា = ផ + ◌្ស + ◌ា (c)បា = ប + ◌ា
	(a)ស៊ី = ស + ◌៊ + ◌ី (b) ប៉័ង = ប + ◌៉ + ◌័ + ង
	(a)ភិ = ភ + ◌ិ (b)តឺ = ត + ◌ឺ (c)តឹ = ត + ◌ឹ

FIGURE 6. Samples of Khmer character clusters and ligatures: base consonants (red), consonant subscripts (blue), diacritics (purple), and vowels (green). Best viewed in color.



(a) No resizing

(b) Resizing factor of two

(c) Resizing factor of four

(d) Resizing factor of eight

FIGURE 7. Effect of image downsizing by preserving an aspect ratio on the small-sized characters in red boxes. (1) Original size (122 × 941 pixels). (2) Downsizing factor of two. (3) Downsizing factor of four. (3) Downsizing factor of eight. All images are displayed with the same scale for illustration purpose. Best viewed in color.

encoder and an attention-based decoder. Similarly, Zhan and Lu [20] proposed an attention-based recognition network named ESIR, which uses an iterative image rectification layer and a modified ResNet-53 as a visual feature extractor. Baek et al. [16] experimented with various model design choices and found TRBA (TPS-ResNet-BiLSTM-Attention) to be the best-performing model, followed by TRBC (TPS-ResNet-BiLSTM-CTC). Sheng et al. [34] proposed a no-recurrence seq2seq model (NRTR), consisting of a modality-transform block (MDT), Transformer encoders, and Transformer decoders. The MDT module maps a 2D input to a one-dimensional (1D) sequence. Methods, such as ASTER, ESIR, TRBC, and TRBA, require input images to be resized to a fixed size to apply STN rectification, whereas rectification-free methods, such as CRNN, GRCNN, and Rosetta, do not necessarily impose such a restriction. The latter methods are therefore capable of handling arbitrarily long input images. In addition, compared with the basic Transformer or attention-based decoder, the CTC decoder used by CRNN, GRCNN, Rosetta, and TRBC is less sensitive to input width [4].

The methods cited above are based on 1D feature maps, which may fail to recognize irregular texts [28]. Therefore, Lee et al. [17] introduced a self-attention STR network (SATRN) that uses 2D self-attention encoders and Transformer decoders. Ly et al. [35] proposed a 2D self-attention convolutional recurrent network (2D-SACRN) for the HWR task, in which 2D self-attention layers are injected into the convolutional neural network (CNN) module directly. Taking advantage of the pretrained vision Transformer (ViT) models, Atienza [15] proposed ViTSTR, a ViT-based approach for STR. ViTSTR directly maps an input image to output tokens by using only an encoder. Similarly, Li et al. [18] proposed TrOCR, a Transformer-based OCR method using a pretrained ViT model. The Transformer-based approaches, such as SATRN, ViTSTR, and TrOCR, are segmentation-free methods that do not possess prior knowledge of character boundaries before recognition. As a result, these methods typically require input images to be resized to a fixed size. However, this restriction may not be suitable for handling non-Latin scripts without explicit boundaries, where textline images can be arbitrarily long and have a low height-to-width ratio. This poses a challenge as it may result in distortion or loss of information in the textlines, potentially affecting the accuracy of recognition for such scripts.

B. WORK ON KHMER AND LOW-RESOURCE NON-LATIN TEXT RECOGNITION

Sok and Taing [36] proposed a Khmer printed text recognition system that is composed of three steps: character segmentation using edge detection, character classification using a support vector machine (SVM), and rule-based character assembly. Valy et al. [21], [24], [26] proposed text recognition methods and applied them to the historical handwritten Sleuk Rith dataset. These methods use both a CNN and a 2D recurrent neural network (RNN) to extract 2D spatial-visual and sequential features. Recently, Buoy et al. [6] proposed an attention-based seq2seq approach that uses both a CNN and an RNN for Khmer printed text recognition, achieving an improved CER, compared with that of Tesseract [5].

Fujii et al. [8] proposed a textline recognition framework with script identification for multilingual scripts including Khmer, which uses only a CNN feature extractor and CTC transcription. The photo OCR performance is significantly higher on Latin and Cyrillic than on non-Latin scripts. Similarly, Ignat et al. [37] performed an OCR benchmark for 60 languages including the Khmer script on both synthetic and real datasets by using Tesseract 4.0. It achieved the highest OCR accuracy for Latin and Cyrillic scripts, but relatively poor OCR accuracy for Perso-Arabic, North and South Indic, and Southeast Asian scripts, indicating that more training data and considerable attention are needed.

Variants of the CRNN architecture or the basic CRNN model have been applied to other non-Latin languages, including Vietnamese [13], Indian languages [9], [10], Arabic [9], [12], Urdu [12], and Chinese [7]. Le et al. [11] proposed an attention-based seq2seq approach that uses both a CNN and an RNN for the Vietnamese HWR task. Thus, the existing methods of non-Latin text recognition are dominated by CRNN and attention-based seq2seq architectures using 1D visual feature maps.

IV. DATASETS

Following a practice that is common in large-scale OCR [38] and considering the scarcity of real labeled data of the low-resource Khmer script, we synthetically generated

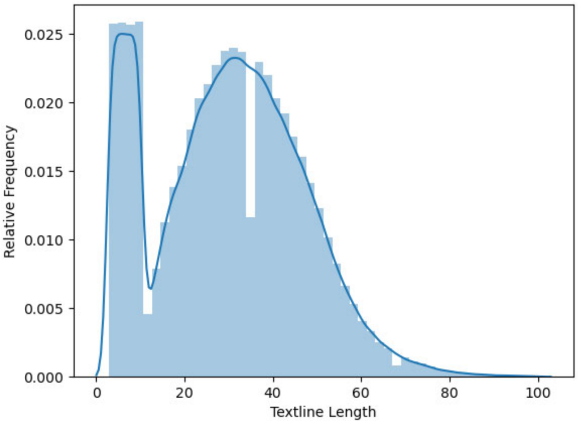


FIGURE 8. Textline length (number of characters) distribution of the synthetic training scene text and document OCR datasets. The training text corpus exhibits a bimodal distribution of textline lengths, encompassing both shorter numbers and longer texts.

datasets of document OCR text and scene text. In both cases, source textline data were converted to textline images and we used the Khmer corpus and fonts provided by Tesseract [5]. Since the source text data lacks numerical content, additional random short textlines containing Khmer and Arabic numbers were included as well. The resulting bimodal distribution of textline lengths (number of characters) is shown in Fig. 8, illustrating two categories: numbers and texts. Several samples from the training, validation, evaluation, and fine-tuning datasets are presented in Fig. 9. The list of the used datasets in this study and their sizes is summarized in Table 1.

For the synthetic document OCR training dataset, we used TextRecognitionDataGenerator[1] to generate 1.5 million textline images with plain white backgrounds. We applied random data augmentation to a mini-batch during training. The data augmentation techniques that we applied include erosion, addition of noise blobs, text thinning and thickening, blurring, perspective distortion, rotation, deformation, and image concatenation.

For the synthetic scene text training dataset, we used SynthTIGER [38] to generate 1.3 million randomly augmented images. During training, the scene text images were randomly concatenated with the augmented document OCR text images to increase text diversity, complexity, and length.

To identify the optimal models and configurations, we used a synthetic scene text validation dataset, consisting of 10,000 images. This validation dataset was generated by using SynthTIGER [38] and the KHPOS corpus.[2] The textline length (character count) distribution of the KHPOS corpus is shown in Fig. 10, and Fig. 11 provides some examples of short and long textline images from the validation dataset.

For performance benchmarking, we used the Khmer ID card dataset, consisting of 1,500 images captured by smartphone cameras. The images in this dataset are heterogeneous with respect to condition, quality, resolution, blurring, and lighting. Perspective distortion, deformation, erosion, pixelation, and noise are common. Each ID card image was manually labeled by extracting field data, such as the ID number and full name, and the CRAFT text detector[3] was used for extracting the bounding boxes of text.

TABLE 1. List of the used datasets and their sizes. *: machine printed. **: historical and handwritten.

Dataset	Type	Training	Validation	Evaluation
STR & OCR*	Synthetic	2.8M	10,000	-
Khmer ID Card*	Real	-	-	1,500
KHOB*	Real	-	-	336
Sleuk Rith**	Real	71,250	-	3,750

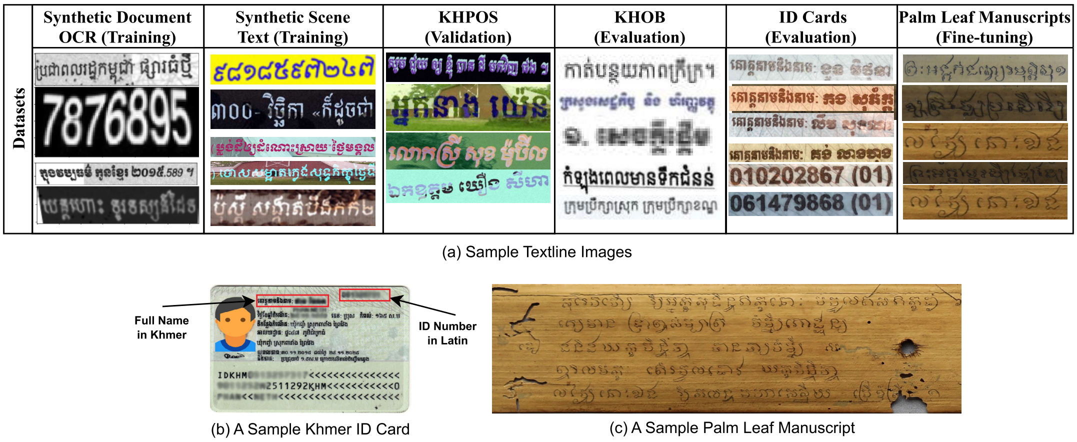
In addition, we also evaluated our proposed 2D models' performance on the publicly available KHOB dataset,[4] which comprises manually annotated textline images of PDF documents. Compared to the Khmer ID card dataset, the images in the KHOB dataset have relatively clean backgrounds but poorer quality due to compression that significantly affects the resolution of small-sized characters, such as subscripts, vowels, and diacritics. After excluding images with insufficient resolution or those containing Latin text, the resulting collection comprises 336 textline images.

