A model for posttransliteration suggestion for balinese palm leaf manuscript with text generation and lstm model

M W A Kesiman\textsuperscript{1} and I M D Maysanjaya\textsuperscript{1}

\textsuperscript{1}Virtual Vision, Image, and Pattern Research Group (VVIP-RG)
Faculty of Engineering and Vocational, Universitas Pendidikan Ganesha

E-mail: antara.kesiman@undiksha.ac.id

Abstract. The main challenge found in building an automatic transliteration system for Balinese palm leaf manuscript (Lontar) collections is that the recognition error in a small portion of glyphs of Balinese script can affect the results of transliteration widely. This is due to the fundamental nature of Balinese script which is a complex alphasyllabic script. This paper presents an initial proposition for a general scheme and model for suggesting several possible transliteration with text generation and LSTM for Lontar collection. The Edit-Insert-Replace model was proposed to be applied on the existing word collection dataset and a Bidirectional LSTM model with a specific feature extraction method was built for the training process of post transliteration suggestion module. This module will help in suggesting several possible transliterations based on the initial transliteration from the previous system.

1. Introduction
The Balinese palm leaf manuscripts, known as Lontar collection, is one of the invaluable nation's cultural heritages [1–3]. Unfortunately, Lontar collection did not last well against the time. Various threats of physical and chemical damage from palm leaves and inadequate storage for collections are often the main causes of degradation for Lontar collections. Besides, the influence of socio-cultural aspects of the Balinese society makes Lontar collection not very popular as a source of knowledge for the younger generation. Not many Balinese people are able to write and read the Balinese script used in the Lontar collection. This has made Lontar begin to be forgotten by most of the generations, or only stored without ever knowing or reading the values of the knowledge written in it [1].

Efforts to save and preserve the Lontar collection have begun [4]. The Balinese government's agency for the Balinese language has intensively carried out to remind the public of the importance of Lontar while providing maintenance services for ancient Lontar collections. With the rapid development of the digital technology era since three or two decades ago, efforts to save, conserve and disseminate the content of valuable knowledge in Lontar has begun with several proposals for Lontar digitization projects. But, most of the Lontar digitization efforts were only limited to storing digital images from the Lontar collection. One international project that seeks to not only digitize the Lontar collection, but also to build a document analysis system for Lontar is the AMADI (Ancient Manuscript Digitization and Indexation) Project [4]. Since the AMADI Project was started in 2014, the research roadmap has a general goal of building a system that can help the process of disseminating the content of Lontar collection through digital technology to a wider audience.

One of the main features that have been built within the framework of the AMADI Project is the OCR engine and automatic transliteration module to convert the Balinese script in Lontar into Latin...
The main problem and challenge with this automatic system, found in the results of OCR engine and transliteration module, is that the recognition error in a small portion of glyphs of Balinese script can affect the results of transliteration widely. This is due to the fundamental nature of the Balinese script which is a complex alpha syllabic script[6]. A glyph (letterform) can map to more than one syllable of the Latin alphabet. To assist users by suggesting several possible transliteration from the initial OCR-Transliteration system, a post transliteration suggestion module is needed. This research presents an initial study and proposition for a general scheme and model for suggesting several possible transliteration with text generation and LSTM for Lontar collection Figure 1.

![Figure 1. The general flow of the transliteration process for lontar collection.](image)

The main thematic of this research is motivated by the urgency of the real challenges lies in the preservation and dissemination of the nation's cultural knowledge stored in Lontar collection, across generations with the support of digital technology and computer vision which are currently developing rapidly. This paper is organized as follows: Section 2 will describe briefly the Lontar collection, the previous work on the Lontar transliteration system, and the corpus and dataset for building the transliteration system for Lontar. Our proposed model and method of post transliteration suggestion with text generation will be presented in Section 3. Section 4 will describe the results of our experiments. Finally, some conclusions and future works will be given in Section 5.

2. Transliteration of Balinese Palm Leaf Manuscript

2.1. Balinese Palm Leaf Manuscript (Lontar)
In most countries in the world, ancient manuscripts are usually found in the form of written text on paper or papyrus. Generally, these manuscripts contain a variety of very important information about the history of world civilization concerning various aspects of life. In Asian countries, more specifically in Southeast Asian countries such as Indonesia, Thailand, and Cambodia, most of the ancient manuscripts found were written on palm leaves [7–9]. The writing in Lontar is in Balinese script, with old Balinese language, mixed of Kawi and Sanskrit. Lontar collections found in Indonesia hold invaluable information regarding the nation’s social and cultural history, including the religion, traditional ceremonies, customary law, art and literature, architecture, including matters relating to traditional medicine, agriculture, and buying and selling systems in trade.

