Page Segmentation for Historical Handwritten Document Images Using Conditional Random Fields

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

Abstract—In this paper, we present a Conditional Random Field (CRF) model to deal with the problem of segmenting handwritten historical document images into different regions. We consider page segmentation as a pixel-labeling problem, i.e., each pixel is assigned to one of a set of labels. Features are learned from pixel intensity values with stacked convolutional autoencoders in an unsupervised manner. The features are used for the purpose of initial classification with a multilayer perceptron. Then a CRF model is introduced for modeling the local and contextual information jointly in order to improve the segmentation. For the purpose of decreasing the time complexity, we perform labeling at superpixel level. In the CRF model, graph nodes are represented by superpixels. The label of each pixel is determined by the label of the superpixel to which it belongs. Experiments on three public datasets demonstrate that, compared to previous methods, the proposed method achieves more accurate segmentation results and is much faster.

Keywords—Conditional Random Field; Page Segmentation; Historical Document Image; Autoencoder; Superpixel

I. INTRODUCTION

Page segmentation is an important initial step for document image analysis and understanding. The goal is to split a document image into regions of interest. Compared to segmentation of machine printed document images, page segmentation of historical document images is more challenging due to many variations such as layout structure, decoration, writing style, and degradation. Our goal is to develop a generic segmentation method to delimit text blocks on historical document images. We consider the segmentation problem as a pixel-labeling problem, i.e., for a given document image, each pixel is labeled as one of the predefined classes.

In our previous work, we proposed a hand-crafted feature based method [4] and a feature learning based method [7]. Both of them are based on local features, i.e., a classifier is trained on the features of the labeled pixels, then the classifier is used to label unseen pixels. Like other methods [11], [15], [19], one limitation of these methods is that the contextual information of pixels is not used for segmentation. In order to achieve better segmentation, contextual information is needed. By considering the labels of neighboring image patches helps to decide the label of a given patch. For example, an image patch contains text may have similar local features as an image patch contains noise which is outside of text blocks. The two patches are ambiguous, but they are identifiable inside their context, e.g., most of the neighbors of the text patch are text and most of the neighbors of the noise patch contain background information. Another drawback of the previous methods is that pixels are labeled independently, i.e., for a given pixel $x_i$, its label $y_i$ is predicted as $\hat{y}_i = \arg \max_{y_i \in \mathcal{Y}} P(y_i | x_i)$, where $\mathcal{L}$ is the predefined label set. In contrast, in the proposed method, for a given image $X$, instead of labeling each pixel, we predict the labels $Y$ of all the pixels jointly, such that $Y = \arg \max_{Y \in \mathcal{Y}} P(Y | X)$, where $\mathcal{Y}$ is defined as all possible labelings over the pixels. To this end, we propose to adapt Conditional Random Field (CRF) to the page segmentation problem.

The CRF model is originally proposed for segmenting and labeling 1-D text sequence [13] which has been applied for diagram recognition [18], handwritten document recognition [10], text/non-text stroke classification in online handwritten notes [9], and structure detection in nature images [12]. Unlike Markov Random Field (MRF) and other generative models which are based on joint probability $P(X, Y)$ over observation $X$ and label configurations $Y$, CRF directly models the conditional probability $P(Y | X)$ of label configurations given observation. Generative models require the independence assumption of the features of the observation. In contrast, CRF depends on arbitrary dependence features of the observation. Compared to other discriminative models, such as logistic regression and support vector machines which only model labeled pixels independently, CRF models not only dependencies between observation and label configurations, but also dependencies between labels.

Labeling all pixels in an image is inefficient and redundant. To reduce the time complexity, in this work, we perform labeling at superpixel level. In our CRF model, graph nodes are represented by superpixels. Features of each superpixel is represented by the features of its central pixel. Then the label of each pixel is determined by the label of the superpixel to which it belongs.

The rest of the paper is organized as follows. Section II gives an overview of some related work. Section III de-
scribes the proposed method. Section IV reports the experimental results and Section V presents the conclusion.

II. RELATED WORK

This section reviews some representative state-of-the-art methods for historical document image segmentation. Bukhari et al. [2] used the normalized height, foreground area, relative distance, orientation, and neighbourhood information of the Connected Components (CCs) as features to train a multilayer perception classifier. Then the classifier was used to classify CCs to the relevant class of text. Cohen et al. [8] convolved images with different filters to extract local orientation of the pixels. The CCs which 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.