To assess the robustness of the proposed approach on other input modalities, we fine-tuned our proposed 2D models on the historical Sleuk Rith dataset[5] and compared the CER with that achieved by the state-of-the-art (SOTA) model [21]. This dataset consists of 657 manually annotated rectangular pages, and contains the equivalent of 75,000 word images. Each page is made of a dried palm leaf on which letters in the ancient Aksar Kham font were carved with a sharp, pointy stylus. Additional ink was applied to make the text and background colors distinct. The manuscripts have been preserved from generation to generation. The writing varies according to the time of creation and differs significantly from modern Khmer writing [21]. The Sleuk Rith dataset does not have a predefined train-test split, so we randomly divided it into training and test sets. The training set consisted of 95% of the samples, while the remaining 5% were allocated to the test set. Therefore, it is important to take this into consideration when comparing the performance with the SOTA model [21].

V. PROPOSED TEXT RECOGNITION ARCHITECTURE

In this section, we discuss our proposed unified and modular convolutional Transformer-based approach for Khmer text recognition. Our proposed approach is unified because it can handle both the 2D and 1D feature maps produced by the CNN module without changing the architecture. It is modular because certain modules, such as the CNN module,

[1] https://github.com/Belval/TextRecognitionDataGenerator
[2] https://github.com/ye-kyaw-thu/khPOS
[3] https://github.com/clovaai/CRAFT-pytorch
[4] https://github.com/EKYCSolutions/khmer-ocr-benchmark-dataset
[5] https://github.com/donavaly/SleukRith-Set



(a) Sample Textline Images

(b) A Sample Khmer ID Card

(c) A Sample Palm Leaf Manuscript

FIGURE 9. **(a) Some preview images. (b) A sample Khmer ID card. (c) A sample palm leaf manuscript page.**

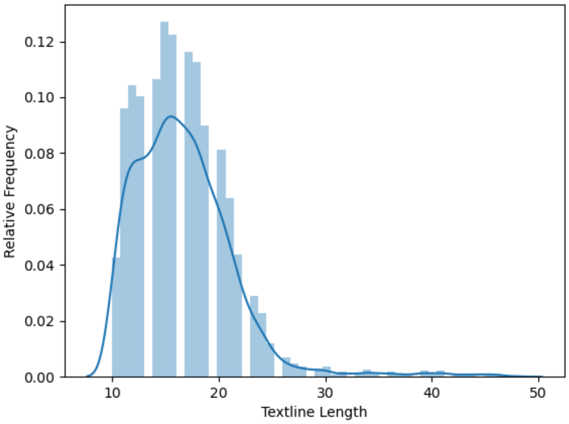


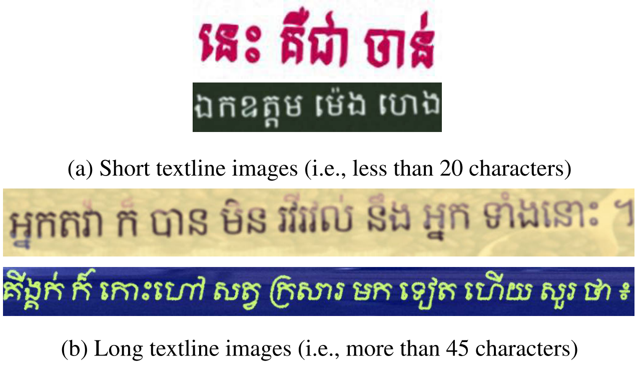
FIGURE 10. **Textline length (number of characters) distribution of the validation text corpus (KHPOS).**

can be included or excluded. Nevertheless, for vision tasks, the CNN module is usually included because it can suppress background artifacts by extracting more abstract features and can reduce the computational burden on the subsequent Transformer encoders [17]. The proposed approach can be used as a baseline method for other low-resource non-Latin scripts with similar features.

A. TRANSFORMER-BASED RECOGNITION MODEL

Similarly to TrOCR [18] and SATRN [17], the Transformer-based architecture used in this study consists of an encoder and a decoder, as shown in Fig. 12. The encoder is composed of a CNN module, a patching module, and Transformer encoder units. For 2D feature maps, the CNN module produces the feature maps by downsampling the input width and height by a factor of four, whereas for 1D feature maps, the input height is downsampled to a unit height. Mathematically, the feature maps, $\boldsymbol{F} = (\boldsymbol{F}_{1,1}, \ldots, \boldsymbol{F}_{1,W'}, \ldots, \boldsymbol{F}_{H',W'})$, $\boldsymbol{F}_{i,j} \in \mathcal{R}^D$, are given by

$$\boldsymbol{F} = \text{CNN}(\boldsymbol{I}), \tag{1}$$



(a) Short textline images (i.e., less than 20 characters)

(b) Long textline images (i.e., more than 45 characters)

FIGURE 11. **Examples of short and long textline images in the validation dataset.**

where $\boldsymbol{I}$ is a grayscale image of $(H \times W)$ pixels and CNN is a CNN module. $H' = \frac{H}{4}$, $W' = \frac{W}{4}$, and D are the height, width, and dimension of the resulting feature maps, $\boldsymbol{F}$. In case of 1D feature maps, the CNN module reduces H' to one through the convolutional or pooling layers.

The patching module converts the feature maps from the CNN module to a position-aware sequence of vectors by splitting the feature maps into small non-overlapping patches. These patches are then linearly projected and position embeddings are added before they are passed to the Transformer encoders. The resulting patched feature maps, $\boldsymbol{P} = (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_{\frac{H'}{k_1}\frac{W'}{k_2}})$, $\boldsymbol{P}_i \in \mathcal{R}^D$, are given by

$$\boldsymbol{P} = \text{PATCH}_{\text{ENC}}(k_1, k_2)(\boldsymbol{F}), \tag{2}$$

where $\text{PATCH}_{\text{ENC}}(k_1, k_2)$ is a patching function which takes a ViT-like patch size (k_1, k_2) as an input. The patching module reduces the feature maps, $\boldsymbol{F}$, by k_1 and k_2 in the vertical and horizontal directions, respectively. To extract spatial feature dependencies, $\boldsymbol{P}$ are fed to a stack of standard Transformer encoder units to produce $\boldsymbol{T} = (\boldsymbol{T}_1, \ldots, \boldsymbol{T}_{\frac{H'}{k_1}\frac{W'}{k_2}})$, $\boldsymbol{T}_i$ in $\mathcal{R}^D$,

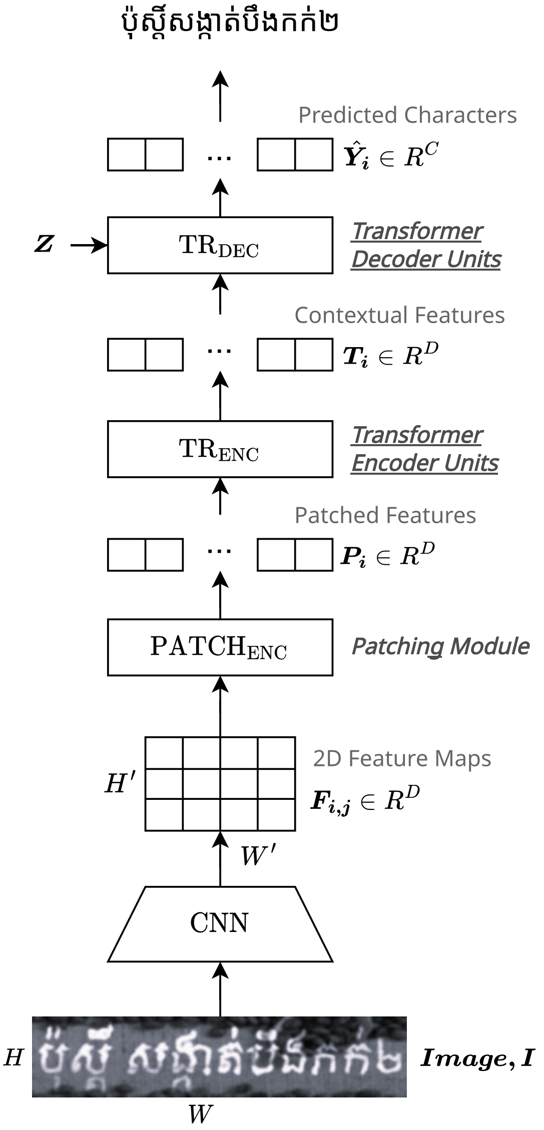


FIGURE 12. **The overall Transformer-based encoder-decoder architecture. The CNN module takes an input image, I, and outputs 2D feature maps, F, that are downsampled by the PATCH$_{\mathrm{ENC}}$ module to generate P. To capture spatial feature dependencies, P are fed to the Transformer encoder units, TR$_{\mathrm{ENC}}$, to obtain T. T and a context sequence, Z, are fed to the Transformer decoder units, TR$_{\mathrm{DEC}}$, to obtain the predicted characters, $\hat{Y}$.**

as given by

$$T = \mathrm{TR}_{\mathrm{ENC}}(P), \tag{3}$$

where TR$_{\mathrm{ENC}}$ is a stack of standard Transformer encoder units.