2.2. Previous work on Balinese Palm Leaf Manuscript Transliteration
This research will be based on specific research findings that have been carried out previously within the framework of the AMADI Project [4]. Several document image analysis tasks were already investigated and implemented under the AMADI Project, such as the dataset construction and ground truth creation protocol[4], the text line segmentation methods [10], the isolated character recognition system [2,11], the automatic transliteration [5,6], the word recognition system [1], as well as a word spotting system for Lontar collection [3]. Especially for the task of Lontar transliteration, the project has reported several research results regarding the final performance obtained from the segmentation based text recognition with OCR system and automatic transliteration of Lontar images into text with the Latin alphabet [2,5,6,11]. The segmentation free word transliteration with the LSTM model was also previously investigated [1]. The results of the transliteration still need to be improved.
In this study, the specific objective to be achieved is to build a post transliteration suggestion module to be integrated after the initial transliteration process for Lontar. This proposed model is expected to be able to help in suggesting several possible transliterations based on the initial transliteration from the previous system. It is hoped that the LSTM model can be used to predict and to suggest several possible transliterations by using the text generation model from the existing collection of transliterated texts of Lontar.

2.3. Corpus and Dataset for Balinese Palm Leaf Manuscript Transliteration

The digital image corpus of Lontar used in this study comes from the AMADI_LontarSet corpus[4]. The image samples of Lontar of AMADI_LontarSet were collected from 23 different Lontar collections, originating from 5 (five) different locations (regions). Two collections came from 2 (two) museums and 3 (three) other collections came from 3 (three) private family Lontar collections. Of the 23 collections, 10 (ten) collections were randomly selected from the Museum Gedong Kirtya, Singaraja, Regency of Buleleng, North Bali; 4 (four) collections were from the MuseumBali, Denpasar, South Bali; 7 (seven) collections were the private family collections from Village of Jagaraga, Regency of Buleleng; and 2 (two) other collections were private family collections from Village of Susut, Regency of Bangli and from Village of Rendang, Regency of Karangasem. From the entire collections, 393 pages of digital images of Lontar were captured.

In one Lontar page, most of it consists of 4 (four) lines of text, as well as 4 (four) transcribed text from the ground truth dataset. But, not all of the Lontar pages in the corpus contain text. Several Lontar pages only contain pictures or are just blank pages as markers for the start and end of a collection. Thus, from the 393 pages of Lontar images, 1,458 lines of transcribed text were obtained. From the whole corpus of Lontar images and transliteration text, a standard dataset has been set for the needs of word recognition and word spotting research in the previous work [4,12,13]. This dataset has been validated by the Balinese philologist, who validates the transliteration text of each word segment in Lontar images. In this dataset, there are 25,497 word segments of Lontar, consisting of 7,208 different (unique) words with 52,256 total characters from the entire word collection. The dataset for text generation of Post-OCR-Transliteration correction will further be built from these 7,208 different word collections.

3. Proposed Model and Method

3.1. Model for Text Dataset Generation

Before constructing a dataset for transliteration correction, the following should be noted based on our existing 7,208 words dataset:

1. The entire text format of the word collection would be changed and uniformed in a lower case format.
2. Extracted from the word text collection, there were 38 unique characters (as the number of vocabulary), including the NEW LINE character (\n') and the STAR character (*'), which were intentionally added for the needs of characters padding when the input word size was not the same length for the LSTM train model later.
   \['\n', '\', ' ', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'r', 's', 't', 'u', 'w', 'y', 'ã', '‹', '*

3. All unique characters would be mapped with an integer label from number 0 to 37, sorted according to the alphabetical order of the character.

Char To Int:
\{'\n': 0, '\': 1, ':': 2, ',': 3, '0': 4, '1': 5, '2': 6, '3': 7, '4': 8, '5': 9, '6': 10, '7': 11, '8': 12, '9': 13, 'a': 14, 'b': 15, 'c': 16, 'd': 17, 'e': 18, 'g': 19, 'h': 20, 'i': 21, 'j': 22, 'k': 23, 'l': 24, 'm': 25, 'n': 26, 'o': 27, 'p': 28, 'r': 29, 's': 30, 't': 31, 'u': 32, 'w': 33, 'y': 34, 'ã': 35, 'č': 36, '*': 37\}

The dataset for post transliteration suggestion was built using the **Edit-Insert-Replace** model, to be applied on the existing word collection dataset. For each word in the dataset, it will generate [(N*3) + N] = (N*4) pairs of input words and target words. N is the length (number of characters) of the original
word. In this case, the *Edit-Insert-Replace* model would only generate the input word, while the target word was the same for all word pairs, that is the original word from the collection. A total of \((N*3)\) generated words were input words modified by the *Edit-Insert-Replace* model at each character position (from positions 0 to N-1). Meanwhile, N other words were original input words that have not undergone modification, they remained as the same original word, Figure 2.

