Pattern Recognition Letters 133 (2020) 8–16

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Nom document digitalization by deep convolution neural networks

Kha Cong Nguyen, Cuong Tuan Nguyen*, Masaki Nakagawa

Department of Computer and Information Sciences, Tokyo University of Agriculture and Technology, 2-24-16 Naka-cho, Koganei-shi, Tokyo 184-8588, Japan

ARTICLE INFO

Article history:
Received 16 July 2019
Revised 21 December 2019
Accepted 16 February 2020
Available online 19 February 2020

ABSTRACT

Nom is an ancient script used in Vietnam until the current Latin-based Vietnamese alphabet became common, and a large number of ancient Nom documents are in existence. Due to the gradual degradation of Nom documents and a decrease in the number of scholars who can understand them, a system to digitalize Nom documents is urgently necessary. This paper presents a segmentation-based method for digitalizing Nom documents using deep convolution neural networks. Nom pages are preprocessed, segmented into isolated characters, and then recognized by a single-character OCR. The structure of the U-Net is applied to create segmentation maps and extract character regions from them. Subsequently, we propose coarse and fine combined classifiers to recognize each character pattern. The results by the best classifier are revised by a decoder using a langue model. The decoder is the same as the connectionist temporal classification decoder used in end-to-end text recognition systems. Compared with the traditional segmentation method using projection profiles and the Voronoi diagram ($IoU = 81.23\%$), the segmentation method using the deep convolution neural network produces a better result ($IoU = 92.08\%$) for detecting character regions. The proposed CNN models for recognizing segmented character patterns outperforms the traditional models using the modified quadratic discriminant function and the learning vector quantization with the recognition rate of 85.07%. The combination of coarse and fine classifiers, the training dataset with salt and pepper noises, and the attention layer are the key factors in the recognition rate improvement.

1. Introduction

Nom is an ancient script used in Vietnam until the current Latin-based Vietnamese alphabet became common. From the tenth century to the twentieth century, all of the documents in Vietnam were recorded by Nom, so that tens of thousands of Nom documents are stored in families, pagodas, churches, and libraries. We face a high risk that the invaluable Vietnamese history would be lost and could not be accessed by the next generations because Nom documents are gradually degrading, most of them have not yet been digitized, and the number of scholars who can read Nom documents is getting smaller. Fig. 1 shows a typical Nom document scanned by the National Library of Vietnam. The Nom pages often include boundary lines and rule lines, and are usually noisy and degraded.

Recently, several institutes and libraries started projects to digitalize Nom documents for reserving this heritage such as the National Library of Vietnam [1], the General Library of Thua Thien Hue and the Temple University Library [2], the Tue Quang wisdom light foundation [3] and so on. Nevertheless, they share the common drawback that they do not have highly accurate OCRs for recognizing Nom characters after scanning Nom documents so that the digitalizing process mostly depends on the interpretation by Nom experts. Now all over the world, however, the number of experts who can comprehend Nom script is less than 100, and most of them are aged.

Nom has a large character set of tens of thousands of characters. It has Chinese origin and new characters were composed of characters or their radicals [4]. About 60% of Nom characters were invented by Vietnamese people. This proportion increased while characters of Chinese origin decreased, as the documents were made later.

Previously, we developed a system to recognize Nom characters on the documents scanned by the National Library of Vietnam [5]. In the system, Nom documents are firstly preprocessed and binarized, then segmented by the method based on projection profiles and the Voronoi diagram. The segmented patterns are recognized by OCRs using generalized learning vector quantization (GLVQ) and modified quadratic discriminant function (MQDF). Finally, the system provides a GUI for users to revise the recognition results and

* Corresponding author.
E-mail address: ntcuong2103@gmail.com (C.T. Nguyen).

https://doi.org/10.1016/j.patrec.2020.02.015
0167-8655/

Fig. 1. A typical Nom page.

save the results to text files. Although we applied the histogram analysis method to remove boundary lines and rules lines in the Nom documents, many noises still remain, so that the above segmentation method often fails and requires the user's corrections. Another problem is that the recognition rates of the OCRs are still low due to the lack of adequate training patterns for a large number of the Nom categories. The OCRs have not yet been combined with a language model for correcting wrong recognition results. Although the system allows users to automatically convert scanned Nom pages into text files, they have to check the recognized results again. This makes the digitalization process still consuming a huge human resource.

To resolve the above problems, we first propose a character extraction method based on the U-Net structure [6]. We train the models with synthetic data from Nom fonts. To create artificial Nom pages, we generate single character patterns from fonts and paste them in different scales to empty pages. The ground-truths of character regions are convex-hulls of character patterns instead of rectangle boxes to reduce the overlap between character regions. Because the trained models sometimes produce touching character regions, the marker watershed method is applied to separate them.

Since Nom script includes tens of thousands of categories, and we do not have real training patterns for Nom, the OCRs using simple features like directional features are ineffective. We propose a combination of a coarse classifier and a fine classifier to predict an input category by their output probabilities. We name this coarse and fine combined classifier. The coarse classifier calculates the probability of an input pattern to be in a super category (described in Section 4) while the fine classifier calculates the output probability for each fine category (character category). The coarse classifier and the fine classifier share the same feature extractor.

To recognize a Nom page, which was written in the vertical direction from top to bottom, left to right. We first segment the Nom page into character regions, then we group them into text lines based on their positions. We apply the coarse and fine combined character recognizer to each character region in a text line and concatenate the recognition candidates for each character into a sequence for the text line. Then, we apply the beam search decoder with a language model of Nom for revising the final recognition results.

The remainder of the paper is organized as follows. Section 2 presents related work. Section 3 shows the character region extraction method. Section 4 describes the proposed coarse and fine combined classifier. Section 5 describes the beam search decoder with a language model for Nom documents. Section 6 presents the results of the experiment and Section 7 draws the conclusion.