Panchkriangkrai et al. [15] proposed a character segmentation and retrieval system of Japanese historical woodblock printed books. Text lines were separated by using vertical projection on binarized images. Rule-based integration was applied to merge or split the CCs to extract kanji characters.

We proposed a superpixel based page segmentation method in [7]. Superpixels were considered as basic units of segmentation. An image was first oversegmented into superpixels. Then, each superpixel was represented by the features of its central pixel. An SVM was used to classify superpixels into different classes. Like other methods [3], [4], [6], [11], [15], [19], one limitation is that the method labels each superpixel only based on its local information. The contextual information was not used for the labeling. In order to combine local and contextual information to improve segmentation, in this paper, we propose a CRF based method.

III. METHODOLOGY

The proposed method consists of two steps. In the first step, features are learned from pixels with an unsupervised learning method. In the second step, an image is represented as a graph, where each node is represented by a superpixel. Then the superpixels are labeled into different classes with a CRF model. Fig. 1 gives the whole workflow of the method.

A. Feature learning

The architecture of the feature learning method is based on the work as presented in [6]. To learn features, we use a neural network with one hidden layer as an autoencoder (AE). The AE 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 $\sigma(W_2 \sigma(W_1 x)) = \hat{x}$, where $x$ is the input vector, the output $\hat{x}$ is similar to $x$ and $\sigma(\cdot)$ is the activation function. We choose $\sigma(\cdot)$ to be the soft-sign function, such that $\sigma(x) = \frac{x}{1+|x|}$. $W_1$ 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 the reconstruction error, $W_1$ is used to compute the learned features, i.e., the mapping from input vector $x$ to feature vector $z$ where $z = \sigma(W_1 x)$. The input vector $x$ is the concatenation of each pixel’s RGB values of an image patch. A single layer AE is illustrated in Fig. 2. The first neural layer encodes the inputs, and the second neural layer, which is used only during the training phase, reconstructs the inputs from the encoded values.

Convolutional autoencoders (CAE) [6] are stacked autoencoders. Layers in CAE excepted for the top, are convolved. This allows to cover a larger area while keeping the number of weights of the neural network small enough to have an acceptable training time. In case of stacked autoencoders, the first layer of the CAE takes raw pixel data as input. The other layers take the output of the previous layer as input. 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. 10 millions $5 \times 5$ pixels image patches $P^{(1)}$ are randomly selected from the training set. The number of hidden neurons of the AE is set to 40.

![Fig. 1: Page segmentation workflow.](image)

![Fig. 2: Illustration of a single-layer autoencoder.](image)
Second Level. We use the learned feature mapping function of the first level and convolve them with larger image patch to learn high-order features. A $15 \times 15$ pixels image patch $P^{(2)}$ is composed with $3 \times 3$ patches $P^{(1)}$ without overlapping. The input of each $P^{(1)}$ is denoted by $x_n^{(1)}$, where $n$ is the patch number. The vector $x^{(2)}$ is the concatenation of $\sigma(W_1^{(1)}x_1^{(1)}), \cdots, \sigma(W_1^{(1)}x_n^{(1)})$. The number of hidden neurons 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. Such that $P^{(3)}$ covers $45 \times 45$ pixels. The AE has $20$ hidden neurons and is trained on $10$ millions patches.

The superiority of using the learned features compared to hand-crafted features for the page segmentation task is demonstrated in [6]. The implementation of the CAE is derived from our publicly available toolbox N-light-N [16].

B. Conditional random field

Conditional Random Field (CRF) is a discriminative graph model [13] which directly models the conditional distribution of observations over labels. For a given image, let $X$ be the observation over a set of nodes $S$, such that $X = \{x_i\}$, $i \in \{1,2,\ldots,n\}$. Where $x_i$ is the observation of node $s_i$, $s_i \in S$, and $|S| = n$. Let $Y$ be a set of random variables over the nodes, such that $Y = \{y_i\}$, $y_i \in \mathcal{L}$, where $y_i$ is the label of node $s_i$ and $\mathcal{L}$ is the label set. We define $\mathcal{L} = \{P,B,T,D\}$ for periphery, background, text block, and decoration respectively. Each node is connected to its eight nearest neighbors. The topology of the CRF model is given in Fig. 3. In the graph, each node is represented by a superpixel. The superpixels are generated by grouping pixels into perceptually meaningful patches that belong to the same object. In this work, we apply the simple linear iterative clustering (SLIC) algorithm [1] to oversegment images into superpixels. The superiority of SLIC compared to other superpixel methods for the page segmentation task is demonstrated in [7]. The features $x_i$ of each superpixel $s_i$ is represented by its central pixel’s features. The features are extracted by using the method as described in Section III-A.

