Page Segmentation for Historical Document Images Based on Superpixel Classification with Unsupervised Feature Learning

Kai Chen*, Cheng-Lin Liu†, Mathias Seuret*, Marcus Liwicki*, Jean Hennebert*‡, and Rolf Ingold*
*DIVA, University of Fribourg, Switzerland, Email: {firstname.lastname}@unifr.ch
†NLPR, Institute of Automation of Chinese Academy of Sciences, China, Email: liucl@nlpr.ia.ac.cn
‡University of Applied Sciences, HES-SO//FR, Fribourg, Switzerland, Email: jean.hennebert@hefr.ch

***Abstract*—In this paper, we present an efficient page segmentation method for historical document images. Many existing methods either rely on hand-crafted features or perform rather slow as they treat the problem as a pixel-level assignment problem. In order to create a feasible method for real applications, we propose to use superpixels as basic units of segmentation, and features are learned directly from pixels. An image is first oversegmented into superpixels with the simple linear iterative clustering (SLIC) algorithm. Then, each superpixel is represented by the features of its central pixel. The features are learned from pixel intensity values with stacked convolutional autoencoders in an unsupervised manner. A support vector machine (SVM) classifier is used to classify superpixels into four classes: *periphery*, *background*, *text block*, and *decoration*. Finally, the segmentation results are refined by a connected component based smoothing procedure. Experiments on three public datasets demonstrate that compared to our previous method, the proposed method is much faster and achieves comparable segmentation results. Additionally, much fewer pixels are used for classifier training.**

***Keywords*-page segmentation; layout analysis; historical document image; superpixel; SLIC; autoencoder**

I. Introduction

Page segmentation is considered as an important initial step for document image analysis and understanding. It aims at splitting a page image into regions of interest and distinguishing text blocks from the regions (Figure 1). Page segmentation on historical document images is challenging due to many variations such as layout structure, decoration, writing style, and degradation. Our goal is to develop a generic, flexible, and robust segmentation method to delimit text blocks, text lines, and eventually isolated words.

Some page segmentation methods have been proposed in the literature. These systems work on pixel level and rely on hand-crafted features [2], [3], [4], [14], prior knowledge [12], [15], [17], or models that combine hand-crafted features with domain knowledge [7], [11].

In this paper, we propose a novel page segmentation method based on superpixel classification with unsupervised feature learning. The goal is to produce a pixel-level segmentation of document images. In our previous methods [4], [6], we consider the page segmentation problem as a pixel labelling problem, i.e., each pixel is classified as either *periphery*, *background*, *text block*, or *decoration*. However, the pixel labeling methods are computationally expensive. In this work, rather than using pixels, superpixels, i.e., small regions obtained from an over segmentation are considered as the elementary units of our page segmentation task. Then features are learned directly from randomly selected image patches by using stacked convolutional autoencoders. With a support vector machine (SVM) trained with the features of the central pixels of the superpixels, an image is segmented into four regions as mentioned above. Finally, the segmentation results are refined by a connected components based smoothing procedure. The advantages of the proposed method are: (1) The page segmentation is efficient compared to the previous methods. It is on average 394 times faster to segment an image of 489×742 pixels, which makes the method feasible for real applications. (2) With much fewer pixels for classifier training, the proposed method achieves comparable segmentation results. (3) In the previous method [6], pixels are randomly selected for classifier training. These pixels may contain redundancy. In contrast, the proposed method selects only the central pixels of the superpixels as training samples. These pixels are informative and representative, which is demonstrated in

Figure 1: Page segmentation result. The left is the original image. The right is the segmentation result by using the proposed method. The colors: black, white, blue, and red are used to represent: *periphery*, *background*, *text block*, and *decoration* respectively.

978-1-5090-1792-8/16 $31.00 © 2016 IEEE
DOI 10.1109/DAS.2016.13

the experiments.

In summary, our method differs from traditional page segmentation methods in three aspects. (1) By applying superpixels algorithm, the run time of segmentation is reduced dramatically. (2) The features are learned directly from the pixels without supervision. (3) Preprocessing (e.g., binarization, connected components extraction) and prior knowledge of layout are not needed.

The rest of the paper is organized as follows. Section II gives an overview of some related work in page segmentation for historical document images. Section III describes the proposed method. Section IV reports the experimental results and Section V presents the conclusion.

