Character and Text Recognition of Khmer Historical Palm Leaf Manuscripts

Dona Valy[a,b], Michel Verleysen[a], Sophea Chhun[b], and Jean-Christophe Burie[c]
[a]*ICTEAM Institute, Université catholique de Louvain, Belgium*
[b]*Department of Information and Communication Engineering, Institute of Technology of Cambodia, Cambodia*
[c]*Laboratoire Informatique Image Interaction (L3i), University of La Rochelle, France*
{dona.valy,michel.verleysen}@uclouvain.be, sophea.chhun@itc.edu.kh, jcburie@univ-lr.fr

***Abstract*—This paper presents methods for two historical document analysis tasks on digitized Khmer palm leaf manuscripts. The first task consisting of isolated character recognition is conducted utilizing different types of neural network architectures such as CNN, LSTM-RNN, and a combination of both. The second task focuses on recognizing word/text image patches of variable length and simultaneously localizing each glyph in the text image. For this task, according to the characteristic of Khmer writing system, both one-dimensional and two-dimensional RNN are used.**

Keywords—historical document analysis, palm leaf manuscript, neural network

I. Introduction

Historical document analysis has been one of the most challenging problems and has received a lot of attention recently from researchers around the world. Recognition of segmented individual characters, words, and even the whole text line is crucial and is considered to be one of the most important steps in the document analysis pipeline. We have seen many proposed approaches focusing on character and text recognition in the literature on popular languages and writings such as Latin [1, 2], Chinese [3, 4], and Arabic [5, 6]; however, researches on other languages, from Southeast Asian countries specifically Cambodia for example, are still trivial in comparison.

The recent availability of datasets constructed from digitized Khmer palm leaf documents opens ways to experimental studies on the analysis tasks of this language. Machine learning techniques, deep neural networks in particular, have become more and more well-known to be the solution for this type of problems. This is owing to their ability to automatically learn desirable representations of text images instead of having to design manually handcrafted features. Some eminent deep learning architectures notably the Convolutional Neural Networks (CNN) including AlexNet [7], VGGNet [8], and ResNet [9] are considered to be state-of-the-art approaches for image classification problems. However, due to its depth, these types of deep architectures normally require significantly large amount of data (for example tens of millions of samples) to train and tend to overfit on smaller datasets.

For sequence modeling such as handwritten text recognition, Recurrent Neural Networks (RNN) seem to be one of the best options. Some of the most popular RNN architectures include Gated Recurrent Unit (GRU) [10] and Long Short-Term Memory (LSTM) [11]. Two-dimensional RNN [12] which is an extension designed to perform on images is also gaining popularity. For this task, a classifier called CTC (Connectionist Temporal Classification) [13] is often used in combination with the RNN due to its ability to maximize the probability of the target text without knowing the alignment information (between characters in the text image and the positions of those characters in the ground truth transcription) which most datasets lack of. However, if the alignment is present, it may be helpful to integrate this information in the training process of the network model so that its recognition performance can be improved.

In this paper, we present different approaches for two document analysis tasks on medium size datasets of digitized Khmer palm leaf manuscripts: isolated character classification and word text recognition. For the first task, different methods are presented keeping in mind the size of the training set. The second task focuses on the recognition of the whole text in the word image. For this task, we propose approaches which incorporate the spatial alignment information of each glyph in the image and also the characteristics of Khmer writing.

The remainder of this paper is structured as follows. Section II describes Khmer palm leaf manuscripts by detailing their characteristics and challenges they may cause and also mentions the new dataset constructed from such documents. In Section III, we present methods for the first task, isolated character classification. Section IV describes the proposed systems for the second task which is to recognize text images. Experimental

Fig. 1. Sample images of digitized palm leaf document pages

results are shown and discussed in Section V. Finally, conclusions are drawn in Section VI.

II. Khmer Palm Leaf Manuscripts

Khmer palm leaf manuscripts (Fig. 1) are made from dried palm leaves which are cut into a long rectangular shape and are bound and tied together to create a binding like a book. On each page, letters are carved using a sharp edge stylus. Ink is then applied on the freshly engraved text to make it more noticeable resulting in dark text on light brown or yellowish color of the dried palm leaves as background. Palm leaf documents store valuable content on various domains and fields of study which has been passed on from generations to generations.