2. Related work

Usually, there are three approaches for digitalizing documents to text: the segmentation-based approach [5,7,8], the text location and recognition approach [9,10] and the end-to-end text sequence recognition approach with Convolution Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) [11–13]. In the segmentation-based approach, documents are segmented to character regions, and an OCR is applied to recognize them. In the text localization and recognition approach, CNNs are used to extract feature maps of input images and character regions are proposed on the maps by a Region Proposal Networks (RPN). Then each character region is recognized by a classifier. In the third approach, the feature maps extracted by CNNs can also be fed into RNNs and a transcription decoder is used to decode time step outputs of the RNNs to text sequences. The second approach employs the end-to-end training process. Both the location loss and the recognition loss are optimized once, so that it requires a huge number of training patterns, which is often serious for historical document recognition. Moreover, the combination of ten-thousand category OCR into the end-to-end model makes the whole network hard to converge and complicated to analyze. The third approach is even more sophisticated and requires a huge number of patterns for training. Both the second and the third approaches are also very sensitive to variations in the handwriting styles.

In this paper, we focus on the first approach, but we also borrow a transcription decoder with a language model in the third approach for correcting the recognition results. We separate the end-to-end process into two steps: location and recognition, and train them separately.

In the segmentation-based approach, text pages are segmented into text lines based on projection profiles, the Hough transform, smearing [14] and then separated into single characters by analyzing the fore and background of text line images [15]. Another approach is based on local features [16]. The authors generated template images and extracted SIFT features for multiple-size sliding windows and matched with template images. Voting and geometric verification algorithms are used to decide final results. Baek et al. (2019) applied the U-Net based network for character region detection. They utilize not only character regions but also the affinity between characters [17]. Due to the lack of real training data, they trained the model for synthetic images first, then used it to estimate character regions for real images.

As for large category classification, many methods group objects into coarse categories before classifying them into fine categories, which is known as coarse-to-fine classification. Cevahir et al. (2016) combined deep belief nets and deep auto-encoder neural network models to firstly classify large-scale e-commerce data into five super classes, and then classify them into 28,338 fine categories [18]. They utilized all textual contents such as titles and descriptions but ignored image contents of products. Yan et al. (2015) applied a hierarchical deep convolution neural network (HD-CNN) for large scale visual recognition [19]. They first pre-trained a CNN model for a fixed number of coarse categories (5, 9, 14 and 19). Then, they pre-trained a model for each coarse category. After both the coarse category network and the fine category networks for the coarse categories were properly pre-trained, they fine-tuned the complete HD-CNN. Because the number of parameters in the fine category networks grew linearly with the number of coarse categories, they compressed the parameters of the fine category networks by K-mean clustering.

Pre-training a large number of fine category networks, however, is time-consuming, so that it is hard to apply for recognizing Nom.

Jie et al. (2017) considered coarse categories and fine categories for weakly supervised learning [20]. They had more training data labeled with coarse categories but fewer data labeled with fine categories. From such training data, they trained a model that could classify a new image into one of the fine categories. This is not the combination of a coarse classifier and a fine classifier, but we are inspired by the method of how coarsely labeled data can help fine label classification.

As for text-sequence decoding, there are many text sequence decoding algorithms proposed so far. The basic algorithm is the best path decoding that takes the most likely candidate at each time step. Graves et al. (2009) proposed the token passing algorithm to search the most probable sequence from the output matrix in a dictionary word [21]. The algorithm constrains the output to a context dictionary, so it cannot handle arbitrary character sequences. In 2012, he also introduced the beam search decoding method that integrated a language model for arbitrary character sequences [22]. The beam search algorithm expands all possible next steps and keeps the k most likely, known as the beam width.

3. Character extraction method

To extract character images, we follow the structure of U-Net to classify each pixel in Nom pages into background pixels and pixels in character regions. The architecture of the network includes an encoder to capture context and a symmetric expanding decoder to relocate precise locations. The network learns to predict the character regions as polygons instead of rectangles so that it reduces the touching between character segmentation results.

Although we follow the U-Net architecture, we consider a few different encoders such as VGG-16 [23], Resnet-50 [24] or Inception-ResNet-V2 [25] as shown in Appendix A to down-sample training images to the small sizes of feature maps which contain the information of character regions. Assuming the encoder here is a VGG-16 based structure, the corresponding decoder up-samples the feature maps and produces the segmentation maps. Fig. 2 shows the character extraction network, including the down-sampled feature maps (boxes in sky-blue) by the encoder, the up-sampled feature maps (boxes in yellow) by the decoder, and the network operations (colored arrows). The input size of the network is set to 512× 512. Following each convolution layer is batch normalization layer, which computes the mean and the standard deviation of all the output feature maps and then normalizes them. That makes all feature maps have the same range and zero mean, so that helps the training process of the next layer not have to learn offsets of data, which is known as the covariate-shift problem. The information of the former convolutional layers in the encoder is passed to the up-sampling layers in the decoder, which can avoid the vanishing gradient problem. In detail, the features by each multi-level down-sampling layer in the encoder are concatenated to those of the same size by the corresponding multi-level up-sampling layer in the decoder as shown in the green arrows in Fig. 2. This concatenation is known as the skip connection.

Character regions produced by trained models, still sometimes touch each other, especially at the confusing boundary pixels of convex-hull character regions. The produced segmentation maps are finally segmented by the marker-based watershed algorithm. Using the erosion morphology, we can locate the sure centers of character regions (markers). We apply the watershed algorithm with decided markers to segment touching character regions.

