Data Augmentation and Text Recognition on Khmer Historical Manuscripts

Dona Valy[a], Michel Verleysen[b], and Sophea Chhun[a]
[a]Department of Information and Communication Engineering, Institute of Technology of Cambodia, Cambodia
[b]ICTEAM Institute, Université catholique de Louvain, Belgium
dona@itc.edu.kh, michel.verleysen@uclouvain.be, sophea.chhun@itc.edu.kh

***Abstract*— Analysis and recognition of historical documents faces many challenges, one of which is the scarcity of the ground truth data needed for most machine learning techniques, deep learning in particular. In this paper, we present a novel approach which significantly augments the word image samples generated from an existing dataset of Khmer ancient palm leaf manuscripts. Instead of segmenting real Khmer words, we combine the annotated glyphs into groups called sub-syllables. A new text recognition method is also proposed to take into account the spatially complex structure of Khmer writing. The proposed method is composed of two main modules: a feature generator and a decoder. The generator utilizes convolutional blocks, inception blocks, and also a bi-directional LSTM to encode information extracted from the input image so that it can be decoded by the attention-based decoder to predict the final text transcription. Experiments are conducted on a new dataset of groups of sub-syllables constructed from annotated glyphs of the SleukRith Set.**

Keywords—historical document analysis; palm leaf manuscript; neural network; data augmentation

I. INTRODUCTION

For centuries, palm leaves have been a popular writing medium especially in many Southeast Asian countries. Palm leaf manuscripts hold valuable recordings on subjects of numerous fields of study such as religion, astronomy, architecture, law, and even scientific findings. For the past decades, there have been ongoing efforts to recover and to preserve these treasured documents. Since the era of digital media has arrived, digitizing such cultural heritage becomes one of the necessary steps in terms of protecting the documents and their contents from destruction. To widen the accessibility of digitized palm leaf documents even more by enabling word indexing and search capability, document image analysis (DIA) system is very much needed.

In this work, we showcase palm leaf manuscripts found in Cambodia. These documents are written with a particular language called Khmer. Some sample images of Khmer palm leaf manuscripts are shown in Fig. 1. Khmer script originates from Pallava script of India and has been evolving from era to era by religious and cultural influences of scripts such as Pali and Sanskrit. Khmer script is written horizontally from left to right and then downwards when there is no more horizontal space. Khmer is a syllabic alphabet (i.e. each syllable is presented by a consonant letter followed by an inherent vowel) consisting of different types of symbols including consonants, vowels (dependent and independent), diacritics, and other special characters. Each word must start with either a consonant or an independent vowel. There is no word separator in Khmer script. Word segmentation is done according to syntactical rules and sometimes the semantical context of the sentence. Forming a Khmer word can be challenging since there exist certain irregular characters such as dependent vowels, sub-consonants, and diacritics. These types of characters can change shape when merged with other characters or be composed of multiple disconnected parts. Another reason behind the irregularity of such characters is that in writing they can be positioned at either the top, the bottom, on the left, or on the right of their adjacent characters. Spatial positions of characters are therefore very crucial in reading, understanding, and recognizing Khmer words.

Recently, deep learning has gained its popularity in solving DIA tasks especially text recognition on various languages each with its own difficulties. Recognition systems on Latin text are the most primitive ones where the use of convolutional and recurrent neural networks proves to be successful and achieves great performance on standard datasets [1, 2, 3, 4, 5]. Documents written in Chinese and Japanese show numerous challenges one of which is the large number of character classes and vocabularies [6, 7, 8, 9] while handwritten documents in Arabic pose a problem with regards to the cursive nature of the writing and the visual similarities of the characters [10, 11].