Due to natural aging, to preserve old palm leaf documents, digitalization through digital imaging is needed. Some efforts have already been made in order to collect and digitalize remaining documents in Cambodia [14].

A. Challenges from Khmer Palm Leaf Documents

One of the main challenges of working with ancient palm leaf manuscripts is the presence of different types of degradations and noises. This poses a big problem especially for the preprocessing tasks such as binarization which cannot produce acceptable results as mentioned in [15]. The later steps of the pipeline have to be therefore binarization free.

Another difficulty is caused by the characteristics of the language itself. Khmer is one of the most complex languages in the world. Its alphabet consists of many symbols some of which are very similar to each other. Certain symbols can only be distinguished by a mere tiny stroke or hole. This, most of the time, creates character ambiguity which is even more apparent in old handwritten form. Fig. 2 shows the ambiguity and similarity between three consonants.

The sequential order of characters composing a word is also irregular. For instance, a word must start with a consonant and may be followed by vowels. However, the physical positions of the vowels can either be on the left, on the right, on top, or under the starting consonant. Some particular vowel symbols even consist of multiple parts which are placed simultaneously at multiple locations around the consonant. Fig. 3 illustrates this case. The vowel symbol AEU is composed of two parts which are located on the left and on top of the main consonant SA. To emphasize even more, we also simulate how an English word could have been written in Khmer style in Fig. 4.

B. SleukRith Set

SleukRith set [14] is a collection of annotated data created from 657 pages of digitized Khmer palm leaf manuscripts. It is composed of three types of annotated data: characters or glyphs, words, and lines. To annotate each character, a polygon boundary of that character has to be drawn manually by defining a list of its vertices. A label is also given to each annotated character. After the annotation of all characters is complete, words are annotated by grouping together annotated characters composing those words keeping intact the order of how the words are spelt. Labels are afterward assigned to each word. For each document page, the annotated information is stored in an xml file. Individual character and word image patches with their respective labels and transcriptions can be generated.

Fig. 2. Similarity between three Khmer symbols (from left to right KO, TA, and PHO)

Fig. 3. Irregular sequential ordering of symbols in a word. The different order of the position of the characters in the text image (top) and the Unicode sequence of the text (bottom)

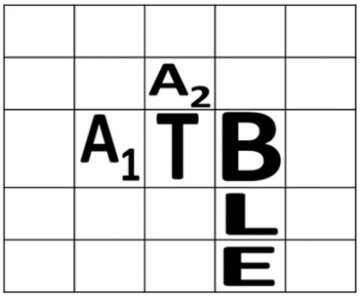

Fig. 4. Simulation of how the word "TABLE" could have been written in Khmer style (vowel 'A' composed of two parts which are placed on top and on the left side of 'T', consonant 'L' written as a sub-from under 'B', and vowel 'E' placed under 'B' and the sub-form of 'L')

III. Isolated Character Classification

A. Isolated Character Dataset

The dataset for this task consists of 203,875 sample images of isolated character patches extracted from SleukRith set. The dataset is divided into 113,206 sample images for the training set and 90,669 sample images for the test set. We do not take into account small symbols such as punctuations and diacritics and also remove symbols with too few occurrences. The resulting set consists of a total number of 111 classes. Each image patch is gray scaled, resized to be 48 by 48 pixels, and then normalized using histogram stretching technique (Fig. 5).

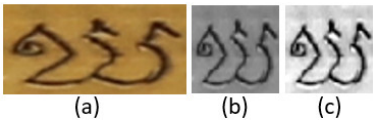

Fig. 5. (a). Original image, (b). Gray scaled and resized, (c). Normalized

B. *Neural Network Architectures*

We present four network architectures for this task. The choice of the hyper parameters of each model presented in this paper is empirical. We focus on the feasibility of different types of neural network architectures on the recognition problem of Khmer palm leaf manuscripts rather than a computationally intensive fine tuning of their structure.