II. Related Work

Some page segmentation methods for historical document images have been proposed. Most of them are based on image preprocessing such as binarization, connected components (CCs) extraction on pixel level, off-the-shelf classifiers trained on hand-crafted features, and prior knowledge.

Van Phan et al. [17] used the area Voronoi diagram to represent the neighborhood and boundary of CCs. By applying predefined rules, characters were extracted by grouping adjacent Voronoi regions. Finally, recursive x-y cut was applied to refine the segmentation. Bukhari et al. [2] used the normalized height, foreground area, relative distance, orientation, and neighbourhood information of the CCs as features to train a multilayer perceptron classifier. The trained classifier was used to classify CCs to the relevant class of text. Similarly, Cohen et al. [7] convolved images with different filters to extract local orientation of the pixels. The CCs constituted of pixels with horizontal orientation were considered as text lines. Based on prior knowledge, noise CCs were removed. Features such as bounding box size, area, stroke width, estimated text lines distance were used to label each CC into text or non-text by using an energy minimization method.

In our previous work [6], we used stacked convolutional autoencoders to learn features directly from pixels. The learned features were used to train an SVM. With this SVM, pixels were classified into four classes, i.e., *periphery*, *background*, *text block*, and *decoration*. Since the method was based on pixel-level classification, it was very slow (cf. Table I). On a cluster,[1] it took more than 20 minutes to segment an image with 550×850 pixels. Therefore, the pixel-level method is not applicable on standard environments.

III. System description

The proposed method consists of four steps. The first step relies on superpixel algorithms, i.e., we segment an image into superpixels. In the second step, features are learned with the unsupervised learning method as presented in [6]. In the third step, the learned features are used to train an SVM. With this SVM, the superpixels are classified into: *periphery*, *background*, *text block*, and *decoration*. Finally, we refine the segmentation results by eliminating the connected components (CCs) with predefined rules, where CCs are generated based on the predicted labels of the pixels. Figure 2 gives the whole workflow of our method.

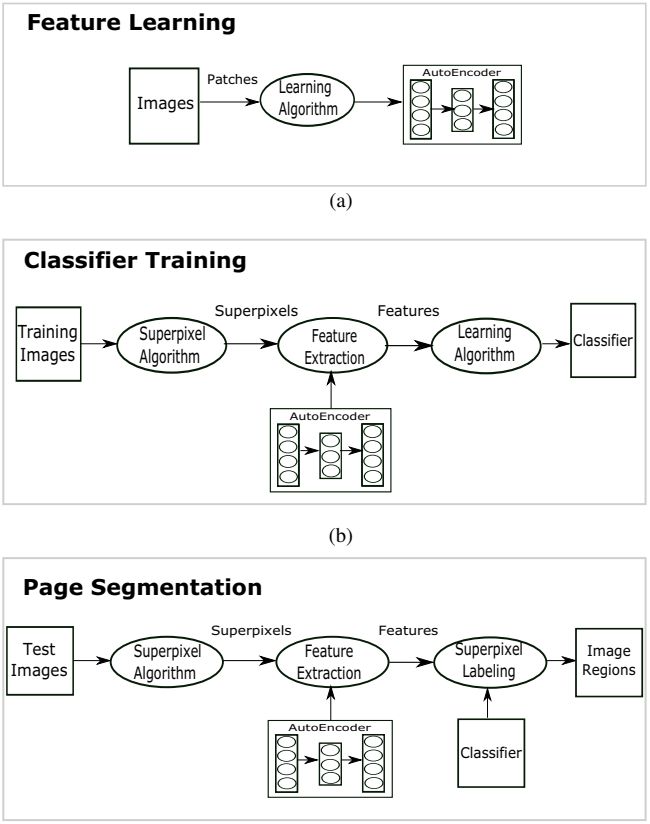


Figure 2: Page segmentation workflow.

A. Superpixel

Superpixel algorithm aims at grouping pixels into perceptually meaningful patches that belong to the same object. The motivation of embedding the superpixel algorithm into our system is to reduce the computational complexity without degrading the quality of the segmentation. For the purpose of increasing the speed, instead of predicting each pixel's class, the method predicts each superpixel's class.

Superpixels have been used in many computer vision applications [10], [16], [18]. They also have been applied for page segmentation for historical document images. Cohen et al. [7] grouped pixels into superpixels by using the Earth Mover's Distance metric as the superpixel distance in the CIE-Lab color distance. Mehri et al. [14] generated superpixels by using the simple linear iterative clustering (SLIC) [1] algorithm.