![Diagram](https://via.placeholder.com/150)

**Figure 2.** Edit-Insert-Replace model for text generation.

Table 1 shows an example of the *Edit-Insert-Replace* model from one original word “brahma”, with the length (number of characters) N = 6. The number of word pairs to be generated is 24 word pairs.

| No | Modification Char Position | Modification Model | Input Word | Target Word | No | Modification Char Position | Modification Model | Input Word | Target Word |
|----|-----------------------------|--------------------|------------|-------------|----|-----------------------------|--------------------|------------|-------------|
| 1  | 0                           | Edit               | yraham      | brahma      | 13 | 3                           | Edit               | bra*ma     | brahma      |
| 2  | 0                           | Insert             | 6brahma     | brahma      | 14 | 3                           | Insert             | bra7hma    | brahma      |
| 3  | 0                           | Delete             | rahma       | brahma      | 15 | 3                           | Delete             | brahma     | brahma      |
| 4  | 0                           | No                 | brahma      | brahma      | 16 | 3                           | No                 | brahma     | brahma      |
| 5  | 1                           | Edit               | bkahma      | brahma      | 17 | 4                           | Edit               | brahga     | brahma      |
| 6  | 1                           | Insert             | b*rahma     | brahma      | 18 | 4                           | Insert             | brah*ma    | brahma      |
| 7  | 1                           | Delete             | bahma       | brahma      | 19 | 4                           | Delete             | braha      | brahma      |
| 8  | 1                           | No                 | brahma      | brahma      | 20 | 4                           | No                 | brahma     | brahma      |
| 9  | 2                           | Edit               | br1hma      | brahma      | 21 | 5                           | Edit               | brahm,     | brahma      |
| 10 | 2                           | Insert             | brrahma     | brahma      | 22 | 5                           | Insert             | brahm0a    | brahma      |
| 11 | 2                           | Delete             | brhma       | brahma      | 23 | 5                           | Delete             | brahm      | brahma      |
| 12 | 2                           | No                 | brahma      | brahma      | 24 | 5                           | No                 | brahma     | brahma      |

The text generation process with the *Edit-Insert-Replace* modification model was only carried out on words that had a length (number of characters) of at least 2 (two) characters. Thus, from a collection of 7,208 unique words, 180,060 generated words were finally obtained. Likewise, during the test process, the post transliteration suggestion process would only be carried out on words that have a length (number of characters) of at least 2 (two) characters. For words that contain only 1 (one) character, the
initial transliterated text was assumed to be correct (usually most of them are the punctuation marks, namely CECEK, or unit numbers 0-9).

3.2. Feature Extraction Method

After generating the modified input words, the feature for post transliteration suggestion would be extracted. The extracted features from each word was a combination of 2 (two) characters before and after the position of a character that would be checked for correction. So that the vector size of this basic feature was 4, from each character position in word input. To fulfill this condition, all input words were padded first with ** character at the beginning and at the end, so that all characters from the first position to the last character of the word could meet the requirements of 2 (two) characters before and after the feature that would be extracted.

As an example, for the word “brahma” in modification character position = 0, feature extraction would be carried out on each modified word that had been built in the following steps:

1. Applying modification for original word, and padding of character “**”

   Original word : seq_in = ***+brahma+*** = **brahma**
   Edit model : seq_edit = ***+ yrahma +*** = **yrahma**
   Insert model : seq_insert = ***+6brahma +*** = **6brahma**
   Delete model : seq_del = ***-yrahma+*** = **yraham**
   Target word = brahma

2. Mapping character into numerical value of the character with Char To Int:

   Original word : convert_in = [37, 37, 15, 29, 14, 20, 25, 14, 37, 37]
   Edit model : convert_edit = [37, 37, 34, 29, 14, 20, 25, 14, 37, 37]
   Insert model : convert_insert = [37, 37, 10, 15, 29, 14, 20, 25, 14, 37, 37]
   Delete model : convert_del = [37, 37, 10, 15, 29, 14, 20, 25, 14, 37, 37]
   Target word : out_label = [15, 29, 14, 20, 25, 14]