However, to the best of our knowledge, not much study has been done for Khmer text recognition dealing with the challenges that Khmer script provides. In this paper, we propose a complete handwritten text recognition scheme for documents written in ancient Khmer scripts. The contributions made in this work are summarized as follows. (1) We develop a data augmentation technique by introducing a new type of data called sub-syllables constructed on the average size SleukRith set. (2) We design a text recognition

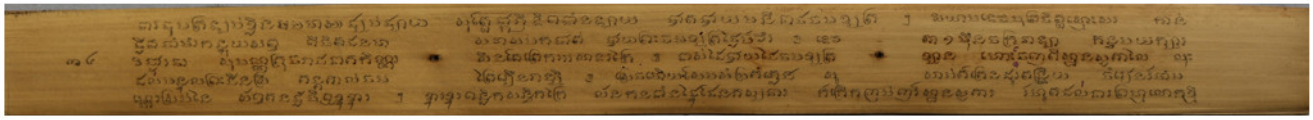

Fig. 1. A sample image of digitized Khmer palm leaf manuscripts

978-1-7281-9966-5/20/$31.00 ©2020 IEEE
DOI 10.1109/ICFHR2020.2020.00024

pipeline consisting of a feature generator utilizing convolutional combined with inception and recurrent blocks and a decoder equipped with attention mechanism. (3) We propose a learnable feature map called glyph class map (GCM) containing both the identity and spatial information of each glyph in the input image which are necessary in recognizing spatial dependent Khmer characters.

II. Datasets and Data Augmentation

A. Digital Corpus and the SleukRith Set

Datasets for ancient written text recognition algorithms are of fundamental interest for the training of machine learning methods as well as for benchmarking existing recognition systems. In [12], the SleukRith Set, the first dataset constructed on digital images of Khmer palm leaf manuscripts, was introduced. The corpus of the SleukRith Set includes recently digitized manuscripts in addition to existing digital data found in various libraries and institutions in Cambodia. The SleukRith Set consists of 657 pages which have been annotated to construct, in a bottom-up manner, three types of data: glyphs, words, and text lines. In this paper, a new type of data called "sub-syllable" is added to the SleukRith Set. It will be described in detail in Section C.

B. Glyphs and Words

In the SleukRith Set, glyphs are the smallest unit which represents a single connected component. In Khmer writing, certain characters form new connected shapes when they are written close to each other while some other characters are composed of multiple disconnected parts. Each glyph therefore can be a part of a character, a whole character, or a group of multiple characters depending on how those characters are written. To annotate glyphs, polygon boundaries are used to encapsulate the elaborate form of Khmer characters. Each glyph is given its corresponding text label. Word annotation is performed afterwards by combining segmented glyphs. The transcription of each word is specified. A rectangle image patch of a word can be extracted where the top-left vertex and the bottom-right vertex of the rectangle are $(min(x_i), min(y_i))$ and $(max(x_i), max(y_i))$ respectively (x_i and y_i are coordinates of the vertices of the polygon boundaries of all glyphs composing the word).

C. Data Augmentation

One of the main difficulties in creating useful data from palm leaf manuscripts is that the number of digitized pages is still limited. Current machine learning approaches require much more data to train, and the number of annotated words in the SleukRith Set is still limited and insufficient. Moreover, some manuscripts are written in Pali which, even though it uses Khmer alphabet, is a completely different language from Khmer. It requires the knowledge of the language to identify the word separation, and therefore word annotation and recognition cannot be achieved easily for those manuscripts.

To solve both of these problems, in this paper, we introduce a new type of data added to the SleukRith Set called "sub-syllables". A sub-syllable refers to a group of glyphs which satisfies the following criteria: (1) A sub-syllable must be a cluster of glyphs belonging to the same text line and containing at least one main consonant or be a standalone digit, a punctuation, or an independent vowel, (2) in the case where they are a cluster of glyphs, they must not begin with irregular characters such as dependent vowels, sub-consonants, or diacritics, and (3) each sub-syllable must represent the smallest possible combination of glyphs which satisfies criteria (1) and (2). A new label is also needed to be specified for the created sub-syllable. Any Khmer word (or any word written using Khmer alphabet) can therefore be formed by concatenating its composing sub-syllables. Also, the order of the sub-syllables composing a word is now a straightforward left-to-right sequence.