The decoder module consists of Transformer decoder units; however, in contrast to a basic Transformer decoder, it uses static position encoding for predicting the output of variable length. The autoregressive decoder module takes T as input and outputs a probability distribution sequence over 131 characters, including Khmer characters, common foreign symbols (such as hyphen, space, and period), and three special tokens (namely PAD for padding, EOS for end of sentence, and SOS for start of sentence). The class distribution sequence, $\hat{Y} = (\hat{Y}_1, \ldots, \hat{Y}_L)$, $\hat{Y}_i \in \mathcal{R}^C$, and Loss are given by

$$\hat{Y} = \mathrm{TR}_{\mathrm{DEC}}(T, Z) \tag{4}$$

$$\mathrm{Loss} = \mathrm{CE}(\hat{Y}, Y), \tag{5}$$

where TR$_{\mathrm{DEC}}$ is a stack of Transformer decoder units and $Y = (y_1, \ldots, y_L, EOS)$ is a target sequence. $Z = (SOS, y_1, \ldots, y_L)$ is a context sequence. L is the prediction length and C is the number of prediction classes which is 131. CE is a cross-entropy loss function.

B. IMAGE CHUNKING AND MERGING METHOD

The position embedding in the patching module requires a maximum length, which is usually derived from training data. In the context of the Khmer script and other languages without explicit word boundaries, there is no notion of maximum text length: the text can be arbitrarily long. Using a fixed maximum text length leads to two drawbacks: (1) the inability to generalize to textline images longer than the training images, and (2) high self-attention complexity in TR$_{\mathrm{ENC}}$ for long textline images.

The chunking and merging technique was originally introduced by Diaz et al. [4] as a strategy to handle arbitrarily long Latin textline images by splitting each input image into overlapping chunks. Additional padding was added to the first and last chunks to make all chunks in the batch share the same width for mini-batch training. However, Diaz et al. [4] limited this chunking strategy to the 1D CTC-based models only.

We adopted, adapted, and incorporated the chunking and merging technique into the 2D Transformer-based Khmer text recognition to reduce the self-attention complexity for arbitrarily long textline images from a quadratic to a linear relation. As depicted in Fig. 13, if we consider an input that is five times wider (denoted as $5W$), the self-attention complexity is $O(5^2C)$ as compared to $O(C)$ for an input width of W. However, if we divide the input with a width of $5W$ into five consecutive chunks, each having a width of W and a self-attention complexity of $O(C)$. The features of each chunk are then processed independently and subsequently merged together. This approach results in the overall self-attention complexity of only $O(5C)$, effectively reducing the computational burden by five times. As quantitatively shown in Fig. 14, the encoder complexity is reduced from 32.3 billion floating-point operations (FLOPs) to only 16.8 billion FLOPs when the chunking technique is applied at a textline width of 2000 pixels. Therefore, by employing the chunking and merging technique, it becomes feasible to handle input of arbitrary length without the need for resizing it to a fixed size and the resolution of the small-sized characters is not undesirably affected.

The proposed 2D Transformer-based architecture, incorporating the chunking and merging technique, is illustrated in Fig. 15. The figure depicts an image, denoted as I, divided into two overlapping chunks: Chunk (1) and Chunk (2).

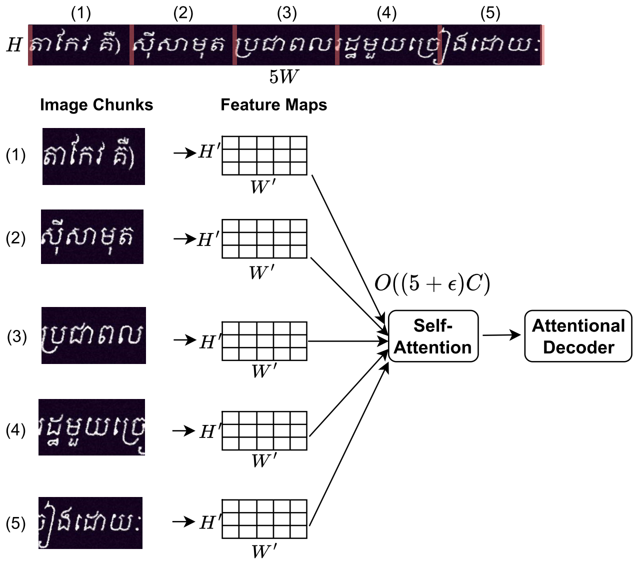


FIGURE 13. Self-attention complexity reduction by the image chunking technique. By splitting a long textline image (e.g., 500 pixels) with a width of $5W$ into five smaller chunks, each of which has a width of W (e.g., 100 pixels), the self-attention complexity is reduced from a quadratic relation, $O(5^2C)$, to a linear relation, $O((5+\epsilon)C)$ where ϵ is a tiny overlapping margin (e.g., zero pixel). Vertical dashed lines denote chunk boundaries. Best viewed in color.

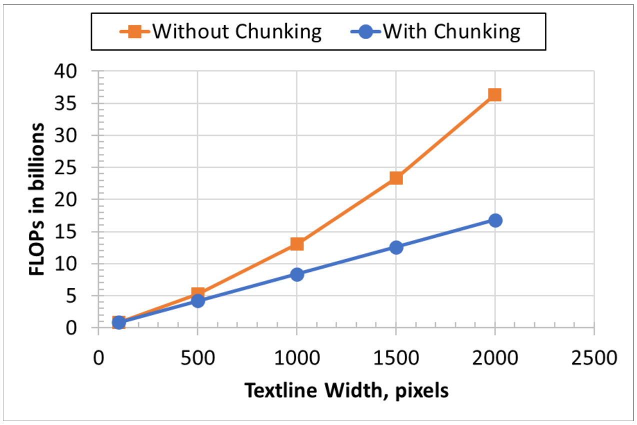


FIGURE 14. The encoder complexity comparison: chunking vs. no chunking. The graph is based on a VGG feature extractor with a patch size (k_1, k_2) of (3,1). A chunk width of 100 pixels is used, assuming an overlapping margin, ϵ, of 0. Best viewed in color.

The features of Chunk (1) and Chunk (2) are processed independently and finally merged together before decoding, as indicated by the blue and green components in the figure. By consolidating the features of the chunks before character decoding, rather than decoding characters independently in each chunk and then merging the resultant characters, the Transformer decoder gains the ability to utilize all previously decoded characters across all chunks for predicting the next character. Additional overlapping margins are introduced to mitigate image discontinuity and counteract undesirable boundary effects, as illustrated in the same figure.

Unlike in NLP tasks, such as machine translation, where long-range feature dependencies are crucial, local visual feature dependencies play a more significant role in accurately predicting characters in text recognition. By dividing an image into smaller independent chunks, we effectively mitigate long-range dependencies and focus on modeling local dependencies instead.

In addition, the chunking technique also enhances efficiency of mini-batch processing during training, as shown in Fig. 16. When the chunking technique is applied, the input images in a mini-batch do not require padding to the maximum width, as in Fig. 16(a), but only to a much smaller chunk width, as in Fig. 16(b). Self-attention computations are performed at the chunk level instead of the whole image level. This leads to enhanced localized spatial connections and a decrease in unnecessary computations within the padded regions.

VI. DESIGN OF EXPERIMENTS, RESULTS, AND ANALYSES

In this section, we discuss the experimental setup, present the results, and provide the analyses. We use the synthetic validation dataset to identify the optimal chunk width (i.e., W) and patch size (i.e., k_1, k_2) and to assess the impact of the chunking technique in Section VI-A. The baseline models were also set up for performance benchmarking in the same section.

To assess the model accuracies on the real Khmer datasets and other scripts, we compare the proposed 2D approach with the baseline models on the Khmer ID card and KHOB datasets, followed by transfer learning on the Sleuk Rith dataset in Section VI-B and other low-resource non-Latin scripts in Section VI-C. Finally, we provide two crucial analyses: performance versus textline length and failure cases in Section VI-D.

Following Baek et al. [16], we used two modified CNN backbones, namely VGG [39] and ResNet [40], in our experiments. In principle, any published CNN architectures can be employed as a feature extractor, and enhancing a backbone's complexity typically results in a moderate recognition gain [4]. Nevertheless, VGG, characterized by its lower complexity, and ResNet, known for its higher complexity, are commonly used as a CNN feature extractor in text recognition. Using the VGG and ResNet backbones allows the recognition assessment of the backbone complexity on the Khmer script in this study. The detailed specifications of the modified VGG and ResNet architectures are provided in Tables 2 and 3, respectively. In all cases, input images were resized to a fixed height by preserving the aspect ratio and allowing the images to be arbitrarily long, to handle writing without explicit word boundaries. No assumption was made about the maximum image width, which is often assumed in most Latin scene text recognition methods. As a result, a long textline image will have a higher number of chunks, compared with a short textline image. For a textline image that is shorter than a chunk width, zero padding is added.