Given an image $X$, we can score the label configuration $Y$ as:

$$
\psi(X,Y,\lambda) = \sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_i j_f(X,Y,i)
$$

(1)

where $f(\cdot)$ is a feature function which takes image observation $X$, label configuration $Y$, and node index $i$ as input and outputs a real-valued number. And $\lambda$ is a set of weights associated to the feature functions. The number of feature functions is denoted as $m$ and the number of nodes is denoted as $n$. We can transform these scores into probability $P(Y|X;\lambda)$ as:

$$
P(Y|X;\lambda) = \frac{1}{Z(X,\lambda)} \exp(\psi(X,Y,\lambda))
$$

(2)

where $Z(\cdot)$ is the normalization factor which is the sum over all the possible labelings $Y$, such that $Z(X,\lambda) = \sum_{Y \in \mathcal{Y}} \exp(\psi(X,Y,\lambda))$.

In our work, two kind of feature functions are defined: local feature function and contextual feature function. We use discriminative classifiers to model these feature functions. The Multilayer Perceptron (MLP) is used to create the feature functions due to its performance. However, it could be replaced by other classifiers.

Local feature function. For a given node, this function measures the compatibility between a given label and its local features. It takes into account features extracted on the node. The features are extracted by using the autoencoders as described in Section III-A. We use the autoencoder features and ground-truth label of the superpixels on the training images to train an MLP. For a given node $s_i$, the MLP outputs a vector of scores, $c_{ij}, j \in \{1,2,\ldots,|\mathcal{L}|\}$, where $\mathcal{L}$ is the label set. Each $c_{ij}$ presents the probability of label $l_j$ associated to the node $s_i$. The higher value means that label is more likely to be attached to the node. The weights $\lambda_i$ of the local function $f_{i}(\cdot)$ is a vector of size $|\mathcal{L}|$.

Contextual feature function. For a given node, the function describes the compatibility of its label and the labels of its neighbors. The neighbors is determined by measuring the euclidean distance of the coordinates. For a given node $s_i$, we take the $N$ nearest nodes as its neighbors, where $N = 8$. The number of neighbors $N$ is empirically chosen on our validation set to reach a trade off between accuracy and CPU load. We use an MLP to model the contextual function $f_{c}(\cdot)$. To train the MLP, the feature vector of each node is the output of the local feature function of the nine nodes, i.e., the node with its eight neighbors. The MLP outputs a vector of scores, $d_{ij}, j \in \{1,2,\ldots,|\mathcal{L}|\}$. Each $d_{ij}$ presents the probability of label $l_j$ assigned to the node $s_i$, given the label configurations of its neighbors. The weight $\lambda_i$ of the contextual function $f_{c}(\cdot)$ is a vector of size $|\mathcal{L}|$.

Inference. The inference method is similar to the methods described in [9], [14]. For a given image $X$, in order to
find the optimal label configuration without computing the normalization factor $Z(\cdot)$ for the whole document image, we apply the Iterated Conditional Mode (ICM) [17] to find an approximate solution of:

$$\hat{Y} = \arg\max_{Y \in \mathcal{Y}} P(Y|X; \lambda)$$

(3)

where $\mathcal{Y}$ is defined as all possible labelings over the nodes.

The idea of ICM algorithm is to sequentially maximize the local conditional probability of each node. ICM algorithm assumes that $P(Y|X; \lambda)$ can be approximately maximized by maximizing local conditional probability $P(y_i| y_{N_i}, x; \lambda)$ at each node $s_i$, such that,

$$P(y_i| y_{N_i}, x; \lambda) = \frac{\exp(\psi(y_i, x; \lambda))}{\sum_{y_i'} \exp(\psi(y_i', x; \lambda))}$$

(4)

where $\psi(\cdot)$ is defined in Eq. 1. According to this assumption, we first label each node by using the local feature function, i.e., for each node, the label gives the highest output is attached to that node. Then we update the labels of all the nodes iteratively. At iteration $t$, for each node $s_i$, we select the label $l$ which maximizes the local probability $P(y_i| y_{N_i}, x; \lambda)$. To compute the local probability, at the first iteration, we take the output of the local feature function as the input of the contextual feature function. Then we combine the outputs of the two feature functions to get the local probability $P(y_i| y_{N_i}, x; \lambda)$. For the next iterations, i.e., $t > 2$, the input of the contextual feature function is $P(y_i^{(t-1)}| y_{N_i}^{(t-2)}, x; \lambda)$ and the input of the local feature function does not change. This updating process is repeated until convergence of the label configuration. Fig. 4 depicts the workflow of the inference method.