The first network (Fig. 6) is CNN based and is composed of two pairs of convolutional (12 and 24 of 5 by 5 filters) and max pooling layers (2 by 2 window size with 2 by 2 strides for both layers). The output from each convolutional part is activated by Relu non-linearity. A drop out with dropped probability of 0.2 is applied after each max pooling. The output from the last max pooling is flattened and followed by a fully connected layer (with 1024 hidden units) also activated by Relu. A drop out with dropped probability of 0.6 is also applied afterward. Finally, the final output with a softmax activation is produced.

The second network (Fig. 7) is recurrent. First, the input image is transformed into a sequence of one-pixel columns. Each column is then fed into two layers of LSTM cells with 512 hidden units each. The last time step output from the last LSTM layer is connected to the final output layer which is then activated with a softmax function.

The third network (Fig. 8) is also RNN based. However, the input image is transformed simultaneously into a sequence of one-pixel columns and a sequence of one-pixel rows. Each sequence is fed separately into two distinct pairs of LSTM layers which are similar to the ones in the second network. The last time step output from the last layer of each pair is concatenated before being fed into the softmax activated final output layer.

The fourth network is a combination of the convolutional and max pooling part in the first network and the recurrent part in the third network (Fig. 10). The input is first fed into the two convolutional and max pooling pairs. The output from this first two layers is split row wise and column wise along the height and width dimension respectively (the depth or channel dimension is flattened). Similar to the third network, we feed the row wise and column wise sequences to two different two-layer LSTMs (again with 512 hidden units in each layer). The outputs from the last time step of the last layer of the two LSTMs are concatenated and finally are followed by the last output layer activated by a softmax function. Like previous architectures, a drop out (with dropped probability of 0.2) is applied after each max pooling and also after each LSTM layer.

IV. WORD/TEXT RECOGNITION

A. *Annotated Word Dataset*

The dataset for this task is composed of word image patches also generated from SleukRith set. The rectangle bounding box of each word is the bounding box of the union of all polygon boundaries of its characters. There are 16,245 sample images in the training set and 7,764 sample images in the test set. All word image patches are in gray scale and are normalized by resizing so that they are of the same height (of 72 pixels) but still with variable width.

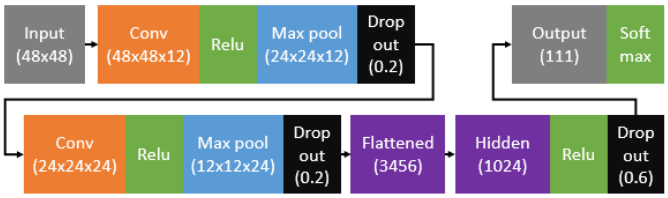

Fig. 6. Network 1.1: CNN based

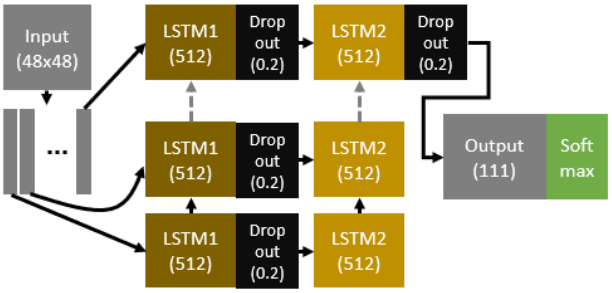

Fig. 7. Network 1.2: column wise LSTM

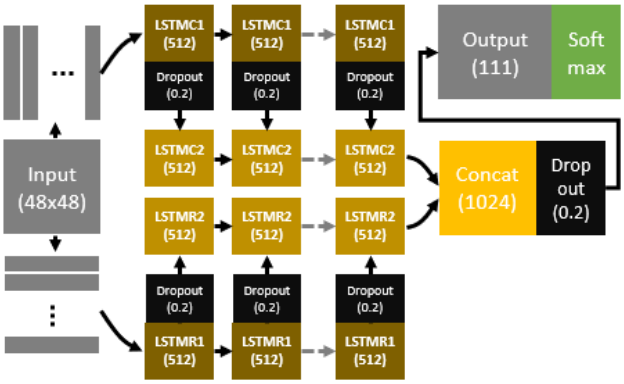

Fig. 8. Network 1.3: column wise and row wise LSTM

To take into account the character annotation information, for each word, a character-class map is built. The word image patch is divided into grid like cells of c_h by c_w pixels where c_h and c_w are the height and the width of each cell respectively. Padding may be applied to ensure that all cells are of equal size (after padding the width of the word image patch must be divisible by c_w, and the height must be divisible by c_h). Each cell of the character-class map is then assigned to one and only one character-class which contains the most pixels in that cell. From all the annotated words in the dataset, 134 character-classes are found including one additional token class for cells containing only blank space or background pixels. An example of how a character-class map is built is shown in Fig. 9.