As for model training, we used the Adam optimizer with an initial training rate of 10^{-4} for the first 15 epochs for fast

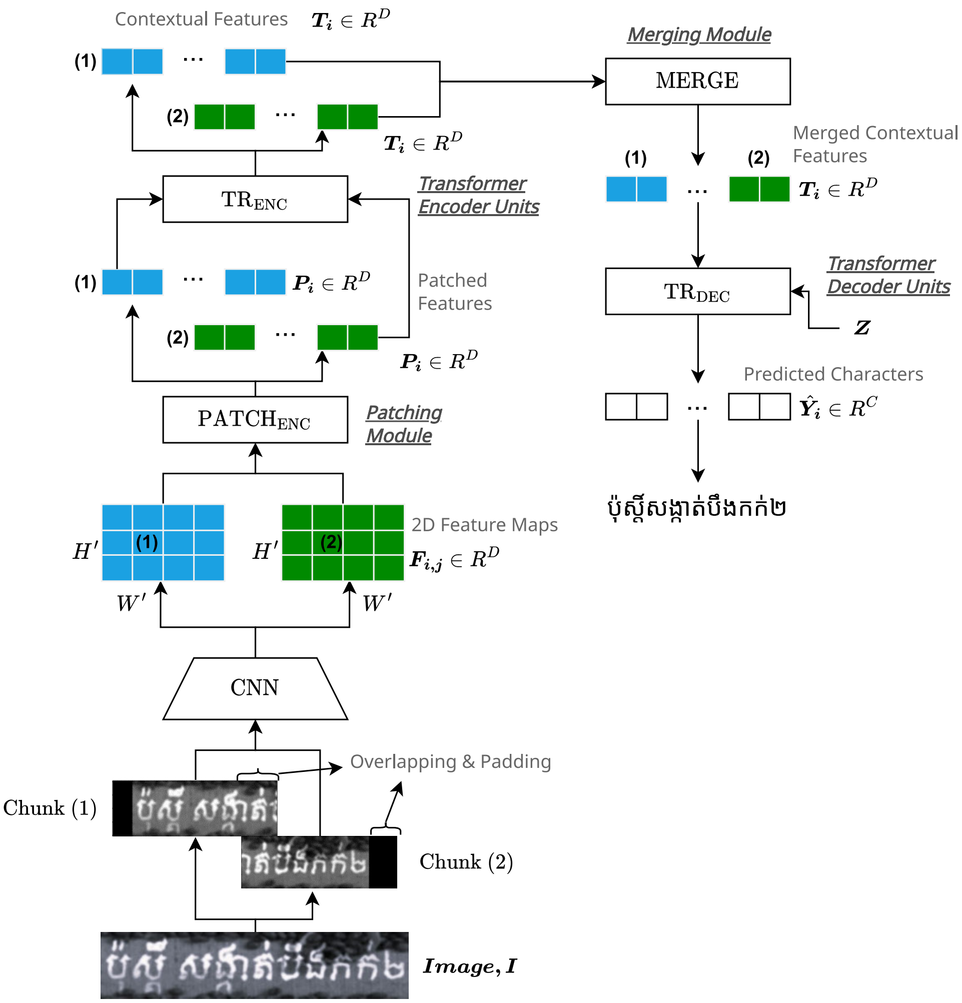


FIGURE 15. The proposed 2D Transformer-based encoder-decoder architecture, incorporating the chunking and merging technique. An input image, I, is split into two smaller chunks (i.e., Chunk (1) and Chunk (2) in this case). The chunks (blue vs. green) are independently processed by the PATCH$_{\text{ENC}}$ and the TR$_{\text{ENC}}$. However, the contextual features, T, are merged and position-encoded in the MERGE module. The merged T and a context sequence, Z, are fed to the Transformer decoder units, TR$_{\text{DEC}}$, to obtain the predicted characters, $\hat{Y}$. Best viewed in color.

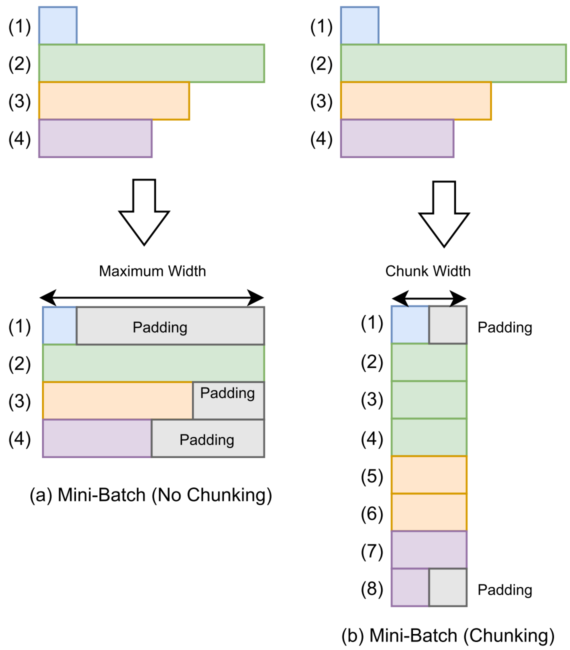
convergence, followed by cyclic learning between 10^{-4} and 10^{-5} for another 15 epochs for stabilization, and then cyclic learning between 10^{-5} and 10^{-6} for the remaining epochs. In each epoch, random samples comprising 100,000 images were selected. The training lasted for 100 epochs. A standard multi-class cross-entropy loss was used in all experiments.

Regarding the evaluation metric, in contrast to the Latin script, computing a CER on the Khmer script requires an extra conditioning step because of the non-canonical order of characters mentioned earlier. Therefore, we applied the character normalization described in Section II before computing a CER. Since the Khmer script does not have any explicit word boundaries, only CER is used as an evaluation metric. Although it is feasible to apply a word segmentation algorithm to both the predicted and ground-truth texts before calculating a word error rate (WER), errors arising from word segmentation can distort the accuracy of the resulting WER. It should be mentioned that a WER is always greater than or equal to a CER for a given Khmer text.

A. OPTIMAL MODEL CONFIGURATIONS AND BASELINE MODELS

1) EXPERIMENT SETUP

We begin by evaluating the impact of the chunking and merging technique and determining the optimal settings, namely patch size (i.e., k_1, k_2) and chunk width (i.e.,W). For this purpose, we utilize the synthetic validation dataset, as previously described in Section IV. In addition to our proposed 2D models, we also set up the 1D baseline models to assess the effect of the chunking technique, determine the optimal chunk width, and make performance comparisons. We also set up other baseline models based on the existing CTC-based and attention-based methods for the purpose of performance comparisons.



(a) Mini-Batch (No Chunking)

(b) Mini-Batch (Chunking)

FIGURE 16. Illustration of image chunking operation. There are four textline images of different widths. Without image chunking (a), the four input images are padded to the maximum width. This results in unnecessary computations in the padded regions. With image chunking (b), the four input images are split into eight smaller chunks, only the first and last of which are padded to a much smaller chunk width for efficient mini-batch training. Best viewed in color.

TABLE 2. The modified VGG architecture [16].

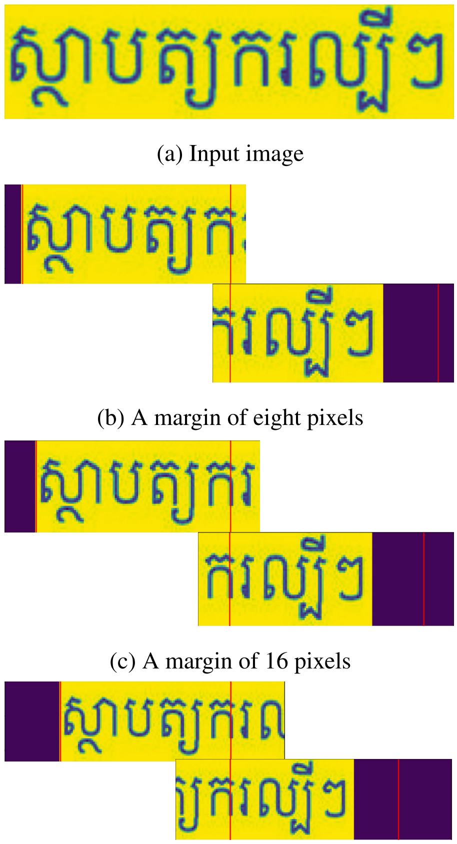
Layer	Configuration	Output (H'×W'×C)
Input	grayscale	48×132×1
Conv1	c:64 k:3×3 s:1×1 p:1×1	48×132×64
Pool1	k:2×2 s:2×2	24×66×64
Conv2	c:128 k:3×3 s:1×1 p:1×1	48×132×128
Pool2	k:2×2 s:2×2	12×33×128
Conv3	c:256 k:3×3 s:1×1 p:1×1	12×33×256
Conv4	c:256 k:3×3 s:1×1 p:1×1	12×33×256
Pool3	k:1×2 s:1×2	6×33×256
Conv5	c:512 k:3×3 s:1×1 p:1×1	6×33×512
BatchNorm1	-	12×33×512
Conv6	c:512 k:3×3 s:1×1 p:1×1	6×33×512
BatchNorm2	-	12×33×512
Pool4	k:1×2 s:1×2	3×33×512
Conv7	c:512 k:3×3 s:1×1 p:1×1	2×32×512

For the 1D baseline models (i.e., those using 1D CNN feature maps), the resulting feature maps from both CNN architectures (VGG and ResNet) have a unit height, with a fixed height of 42 pixels. For each CNN architecture, we set up three models using three different chunk widths (W): 64, 100, and 128 pixels.

For our proposed 2D models (i.e., those using 2D CNN feature maps), we dropped the last two MaxPooling layers (i.e., Pool3 and Pool4) from the VGG architecture, and the last MaxPooling layer (i.e., Pool3) and the last two convolutional layers (i.e., Conv6 and Conv7) from the ResNet architecture. The resulting feature maps from the VGG and ResNet have 12 units of height, using a fixed height of 48 pixels. For each CNN architecture, we set up three models using asymmetric patches (i.e., k_1, k_2) of (2,1), (3,1), and (4,1), which downsample the feature maps in the height dimension by factors of two, three, and four, respectively. Only asymmetric patches were used because textline images

TABLE 3. The modified ResNet architecture [16].