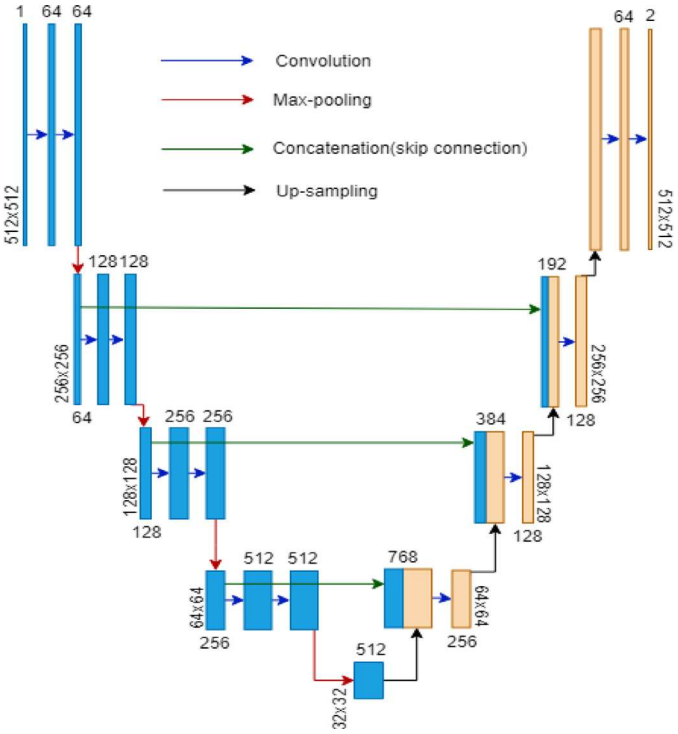


Fig. 2. Character region extraction network with the VGG-16 decoder.

4. Coarse and fine combined classifier

We present a novel architecture of combining a coarse and fine classifier for a huge number of categories (denoted by CF_Combined). We apply a coarse classifier to classify an input pattern into a super category and a fine classifier to classify it into a particular class known as the fine category. Then, we multiply the coarse category probability and the fine category probability to predict the fine category probability. The coarse classifier is also used to guide the backpropagation for training the fine classifier. We will show that this architecture is better than just a single fine classifier for a large category set or a simple sequential architecture of the coarse-to-fine classifier.

4.1. Coarse category formation and category labeling

When Truyen et al. (2016) made the first attempt to make Nom OCRs, they pointed out that Nom script includes at least 32,695 categories based on studies on Nom fonts and publications from the Vietnamese Nom Preservation Foundation [5].

To create supergroups (called coarse categories), we group the 32,695 Nom categories (called fine categories) by the K-means algorithm. Since Nom is composed of Chinese characters or their radicals, we consider the number of clusters K to be more or less of the number of radicals in Nom, i.e., 304 [4]. However, if we group Nom categories by radicals, some characters in a group have very complicated structures while others have simple structures, with the result that coarse category classification may not correspond to the radical groups. Therefore, we use just this number to guide us to find the best number.

We follow the same process for feature extraction as the previous work [5]. We normalize Nom character patterns by non-linear line density projection interpolation (LDPI). Then, we extract directional features of 512 dimensions by the normalization-cooperated gradient feature (NCGF) method. To make features be more discriminative so that the clustering process by K-means is more effective, we reduce the dimension of original features to 160 by the

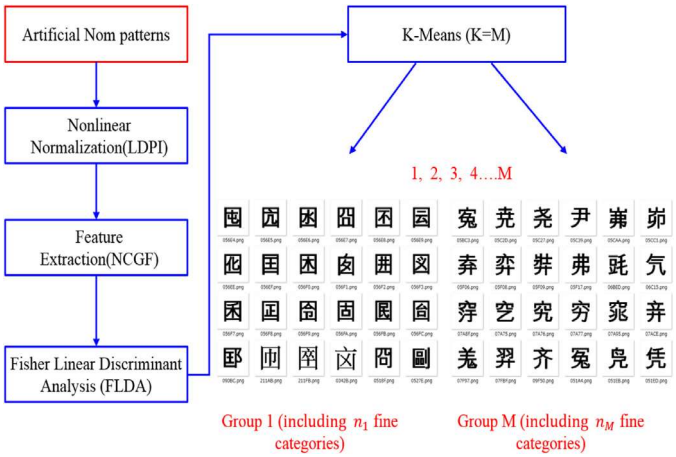


Fig. 3. Coarse category formation.

Fisher Linear Discriminant Analysis (FLDA). The clustering process to create coarse categories is shown in Fig. 3.

4.2. Coarse and fine combined classifier

The stem block in the Inception-ResNet-V2 [25] which is a VGG-16 liked structure, is combined with two fully connected layers and a softmax layer to form the coarse classifier as shown in the green color blocks in Fig. 4. The probability of an input pattern (x) assigned to a coarse class C_m, $m \in \{1,...M\}$, by the coarse category classifier after the softmax layer is $P_c(C_m|x)$. The probability of a fine category F_n, $n \in \{1,...,N = 32{,}965\}$ corresponding to the coarse category C_m is $P_c(F_n|C_m,x)$:

$$P_c(F_n|C_m, x) = P_c(C_m|x) if\ F_n \in C_m \tag{1}$$

The coarse category feature extractor is frozen its weights and is used to extract features for the fine category classifier as shown in the carrot color blocks in Fig. 4. The output probability of the fine category classifier $P_f(F_n|x)$ is multiplied with the output from the coarse classifier to create the final probability $P(F_n|x)$:

$$P(F_n|x) = \frac{P_f(F_n|x) * P_c(F_n|C_m, x)}{\sum_{n=1,\ m=1,\ F_n \in C_m}^{N,\ M} P_f(F_n|x) * P_c(F_n|C_m, x)} \tag{2}$$

The categorical cross entropy loss will be back-propagated to the shallow layer in the fine category classifier as shown by the blue arrow in Fig. 4.