Adjacent sub-syllables can be joined together to form a group representing a synthetic word. Let n-SG denote a group of sub-syllables where n is the number of sub-syllables composing it. A text line containing N sub-syllables might therefore be able to produce up to $(N - n + 1)$ n-SGs. Two neighboring sub-syllables are not grouped together if the distance between them is greater than a predefined threshold. This is to take into account blank gaps which often appear between phrases in the manuscripts. Fig. 2 illustrates a comparison between word separation and sub-syllable separation. As shown, eight sub-syllables can be extracted from the text. We can group therefore those sub-syllables into n-SGs. For $n = 2$, the following set of 2-SGs can be generated: $\{(1,2), (2,3), (3,4), (5,6), (6,7), (7,8)\}$ i.e. we can obtain six synthetic words instead of only four real words. Note that the sub-syllable 4 and 5 cannot be grouped together since there is a large gap between them.

D. Text Lines

Each annotated glyph is given an ID corresponding to the line which it belongs to. The area of a text line can therefore be constructed by performing a union operation on all its glyph polygon boundaries. Due to the irregularity of how certain glyphs are positioned in the text sequence, the text transcription cannot be generated by simply using a left-to-right sequential concatenation of the labels of all glyphs in the line. Another benefit of annotated sub-syllables is that they can be used to generate line transcriptions. Similar to annotated words, a rectangle boundary of a sub-syllable is built using the polygon boundaries of the component glyphs. We then sort all sub-syllables in a line by their horizontal positions and produce the final line transcription by

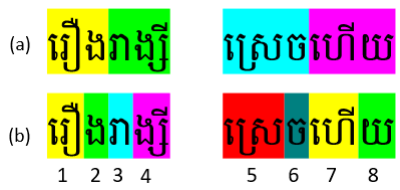

Fig. 2. (a) Word separation, (b) Sub-syllable separation

concatenating the label of each sub-syllable according to this sorted order.

III. Text Recognition

We propose a complete pipeline of text recognition system which is used to recognize a short image patch consisting of a group of glyphs and to return its corresponding text transcription. As illustrated in Fig. 3, the proposed system comprises two main modules: a generator which extracts feature maps containing the context describing the input image and a decoder which maps the variable length feature maps and interprets them into variable length output text sequences. To capture the spatial information as well as the identity of each glyph in the image patch, the generator outputs a map called glyph class map. In addition, a second feature map is also produced by the generator to acquire extra abstract information which is used to help decode the text transcription.

A. Glyph Class Map

A Glyph Class Map (or GCM in short) was first introduced as a character-class map [13] and then modified to use resizing instead of padding [14]. Let's suppose an image patch of a sub-syllable group I composing of n glyphs, and B_i $(0 \leq i < n)$ represents the region bound by the polygon boundary of the i^{th} glyph g_i. In each region B_i, we replace the value of each pixel by a new value v_i $(0 < v_i \leq N_{gc}$ where N_{gc} is the number of glyph classes) corresponding to the class of the glyph g_i. A new image I' with the same dimension as I is created by forming the union of all regions B_i. An additional value $(v_{blank} = 0)$ is used to fill in the background region of I' where no glyph pixels are assigned to. To obtain the final GCM, the image I' is downsampled by applying nearest-neighbor interpolation.

B. Feature Generator

This module takes as input a grayscale image patch of a sub-syllable group and produces two outputs: a corresponding GCM and an addition feature map. The generator starts with a convolutional block given its effective capability in extracting visual features from images. The block comprises of two basic convolutional layers to output a low-level feature map whose dimension is reduced by passing through a maxpooling layer.

Next, we utilize two inception blocks following the previous convolutional block. Each inception block is composed of two inception layers and again at the end of the block, a maxpooling layer is applied to down sample the dimension of the output map. The proposed inception layer in this work is derived from the inception module introduced by Szegedy, et al. [15]. The principle idea of the inception layer is that multiple convolutional layers with different kernel sizes in a single module can be used as a feature extractor to capture multi-scale contextual information from the input. In another word, a single inception layer contains multiple inception paths, and from these paths, feature maps of different scales are combined to approximate more complicated feature maps which can otherwise be achieved only through larger filters and more layers. The architecture of the inception layer used in our proposed system is shown in Fig. 4.