Layer	Configuration	Output (H'×W'×C)
Input	grayscale	48×132×1
Conv1	c:32 k:3×3 s:1×1 p:1×1	48×132×32
Conv2	c:64 k:3×3 s:1×1 p:1×1	48×132×64
Pool1	k:2×2 s:2×2	24×66×64
Block1	[c:128 k:3×3 s:1 × 1 p:1 × 1 c:128 k:3 × 3 s:1 × 1 p:1 × 1] × 1	24×66×128
Conv3	c:128 k:3×3 s:1×1 p:1×1	24×66×128
Pool2	k:2×2 s:2×2	12×33×128
Block2	[c:256 k:3 × 3 s:1 × 1 p:1 × 1 c:256 k:3 ×3 s:1 × 1 p:1 × 1] × 2	12×33×256
Conv4	c:256 k:3×3 s:1×1 p:1×1	12×33×256
Pool3	k:2×2 s:1×2 p:1×0	6×34×256
Block3	[c:512 k:3 × 3 s:1 × 1 p:1 × 1 c:512 k:3 × 3 s:1 × 1 p:1 × 1] × 5	6×34×512
Conv5	c:512 k:3×3 s:1×1 p:1×1	6×34×512
Block4	[c:512 k:3 × 3 s:1 × 1 p:1 × 1 c:512 k:3 × 3 s:1 × 1 p:1 × 1] × 3	6×34×512
Conv6	c:512 k:2×2 s:1×2 p:1×0	3×35×512
Conv7	c:512 k:2×2 s:1×1 p:1×1	2×34×512



(a) Input image

(b) A margin of eight pixels

(c) A margin of 16 pixels

(d) A margin of 32 pixels

FIGURE 17. Image chunks with different overlapping margins. (a) Input image. (b) Eight pixels. (c) 16 pixels. (d) 32 pixels. The red lines represent margin boundaries. A fixed height of 48 pixels and a chunk width of 100 pixels are used.

generally have a low height-to-width ratio. The optimal chunk width from the 1D baseline experiments was used. As for the overlapping margin, Fig. 17 shows that an overlapping margin of 16 pixels can sufficiently cover a Khmer character's width while the margins of eight and 32 pixels are too small and too large, respectively. For both the 1D and 2D experiments, an overlapping margin of 16 pixels was, thus, used. Nonetheless, an overlapping margin of 32 pixels or larger can also be employed to enhance feature continuity and context at chunk boundaries, although this may lead to additional complexity.

As for the other baseline models, we set up three representative baseline CTC-based and attention-based models: CRNN [30], TRBA (TPS-ResNet-BiLSTM-Attention) [16], and TRBC (TPS-ResNet-BiLSTM-CTC) [16]. To avoid resizing input images to a fixed size, thin-plate spline (TPS) transformation was dropped from TRBA and TRBC. Chunking was not applied to these baseline models.

2) RESULTS AND ANALYSES

In this section, we present the performance results of different model configurations on the synthetic validation dataset to identify optimal configurations for the subsequent evaluations on the real datasets. As shown in Table 4, the 1D baseline models consistently outperformed the baseline CRNN, TRBA, and TRBC on the synthetic validation dataset. Compared with a VGG backbone, using a ResNet backbone further improved the CERs at the cost of a large increase in the number of FLOPs and the number of parameters. This is because the ResNet backbone has a greater receptive field and can extract richer semantic features than the VGG. The results also indicate that localizing self-attention by means of the modified image chunking and merging strategy consistently led to a further reduction in the CERs (up to 0.3%). The choice of chunk width had a marginal impact ($\leq 0.1\%$) on the resulting CERs, although a chunk width of 100 pixels appeared to be optimal.

By using the optimal chunk width from the 1D baseline experiments, Table 4 shows that our proposed 2D models achieved lower CERs, compared with the baseline CRNN, TRBA, and TRBC. Compared with the 1D baseline models, our proposed 2D models achieved improved CERs regardless of the backbone, which suggests that explicitly capturing local 2D spatial dependencies is beneficial for Khmer character recognition. The performance of our proposed 2D models using a VGG was comparable with that of the 1D baseline models using a ResNet. This shows that a deep CNN implicitly encodes local spatial dependencies through its depth, whereas a shallow CNN needs to explicitly model spatial relations through self-attention layers. Nevertheless, the former is computationally more expensive: the 1D baseline models using a ResNet performed approximately more than twice as many FLOPs as our proposed 2D models using a VGG. For 2D cases, the choice of patch size had only a marginal impact ($\leq 0.1\%$) on the CERs, although a (3,1) patch appeared to be optimal. Therefore, a patch size of (3,1) was used in the subsequent experiments.

B. PERFORMANCE ON THE KHMER ID CARD, KHOB, AND SLEUK RITH DATASETS

1) EXPERIMENT SETUP

In the previous section, we identified the optimal 1D baseline models as well as our proposed optimal 2D models. In this section, we assess the performance of these models on various real Khmer datasets, encompassing different modalities, such as the Khmer ID card, KHOB, and Sleuk Rith datasets described in Section IV. For the Khmer ID card and KHOB datasets, we directly applied the optimal models from Section VI-A without any fine-tuning training. Evaluating the performance on these real datasets is crucial, as it provides insights into how well the models, trained solely on synthetic data, can generalize in real-world settings. This becomes particularly significant for low-resource non-Latin scripts, where only limited labeled data is available for fine-tuning.

Regarding the historical, handwritten Sleuk Rith dataset, both the baseline models and our proposed optimal 2D models underwent fine-tuning using transfer learning. The experiments were conducted with two main objectives. The first objective is to evaluate our proposed 2D approach on historical handwritten modality in comparison to the baseline models and the SOTA model by Valy et al. [21]. The second objective is to demonstrate the effectiveness of transfer learning from the printed text modality to the handwritten text modality.

2) RESULTS AND ANALYSES

Regarding the Khmer ID card dataset, the text boxes containing full names in Khmer and ID numbers in Latin were detected and cropped using the CRAFT text detector, without any manual correction. The average height of the cropped images is 154 pixels, with a standard deviation of 36 pixels. In Fig. 9, the textline for the Khmer name field is depicted, written in two calligraphic fonts known for their inherent difficulty in reading, often due to ambiguous characters and ligatures. Furthermore, since the ID card images were captured using smartphone cameras, they exhibit diverse conditions, varying in quality, resolution, blurring, and lighting. Due to the nature of the Khmer script not being explicitly trained in the CRAFT text detector, there are instances where the text detector partially misses certain characters, such as subscripts, vowels, or diacritics, either below or above the base characters. Notwithstanding these challenges, our proposed 2D models achieved CERs that were approximately 50% or more lower than those of the baseline Tesseract OCR, CRNN, TRBC, and TRBA for both the name and number fields, as demonstrated in Table 5. The substantial disparities in CERs between the full name in Khmer and the ID number in Latin can be attributed to the inherent complexity of the Khmer script compared to Latin. Once again, the enhancements in CERs accomplished

TABLE 4. Results of our proposed 2D models and the baseline models on the synthetic validation dataset. Bold: highest/lowest. Italic: second highest/lowest.

Model	Backbone	Feature Maps	Decoder	Params(M)	FLOPs(G)[1]	CER (%)
CRNN [30] (Baseline)	VGG	1D	CTC	7.89	0.77	2.96
TRBC [16] (Baseline)	ResNet	1D	CTC	47.19	6.32	2.44
TRBA [16] (Baseline)	ResNet	1D	Attention-based	47.98	6.32	1.98
1D Baseline (No Chunking)	VGG	1D	Tr. Decoder[4]	21.74	1.03	1.53
1D Baseline (W=64)[2]	VGG	1D	Tr. Decoder	21.74	1.92	1.30
1D Baseline (W=100)[2]	VGG	1D	Tr. Decoder	21.74	1.34	1.24
1D Baseline (W=128)[2]	VGG	1D	Tr. Decoder	21.74	1.63	1.34
1D Baseline (No Chunking)	ResNet	1D	Tr. Decoder	**60.44**	6.59	0.94
1D Baseline (W=64)[2]	ResNet	1D	Tr. Decoder	**60.44**	12.64	0.70
1D Baseline (W=100)[2]	ResNet	1D	Tr. Decoder	**60.44**	8.61	0.62
1D Baseline (W=128)[2]	ResNet	1D	Tr. Decoder	**60.44**	10.39	0.64
Our Proposed 2D K(2,1)[3]	VGG	2D	Tr. Decoder	23.40	4.93	0.93
Our Proposed 2D K(3,1)[3]	VGG	2D	Tr. Decoder	23.63	4.22	0.85
Our Proposed 2D K(4,1)[3]	VGG	2D	Tr. Decoder	23.88	3.88	0.90
Our Proposed 2D K(2,1)[3]	ResNet	2D	Tr. Decoder	58.70	**19.38**	**0.51**
Our Proposed 2D K(3,1)[3]	ResNet	2D	Tr. Decoder	58.93	*18.67*	**0.51**
Our Proposed 2D K(4,1)[3]	ResNet	2D	Tr. Decoder	*59.18*	18.33	*0.53*

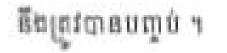
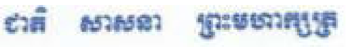
[1] Inference FLOPs evaluated on a 42 × 100 input for 1D cases and a 48 × 100 input for 2D cases.
[2] Using an overlapping margin of 16 pixels.
[3] Using a chunking width and margin of 100 and 16 pixels, respectively.
[4] Transformer decoder.