The coarse and fine combined classifier is combined with a spatial attention layer. Assuming that the output feature map y of the last convolution layer in the Inception-ResNet-V2 has the size of (w, h, d) where w, h and d are the width, the height and the depth of the feature map, we calculate the spatial weight matrix W along feature deep dimension d as follows:

$$W(w, h, d) = \left[\frac{e^{y(w,h,d_k) - max_d(y)}}{\sum_k e^{y(w,h,d_k) - max_d(y)}} \right] \tag{3}$$

where $max_d\ (y)$ is the maximum value of the feature map along the axis d, d_k is the k^{th} element along the d axis. The feature map of the last convolution layer is weighted by adding the weight matrix.

Due to the binarization process, some details of character patterns have vanished while some noises are added by writers or later when preserving. To make the training dataset noised as the testing set, we add salt and pepper noises to the training images. The spatial attention layer helps the model to pay more attention to the details of character patterns rather than the noises on character regions.

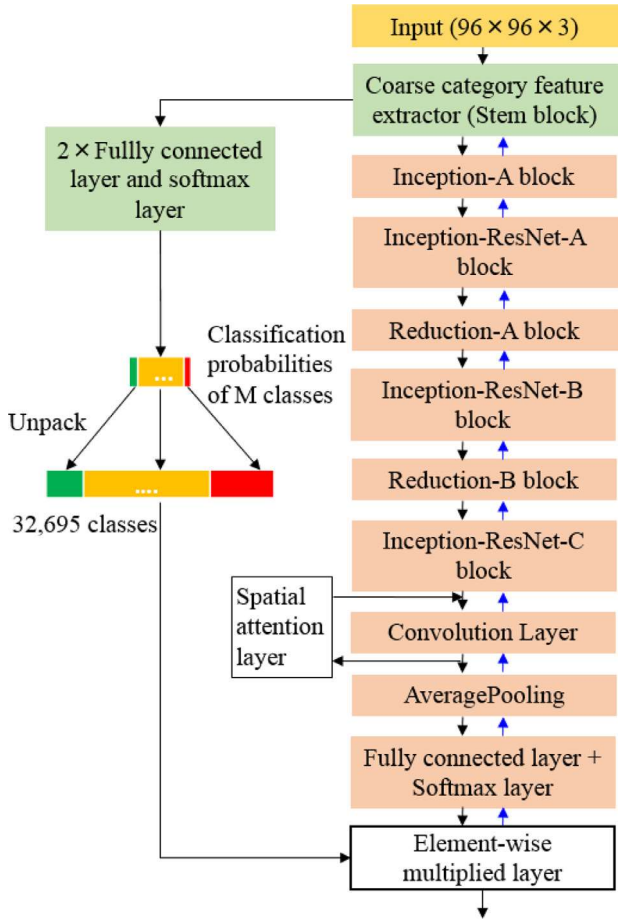


Fig. 4. Coase and fine combined classifier.

5. Beam search decoder with a language model

To find the best path through the sequence of recognition candidates for each character pattern in a text line, we apply a beam search decoder with a language model to decode the text. We use the unigram and the bigram for the beam search decoder. We used unigram and bigram because the collected corpus is not so large. Using trigram may improve the recognition rate, but trigram sequences are less frequent than unigram or bigram sequences. The occurrences of four-gram or higher are even fewer, so this approach is not robust because of the huge number of Nom categories in spite of the limited corpus. Assuming that $P(c_1)$ is the unigram probability of the candidate c_1 and $P(c_1|c_2)$ is the bigram probability of candidate c_2, the probability $P_{LM}(c_1,c_1...c_n)$ of a candidate sequence $c_1,c_1...c_n$ is shown in the following equation:

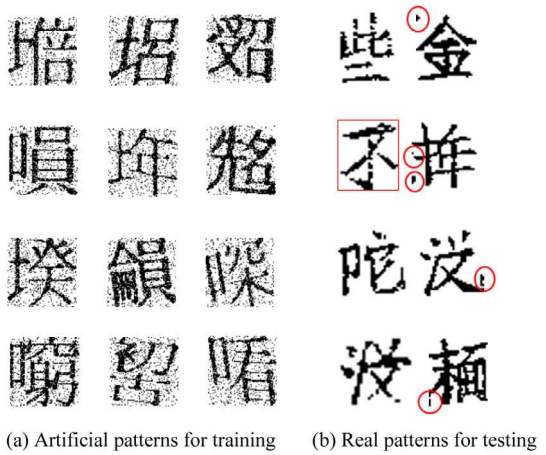
$$P_{LM}\ (c_1,\ c_2 \ldots c_n) = P(c_1) \times P(c_2|c_1) \times P(c_n|c_{n-1}) \tag{4}$$

The score $P(b)$ of the beam b will be multiplied with its context probability $P_{LM}(b)$ to create the overall score P and take the best sequence:

$$P = P(b) \times P_{LM}(b) \tag{5}$$

6. Experiment

In this section, we present the training and testing datasets for the OCRs and the character region extraction models. Then, we show their performances on the prepared datasets. The results of the beam search decoder with a Nom language model is also shown at the end of this section.



(a) Artificial patterns for training (b) Real patterns for testing

Fig. 5. Training and testing patterns for OCRs.

6.1. Training dataset and testing dataset

To train the OCRs for single characters, we prepared artificial Nom character patterns since we do not have a sufficient number of real patterns. We made about 1000 artificial patterns as shown in Fig. 5 (three left columns) for each fine category, using image deformation methods of the affine, rotate, shear, shrink, and prospective methods from 27 Nom fonts such as Nom Na Tong, Nom Khai, Nom Minh and so on [5,26]. In total, we have 28,035,360 artificial training patterns for 32,695 categories. We also added salt and pepper noises to the above-generated images to make training patterns, which is the key factor for improving the recognition rate on the testing dataset as analyzed later.