Formally, the input image I of dimension $H_I \times W_I \times 1$ becomes a feature map $\mathcal{M}$ with dimension $H_\mathcal{M} \times W_\mathcal{M} \times F_\mathcal{M}$ after it passes through the convolutional block and then the two inception blocks (H and W represents the height and the width dimensions respectively). The input image is grayscale (only one channel) hence its third dimension is 1, and $F_\mathcal{M}$ is the size of the output feature. The height and width dimension of the output feature map is reduced in size by a factor of 2^N ($H_\mathcal{M} = H_I/2^N, W_\mathcal{M} = W_I/2^N$) due to the application of maxpooling, where N is the number of the 2×2 maxpooling layers used in the generator.

In order to also encode the temporal context which is significant especially in the complex writing such as Khmer whose letters are position dependent, the feature map $\mathcal{M}$ is transformed into a one dimensional sequence of length $H_\mathcal{M} * W_\mathcal{M}$ whose each element is a vector of size $F_\mathcal{M}$. To process the sequence, we utilize a Bi-directional LSTM (BLSTM), a combination of two LSTM layers which read the sequence in opposite directions to encode both forward and backward

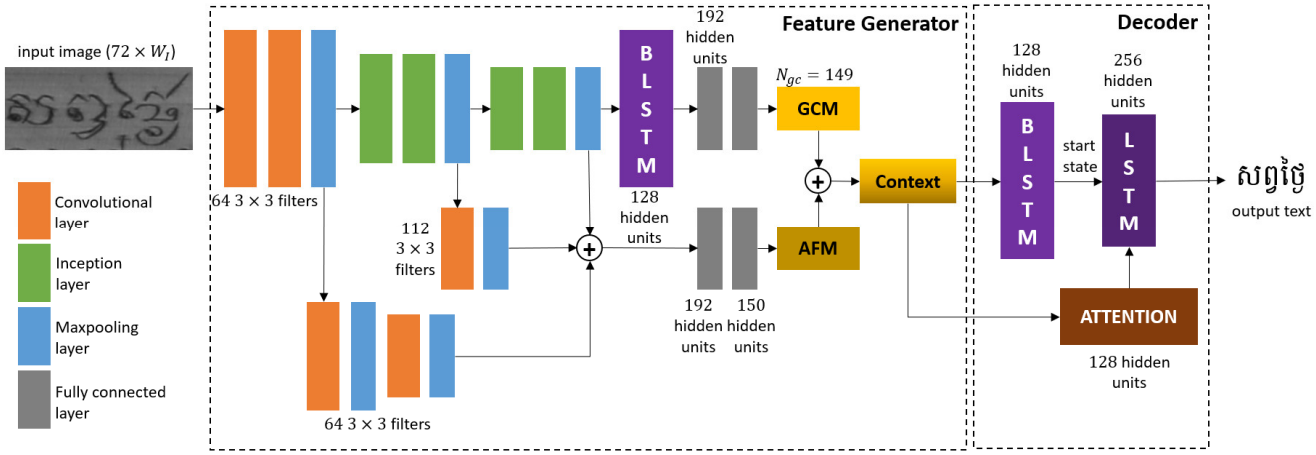

Fig. 3. Overview of the architecture of the proposed text recognition system and its implementation details

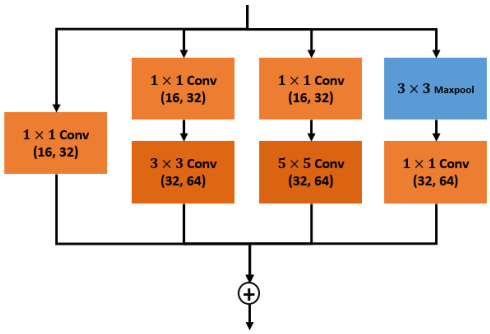

Fig. 4. Detailed architecture of the inception layer (values in the parenthesis are the numbers of filters corresponding respectively to the first and the second inception blocks)

dependencies. The BLSTM outputs two sequences whose corresponding elements are concatenated to return back to a single sequence of feature vectors. Each feature vector is passed on to a fully connected layer and then to a final output layer activated by a softmax to produce the probability distribution for each glyph class. The output sequence is finally transformed back to its previous dimension $H_{\mathcal{M}} \times W_{\mathcal{M}} \times N_{gc}$ to represent the predicted GCM where N_{gc} is a total number of glyph classes.