TABLE 5. Character error rate (CER in %) results on the Khmer ID card, KHOB, and Sleuk Rith datasets. Bold: lowest. Italic: second lowest.

Model	Khmer ID Card		KHOB	Sleuk Rith[1]
	Name	Number		
Valy et al. [21][2]				6.16
Tesseract OCR (Baseline)	27.78	5.64	11.37	94.88
CRNN [30] (Baseline)	7.34	2.36	6.17	4.81
TRBC [16] (Baseline)	6.80	0.76	5.42	3.26
TRBA [16] (Baseline)	7.23	3.53	7.77	3.80
1D-VGG Baseline	4.31	1.61	3.32	2.92
1D-ResNet Baseline	4.24	**0.57**	**2.55**	**2.46**
Our Proposed 2D-VGG	*3.47*	1.27	*3.03*	*2.66*
Our Proposed 2D-ResNet	**3.00**	*0.62*	3.41	**2.46**

[1] Fine-tuning from the respective Khmer script model.
[2] SOTA model on the Sleuk Rith dataset.

TABLE 6. Failure cases of the 2D-ResNet model due to low resolution caused by image compression and double scanning. The first row contains the input images. The second row is the corresponding ground-truths. The 3rd until the 6th rows are the predicted texts from the different models. CERs are provided in the bracket. The errors are highlighted in red. Best viewed in color.

	Failure Cases	
Cases		
Label	នឹងត្រូវបានបញ្ចប់ ។	ជាតិ សាសនា ព្រះមហាក្សត្រ
1D-VGG	នឹងត្រូវបានបញ្ចប់ ៕ (5%)	ជាតិ សាសនា ព្រះមហាក្សត្រ (0%)
2D-VGG	នឹងត្រូវបានបញ្ចប់ ។ (0%)	ជាតិ សាសនា ព្រះមហាក្សត្រ (0%)
1D-ResNet	នឹងត្រូវបានបញ្ចប់ ។ (0%)	ជាតិ សាសនា ព្រះមហាក្សត្រ (0%)
2D-ResNet	នឹងត្រូវបាន1/ញ្ចប់ ។ (10%)	ជាតិ . សាសនា ព្រះមហាក្សត្រ (8%)

by the proposed 2D models over the 1D baseline models highlight the robustness obtained from capturing local 2D spatial dependencies in this real dataset.

Unlike the Khmer ID card dataset, the KHOB dataset benefits from relatively clean backgrounds as it is derived from PDF documents. However, the dataset suffers from lower resolution caused by image compression, which disproportionately affects small-sized characters, such as vowels, subscripts, and diacritics. The average height of the cropped images in this dataset is merely 27 pixels, with a standard deviation of 6 pixels. Our proposed 2D models demonstrated a significant reduction of approximately 50% or more in CERs compared to the baseline Tesseract OCR, CRNN, TRBC, and TRBA, as indicated by the data presented in Table 5. Our proposed 2D-VGG model exhibited superior performance in terms of CER when compared to the 1D-VGG baseline model. However, the 1D-ResNet baseline model outperformed our proposed 2D-ResNet model. This is because unlike the shallow or 1D models, the deep 2D model with a large receptive field requires enough vertical resolution to extract meaningful feature dependencies. In other words, 2D feature modeling can be likened to zooming in on the details of visual features, which is useful for accurate recognition when there are enough vertical details (i.e., resolution). In the case of low resolution images, zooming out, akin to 1D modeling, is more useful for recognition. In addition, the legibility of certain Khmer characters, such as diacritics, vowels, and subscripts is very sensitive to image resolution as illustrated in Fig. 7. Table 6 presents sample failure cases of the 2D-ResNet model caused by low resolution caused by image compression and double scanning, along with recognition results from other models.

So far, we have observed that our proposed 2D models trained solely on synthetic data can effectively generalize on the real printed text modality without requiring fine-tuning,

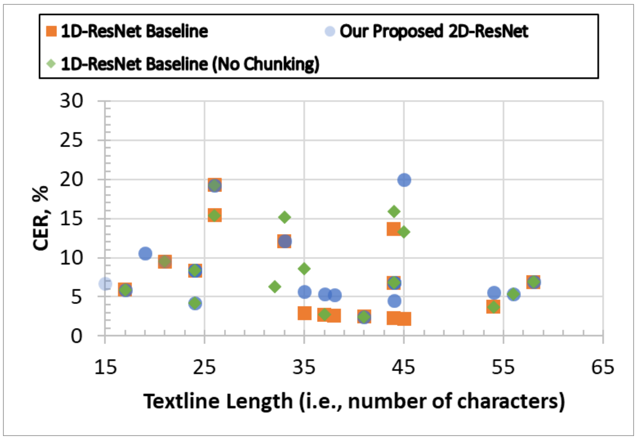


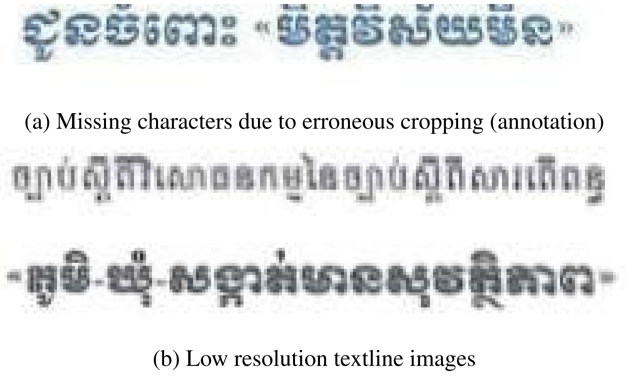
FIGURE 18. The cross-plot of image-level CERs vs. textline length (i.e., character count) for our proposed 2D model using a ResNet and the 1D-ResNet baseline models. The chunking technique was applied in both the *1D-ResNet baseline* and *Our proposed 2D-ResNet* models. Best viewed in color.

as demonstrated in comparison to the baseline models. In this part, we focus on evaluating the performance of our proposed 2D models on the Sleuk Rith dataset. Due to the unique characteristics of the Sleuk Rith dataset, which comprises historical handwritten palm leaf documents, the modality of the dataset differs significantly from the synthetic training data. Consequently, the models trained on synthetic data cannot be directly applied without undergoing the process of fine-tuning. Both the baseline models (excluding Tesseract) and our proposed 2D models underwent fine-tuning using a small training set specific to the Sleuk Rith dataset. Subsequently, the performance of these fine-tuned models was evaluated on a randomly selected test set. As demonstrated in Table 5, both the fine-tuned baseline models and our fine-tuned models achieved lower CERs compared to the current SOTA model by Valy et al. [21]. Furthermore, our proposed 2D models exhibited superior performance to the baseline CRNN, TRBC, and TRBA models by a margin of up to 2.4%, while also attaining slightly lower CERs when compared to the 1D baseline models. The results highlight the robustness of our proposed 2D approach and the feasibility of transfer learning from Khmer printed text to historical handwritten text with a limited labeled dataset.

C. PERFORMANCE ON OTHER RELATED SCRIPTS: THAI, LAOS, BURMESE, VIETNAMESE, AND HINDI SCRIPTS

1) EXPERIMENT SETUP

The preceding sections focus on evaluating the performance of our proposed 2D models specifically on the Khmer script. In this section, we shift our attention to assessing the robustness and transferability of the proposed 2D models beyond the Khmer script. We accomplished this by evaluating the fine-tuning performance on other low-resource scripts that share similar features with the Khmer script and have limited labeled data available for training. We performed fine-tuning on the baseline models and our proposed 2D models using Thai, Laos, Burmese, Vietnamese, and Hindi scripts. These particular scripts were selected to represent the diverse range of low-resource non-Latin scripts that share similar features with the Khmer script. For each script, we created a small-scale dataset consisting of 200,000 images containing document OCR and scene text. From this dataset, a randomly selected 5% was reserved as a test set for evaluation purposes. The performance of our proposed 2D models was then compared against Tesseract and the fine-tuned baseline models. It is worth noting that since Tesseract 4.0 already supports the selected scripts, no additional fine-tuning was required for Tesseract.



(a) Missing characters due to erroneous cropping (annotation)

(b) Low resolution textline images

FIGURE 19. Some examples of recognition failures in the KHOB dataset. (a) Missing characters due to erroneous cropping (annotation). (b) Low resolution textline images.

2) RESULTS AND ANALYSES

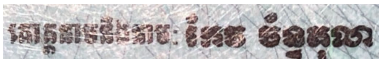
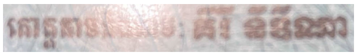
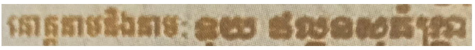
Despite Tesseract being trained with data augmentation techniques [5], the resulting CERs for all the scripts were approximately one order of magnitude higher compared to the CERs achieved by the fine-tuned baseline models and our fine-tuned models. This performance difference is demonstrated in Table 7. A similar observation was made by Namysl and Konya [41] in their study on distorted Latin text images. However, the results indicate that Tesseract achieved significantly lower CERs for scripts that have explicit word boundaries, such as Vietnamese and Hindi, compared to scripts without explicit word boundaries. This suggests the recognition challenge caused by implicit word boundaries. Among the baseline TRBC and TRBA, the outcomes indicate a preference for a CTC-based decoder (TRBC) over an attention-based decoder (TRBA), given the same backbone. Across all the scripts evaluated, our proposed 2D models consistently outperformed the baseline Tesseract OCR, CRNN, TRBC, and TRBA and achieved lower CERs compared with the 1D baseline models. This emphasizes the efficacy of our proposed 2D approach in achieving robust generalization beyond the Khmer script. Despite the limited fine-tuning labeled data available for each script, the results presented in Table 7 demonstrate the transfer learning's robustness from the Khmer script to other low-resource non-Latin scripts with similar characteristics. Consequently, the Khmer OCR method can serve as a valuable baseline method for low-resource non-Latin scripts.