[1]Intel Xeon Processor 2.20 GHz with 8 cores.

In this work, we evaluate four state-of-the-art superpixel algorithms [19], [8], [9], [1]. The watershed (WS) algorithm [19] is a gradient ascent approach. In this algorithm, an image is viewed as a topological map where pixel intensity is represented by its gradient. By placing a waterdroplet in any of pixels, it will fall down towards a local minimum. The pixels which fall inside the same local minimum belong to the same superpixel. The mean shift (MS) algorithm [8] is a feature-space analysis technique for locating the maxima of a density function. It is used to find modes in the color or intensity feature space of an image. Pixels that converge to the same mode define the superpixels. In [9], Felzenszwalb and Huttenlocher proposed a graph-based approach. Pixels are considered as nodes on a graph. The minimum spanning trees of constituent pixels are considered as superpixels by applying clustering algorithm. The simple linear iterative clustering (SLIC) [1] is an adaptation of k-means clustering approach, where k is the desired number of approximately equally sized superpixels. Color and spatial proximity are used as distance measure. By limiting the search space to a region proportional to the superpixel's size, the number of distance calculations in the optimization is reduced.

Achanta et al. [1] showed that each of the superpixel algorithm has its own advantages and drawbacks that may be better suited to a particular application. For our application, we define the following criteria for the evaluation of superpixel algorithms: (1) The superpixel algorithm should increase the speed and improve (or not degrade) the quality of segmentation. (2) The generated superpixels should have regular shapes and sizes, since our future work is to create a graph based on the generated superpixels and use a probabilistic graphical model framework, e.g., conditional random field (CRF), to refine the segmentation results.

Regarding the two criteria, SLIC is applied as a preprocessing step of our page segmentation method. Compared to other superpixel algorithms, SLIC reaches a compromise between speed and segmentation quality. Furthermore, the generated superpixels' shapes are regular and approximately equally sized. Benefiting from this property, a graphical model can be applied to refine the segmentation results. The experimental results of the comparison of the four state-of-the-art superpixel algorithms are given in Section IV-B.

B. Feature learning

The architecture of the feature learning method is based on our previous work [6]. We use a single-layer fully-connected neural network as an autoencoder (AE) which learns to reconstruct its input data. Features can be discovered in the hidden layer. Concretely, the AE learns the weights W_1 and W_2, such that $f(W_2 f(W_1 x)) = \hat{x}$, where x is the input vector, the output $\hat{x}$ is similar to x and f is the activation function. We choose f to be the soft-sign function, such that $f(x) = \frac{x}{1+|x|}$. W_1 and W_2 are the weights on the first and second layer respectively. W_1 is used for encoding and W_2 is used for decoding. After using backpropagation to minimize squared reconstruction error, W_1 is used to compute the learned features, i.e., the mapping from input vector x to feature vector z where $z = f(W_1 x)$. In our method, the input vector x is the concatenation of each pixel's RGB values of a $w \times w$ pixels image patch extracted from a document image.

In order to learn high-dimensional feature representations from unlabeled pixels, our feature learning system stacks three levels of AEs. We denote $x^{(k)}$ as the input vector and $W^{(k)}$ as the weights of the AE on the k-th level. Our feature learning strategy on each level is described as follows:

First Level. We randomly select 10 millions 5×5 pixels image patches $P^{(1)}$ from the training set. We set the number of hidden units of the AE to 40.

Second Level. Document images have the property that one part of an image shares similarities with other part [13]. Thanks to this property, we can use the learned feature mapping function of the previous level and convolve them with larger image patch to learn high-order feature representations. A 15×15 pixels image patch $P^{(2)}$ is composed by 3×3 patches $P^{(1)}$ without overlapping. The input vector of each $P_n^{(1)}$ is denoted by $x_n^{(1)}$, where n is the patch number. The input vector $x^{(2)}$ is the concatenation of $f(W_1^{(1)} x_1^{(1)}), \cdots, f(W_1^{(1)} x_9^{(1)})$. The number of hidden units of the second-level AE is 30 and 10 millions randomly selected patches $P^{(2)}$ are used for training.

Third Level. We repeat the same procedure as for the second level. Therefore $P^{(3)}$ covers 45×45 pixels. The AE has 20 hidden neurons and is trained on 10 millions patches.