3. Then the basic feature extraction was carried out at each character position in the word. For example, the feature extracted from seq_edit and convert_edit:

   seq_edit = ***+ yrahma +*** = **yrahma**
   convert_edit = [37, 37, 34, 29, 14, 20, 25, 14, 37, 37]
   feature from convert_edit = [[37, 37, 34, 29], [37, 34, 14, 20], [34, 29, 20, 25],
   [29, 14, 25, 14], [14, 20, 14, 37], [20, 25, 37, 37]]
   with target word out_label = [15, 29, 14, 20, 25, 14]

4. The padding process with character STAR (“**”) was needed to generate the uniform length of feature size and target label from each word. In our experiment, we define max_pad = 10 characters.

   The feature from convert_edit becomes =[[37, 37, 34, 29], [37, 34, 14, 20], [34, 29, 20, 25],
   [29, 14, 25, 14], [14, 20, 14, 37], [20, 25, 37, 37],
   [37, 37, 37], [37, 37, 37], [37, 37, 37], [37, 37, 37],
   [37, 37, 37]]

   and the target word out_label becomes  out_label = [15, 29, 14, 20, 25, 14, 37, 37, 37, 37, 37]

5. Then out_label was converted into a categorical form based on the number of vocabulary letters, (in this case 38 unique characters)

6. The whole process would be performed for each generated input word from the Edit-Insert-Replace model. From the original word input of “brahma” with 24 generated input words, and after the padding process for feature and target label, we finally reshaped the feature into (nb_word, max_pad, 4) and get the features in size X = (24, 10, 4) with the target out_label in size Y = (24, 10, 38).

7. From the total dataset, as we had 180,060 generated words from total words collection, the final feature size was X = (180060, 10, 4) and the final target size was Y = (180060, 10, 38). The X and Y data were then used as input feed for the training process using the LSTM model.

5
3.3. LSTM Model for Post Transliteration Suggestion

By using the Tensor Flow framework, the following Bidirectional LSTM model was built for the training process of post transliteration suggestion model:

```python
model = Sequential()
model.add(Bidirectional(LSTM(100, activation='relu'),input_shape=(max_pad, 4)))
model.add(RepeatVector(max_pad))
model.add(Bidirectional(LSTM(100, activation='relu', return_sequences=True)))
model.add(TimeDistributed(Dense(n_vocab, activation='softmax')))    
model.compile(optimizer='adam', loss='mse')
history = model.fit(X, Y, epochs=100, validation_split=0.2, verbose=1, batch_size=1000, callbacks=callbacks_list)
```

3.4. Model for Word Query Generation

During the test for an input word, the same process of feature extraction was carried out. For one word as a query test, the feature was in size of (1, 10, 4). In order to provide several transliteration suggestions for the post transliteration suggestion process, the original input word query testing were modified using **Edit-Insert-Replace** model during the training process. But for the correction test process, only (N*3) new words query test would be generated from one original word query test. N is the length (number of characters) of the original word query test. Table 2 shows an example of word query test list generated from one original word query test of “dharma” (one expected transliteration suggestion would be “dharma”). From a word query test of “dhrma”, 15 new words query test were generated: ['ohrma', 'pdhrma', 'hrma', 'd.rma', 'dhrma', 'dhrma', 'dhkrma', 'dhma', 'dhraa', 'dhr4ma', 'dhra', 'dhrm2', 'dhrm9a', 'dhrm'].

### Table 2. Wordquery to be generated from word test of ‘dhrma’.

| NO | Modification Char Position | Modification on Model | Generated Word Query | NO | Modification Char Position | Modification on Model | Generated Word Query |
|----|---------------------------|-----------------------|----------------------|----|---------------------------|-----------------------|----------------------|
| 1  | 0                         | Edit                  | Ohrma                | 9  | 2                         | Delete                | dhma                 |
| 2  | 0                         | Insert                | pdhrma               | 1  | 3                         | Edit                  | dhraa                |
| 3  | 0                         | Delete                | hrma                 | 1  | 3                         | Insert                | dhraa                |
| 4  | 1                         | Edit                  | d.rma                | 1  | 3                         | Delete                | dhra                 |
| 5  | 1                         | Insert                | dyhrma               | 1  | 4                         | Edit                  | dhrm2                |
| 6  | 1                         | Delete                | drma                 | 3  | 4                         | Insert                | dhrm9a               |
| 7  | 2                         | Edit                  | dhmma                | 4  | 4                         | Delete                | dhrm                 |
| 8  | 2                         | Insert                | dhkrma               | 5  |                           |                       |                      |