B. *Methods*

The objective function of the approaches for this task is to predict the character-class map of the input word image patch which means to classify each cell of the input patch to its corresponding character-class. Two network architectures are presented for this task. Both models consist of the same two beginning convolutional layers (the filter size is 5 by 5 for both layers and the numbers of filters are 12 and 24 respectively). A max pooling layer of stride 2 follows, so the output gives rise to

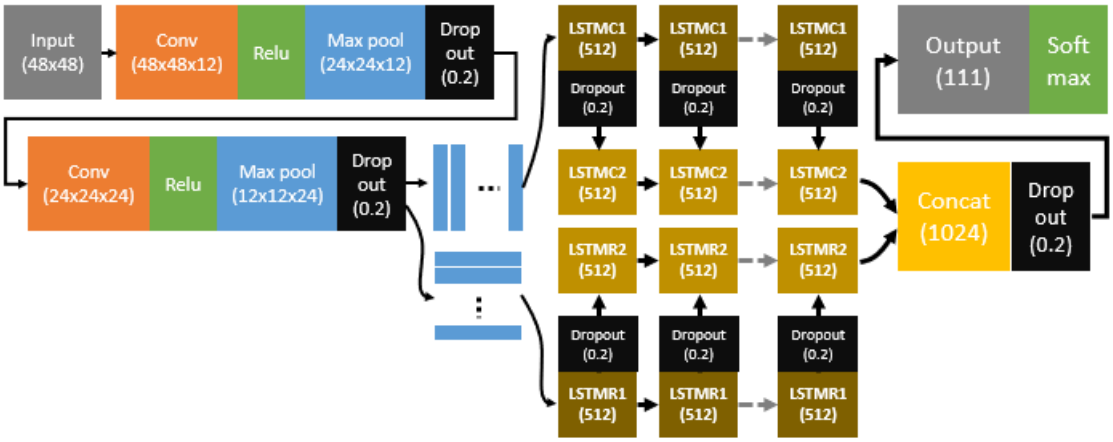

Fig. 10. Network 1.4: a combination of convolutional and recurrent neural network

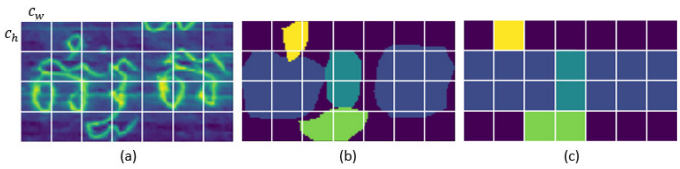

Fig. 9. (a). Original word image patch, (b). Annotated character information in the word: polygon boundaries of all characters, (c). Character-class map

a $½I_h \times ½I_w \times D$ feature map where I_h and I_w are the height and the width of the input image (after possible paddings) and $D = 24$ the depth of the feature map which equals to the number of filters of the last convolutional layer. To match the number of cells in the character-class map ($n_{row} \times n_{col}$ cells where $n_{row} = I_h/c_h$ and $n_{col} = I_w/c_w$), the feature map is also divided into grid like cells where each cell is of equal size of $½c_h$ by $½c_w$ (c_h and c_w must be divisible by 2 and in our experiments, we choose c_h and c_w to be $c_h = c_w = 8$). Therefore, the final dimensions of the feature map are now $n_{row} \times n_{col} \times n_f$ where $n_f = ½c_h.½c_w.D$. Fig. 11 shows the general architecture of both networks.