TABLE 7. Character error rate (CER in %) results on the test sets of Thai, Laos, Burmese, Hindi, and Vietnamese scripts. Bold: lowest. Italic: second lowest.

Models	Thai	Laos	Burmese	Hindi	Vietnamese
Tesseract OCR[1] (Baseline)	38.66	35.62	45.84	22.73	28.62
CRNN [30][2] (Baseline)	3.59	3.63	3.73	2.91	3.60
TRBC [16][2] (Baseline)	4.39	3.12	3.46	2.56	3.53
TRBA [16][2] (Baseline)	3.80	3.31	3.67	2.49	7.56
1D-VGG Baseline[2]	2.23	2.77	3.28	1.90	2.20
1D-ResNet Baseline[2]	*1.32*	*2.29*	**2.60**	*1.42*	**1.69**
Our Proposed 2D-VGG[2]	1.93	2.55	3.01	1.70	*2.01*
Our Proposed 2D-ResNet[2]	**1.31**	**2.19**	*2.61*	**1.30**	**1.69**

[1] No fine-tuning.
[2] Fine-tuning from the respective Khmer script model.

TABLE 8. Some failure cases associated with the failure of the text detector to capture the entire textlines and image distortions. The first row contains the input images. The second row is the corresponding ground-truths. The 3rd until the 6th rows are the predicted texts from the different models. CERs are provided in the bracket. The errors are highlighted in red. Best viewed in color.

	Failure Cases		
Cases			
Label	គោត្តនាមនិងនាមៈ កែវ ច័ន្ទតុលា	គោត្តនាមនិងនាមៈ គីរី នីឌីណា	គោត្តនាមនិងនាមៈ ទុយ ផល្គុនសុភ័ក្ត្រា
1D-VGG	អត្តនាមេនឹងតាមៈ ទឹកគ ច័ន្ទគុណា (41%)	កោត្តតានអាស៊ាម គឺះ ឱមណា (56%)	នោត្តតាមដឹងតាមៈ ឆ្កាយ ៨០១ សុគីក្រោ (56%)
2D-VGG	អោត្តតាមនឹងតាបៈ ទឹក6 ចំន្ទគុណា (51%)	គោត្តនាមនេះម៉ា គឺះ និមណា (41%)	គោត្តតាមនឹងឆាមៈ ៦០១៨ល១សុភព្រេ (47%)
1D-ResNet	សោត្តនាមនឹងកាម.កែវ ចំន្ទតលា (31%)	កោត្តតានាង នាមៈគឺះ នីឌីណា (33%)	នោត្តនាមនឹងនាមៈ ទួយ ៩លនសុគី ក្រោ (36%)
2D-ResNet	សេត្តនាមនឹងខាមៈ កែវ ច័ន្ទតុលា (14%)	គោត្តនាន ការសមៈ គីរី នីឌីណា (22%)	នោត្តនាមនឹងនាមៈ ទួយ ផលនសេគី កា (39%)

D. ADDITIONAL ANALYSES

Finally, we conducted two crucial analyses: performance versus textline length and failure cases. Evaluating the CERs in relation to the textline length (character count) is vital as it helps us determine whether our proposed 2D models can effectively recognize textline images of varying lengths, even those that are arbitrarily long. Similarly, the analysis of failure cases allows us to identify the weaknesses of the models and training data, providing valuable insights for further improvements. Regarding the analysis of performance versus textline length, the KHOB dataset was employed due to its inclusion of longer textline images, providing a suitable basis for evaluation when compared to the Khmer ID card dataset. On the other hand, for the analysis of failure cases, the Khmer ID card dataset was utilized. This dataset contains particularly challenging textline images in natural scenes, making it suitable for assessing the models' weaknesses and areas for improvement.

1) ANALYSIS OF CERS VS. TEXTLINE LENGTH

The correlation between image-level CERs and textline length on the KHOB dataset for our proposed 2D model using a ResNet and the 1D-ResNet baseline models, with and without the application of the chunking technique, is depicted in Fig. 18. According to the figure, it can be inferred that the chunking technique does not negatively impact model performance, even when dealing with long textline images. This suggests that the integration of the chunking and merging technique in the proposed 2D Transformer-based Khmer text recognition method can effectively reduce model complexity when dealing with arbitrarily long input, without sacrificing accuracy.

However, Fig. 18 also demonstrates that higher CERs are observed for textline images containing between 25 and 45 characters. Two primary root causes were identified: (1) missing characters caused by erroneous cropping and (2) low resolution textline images caused by image compression and double scanning, as illustrated in Figs. 19(a) and (b), respectively.

2) ANALYSIS OF FAILURE CASES

We identified the failure cases on the Khmer ID card dataset, some of which are shown in Table 8, as well as the causes. The predicted texts, along with the corresponding CERs from the proposed 2D models and the baseline 1D models, are presented in rows 3 to 6 for each example failure case (column). Two findings can be derived from these results. Firstly, the ResNet-based models outperformed the VGG-based models on these challenging cases. Secondly, the 2D models achieved lower CERs, compared to the 1D models in this dataset.

Two main causes of performance degradation were identified. The first is that the text detector failed to detect some subscripts, below or above vowels, and diacritics, as shown in the first and third cases of Table 8. The second is image distortions caused by light reflection, scratching, and blurring, as shown in all cases of Table 8. While the latter cause is common for general scene text images in natural scenes, the former happens only with the Khmer and non-Latin scripts with complex features.

VII. CONCLUSION AND FUTURE WORK

We present a convolutional Transformer-based text recognition approach for low-resource non-Latin scripts with character stacking, diacritics, ligatures, and writing without explicit word boundaries. Coupled with a modified chunking and merging strategy, the proposed method can handle arbitrarily long textline images without resizing them to a fixed size, reduce the complexity of model training, and model local 2D spatial dependencies. Using the Khmer script as our case study, our proposed 2D models outperformed the baseline models across multiple input modalities. The fine-tuning results on other low-resource non-Latin scripts suggest that OCR methods for other related scripts can be efficiently fine-tuned from the Khmer OCR method by transfer learning, even with limited labeled data. Thus, by analogy with the AI-complete concept, a Khmer OCR method can be considered as non-Latin-complete and can be used as a low-resource non-Latin baseline method.

Future work will involve including additional non-Latin scripts to further validate the proposed approach, and improving feature extraction, particularly for distorted inputs. The design of a joint text detection and recognition system for the Khmer and other non-Latin scripts will also be investigated. In addition, we will perform experiments with real Khmer handwritten data collected from different age groups and professions to further validate the proposed method.

REFERENCES

[1] X. Liu, G. Meng, and C. Pan, "Scene text detection and recognition with advances in deep learning: A survey," *Int. J. Document Anal. Recognit. (IJDAR)*, vol. 22, no. 2, pp. 143–162, Jun. 2019, doi: 10.1007/s10032-019-00320-5.

[2] I. Krylov, S. Nosov, and V. Sovrasov, "Open images V5 text annotation and yet another mask text spotter," in *Proc. Asian Conf. Mach. Learn.*, vol. 157, Nov. 2021, pp. 379–389.

[3] A. Singh, G. Pang, M. Toh, J. Huang, W. Galuba, and T. Hassner, "TextOCR: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text," in *Proc. IEEE/CVF Conf. On Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 8802–8812.

[4] D. Hernandez Diaz, S. Qin, R. Ingle, Y. Fujii, and A. Bissacco, "Rethinking text line recognition models," 2021, *arXiv:2104.07787*.

[5] R. Smith. (2016). *Tesseract Blends Old and New OCR Technology*. [Online]. Available: https://github.com/tesseract-ocr/docs/tree/master/das_tutorial2016

[6] R. Buoy, N. Taing, S. Chenda, and S. Kor, "Khmer printed character recognition using attention-based Seq2Seq network," *HO Chi Minh City Open Univ. J. Sci. Eng. Technol.*, vol. 12, no. 1, pp. 3–16, Apr. 2022, doi: 10.46223/hcmcoujs.tech.en.12.1.2217.2022.

[7] P. Cao, "Recognizing Chinese texts with multi-width feature extractor and attention-based fusion," in *Proc. Int. Conf. Comput. Inf. Sci. Artif. Intell. (CISAI)*, Sep. 2021, pp. 317–321, doi: 10.1109/CISAI54367.2021.00067.

[8] Y. Fujii, K. Driesen, J. Baccash, A. Hurst, and A. C. Popat, "Sequence-to-label script identification for multilingual OCR," in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, Nov. 2017, doi: 10.1109/icdar.2017.35.

[9] S. Gunna, R. Saluja, and C. V. Jawahar, "Towards boosting the accuracy of non-latin scene text recognition," in *Proc. Document Anal. Recognit. ICDAR Workshops*, 2021, pp. 282–293.

[10] S. Gunna, R. Saluja, and C. V. Jawahar, "Transfer learning for scene text recognition in Indian languages," in *Proc. Document Anal. Recognit. ICDAR 2021 Workshops*, 2021, pp. 182–197.

[11] A. D. Le, H. T. Nguyen, and M. Nakagawa, "An end-to-end recognition system for unconstrained Vietnamese handwriting," *Social Netw. Comput. Sci.*, vol. 1, no. 1, pp. 1–7, Jan. 2020, doi: 10.1007/s42979-019-0001-4.