The settings have been tuned in our cross validation procedure to reach a tradeoff between accuracy and CPU load. For the details of stacked convolutional autoencoder, please refer to [6].

C. Classifier training and pixel labeling

Our objective is to label pixels into classes, i.e., *periphery*, *background*, *text block*, and *decoration*. To this end, an SVM is trained with the labels and the learned features (cf. Section III-B) of the central pixels of the superpixels on the training images. The features of a given pixel are the concatenation of the nth-level features $z^{(n)}$ from patches $P^{(n)}$ centered on that pixel where $z^{(n)} = f(W_1^{(n)} x^{(n)})$, $n \in \{1, 2, 3\}$.

To segment an image, we first generate superpixels on that image. Then each superpixel is represented by the feature vector of its central pixel. With the trained SVM, the superpixels are classified into four classes, where the central pixel's class is considered as the class of its corresponding superpixels.

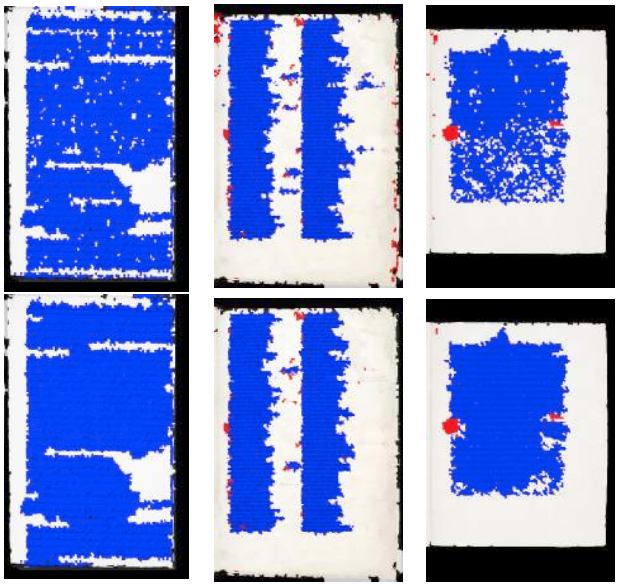
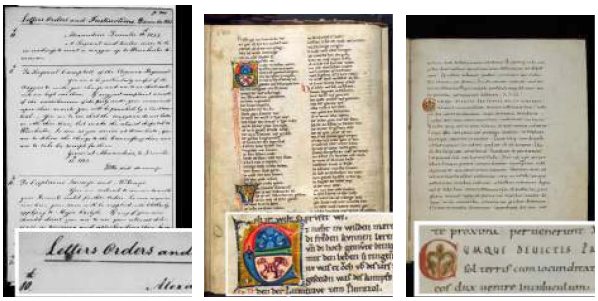


Figure 3: The first row shows the segmentation results without using the post-processing method. The second row shows the segmentation results by using the post-processing method. The colors: black, white, blue, and red are used to represent: *periphery*, *background*, *text block*, and *decoration* respectively.

D. Post-processing

By observing some page segmentation results with the proposed method on randomly selected images in the validation set, we find some reoccurring types of classification. Due to the degradation, in the blocks of the text, some superpixels are misclassified as background. And some superpixels on the background are misclassified as text or decoration. Our objective is to relabel these isolated superpixels. For a given image, based on the segmentation result obtained in Section III-C, we first generate connected components based on the *background* pixels. Then the pixels are relabeled to *text*, if their connected component's size is less than 1% of the size of the image. Similarly, the *non-background* pixels are relabeled to *background*, if their connected component's size is less than 1% of the size of the image. Figure 3 gives some results of the post-processing.



(a) (b) (c)

Figure 4: Example pages from the three dataset.

IV. Experiments

In order to compare the proposed method with our previous pixel-level method [6], we use the same datasets [5], scaling factors[2] α and follow the same evaluation protocol. The original image sizes are: 2200×3400, 2000×3008, and 1664×2496 pixels for the datasets: *G. Washington*, *Parzival*, and *Saint Gall* respectively. Four classes of layout elements are defined in the *Parzival* and *Saint Gall* datasets, i.e., *periphery*, *background*, *text block*, and *decoration*. And three classes are defined in the *G. Washington* dataset, i.e., *periphery*, *background*, and *text block*. To compare the page segmentation methods, three criteria are used for evaluation, i.e., (1) pixel classification accuracy [4], (2) segmentation speed, (3) training set's size.