In addition to the GCM, we also consider an additional feature map (let's denote it AFM) which is useful in decoding the text transcription since this feature can automatically capture other information not already contained in the GCM. As illustrated in Fig. 3, instead of adapting a new path to output such feature, we reuse the features obtained from different levels of the GCM production architecture. This new feature combines information from different levels of abstract and depth of the generator network by concatenating the output feature map after the convolutional block and the feature maps after each of the inception block (including the map $\mathcal{M}$). Prior to the concatenation, to ensure that all feature maps are of the same dimension (height and width), we repetitively apply convolutional and maxpooling layer pair to each of the feature map (except the last map $\mathcal{M}$) until each feature map obtains the dimension of the last map $\mathcal{M}$. The final concatenation of these feature maps produces the AFM. The GCM and the feature map AFM are finally concatenated along the feature dimension and are used as the context to be decoded by the decoder to predict the corresponding text transcription of the input image.

C. Decoder and Attention Mechanism

Let's denote the context extracted from the input image by the generator as $\mathcal{H}$ (the concatenation of GCM and AFM) whose dimension is $H_{\mathcal{H}} \times W_{\mathcal{H}} \times F_{\mathcal{H}}$ where $H_{\mathcal{H}} = H_{\mathcal{M}}$, $W_{\mathcal{H}} = W_{\mathcal{M}}$, and $F_{\mathcal{H}} = F_{GCM} + F_{AFM}$ ($F_{GCM} = N_{gc}$ and F_{AFM} the size of the feature dimension of the feature map AFM). The context $\mathcal{H}$ can also be viewed as a matrix (size $H_{\mathcal{H}} \times W_{\mathcal{H}}$) of feature vectors $h_{i,j}$, $0 \leq i < H_{\mathcal{H}}$ and $0 \leq j < W_{\mathcal{H}}$. Our proposed attention-based decoder utilizes a one directional LSTM to predict the text transcription one character at a time. The decoder produces at each time step t the probability distribution over all characters $Y^{(t)}$ conditioned on the context $\mathcal{H}$ and all previously seen character distributions $Y^{(0)}, Y^{(1)}, \ldots, Y^{(t-1)}$. Again, $Y^{(t)}$ is a character probability distribution predicted by the decoder at time step t i.e. $Y^{(t)} = [y_c^{(t)}]$ where $y_c = p(c) \in [0,1]$ is a probability that the predicted character is likely to be c, $0 \leq c < N_{char} + 3$. Here, we represent each Khmer character by an integer c, and N_{char} is the total number of characters that are used. We also include three additional special characters to represent a start token (c_{start}), an end token (c_{end}), and an unknown character (c_{unk}).

Recall that the prediction of the output character at each time step t is also conditioned on the context matrix $\mathcal{H}$. Since the context $\mathcal{H}$ maintains the spatial information of all glyphs from the input image, to efficiently predict the correct character, the decoder should concentrate specifically on a particular group of context vectors in $\mathcal{H}$ (the group of context vectors which corresponds to the region in the input image where the character is located) rather than to pay its attention equally to all context vectors in $\mathcal{H}$. At each time step t, this weighted context or so-called context with attention (denoted $\mathcal{A}^{(t)}$) is used instead of $\mathcal{H}$. Each context vector $a_{i,j}^{(t)}$ in $\mathcal{A}^{(t)}$ is the corresponding context vector $h_{i,j}$ in $\mathcal{H}$ weighted by a coefficient $\alpha_{i,j}^{(t)}$, $a_{i,j}^{(t)} = \alpha_{i,j}^{(t)} * h_{i,j}$. To compute the coefficient $\alpha_{i,j}^{(t)}$, we use a small neural network with one hidden layer whose input depends on the previous hidden state of the decoder (the hidden state at time step $t-1$) and the corresponding context vector $h_{i,j}$ in $\mathcal{H}$. The output of the network is squashed by a sigmoid function so that each $\alpha_{i,j}^{(t)} \in [0,1]$.