By using the Tensor Flow framework, the following Bidirectional LSTM model was built for predicting process of post transliteration suggestion model:

```python
model = Sequential()
model.add(Bidirectional(LSTM(100, activation='relu'),input_shape=(max_pad, 4)))
model.add(RepeatVector(max_pad))
model.add(Bidirectional(LSTM(100, activation='relu', return_sequences=True)))
model.add(TimeDistributed(Dense(n_vocab, activation='softmax')))    
test_output = model.predict(test_input, verbose=0)
```
4. Results and Discussion
For the training process, of the 180,060 trained data, the data were divided into validation: train on 144,048 samples and validate on 36,012 samples. Figure 3 shows the loss value evolution during the training process until 1300 epochs.

![Figure 3. Loss value evolution during the training process.](image)

With the weighted model obtained from the training process, we performed a test for a word query of “dhrma”. Then the post transliteration suggestion process is carried out for one original word query and for each of the new generated word query to obtain (N*3) = 5*3 = 15 possible transliteration suggestions (Table 3). Table 4 shows 5 (five) other examples of word query with all the possible transliteration suggestions.

### Table 3. All possible transliteration suggestions from a word query “dhrma”.

| No | Generated Word Query | Transliteration Suggestions | No | Generated Word Query | Transliteration Suggestions |
|----|----------------------|-----------------------------|----|-----------------------|-----------------------------|
| 1  | dhrma                | dharma                      | 9  | dhkrma                | dharma                      |
| 2  | ohrma                | oorama                      | 10 | dhma                  | dama                        |
| 3  | pdhrma               | pdira                       | 11 | dhraa                 | dirana                      |
| 4  | hrmA                 | 7rama                       | 12 | dhr4ma                | dharma                      |
| 5  | d.rma                | darma                       | 13 | dhra                  | dara                        |
| 6  | dyhrma               | dihpa                       | 14 | dhrm2                 | dhamma                      |
| 7  | drma                 | drama                       | 15 | dhrm9a                | dhirma                      |
| 8  | dhmma                | damma                       | 16 | dhrm                  | dhi                         |

### Table 4. All possible transliteration suggestions from 5 (five) examples of word query

| No | Word Query | Expected Suggestion | Transliteration Suggestions |
|----|------------|---------------------|-----------------------------|
| 1  | dwwa       | dewa                | dawa; dawa; prwai; wwaa; dawaa; daw; dwwa; dwwca; dumaa; dwwa; dwyt; dwwa; dwy |
| 2  | surga      | suarga              | suarga; muraga; supa; uurga; sarga; suuaa; srrga; sugaa; suerga; sugaa; surya; surara; surra; surel; suara; sura |
| 3  | kawasanin  | kawastanin          | kawastanin: wawasaniing; kawasaini; kawastnnin; kawasinn; kawasanin; kwsannin; kawasang; kawasaniing; karsinini; kawastanin; kawastnnin; kawasani; kawasnannin; kawastanin; kawaran; kawaseenin; kawastanin; kawasinnan; kawasani; kawastanin; kawastan; kawasannin; kawasannin; kawasann; kawasna; kawasninka; kawasani |
| 4  | wennang    | wenang              | wennang; kenang; wewangang; eenang; wanang; weneang; wannag; wennang; wenanang; wenang; wennang; wenang; wenang; weng; weng; wenong; wenong; wenong; wenong; wenong |
| 5  | aksraning  | aksaraning          | krsraning; nassraning; kaaraning; aasraning; anisaning; aasraning; akcraning; akrrraning; akcraning; aksaraning; aksaraning |

0.05 Loss Value
Epoch vs Loss Value
Epoch
1 100 200 300 400 500 600 700 800 900 1000 1100 1200
Loss Value
With several possible transliteration suggestions, this module will be able to help in assisting the user to predict the correct transliteration text from the incorrect initial transliteration result. There are still cases where this module could not produce at least one correct transliteration text from all possible transcriptions it suggests. But this can be improved by continuing the training process with additional data collections which would trigger the additional generated text with our proposed text generation model. This post transliteration suggestion module will be integrated into the AKSALont application (https://aksalont.mudratech.org) which will carry out the initial transliteration process. AKSALont will then provide several transliteration suggestions based on the resulting initial transliteration text.