The proposed method is implemented in Java. All of the experiments are performed on a PC with an Intel Core i7-3770 3.4 GHz processor and 16 GB RAM.

A. Comparison with pixel-level method

The drawback of the pixel-level method [6] is that it is very slow (cf. Table I). Furthermore, because we have to keep all data in a file for classifier training, this method is memory inefficient. In this work, we use SLIC [1] as a preprocessing procedure in order to increase the speed without degrading the quality of the segmentation. The number of the generated superpixels by SLIC is set to $k = 3000$ in order to achieve the compromise between the speed and quality of the segmentation results. Experiments are performed on the images of two resolutions with the scaling factors: $\alpha = 2^{-2}$ and $\alpha = 2^{-3}$. Figure 5 gives some segmentation results by using the pixel-level method and the proposed method. Table I reports on the performances of the page segmentation with the pixel-level method and the proposed superpixel-level method. We observe in comparison to the pixel-level method, the proposed method is more efficient. It is 341 times faster on the *G. Washington* dataset, 639 times faster on the *Parzival* dataset, and 202 times faster on the *Saint Gall* dataset with the scaling factor $\alpha = 2^{-2}$. The proposed method also achieves better segmentation results on the *G. Washington* dataset and comparable segmentation results on the *Parzival* and *Saint Gall* datasets. Furthermore, the proposed method is more memory efficient. In the pixel-level method, pixels are randomly selected on the images. In the proposed method, we choose all the central pixels of the superpixels as training samples. It is shown in the table that with much fewer training pixels, the proposed method achieves comparable segmentation results.

B. Comparison of superpixel algorithms

To evaluate the impact of different superpixel algorithms on our page segmentation method. We compare four state-of-the-art superpixel algorithms: FH [9], MS [8], WS [19],

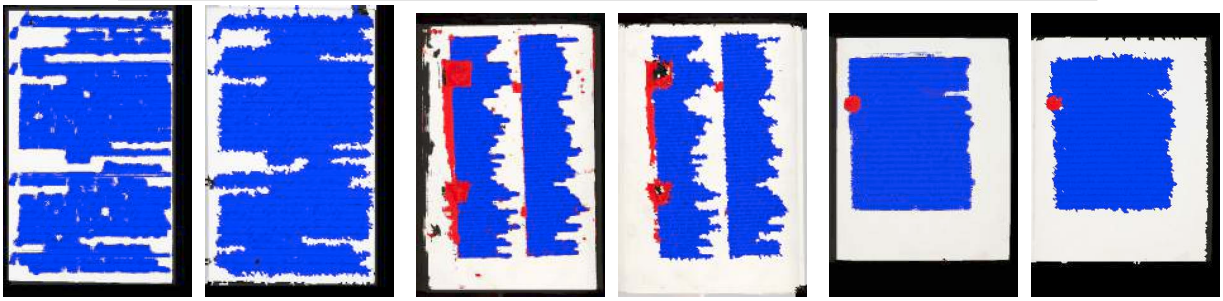
[2]Due to the large size of the images, we scale images to smaller size with a scaling factor $\alpha < 1.0$.

Table I: Summary of segmentation results by using pixel-level and superpixel-level methods. Experiments are performed on two image scaling factors $\alpha \in \{2^{-2}, 2^{-3}\}$. The criteria used for evaluation are: classification accuracy A (%) [4], run time per image T (min.), and the number of training pixels per image N. For the pixel-level method, the number of training pixels per class is denoted as n. The number of superpixels generated by SLIC [1] is predefined as $k = 3000$.

	Pixel-labeling method [6] ($n = 10$k)			Pixel-labeling method ($n = 50$k)			Pixel-labeling method ($n = 100$k)			The proposed method		
	A (%)	T (min.)	N	A (%)	T (min.)	N	A (%)	T (min.)	N	A (%)	T (min.)	N
$\alpha = 2^{-2}$												
G. Washington	81.3	477	30k	85	477	150k	85.8	477	300k	**86.9**	**1.4**	**3k**
Parzival	58.6	594	40k	85.8	594	200k	**87.3**	594	400k	87.2	**0.93**	**3k**
St.Gall	94.3	190	40k	96.4	190	200k	**96.9**	190	400k	95.5	**0.94**	**3k**
$\alpha = 2^{-3}$												
G. Washington	85	114	30k	86.1	114	150k	86.4	114	300k	**89.5**	**0.46**	**3k**
Parzival	86.5	101	40k	95.3	101	200k	**96.1**	101	400k	91.7	**0.91**	**3k**
St.Gall	95.2	41	40k	96.8	41	200k	**97.3**	41	400k	95.7	**0.58**	**3k**