To train the models for character region extraction, we create empty page images of fixed size: 512×512, choose character patterns in the 28,035,360-character dataset randomly and paste the patterns in different scales to the page images as shown in Fig. 6.

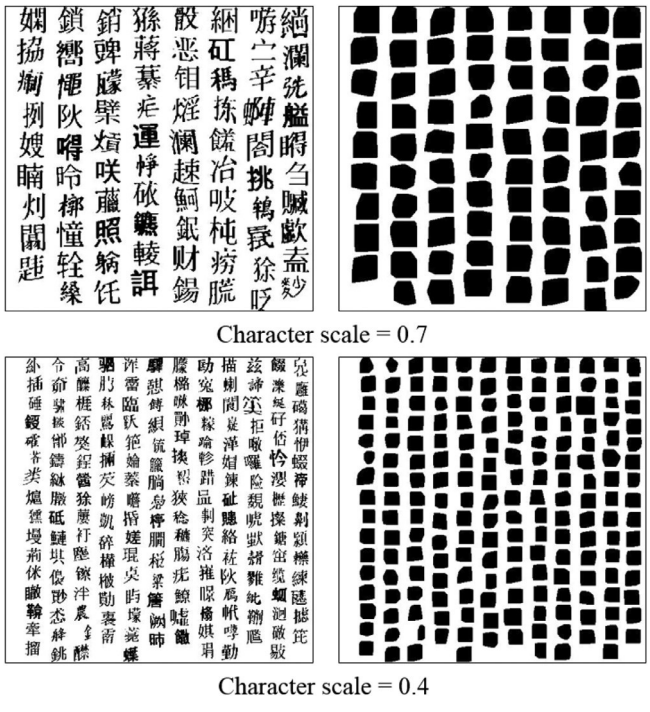


Fig. 6. Training patterns from different scale characters and convex-hull ground-truths.

We find a convex hull boundary of each character pattern to make its ground-truth. The convex hull boundaries are to reduce the touching between characters. In fact, the testing Nom documents do not have many touching cases, but they are written closely. The rectangle bounding box may make the touching among regions of characters. In total, we generate 58,150 Nom page images and their ground-truths. We separate them in a ratio of 0.8:0.2 for training and evaluating, respectively.

The testing dataset for the proposed OCRs was collected from 47-page images from ten real Nom documents which include 13,111 character rectangle bounding boxes in which 11,669 patterns are decoded to 2538 categories as shown in Fig. 5 (two right columns). The 2538 testing categories are scattered in the 32,695 training categories. We collected these real testing patterns by segmenting them based on projection profiles and the Voronoi diagram, applying the MQDF+GLVQ OCR, and then inspecting and revising errors manually as described in [5].

Since pages in a single document usually have similar backgrounds, we choose one page in each document for fine-tuning the character region extraction models which have been trained with the artificial dataset. We use the remaining 37 pages for evaluating the performance of the character region extraction models.

6.2. Evaluation of character extraction method

The encoders of the character extraction models are pre-trained with generated character patterns that are used for training OCRs in Section 6.3. To evaluate the character region extraction models, we employ Intersection over Union (*IoU*) metric as shown in the following equation:

$$IoU = \frac{Total\ pixels\ in\ overlap\ areas}{Total\ pixels\ in\ union\ areas} \tag{6}$$

Because the segmentation regions of the method in [5] are rectangular boxes, after getting the polygonal regions of character we also calculate the *IoU* metric on the rectangles of the polygonal regions and the ground-truths on 37 Nom real pages. The character region extraction models are trained in five epochs with the batch size of 2 and the Adam optimizer [27].

Table 1 shows the pixel level accuracies and the *IoU* metrics of the character region extraction with the different encoders on the evaluating dataset. Since the deeper encoders can take more information about the shapes of characters, they have produced better results.

Table 2 shows the character extraction results by different models on the real Nom dataset of 37 pages. The performance is reduced on the real dataset compared to the evaluating dataset. Increasing the number of scales of characters on Nom pages may help to improve the results. The second column is the evaluation in the polygonal ground-truths while the third column is in the rectangular regions. The applied method is almost ten percent point better than the projection profile and Voronoi diagram based method. In Appendix B, we show some segmentation results on real Nom pages. The first column is the segmentation by the combination of the projection profile and the Voronoi diagram. The second column is the polygonal segmentation maps by the fine-tuned models, and the last column displays the character regions after applying the marker-based watershed.

6.3. Performance of proposed OCRs

Table 3 shows the training accuracy, the top-one accuracy, and the top-ten accuracy on the testing set. CNN-based models such as VGG-16, Inception-ResNet-V2, CF_Combined with $M = 100$, 304 and the model with the spatial attention layer were trained in ten epochs, with the Adam optimizer [27] using the batch size of

Table 1
Character region extraction performance on the evaluating dataset.

Methods	Pixel level accuracy	*IoU*
Character region extraction with VGG-16 encoder	89.25	84.67
Character region extraction with ResNet-50 encoder	93.76	89.41
Character region extraction with Inception-Resnet-V2 encoder	**95.19**	**93.82**

Table 2
Character region extraction on the real testing dataset.

Methods	*IoU* (Polygonal regions)	*IoU* (Rectangular regions)
Projection profiles and Voronoi	_	81.23
Character region extraction with VGG-16 encoder	81.14	79.67
Character region extraction with ResNet-50 encoder	86.54	83.17
Character region extraction with Inception-Resnet-V2 encoder	**92.16**	**90.08**

Table 3
Training accuracy and testing accuracy of OCRs.