At the beginning of the decoding process (at time step $t = 1$), the LSTM of the decoder requires two initial states: the start cell state and the start hidden state. For the start cell state, a vector of all zeros is used. However, for the initial hidden state, to utilize the context vectors $h_{i,j}$ at every location (i,j) as well as to capture the spatial information to predict the first character of the text transcription, another BLSTM is applied. The context matrix $\mathcal{H}$ is transformed into a one-dimensional sequence of length $H_{\mathcal{H}} * W_{\mathcal{H}}$ which is used as the input sequence to the BLSTM to produce two sequences corresponding to both forward and backward directions. The two sequences are concatenated, and finally the initial hidden state of the decoder is computed by calculating the mean of all elements in the concatenated sequence. We feed the decoder as an initial input, a one hot encoding of the start token c_{star} ($y_{c_{start}}^{(0)} = 1, y_{c \neq c_{start}}^{(0)} = 0$) concatenated with the sum of all context vectors $a^{(1)} = \sum_i \sum_j a_{i,j}^{(1)}$ in the weighted context $\mathcal{A}^{(1)}$. For the next time steps, we recursively perform the same process until the decoder produces a distribution which represents the end token c_{end}.

D. Training

We train the proposed text recognition network with data extracted from the SleukRith set. The data is divided into three subsets: the training set, the validation set, and the testing set. Each patch image in the data samples can be

considered to contain synthetic texts since they are groups of sub-syllables which resemble real Khmer words. We select n, the number of sub-syllables in each group, to be 3 and 4. Table I summarizes the statistics of the data in each subset. The number of samples of the real Khmer words which are annotated in SleukRith set is also presented in the table. We can see that by using the technique to create groups of sub-syllables instead of real words, we can augment the size of the dataset significantly (around tenfold).

The proposed network is trained from end to end in a supervised manner. The common approach is to train the generator and the decoder jointly by minimizing the text prediction loss L_{TEXT} which is a cross-entropy loss computed by $L_{TEXT} = -\sum_t \log p(c^{(t)} = \hat{c}^{(t)})$ where $\hat{c}^{(t)}$ is the ground truth output character at time step t. We also want to introduce the GCM loss corresponding to the predicted GCM by the generator. By enforcing the generator to generate the GCM which contains both the identity and the spatial information of all glyphs in the input image, it is expected that this encoded context improves the recognition performance of the decoder. The GCM loss L_{GCM} is also a cross-entropy loss summing over all locations (i, j) in the GCM map computed by $L_{GCM} = -\sum_i \sum_j \log p(g_{i,j} = \hat{g}_{i,j})$ where $g_{i,j}$ is the predicted glyph and $\hat{g}_{i,j}$ is the ground truth glyph at position (i, j). The objective function of the network is therefore $L = L_{TEXT} + \lambda * L_{GCM}$ where $\lambda \geq 0$ (we choose $\lambda = 1$ in our experiments) controls how predicting correct GCM affects the overall performance of the network.

TABLE I. SUMMARY OF NUMBER OF SAMPLES IN EACH SUBSET

Type		Training Set	Validation set	Testing Set
Word Dataset		15,432	813	7,764
Sub-syllable Dataset	3-SG	51,906	4,828	38,090
	4-SG	52,086	4,704	37,592
	Total SG	103,992	9,532	75,682

IV. EXPERIMENTS

A. Evaluation Scenarios

The evaluation is based on two criteria i.e. the generated GCM predicted by the generator and the text transcription output by the decoder. A score $s \in [0,1]$ is calculated for each predicted GCM by $s = \sum_i \sum_j f(g_{i,j}, \hat{g}_{i,j}) / (H_{GCM} * W_{GCM})$ where $f(g_{i,j}, \hat{g}_{i,j}) = 1$ if the predicted glyph $g_{i,j}$ matches the ground truth $\hat{g}_{i,j}$, otherwise it returns 0. The overall score is then computed as the mean of the scores of all predicted GCM over the testing set. To evaluate the text prediction of the network, the character error rate (CER) and the word error rate (WER) are computed.