Table II: Comparison of four state-of-the-art superpixel algorithms on page segmentation for historical document images. The criteria used for evaluation are: classification accuracy A (%) [4], run time per image T (min.), and the number of training pixels per image N. Experiments are performed on the three datasets without using the post-processing procedure. Images are scaled down with the factor $\alpha = 2^{-3}$. The number of superpixels generated by SLIC [1] is predefined as k.

	G. Washington			*Parzival*			*St. Gall*		
	A (%)	T (min.)	N	A (%)	T (min.)	N	A (%)	T (min.)	N
FH [9]	84.9	**0.01**	**400**	86	**0.04**	**400**	74.1	**0.01**	**150**
MS [8]	86.3	0.03	700	88.2	0.19	700	79.8	0.02	300
WS [19]	85.6	0.06	10k	89.4	0.14	1k	88.8	0.03	500
SLIC [1] ($k = 3000$)	87	0.34	3k	91.4	0.89	3k	94.9	0.56	3k
SLIC ($k = 10000$)	88.3	3.23	10k	93.9	5.48	10k	95.5	3.32	10k
SLIC ($k = 20000$)	**89.1**	9.82	20k	**94.5**	12.24	20k	**95.9**	18.19	20k



(a) (b) (c) (d) (e) (f)

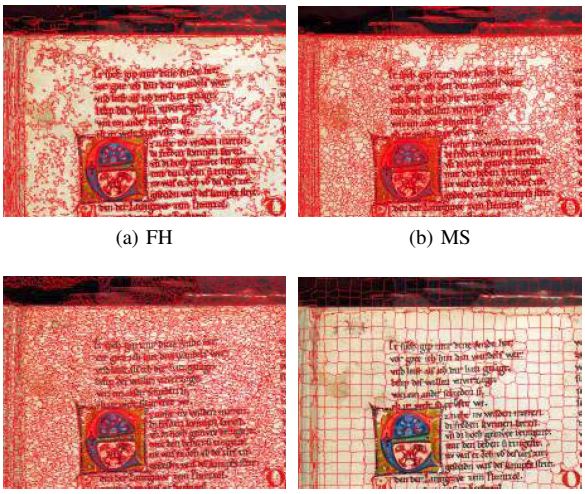
Figure 5: Segmentation results of pages in Figure 4a, 4b, and 4c. The pixel-level segmentation results ($\alpha = 2^{-3}, N = 100$k) are: 5a, 5c, and 5e respectively. The superpixel-level segmentation (with post-processing) results are: 5b, 5d, and 5f respectively. The colors: black, white, blue, and red are used to represent: *periphery*, *background*, *text block*, and *decoration* respectively.

and SLIC [1]. The open source Java library BoofVC[3] is used to generate superpixels with these algorithms. In these experiments, the post-processing method is not performed. Figure 6 shows the visual results of the produced superpixels with these algorithms. Table II reports on the classification accuracy [4], average run time per image, and number of generated superpixels per image of the four superpixel algorithms. It is shown that SLIC outperforms the other three algorithms in classification accuracy. Another advantage of SLIC is that it provides control over the number of generated superpixels. The other algorithms do not provide an explicit control over the number of superpixels. Therefore, with SLIC it is easier to tune the number of superpixels to control the tradeoff between run time and segmentation quality. Moreover, on the historical document images, SLIC produces superpixels with regular sizes and shapes. Benefiting from this property, a graphical model (e.g., CRF) can be applied on the superpixels in order to refine the segmentation results. From the experiments, we conclude that SLIC is more suitable for our page segmentation application on historical document images.