OCR models	Training accuracy (%)	Top-one testing accuracy (%)	Top-ten testing accuracy (%)
MQDF+GLVQ	96.36	69.08	86.03
VGG-16	96.47	76.09	91.42
Inception-ResNet-V2	97.53	79.93	92.13
CF_Combined ($M = 100$)	97.89	82.15	94.17
CF_Combined ($M = 304$)	**98.16**	82.26	94.04
CF_Combined ($M = 304$) with the attention layer and trained with noised data (BEST)	97.62	**85.07**	**94.76**

128. MQDF + GLVQ, VGG-16, Inception-ResNet-V2, CF_Combined ($M = 100, 304$) models are trained with generated patterns without noises. The proposed CF_Combined ($M = 100, 304$) produced the best top-one accuracy, outperformed the single fine classifiers of VGG-16 and Inception-ResNet-V2. CF_Combined models also outperformed the coarse-to-fine model of MQDF + GLVQ. The model by MQDF + GLVQ was over-fitted so that it could not produce a good result for the real dataset. Previously, we trained VGG-16 without the batch normalization but the network could not converge. Therefore, we used the batch normalization for the VGG-16 model. The three CF_Combined models which use coarse classifiers to guide the training for fine classifiers achieve better recognition rates than any single fine category classifiers of VGG-16 or Inception-ResNet-V2 after ten epochs. In the proposed architecture, the coarse classifiers also help the fine classifiers converge quickly as shown in Fig. 7. Moreover, the CF_Combined model ($M = 304$) with the added attention layer and trained with the noised patterns improved the top-one accuracy on the real test set. The improvement on the test set rather than the training set shows that the attention layer and added noises help the network reduce overfitting. Without adding noises to the training data, the CF_Combined model ($M = 304$) with the attention layer does not show its effectiveness. The green and red lines in Fig. 7 shows the losses of the CF_Combined ($M = 304$) and the CF_Combined model ($M = 304$) with the attention layer, trained without noised patterns, in which the training losses of two models are almost similar. We achieved the best recognition rate of 85.07% for Nom on the real testing dataset by the CF_Combined model ($M = 304$) with the added attention layer and trained with the noised patterns (called BEST model).

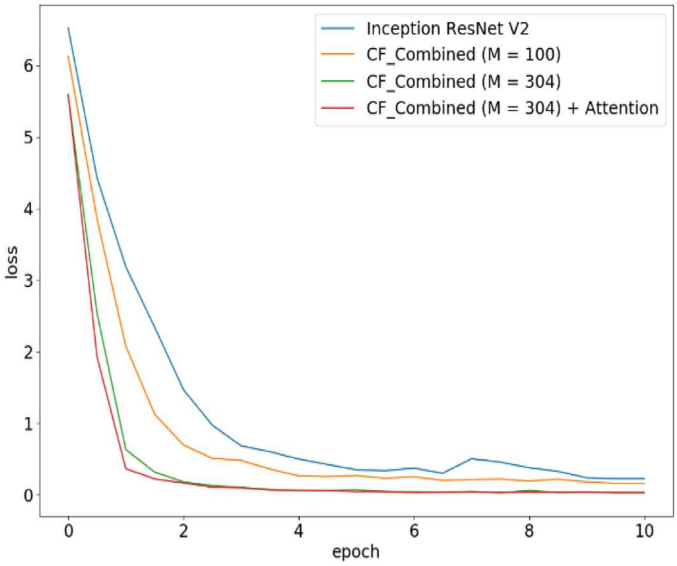


Fig. 7. Training losses of classifiers.

6.4. Evaluation of the performance of the beam search decoder

The language model is built from a dictionary of 26,063 Nom characters in the famous Nom poem named “the tale of Kieu”. The content of experimented text-pages is not included in the language model. We set the beam width to 5 or 10. Table 4 shows the improvement of the beam search decoder combined with the language model for the recognition rate of the best proposed OCR The

Table 4
Beam search decoder for the output sequences of the best CNN model.

Beam width	BEST model	Total time execution (s)
$k = 5$	85.18	177
$k = 10$	85.22	191

increase of the beam width improves the recognition rate a little, but it also incurs a longer time for the decoding process.

7. Conclusion

In this paper, we presented a segmentation-based approach for digitalizing historical documents in Vietnam that have a high risk of permanent disappearance to the next generations. Character regions are extracted from the preprocessed documents by the U-Net based network with different encoders such as VGG-16, ResNet-50, and Inception-ResNet-V2. After that, the character regions are recognized by the coarse and fine combined classifiers. The experiment shows that the proposed method makes the networks converge quickly and produce better recognition rates than single fine classifiers on the dataset of a large number of categories.

We also proposed the attention layer and added noises to the training dataset which helps the classifiers work well on the real testing dataset. The best-archived recognition rate is 85.07% for the real testing Nom dataset. We improve the recognition result by applying a beam search decoder, incorporated a Nom language model.

The approach should be effective for historical documents of Chinese origin. In China, nearly 50,000 categories were used in its history. In Japan, many variants were used for the same Kanji of Chinese origin so that the eventual number of categories are large. The approach would be effective also for other languages of historical documents, where the character set is large but the sample patterns are limited. Even for European historical documents, segmentation-based approach empowered by DNN has been proposed [8,28].

For future work, we plan to find the optimum number of coarse categories to improve the performance of the coarse and fine combined classifiers. For character region segmentation, we consider training the network which predicts the center of each character region as the form of Gaussian distribution to minimize the overlaps among character regions.

Acknowledgments

This work is supported by the Grant-in-Aid for Scientific Research(S)-18H05221 and (A)-18H03597. The authors would like to thank the National Library of Vietnam and the Vietnamese Nom Preservation Foundation for providing the Nom historical document pages. We also thank Prof. B. Indurkhya for discussion and refinement of our method.