The experimental evaluation is performed in two different scenarios.

Scenario 1: we evaluate on the dataset of groups of sub-syllables, the overall flow of the proposed network.

Scenario 2: we evaluate the network on image patches generated from the ground truth of line segmentation. The boundaries from the annotated glyph dataset are also used to split the ground truth text lines into smaller patches containing at most four sub-syllables. The CER is now computed as the Levenshtein distance between the predicted text line transcription (concatenation of all text labels in the image patch sequence) and the ground truth text line transcription.

B. Results

The evaluation results of both scenarios are shown in Table II. On the newly constructed dataset of groups of sub-syllables, the proposed feature generator obtains a sufficiently high score in producing the output GCM. We observe that although the predicted GCM contains some noise and does not match perfectly cell by cell with the ground truth, it is still capable of storing the useful information related to the spatial position, size, and identity of each glyph in the input image. To highlight this and also the effectiveness of the GCM on the attention mechanism of the proposed decoder, Fig. 5 illustrates some sample outputs and the attention map at each time step t ($\alpha_{i,j}^{(t)}$) corresponding to each predicted character. According to these results, it is shown that the generated GCM from the generator serves as a blueprint which contains candidate regions for the decoder to attend to. As expected, by enforcing the generator to learn to produce the ground truth GCM, the spatial information of each glyph annotated in the GCM is helpful in enhancing the attention mechanism as well as improving the predicting capability of the decoder. We measure the performance of the overall network by computing the CER and the WER of the predicted text labels. According to these results on the new dataset, the proposed network is able to achieve a low CER and, as expected, a slightly higher WER since the text labels in the dataset are often long.

We also perform a comparison with the results obtained from the network proposed in [14] (denoted Word-Net) which is trained on the word dataset presented in Table I. According to the results shown in Table III, although the Word-Net performs rather well on annotated word dataset, it does not generalize to sub-syllable dataset which is much bigger. This shows the overfitting problem of the Word-Net. In contrast, the SubSyl-Net which is trained on the expanded dataset using the proposed data augmentation technique generalizes well to all datasets. Also, since the new sub-syllable dataset is significantly larger than the word dataset (see Table I), we therefore keep in mind possible optimization options for the proposed architecture. Some

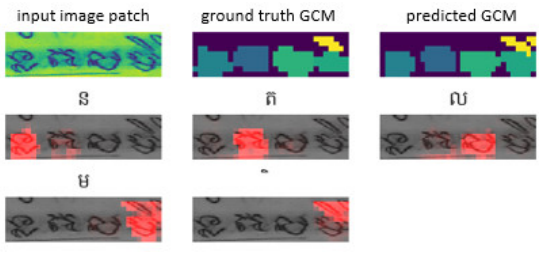

Fig. 5. A sample result showing the predicted GCM and the attention map at each time step (the region highlighted in red is where the decoder attends to)

optimization choices include the use of inception blocks instead of large convolutional layers and the substitution of multi-directional multi-dimensional LSTMs with BLSTM in the recurrent blocks (owing to their comparable performance as discussed in [16]). Compared to the Word-Net, the number of parameters in the proposed network is significantly reduced as illustrated in Table III.