**Parameter estimation.** We estimate the parameter $\lambda$ by maximizing the pseudo-likelihood (PL) of the given image $X$. The PL estimation of parameter $\lambda$ is given as:

$$\hat{\lambda} = \arg\max_{\lambda} \prod_{i=1}^n P(y_i| y_{N_i}, x; \lambda)$$

(5)

where $P(y_i| y_{N_i}, x; \lambda)$ is given in Eq. 4. In the parameter estimation phase, computing the denominator in Eq. 4 involves summing over the labels of node $s_i$, while its neighbors labels $y_{N_i}$ are fixed with the ground-truth labels.

We apply the stochastic gradient descent algorithm to estimate $\lambda$ which maximizes the PL. From Eq. 4 and Eq. 5, the cost function of log PL is given as:

$$J(\lambda) = \sum_{i=1}^n (\psi(y_i, x; \lambda) - \log \sum_{y_i'} \exp(\psi(y_i', x; \lambda)))$$

(6)

where in the first term, $y_i$ is assigned with the ground truth label. Taking the derivatives with respect to $\lambda$ we get:

$$\frac{\partial J(\lambda)}{\partial \lambda} = \sum_{i=1}^n \left( \frac{\partial \psi(\cdot)}{\partial \lambda} - \sum_{y_i'} (P(y_i'| y_{N_i}, x; \lambda) \frac{\partial \psi(\cdot)}{\partial \lambda}) \right)$$

(7)

where for each weight $\lambda_j$ of feature function $f_j(\cdot)$,

$$\frac{\partial \psi(\cdot)}{\partial \lambda_j} = \frac{\partial \psi(y_i, x; \lambda)}{\partial \lambda_j} = f_j(y_i, x_i)$$

(8)

In each iteration, $\lambda_j$ is updated as:

$$\lambda_j = \lambda_j - \eta \sum_{i=1}^n (f_j(y_i, x_i) - \sum_{y_i'} P(y_i'| y_{N_i}, x; \lambda) f_j(y_i', x_i))$$

(9)

where $\eta$ is the learning rate. Note in Eq. 9, the first term in the gradient is the output of the feature function under the true label, and the second term is the expected output of the feature function under the current model.

IV. EXPERIMENTS

To compare the proposed method with the previous methods [6], [7], we use the same datasets [5] and follow the same evaluation protocol. Due to the large size of the images, the images are scaled down with a factor $\alpha = 2^{-2}$. Four classes of layout elements are defined in the Parzival and St. Gall datasets, i.e., periphery, background, text block, and decoration. Three classes are defined in the G. Washington dataset, i.e., periphery, background, and text block. Table I gives the details of the training set TR, test set TE and validation set VA. To compare the page segmentation methods, two criteria are used for evaluation, i.e., pixel-level classification accuracy as defined in [4] and runtime. The proposed method is implemented in Java. All the experiments are performed on a PC with an Intel Core i7-3770 3.4 GHz processor and 16 GB RAM.

In the experiments, we compare the proposed method with the previous methods [6], [7]. In [6], a Support Vector Machine (SVM) is trained on the features of pixels. Then pixels are labeled into different classes with the SVM. In [7], instead of labeling pixels directly, images are first oversegmented into superpixels. Then an SVM is used to classify superpixels into different classes. We also evaluate the performance of using only a local MLP as described in Section III-B. A post processing which takes into account

| Table I: Details of training, test, and validation sets. |
|-----------------|------------------|---------|-------|
| Dataset     | Image size (pixels) | TR        | TE        | VA        |
| G. Washington | 2200 × 3400     | 10       | 5        | 4         |
| St. Gall     | 1664 × 2496     | 20       | 30       | 10        |
| Parzival     | 2000 × 3008     | 24       | 13       | 2         |
Table II: Summary of segmentation results by using pixel-labeling method [6], superpixel-labeling method [7], only using local feature function as described in Section III-B, using local feature function followed by a post-processing method, and the proposed CRF model. Experiments are performed on the images with a scaling factor $\alpha = 2^{-2}$. The criteria used for evaluation are: pixel-level classification accuracy $A(\%)$ as defined in [4] and runtime per image $T$ (sec). The number of superpixels generated by SLIC [1] is defined as $n = 3000$.