The first network utilizes a one-dimensional RNN (1D-RNN) architecture (Fig. 12). Inspired by the architecture in [16], all four directions are taken into account: left to right, right to left, top to bottom, and bottom to top. To keep the architecture as 1D, the feature map cells are flattened column wise or row wise to generate four one-dimensional sequences of length $n_{row}.n_{col}$ where each sequence corresponds to each direction mentioned above. All sequences are then fed into one shared LSTM layer with 512 hidden units to produce four output sequences also of length $n_{row}.n_{col}$ each of which is then reshaped back to be a grid of $n_{row} \times n_{col}$. The four grids are concatenated together along the feature axis to form a single grid of features ($n_{row} \times n_{col} \times 2048$). The next layers of the network are a fully connected layer with 1024 hidden units followed by a softmax activated final output layer to produce the predicted character-class map ($n_{row} \times n_{col} \times n_{class}$ where $n_{class} = 134$).

Due to the characteristics of Khmer writing especially how characters are positioned to form a word, the second network takes into consideration both spatial dimensions of the image. For this reason, the proposed network uses a two-dimensional RNN (Fig. 14). Unlike the first network, diagonal directions are considered instead. Four grids of cells are produced from the input feature map to represent each of the four diagonal directions: top-left to bottom-right, bottom-right to top-left, top-right to bottom-left, and bottom-left to top-right. They share a 2D LSTM layer (whose previous states are given by the previous cells along both horizontal and vertical axes) also with 512 hidden units to output four grids of the same size ($n_{row} \times n_{col} \times 512$) which are then concatenated along the depth or feature axis to get a final grid before feeding it to a fully connected layer and then to the final output layer similar to the architecture of the first network.

We regularize both networks by applying drop outs with dropped probability of 0.2 after the max pooling and of 0.2 and 0.5 before and after the fully connected layer respectively. Since the text image can be big in width which would produce a very long sequence for the LSTM layer, we also use gradient clipping to prevent exploding gradients.

V. Experiments and Results

All networks of the two tasks are implemented using Tensorflow [1]. The implementation of the two-dimensional LSTM in the second task is inspired by tensorflow-multi-dimensional-lstm repository of Philippe Remy [2]. During training, the loss function of each network is minimized using Adam optimizer [17] with initial learning rate of 0.001. Weight parameters in all networks are initialized using a truncated normal distribution with a standard deviation of 0.1 while biases are initialized with constant values of 0.1. The network is trained per mini batch basis (in our experiment we choose a batch size of 300 and 30 for task 1 and task 2 respectively). The training set starts with all samples being shuffled, and after a competition of each epoch, the order of the samples in the training set is then again reshuffled. For every n_{iter} of iterations, we calculate the average loss of all batches and stop the training if the average loss does not improve for N

[1] https://www.tensorflow.org

[2] https://github.com/philipperemy/tensorflow-multi-dimensional-lstm

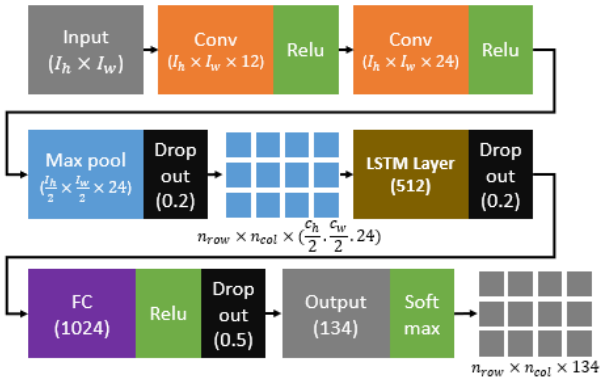

Fig. 11. General architecture of networks of task 2

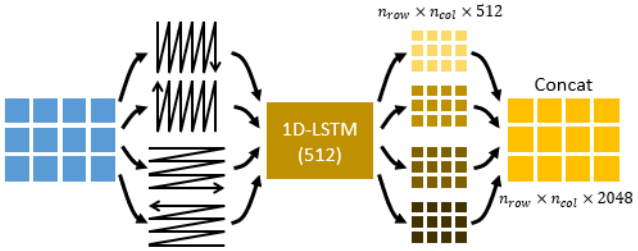

Fig. 12. LSTM Layer of Network 2.1

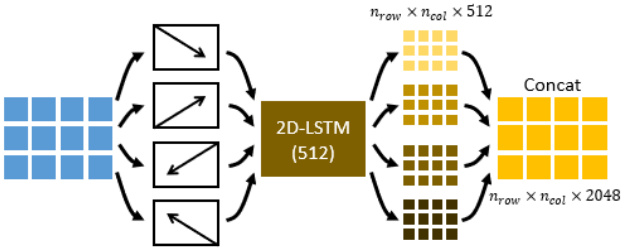

Fig. 14. LSTM Layer of Network 2.2

consecutive tests. In our experiments we choose n_{iter} to be 50 for both tasks and N to be 10 for task 1 and 30 for task 2.