Appendix A. U-Net encoder with different networks

Fig. A1

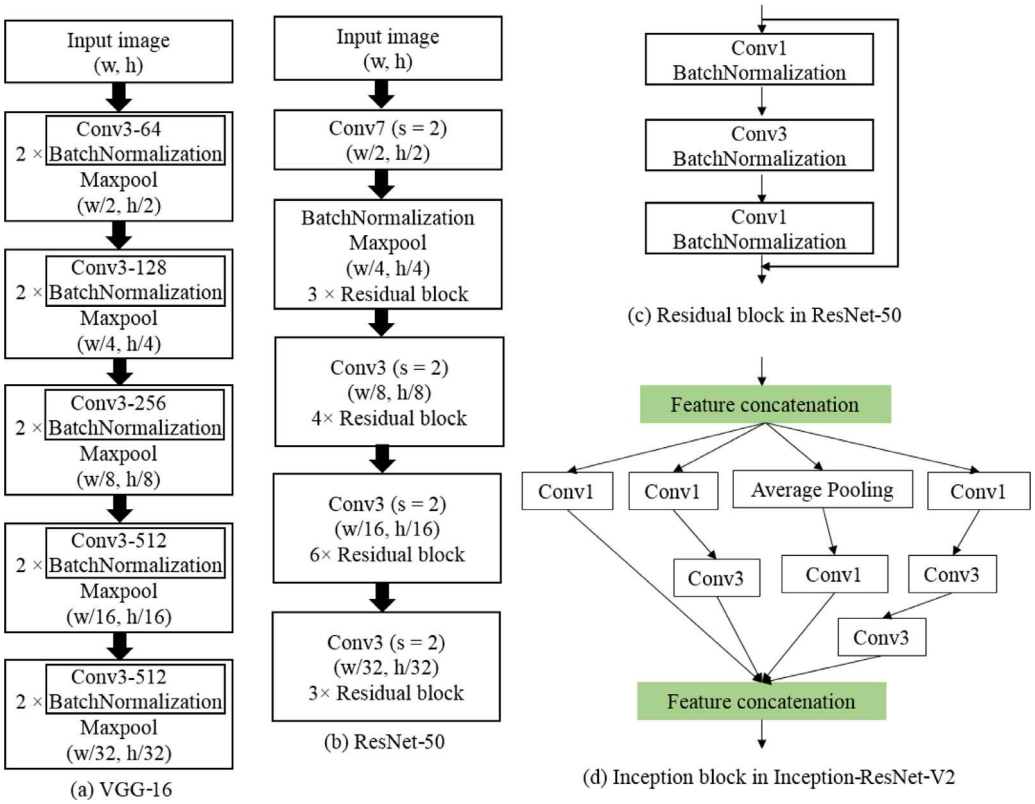


Fig. A1. U-Net encoders with different networks. Inception block is used in Inception-ResNet-V2 as shown in Section 4.2. The convolutional layer parameters are denoted as "Conv (kernel size)-(number of filters) (s = stride)".