<table>
<thead>
<tr>
<th>Method</th>
<th>$G. \text{Washington}$</th>
<th>$Parzival$</th>
<th>$St. \text{Gall}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixel labeling (with SVM) [6]</td>
<td>85.8 28620</td>
<td>87.3 35640</td>
<td>96.9 11400</td>
</tr>
<tr>
<td>Superpixel labeling (with SVM) [7]</td>
<td>86.9 84</td>
<td>87.2 56</td>
<td>95.5 56</td>
</tr>
<tr>
<td>Local MLP</td>
<td>83.3 2</td>
<td>86.4 2</td>
<td>95.8 2</td>
</tr>
<tr>
<td>Local MLP with post-processing</td>
<td>83.3 3</td>
<td>86.6 2</td>
<td>96.4 2</td>
</tr>
<tr>
<td>The proposed CRF model</td>
<td><strong>88.7 47</strong></td>
<td><strong>90.6 45</strong></td>
<td><strong>97.1 48</strong></td>
</tr>
</tbody>
</table>

The proposed CRF model yields more accurate and cleaner results than other methods. (1st column) Input images. (2nd to 7th column) Segmentation results using the pixel-labeling method [6], the superpixel-labeling method [7], the local MLP, the local MLP followed by a post-processing method, the proposed CRF model, and the ground-truth images respectively. The colors: black, white, blue, and red are used to represent: periphery, background, text block, and decoration respectively.

Fig. 5: Segmentation results on test images. (1st row) $G. \text{Washington}$ dataset, (2nd row) $Parzival$ dataset, (3rd row) $St. \text{Gall}$ dataset. The proposed CRF model yields more accurate and cleaner results than other methods. (1st column) Input images. (2nd to 7th column) Segmentation results using the pixel-labeling method [6], the superpixel-labeling method [7], the local MLP, the local MLP followed by a post-processing method, the proposed CRF model, and the ground-truth images respectively. The colors: black, white, blue, and red are used to represent: periphery, background, text block, and decoration respectively.

the neighbors’ contextual information for segmentation is also defined, in order to compare with the proposed CRF based method. In the post-processing, for each node $s_i$, if $C_i^{l_{\text{max}}}/N_{\text{post}} \geq \delta$, then $y_i = l_{\text{max}}$, where $l_{\text{max}} \in \mathcal{L}$, and $C_i^{l_{\text{max}}}$ is the maximal number of label $l_{\text{max}}$ which appears in its $N_{\text{post}}$ nearest neighbors. The number $N_{\text{post}}$ and the threshold $\delta$ are empirically chosen on the validation set, where $N_{\text{post}} = 16$ and $\delta = 0.9$. In the proposed method, features are learned from pixel intensity values with stacked convolutional autoencoders as described in Section III-A. The features are used for the purpose of initial classification with an MLP.

Then a CRF model as described in Section III-B is applied for modeling the local and contextual information jointly to improve the segmentation results.

Table II reports the pixel-level classification accuracy and runtime. Fig. 5 gives some segmentation results. It is shown that the proposed CRF model achieves more accurate segmentation results than other methods. It is clear that the contextual information is useful for page segmentation. Incorporation of the contextual information helps to resolve local ambiguities and achieve locally and globally consistent segmentation. The segmentation results produced by the
The proposed CRF model is cleaner. Moreover, the proposed method is much faster than previous methods.

V. CONCLUSION

In this paper, we present a CRF model to segment images of handwritten historical documents into regions of interest. We formulate the page segmentation as a pixel-labeling problem. Local features are learned with stacked convolutional autoencoders in an unsupervised manner for the purpose of initial labeling. Then a CRF model is applied for modeling the local and contextual information jointly to improve the segmentation results. We evaluate the proposed method on three handwritten historical document image datasets. Experiments show that compared to previous methods, the proposed method not only achieves superior segmentation result but also reduces the runtime. Our future work will focus on integrating other contextual information into the CRF model to improve the segmentation.

ACKNOWLEDGMENT

This work is supported by the Swiss National Science Foundation project HisDoc 2.0 with the grant number: 205120 150173.

REFERENCES