A. *Task 1: Isolated Character Recognition*

We evaluate all four networks according to the top k error rates, i.e., the percentages of incorrectly classified samples over the total number of samples in the test set. The classification is considered to be incorrect if the target label of the character to be predicted is not one of the top k predictions from the network. In this evaluation, we choose $k = 5$ and $k = 1$.

The experimental results of this evaluation are presented in Table I. According to these results, even being purely a recurrent network, which is more suitable in sequence modeling rather than a single object classification, Network 1.2 and 1.3 preform sufficiently well in recognizing isolated characters on Khmer palm leaf documents. Network 1.3 produces a slightly better result than Network 1.2 since it uses sequential information along both horizontal and vertical axes. It is also shown that, Network 1.4 is able to reach a lower error rate (compared to Network 1.1) since on top of using convolutional modules in the shallow layers to extract principle features from the character images, the network also uses the column wise and row wise sequential information in the deeper layers. This illustrates that combining convolutional with recurrent module is a powerful technique to classify Khmer handwritten characters.

TABLE I. EVALUATION RESULTS OF TASK 1

Architecture	Error Rate (%)	
	Top 5	*Top 1*
Network 1.1: CNN	0.65	6.29
Network 1.2: Column LSTM	1.05	8.49
Network 1.3: Row-Column LSTM	0.82	7.00
Network 1.4: Conv-LSTM	**0.46**	**5.01**

B. *Task 2: Word Recognition*

For efficient training, input word image patches are sorted by their width and then are batched together so that all image samples in the same batch have similar width. This saves memory space and computing time by eliminating unnecessary large paddings (we may still need to apply small paddings so that all input images have their widths equal to the maximum width in the batch). Unlike in task 1, after each epoch, the reshuffling is done on the order of batches instead of the order of every single samples. Fig. 13 illustrates this batching mechanism.

Similar to the previous task, we use top k error rate measurement to evaluate the performance of the two proposed networks. Each cell of the target character-class map is predicted by the networks. The error rate of one sample word image is the number of incorrectly identified cells over the total number of cells in that image. We obtain the final error rate of all samples in the test set by averaging the error rate of each sample.

From the results shown in Table II, we can see that the second network which uses two-dimensional LSTM outperforms the first network which uses only one-dimensional LSTM. This is to be expected since the LSTM module of the latter network acquires more information from the previous states in both vertical and horizontal axes. The error rates from both networks drop significantly between the top 1 and top 5 measurements. This illustrates the problem caused by the similarity and the ambiguity of Khmer characters. An example of predicted character-class maps is also given in Fig. 15. By observing this example, we can notice that the character class-map is not very sensitive to error. Connected components in the map (even with some noises) can still be used to obtain the identity of each character and its location in the text image.

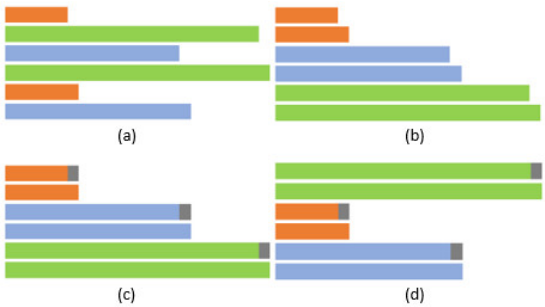

Fig. 13. (a). Initial sample order, (b). Sort by the width of each sample, (c). Pad each sample to the maximum width in the batch, (d). Shuffle batch order

TABLE II. EVALUATION RESULTS OF TASK 2

Architecture	Error Rate (%)	
	Top 5	*Top 1*
Network 2.1: 1D-LSTM	8.46	32.01
Network 2.2: 2D-LSTM	**2.40**	**20.49**