[3]http://boofcv.org/

V. Conclusion

In this paper we presented a novel page segmentation method for color historical document images. The method is based on superpixel classification with unsupervised feature learning. We show that by using SLIC [1] as a preprocessing step, on average, the proposed method is 394 times faster than our previous pixel-level method [6] on three datasets [5] where the scaled image sizes are: 550×850, 500×752, and 416×624 pixels. Moreover, on those scaled images, with almost 99 times fewer training samples, the proposed method



(a) FH (b) MS

(c) WS (d) SLIC

Figure 6: Superpixels produced by four algorithms.

achieves comparable segmentation results. We also compare four state-of-the-art superpixel algorithms as preprocessing step in our page segmentation method. Experiments show that with SLIC, the proposed method achieves superior segmentation results. In addition, the superpixels produced by SLIC on the historical document images have uniform sizes and regular shapes. Benefiting from this property, our future work includes using a graphical model framework (e.g., CRF) to model the dependencies of the superpixels in order to refine the segmentation results.

REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11), pp. 2274-2282, 2012.

[2] S. S. Bukhari, T. M. Breuel, A. Asi, and J. El-Sana, "Layout Analysis for Arabic Historical Document Images Using Machine Learning." in *International Conference on Frontiers in Handwriting Recognition*, pp. 639-644, 2012.

[3] K. Chen, H. Wei, M. Liwicki, J. Hennebert, and R. Ingold, "Robust Text Line Segmentation for Historical Manuscript Images using Color and Texture." in *International Conference on Pattern Recognition*, pp. 2978-2983, 2014.

[4] K. Chen, W. Hao, J. Hennebert, R. Ingold, and M. Liwicki, "Page Segmentation for Historical Handwritten Document Images Using Color and Texture Features." in *International Conference on Frontiers in Handwriting Recognition*, pp. 488-493, 2014.

[5] K. Chen, M. Seuret, W. Hao, M. Liwicki, J. Hennebert, and R. Ingold, "Ground Truth Model, Tool, and Dataset for Layout Analysis of Historical Documents." in *International Society for Optics and Photonics IS&T/SPIE Electronic Imaging*, 940204-10, 2015.

[6] K. Chen, M. Seuret, M. Liwicki, J. Hennebert, and R. Ingold, "Page Segmentation of Historical Document Images with Convolutional Autoencoders." in *International Conference on Document Analysis and Recognition*, pp. 1011-1015, 2015.

[7] R. Cohen, A. Asi, K. Kedem, J. El-Sana, and I. Dinstein, "Robust text and drawing segmentation algorithm for historical documents." in *Proceedings of the International Workshop on Historical Document Imaging and Processing*, pp. 110-117, 2013.

[8] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), pp. 603-619, 2002.

[9] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation." *International Journal of Computer Vision*, 59(2), pp. 167-181, 2004.

[10] D. Hoiem, A. A. Efros, and M. Hebert, "Automatic photo pop-up." *ACM Transactions on Graphics*, 24(3), pp. 577-584, 2005.

[11] J. Ji, L. Peng, and B. Li, "Graph Model Optimization Based Historical Chinese Character Segmentation Method." in *IAPR International Workshop on Document Analysis Systems*, pp. 282-286, 2014.

[12] S. Khedekar, V. Ramanaprasad, S. Setlur, and V. Govindaraju, "Text-Image Separation in Devanagari Documents." in *International Conference on Document Analysis and Recognition*, vol. 3, pp. 1265-1269, 2003.

[13] Y. LeCun, B. Léon, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition." *Proceedings of the IEEE*, 86(11), pp. 2278-2324, 1998.

[14] M. Mehri, N. Nayef, P. Héroux, P. Gomez-Krämer, and R. Mullot, "Learning Texture Features for Enhancement and Segmentation of Historical Document Images." in *Proceedings of the International Workshop on Historical Document Imaging and Processing*, pp. 47-54, 2015.

[15] C. Panichkriangkrai, L. Li, and K. Hachimura, "Character segmentation and retrieval for learning support system of Japanese historical books." in *Proceedings of the International Workshop on Historical Document Imaging and Processing*, pp. 118-122, 2013.

[16] X. Ren and J. Malik, "Learning a classification model for segmentation." in *IEEE International Conference on Computer Vision*, pp. 10-17, 2003.

[17] T. Van Phan, B. Zhu, and M. Nakagawa, "Development of Nom character segmentation for collecting patterns from historical document pages." in *Proceedings of the Workshop on Historical Document Imaging and Processing*, pp. 133-139, 2011.

[18] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework." in *European Conference on Computer Vision*, pp. 211-224, 2010.

[19] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 583-598, 1991.