5. Conclusion
An initial study and proposition for a general scheme and model for suggesting several possible transliterations with text generation and LSTM for Lontar collection are presented in this paper. The dataset for post transliteration suggestion was built using the Edit-Insert-Replace model to be applied on the existing word collection dataset. A Bidirectional LSTM model was built for the training process of the post transliteration suggestion model. This proposed model is to help in suggesting several possible transliterations based on the initial transliteration from the previous system. Future improvement is needed by continuing the training process with more additional data collections for the proposed text generation model.

Acknowledgements
This research is financially supported by Skema Penelitian Terapan DIPA Universitas Pendidikan Ganesha 2020.

References
[1] Kesiman M W A 2019 Word Recognition for the Balinese Palm Leaf Manuscripts 2019 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom) 2019 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom) (Banda Aceh, Indonesia: IEEE) pp 72–6
[2] Kesiman M W A, Pradnyana G A and Maysanjaya I M D 2020 Balinese glyph recognition with gabor filters J. Phys. Conf. Ser. 1516 012029
[3] Kesiman M W A and Pradnyana G A 2019 A Complete Scheme of Word Spotting System for the Balinese Palm Leaf Manuscripts 2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE) 2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE) (Pattaya, Thailand: IEEE) pp 1–5
[4] Kesiman M W A, Burie J-C, Ogier J-M, Wirawan G N M A and Sunarya I M G 2016 AMADI_LontarSet: The First Handwritten Balinese Palm Leaf Manuscripts Dataset 15th International Conference on Frontiers in Handwriting Recognition 2016 15th International Conference on Frontiers in Handwriting Recognition (Shenzhen, China) pp 168–72
[5] Kesiman M W A, Burie J-C and Ogier J-M 2017 A Complete Scheme of Spatially Categorized Glyph Recognition for the Transliteration of Balinese Palm Leaf Manuscripts 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR) 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR) (Kyoto: IEEE) pp 125–30
[6] Antara Kesiman M W, Burie J C, Ogier J M and Grangé P 2018 Knowledge Representation and Phonological Rules for the Automatic Transliteration of Balinese Script on Palm Leaf Manuscript *Computación y Sistemas* 21

[7] Chamchong R and Fung C C 2012 Text Line Extraction Using Adaptive Partial Projection for Palm Leaf Manuscripts from Thailand *2012 International Conference on Frontiers in Handwriting Recognition* 2012 International Conference on Frontiers in Handwriting Recognition (IEEE) pp 588–93

[8] Valy D, Verleysen M, Chhun S and Burie J-C 2017 A New Khmer Palm Leaf Manuscript Dataset for Document Analysis and Recognition – SleukRith Set *4th International Workshop on Historical Document Imaging and Processing* 4th International Workshop on Historical Document Imaging and Processing (Kyoto, Japan)

[9] Kesiman M, Valy D, Burie J-C, Paulus E, Suryani M, Hadi S, Verleysen M, Chhun S and Ogier J-M 2018 Benchmarking of Document Image Analysis Tasks for Palm Leaf Manuscripts from Southeast Asia *Journal of Imaging* 4 43

[10] Kesiman M W A, Valy D, Burie J-C, Paulus E, Sunarya I M G, Hadi S, Sok K H and Ogier J-M 2016 Southeast Asian palm leaf manuscript images: a review of handwritten text line segmentation methods and new challenges *Journal of Electronic Imaging* 26 011011

[11] Kesiman M W A, Prum S, Burie J-C and Ogier J-M 2016 Study on Feature Extraction Methods for Character Recognition of Balinese Script on Palm Leaf Manuscript Images *23rd International Conference on Pattern Recognition* 23rd International Conference on Pattern Recognition (Cancun, Mexico)

[12] Burie J-C, Coustaty M, Hadi S, Kesiman M W A, Ogier J-M, Paulus E, Sok K, Sunarya I M G and Valy D 2016 ICFHR 2016 Competition on the Analysis of Handwritten Text in Images of Balinese Palm Leaf Manuscripts *15th International Conference on Frontiers in Handwriting Recognition* 2016 15th International Conference on Frontiers in Handwriting Recognition 2016 (Shenzhen, China) pp 596–601

[13] Kesiman M W A, Valy D, Burie J C, Paulus E, Suryani M, Hadi S, Verleysen M, Chhun S and Ogier J-M 2018 ICFHR 2018 Competition On Document Image Analysis Tasks for Southeast Asian Palm Leaf Manuscripts *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)* 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR) (Niagara Falls, NY, USA: IEEE) pp 483–8