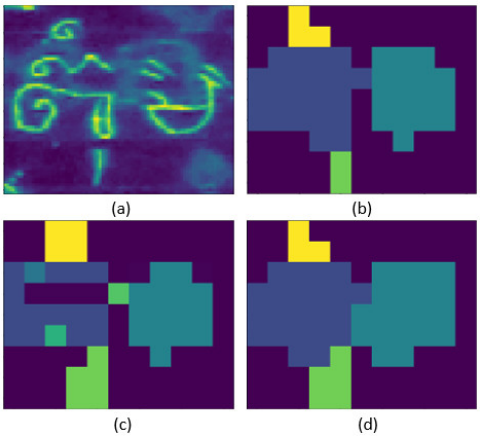

Fig. 15. (a). Original word image, (b). Ground truth character-class map, (c). Result predicted by Network 2.1, (d). Result predicted by Network 2.2

VI. CONCLUSION

In this paper, methods for two document analysis tasks (recognition of character and word image patches extracted from Khmer ancient palm leaf documents) are presented. For the task of classifying isolated characters, we present four network architectures: purely CNN based, RNN on sequence of one-pixel columns, RNN on both column and row sequences, and finally a combination which is both convolutional and recurrent. The results show that both CNN and RNN based architectures perform well enough on this task individually; however, combining both types of architectures proves to be better and more powerful. For the text word recognition task, SleukRith Set provides character annotation which can be used as alignment between character codes in the text transcription and the character positions in the text images. To incorporate this information, we propose approaches whose objective is to output a character-class map for each input word image using both one-dimensional and two-dimensional recurrent neural networks. For this task, it is illustrated in the evaluation results that the latter network performs better. The predicted character-class map can be used as input to produce the final text transcription of the word. A CTC may be used to decode the map in this case. This idea will be tackled in our future work.

ACKNOWLEDGMENT

This research study is supported by the ARES-CCD (program AI 2014-2019) under the funding of Belgian university cooperation and the AMADI project funded by the STIC Asia program implemented by the French Ministry of Foreign Affairs and International Development.

REFERENCES

[1] W. Swaileh, J. Lerouge and T. Paquet, "A Unified French/English syllabic model for handwriting recognition," in *15th International Conference on Frontiers in Handwriting Recognition*, 2016.

[2] T. M. Breuel, "High Performance Text Recognition using a Hybrid Convolutional-LSTM Implementation," in *14th IAPR International Conference on Document Analysis and Recognition*, 2017.

[3] T. Bluche and R. Messina, "Faster Segmentation-Free Handwritten Chinese Text Recognition with Character," in *15th International Conference on Frontiers in Handwriting Recognition*, 2016.

[4] X. Yang, D. He, Z. Zhou, D. Kifer and C. L. Giles, "Improving Offline Handwritten Chinese Character," in *14th IAPR International Conference on Document Analysis and Recognition*, 2017.

[5] M. T. Pavez and S. A. Mahoud, "Offline Arabic handwritten text recognition: a survey," *ACM Computing Surveys (CSUR),* vol. 45, no. 2, p. 23, 2013.

[6] A. Khémiri, A. K. Echi, A. Belaïd and M. Elloumi, "A System for off-line Arabic Handwritten Word Recognition based on Bayesian," in *15th International Conference on Frontiers in Handwriting Recognition*, 2016.

[7] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks.," in *Advances in neural information processing systems*, 2012.

[8] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *arXiv preprint arXiv:1409.1556*, 2014.

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, 2016.

[10] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *arXiv preprint arXiv:1406.1078*, 2014.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[12] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," *Advances in neural information processing systems,* pp. 545-552, 2009.

[13] A. Graves, S. Fernández, F. Gomez and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *23rd international conference on Machine learning*, 2006.

[14] 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 *4th International Workshop on Historical Document Imaging and Processing (HIP)*, 2017.

[15] M. W. A. Kesiman, D. Valy, J.-C. Burie, E. Paulaus, M. Suryani, S. Hadi, M. Verleysen, S. Chhun and J.-M. Ogier, "Benchmarking of Document Image Analysis Tasks for Palm Leaf Manuscripts from Southeast Asia," *Journal of Imaging,* vol. 4, no. 2, p. 43, 2018.

[16] 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*, 2017.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *arXiv preprint arXiv:1412.6980*, 2014.