Appendix B: Segmentation results on real Nom pages

Fig. B1

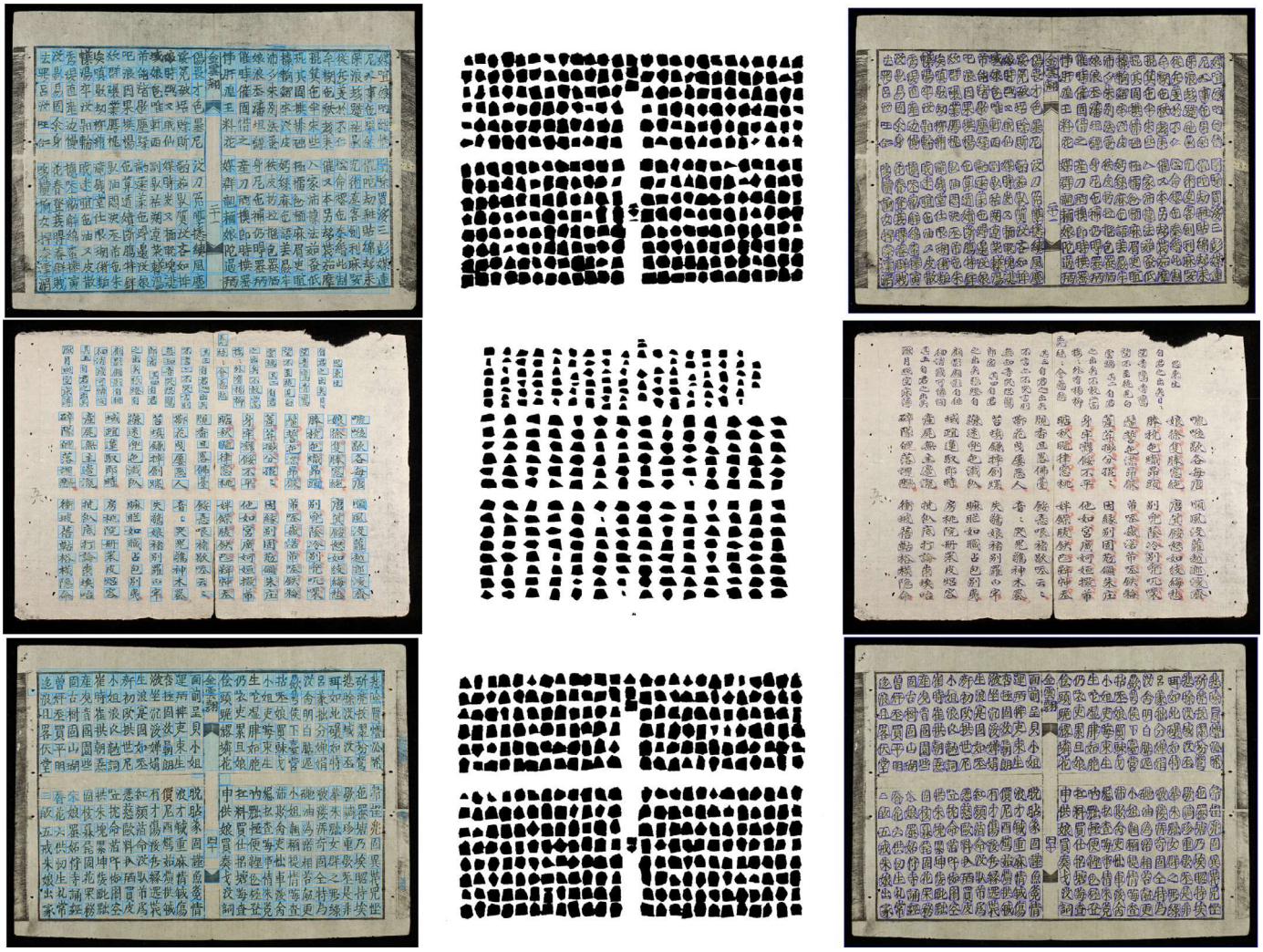


Fig. B1. Some segmentation results on real Nom pages. First column: segmentation by the combination of the projection profile and the Voronoi diagram. Second column: polygonal segmentation maps by the fine-tuned models. Third column: character regions after applying the marker-based watershed.

References

[1] V.J.Y. Shih, T.L. Chu, The han nom digital library, in: The International Nom Conference, Hanoi, The National Library of Vietnam, 2004, pp. 12–14.

[2] N.T. Nhan, C. Mai, An experience from temple university pilot digital project with the general library of thừa thiên huế, A Mini-Conference in Nôm Studies, The National Library of Vietnam, 2015.

[3] T.T. Khanh, T.T. Huyen, A program to translate the chinese taisho tripitaka into english and other western languages, Presentation at United Nations Vesak Day, 2008.

[4] D.A. Dao, Chu nom: origins, formation, and transformations, Nhà Xuất Bản Khoa Học Xã Hội (1979).

[5] T.V. Phan, K.C. Nguyen, M. Nakagawa, A nom historical document recognition system for digital archiving, Int. J. Doc. Anal. Recog. 19 (1) (2016) 49–64.

[6] O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015, pp. 234–241.

[7] M.S. Kim, M.D. Jang, H.I. Choi, T.H. Rhee, J.H. Kim, H.K. Kwag, Digitalizing scheme of handwritten Hanja historical documents, in: Proc. of the 1st International Workshop on Document Image Analysis for Libraries, USA, 2004, pp. 321–327.

[8] I. Gruber, M. Hlaváč, M. Hrúz, M. Železný, Semantic segmentation of historical documents via fully-convolutional neural network, in: International Conference on Speech and Computer, Springer, Cham, 2019, pp. 142–149.

[9] P. Lyu, M. Liao, C. Yao, W. Wu, X. Bai, Mask textspotter: an end-to-end trainable neural network for spotting text with arbitrary shapes, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 67–83.

[10] M. Jaderberg, A. Vedaldi, A. Zisserman, Deep features for text spotting, in: European Conference on Computer Vision, Cham, Springer, 2014, pp. 512–528.

[11] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, C. Sun, An end-to-end textspotter with explicit alignment and attention, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5020–5029.

[12] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, X. Bai, Aster: an attentional scene text recognizer with flexible rectification, IEEE Trans. Pattern Anal. Mach. Intell. (2018).

[13] M. Liao, B. Shi, X. Bai, X. Wang, W. Liu, Textboxes: a fast text detector with a single deep neural network, Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[14] D. Fernández-Mota, J. Lladós, A. Fornés, A graph-based approach for segmenting touching lines in historical handwritten documents, Int. J. Doc. Anal. Recog. 17 (3) (2014) 293–312.

[15] S. Zhao, Z. Chi, P. Shi, Q. Wang, Handwritten chinese character segmentation using a two-stage approach, in: Proceedings of Sixth International Conference on Document Analysis and Recognition, 2001, pp. 179–183.

[16] Q. Zheng, K. Chen, Y. Zhou, C. Gu, H. Guan, Text localization and recognition in complex scenes using local features, in: Asian Conference on Computer Vision, 2010, pp. 121–132.

[17] Y. Baek, B. Lee, D. Han, S. Yun, H. Lee, Character region awareness for text detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9365–9374.

[18] A. Cevahir, K. Murakami, Large-scale multi-class and hierarchical product categorization for an E-commerce giant, in: Proc. COLING, 2016, pp. 525–535.

[19] Z. Yan, H. Zhang, R. Piramuthu, V. Jagadeesh, D. DeCoste, W. Di, Y. Yu, HD-CNN:

hierarchical deep convolutional neural networks for large scale visual recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2740–2748.
[20] L. Jie, G. Zhenyu, W. Yang, Weakly supervised image classification with coarse and fine labels, in: The 14th Conference on Computer and Robot Vision (CRV), 2017, pp. 240–247.
[21] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, 31, 2009, pp. 855–868.
[22] A. Graves, (2012). Sequence transduction with recurrent neural networks. arXiv:1211.3711.
[23] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, International Conference on Learning Representations (ICLR), 2015.
[24] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[25] C. Szegedy, S. Ioffe, V. Vanhoucke, Alexander A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 4278–4284.
[26] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 60.
[27] D. Kingma and J. Ba. (2015). Adam: a method for stochastic optimization. In ICLR. arXiv:1412.6980
[28] S. Stewart, B. Barrett, Document image page segmentation and character recognition as semantic segmentation, in: Proceedings of the 4th International Workshop on Historical Document Imaging and Processing, ACM, 2017, pp. 101–106.