[12] A. Rehman, A. Ul-Hasan, and F. Shafait, "High performance Urdu and Arabic video text recognition using convolutional recurrent neural networks," in *Proc. Document Anal. Recognit. ICDAR Workshops*, 2021, pp. 336–352.

[13] D. P. Van Hoai, H.-T. Duong, and V. T. Hoang, "Text recognition for Vietnamese identity card based on deep features network," *Int. J. Document Anal. Recognit. (IJDAR)*, vol. 24, nos. 1–2, pp. 123–131, Jun. 2021, doi: 10.1007/s10032-021-00363-7.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[15] R. Atienza, "Vision transformer for fast and efficient scene text recognition," in *Proc. Document Anal. Recognit. (ICDAR)*, 2021, pp. 319–334.

[16] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4714–4722, doi: 10.1109/ICCV.2019.00481.

[17] J. Lee, S. Park, J. Baek, S. J. Oh, S. Kim, and H. Lee, "On recognizing texts of arbitrary shapes with 2D self-attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2326–2335, doi: 10.1109/CVPRW50498.2020.00281.

[18] M. Li, T. Lv, J. Chen, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei, "TrOCR: Transformer-based optical character recognition with pre-trained models," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 13094–13102.

[19] B. Shi, M. 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.*, vol. 41, no. 9, pp. 2035–2048, Sep. 2019, doi: 10.1109/TPAMI.2018.2848939.

[20] F. Zhan and S. Lu, "ESIR: End-to-end scene text recognition via iterative image rectification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2054–2063, doi: 10.1109/CVPR.2019.00216.

[21] D. Valy, M. Verleysen, and S. Chhun, "Data augmentation and text recognition on Khmer historical manuscripts," in *Proc. 17th Int. Conf. Frontiers Handwriting Recognit. (ICFHR)*, Sep. 2020, pp. 73–78, doi: 10.1109/ICFHR2020.2020.00024.

[22] H. Kaing, C. Ding, M. Utiyama, E. Sumita, S. Sam, S. Seng, K. Sudoh, and S. Nakamura, "Towards tokenization and part-of-speech tagging for Khmer: Data and discussion," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 20, no. 6, pp. 1–16, Nov. 2021, doi: 10.1145/3464378.

[23] M. Durdin, J. Horton, M. Sok, and R. Ty, "Spoof-vulnerable rendering in Khmer unicode implementations," in *Proc. 1st Int. Conf. Lang. Technol.*, 2019, pp. 137–140.

[24] D. Valy, M. Verleysen, and S. Chhun, "Text recognition on Khmer historical documents using glyph class map generation with encoder–decoder model," in *Proc. 8th Int. Conf. Pattern Recognit. Appl. Methods*, 2019, doi: 10.5220/0007555507490756.

[25] R. Buoy, N. Taing, and S. Chenda, "Khmer word search: Challenges, solutions, and semantic-aware search," 2021, *arXiv:2112.08918*.

[26] D. Valy, M. Verleysen, S. Chhun, and J.-C. Burie, "Character and text recognition of Khmer historical palm leaf manuscripts," in *Proc. 16th Int. Conf. Frontiers Handwriting Recognit. (ICFHR)*, Aug. 2018, pp. 13–18, doi: 10.1109/ICFHR-2018.2018.00012.

[27] A. Aberdam, R. Litman, S. Tsiper, O. Anschel, R. Slossberg, S. Mazor, R. Manmatha, and P. Perona, "Sequence-to-sequence contrastive learning for text recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15297–15307, doi: 10.1109/CVPR46437.2021.01505.

[28] X. Chen, L. Jin, Y. Zhu, C. Luo, and T. Wang, "Text recognition in the wild," *ACM Comput. Surveys*, vol. 54, no. 2, pp. 1–35, Mar. 2022, doi: 10.1145/3440756.

[29] H. Wang, C. Pan, X. Guo, C. Ji, and K. Deng, "From object detection to text detection and recognition: A brief evolution history of optical character recognition," *WIREs Comput. Statist.*, vol. 13, no. 5, Sep. 2021, doi: 10.1002/wics.1547.

[30] 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.*, vol. 39, no. 11, pp. 2298–2304, Nov. 2017, doi: 10.1109/TPAMI.2016.2646371.

[31] J. Wang and X. Hu, "Gated recurrent convolution neural network for OCR," in *Proc. NeuRIPS*, 2017, pp. 334–343.

[32] F. Borisyuk, A. Gordo, and V. Sivakumar, "Rosetta," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, doi: 10.1145/3219819.3219861.

[33] M. Jaderberg, K. Simonyan, and A. Zisserman, "Others spatial transformer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.

[34] F. Sheng, Z. Chen, and B. Xu, "NRTR: A no-recurrence sequence-to-sequence model for scene text recognition," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 781–786, doi: 10.1109/ICDAR.2019.00130.

[35] N. T. Ly, H. T. Nguyen, and M. Nakagawa, "2D self-attention convolutional recurrent network for offline handwritten text recognition," in *Proc. Document Anal. Recognit. (ICDAR)*, 2021, pp. 191–204.

[36] P. Sok and N. Taing, "Support vector machine (SVM) based classifier for Khmer printed character-set recognition," in *Proc. Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA) Asia–Pacific*, Dec. 2014, pp. 1–9, doi: 10.1109/APSIPA.2014.7041823.

[37] O. Ignat, J. Maillard, V. Chaudhary, and F. Guzmán, "OCR improves machine translation for low-resource languages," in *Proc. Findings Assoc. Comput. Linguistics (ACL)*, 2022, pp. 1–11, doi: 10.18653/v1/2022.findings-acl.92.

[38] M. Yim, Y. Kim, H.-C. Cho, and S. Park, "Synthtiger: Synthetic text image generator towards better text recognition models," in *Proc. Document Anal. Recognit. (ICDAR)*, 2021, pp. 109–124.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. On Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[41] M. Namysl and I. Konya, "Efficient, lexicon-free OCR using deep learning," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 295–301, doi: 10.1109/ICDAR.2019.00055.

RINA BUOY (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in petroleum engineering from Universiti Teknologi PETRONAS, Malaysia, in 2012 and 2022, respectively, and the M.S. degree in IT engineering from the Royal University of Phnom Penh, Cambodia, in 2022. He is currently pursuing the Ph.D. degree with Osaka Metropolitan University, Osaka, Japan.

From 2012 to 2019, he was a Petroleum Engineer and a Research and Development Team Lead with Three60 Energy Asia (formerly, LEAP Energy), Kualar Lumpur, Malaysia. Since 2020, he has been a Machine Learning Engineer with the Techo Startup Center, Ministry of Economy and Finance, Cambodia. His research interests include numerical simulation, stochastic optimization, document recognition and analysis, natural language processing, and machine learning in production.

Mr. Buoy was a recipient of the Vice-Chancellor Award, in 2012, and the Convocation Award, in 2022, from Universiti Teknologi PETRONAS.

MASAKAZU IWAMURA (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in engineering from Tohoku University, Japan, in 1998, 2000, and 2003, respectively. He is currently an Associate Professor with the Department of Core Informatics, Graduate School of Informatics, Osaka Metropolitan University. His research interests include text and object recognition and visually impaired assistance. He received awards, including the IAPR/ICDAR Young Investigator Award, in 2011; the Best Paper Awards from IEICE, in 2008 and 2021; the IAPR/ICDAR Best Paper Award, in 2007; the IAPR Nakano Award, in 2010; the ICFHR Best Paper Award, in 2010; the MVA Best Paper Award, in 2017; and the ASSET Best Paper Award, in 2023. He was the Vice-Chair of the IAPR Technical Committee 11 (reading systems), from 2016 to 2018. He has been an Associate Editor of *International Journal on Document Analysis and Recognition*, since 2013. He was an Associate Editor and the Associate Editor-in-Chief of *IEICE Transactions on Information and Systems*, from 2017 to 2021 and from 2021 to 2023, respectively.

SOVILA SRUN received the bachelor's degree in information science and computer engineering, the Engineering degree in computers and automated systems software, and the Ph.D. degree in basic theory of informatics from the Taganrog Institute of Technology, Southern Federal University, Russian Federation, in 2005, 2006, and 2010, respectively. Since 2010, he has been the Head of the Department of Information Technology Engineering, Royal University of Phnom Penh (RUPP), a Coordinator with the master's and Ph.D. in Information Technology Engineering, and the Director of the National Incubation Center of Cambodia, RUPP. He works closely with students and researchers in artificial intelligence, image processing, and data mining.

KOICHI KISE received the B.E., M.E., and Ph.D. degrees in communication engineering from Osaka University, Osaka, Japan, in 1986, 1988, and 1991, respectively. From 2000 to 2001, he was a Visiting Professor with the German Research Center for Artificial Intelligence (DFKI), Germany. He has been the Director of the Japan Laboratory, DFKI, since June 2022. He is currently a Professor with the Department of Core Informatics and the Director of the Institute of Document Analysis and Knowledge Science (IDAKS), Osaka Metropolitan University, Japan. His major research interests include analysis, recognition, and retrieval of documents, images, and human activities. He was a member of the IAPR conferences and meetings committee. He is a member of ACM, IEICE, IPSJ, IEEJ, ANLP, and HIS. He received awards, including the Best Paper Award from IEICE, in 2008; the IAPR/ICDAR Best Paper Awards, in 2007 and 2013; the IAPR Nakano Award, in 2010; the ICFHR Best Paper Award, in 2010; the ACPR Best Paper Award, in 2011; and the ASSET Best Paper Award, in 2023. He was the Chair of the IAPR Technical Committee 11 (reading systems). He is the Editor-in-Chief of the international journal of document analysis and recognition.