TABLE II. EXPERIMENTAL RESULTS CONDUCTED ACCORDING TO THE THREE SCENARIOS

Scenario	GCM Score (%)	CER (%)	WER (%)
1	87.12	6.16	26.23
2	-	7.81	-

TABLE III. COMPARISON BETWEEN THE WORD-NET AND THE PROPOSED NETWORK

Network	Number of parameters (millions)	Dataset	CER (%)	WER (%)
Word-Net [14]	9.18	Words	3.80	11.81
		Groups of sub-syllables	49.62	96.42
Proposed	2.37	Words	6.88	19.33
		Groups of sub-syllables	6.16	26.23

V. CONCLUSION

In this paper, we propose a text recognition approach designed to take into account characteristics of Khmer writing on ancient palm leaf documents. The network in the proposed method uses a generator to extracts a glyph class map and also an additional feature map so that these encoded features can be decoded by an attention-based decoder to output a corresponding text label of the input image patch. To train the network, instead of using the relatively small annotated word dataset from the SleukRith set, we construct a new significantly larger dataset composing of groups of sub-syllables. The experiments conducted on this new dataset show that the proposed network performs promisingly well. To apply this work however to the whole document page, a method to efficiently extract text lines as well as to segment those lines into smaller patches is still needed. This work can also be extended and improved by adopting a language model as a post-processing step which we will leave for future work.

ACKNOWLEDGMENT

This research study is supported by the Higher Education Improvement Project (HEIP) and the ARES-CCD under the funding of Belgian university cooperation.

REFERENCES

[1] F. P. Such, D. Peri, F. Brockler, H. Paul and R. Ptucha, "Fully Convolutional Networks for Handwriting Recognition," in *16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2018.

[2] J. A. Sanchez, V. Romero, A. H. Toselli and E. Vidal, "ICFHR2016 competition on handwritten text recognition on the READ dataset," in *15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2016.

[3] H. Ding, K. Chen, Y. Yuan, M. Cai, L. Sun, S. Liang and Q. Huo, "A compact CNN-DBLSTM based character model for offline handwriting recognition with Tucker decomposition," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017.

[4] T. Bluche, J. Louradour and R. Messina, "Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017.

[5] D. Castro, B. L. Bezerra and M. Valença, "Boosting the deep multidimensional long-short-term memory network for handwritten recognition systems," in *16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2018.

[6] W. Wang, J. Zhang, J. Du and Y. Zhu, "DenseRAN for Offline Handwritten Chinese Character Recognition," in *16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2018.

[7] N. T. Ly, C. T. Nguyen and M. Nakagawa, "Training an End-to-End Model for Offline Handwritten Japanese Text Recognition by Generated Synthetic Patterns," in *16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2018.

[8] C. K. Nguyen, C. T. Nguyen and N. Masaki, "Tens of Thousands of Nom Character Recognition by Deep Convolution Neural Networks," in *4th International Workshop on Historical Document Imaging and Processing*, 2017.

[9] Y.-C. Wu, F. Yin, Z. Chen and C.-L. Liu, "Handwritten Chinese text recognition using separable multi-dimensional recurrent neural network," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017.

[10] A. Lawgali, "A survey on arabic character recognition," *International Journal of Signal Processing, Image Processing and Pattern Recognition,* vol. 8, no. 2, pp. 401-426, 2015.

[11] C. Clausner, A. Antonacopoulos, N. Mcgregor and D. Wilson-Nunn, "ICFHR 2018 Competition on Recognition of Historical Arabic Scientific Manuscripts–RASM2018," in *16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2018.

[12] D. Valy, M. Verleysen, S. Chhun and J.-C. Burie, "A New Khmer Palm Leaf Manuscript Dataset for Document Analysis and Recognition: SleukRith Set," in *The 4th International Workshop on Historical Document Imaging and Processing*, 2017.

[13] D. Valy, M. Verleysen, S. Chhun and J.-C. Burie, "Character and Text Recognition of Khmer Historical Palm Leaf Manuscripts," in *Th 16th International Conference on Frontiers in Handwritting Recognition*, 2018.

[14] D. Valy, M. Verleysen and S. Chhun, "Text Recognition on Khmer Historical Documents using Glyph Class Map Generation with Encoder-Decoder Model," in *Proceedings of ICPRAM 2019*, 2019.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going Deeper with Convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[16] J. Puigcerver, "Are multidimensional recurrent layers really necessary for handwritten text recognition?," in *14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017.