Learning Text-line Segmentation using Codebooks and Graph Partitioning

Le Kang, Jayant Kumar, Peng Ye, David Doermann
Institute for Advanced Computer Studies
University of Maryland, College Park, MD, USA
{*lekang, jayant, pengye, doermann*}*@umiacs.umd.edu*

***Abstract*—In this paper, we present a codebook based method for handwritten text-line segmentation which uses image-patches in the training data to learn a graph-based similarity for clustering. We first construct a codebook of image-patches using K-medoids, and obtain exemplars which encode local evidence. We then obtain the corresponding codewords for all patches extracted from a given image and construct a similarity graph using the learned evidence and partitioned to obtain text-lines. Our learning based approach performs well on a field dataset containing degraded and un-constrained handwritten Arabic document images. Results on ICDAR 2009 segmentation contest dataset show that the method is competitive with previous approaches.**

***Keywords*-text-line; segmentation; learning; codebook;**

I. Introduction

Text-line extraction is an important step for many document processing tasks such as word/character recognition [1], layout-analysis [2] and skew estimation [3]. Unlike printed documents, the lines in handwritten documents are often non-uniformly skewed and curved. Moreover, overlapping spatial envelopes of text-lines, touching of characters across lines, and irregularity of layout and character shapes originated from the variability of writing styles make the problem more challenging. Recent work has focused on addressing each of these issues individually but a unified framework to take into account all the challenges associated with handwriting is still desired. For example, methods based on level-sets [4] are effective but computationally slow, methods based on connected-components (CCs) are fast but are challenged by touching components and overlapping lines [5]. Similarly, projection-based methods [6] cannot handle overlapping lines or touching, and perform poorly when there is large variation in character or word dimensions. In [7], the authors report that the method which gave almost 100 percent accuracy on ICDAR 2009 competition data set [8], performed poorly on field data of degraded handwritten Arabic documents, and an ensemble of multiple methods was required to obtain a reliable segmentation/recognition accuracy.

Although the last few years have seen tremendous growth and success of learning based methods for object recognition and image segmentation [9], existing methods for text-line segmentation tend to use un-supervised approaches. Because training data are difficult to obtain, most systems use only a small validation set for tuning parameters. Earlier methods were primarily targeted at printed and a limited class of handwritten documents where encoding heuristics in an un-supervised setting produced acceptable results. But for unconstrained handwritten documents, it has been difficult to encode all of the knowledge, and the performance using unsupervised methods has not been satisfactory for many data sets. When the images are degraded, many existing methods fail even for printed and less complex handwritten documents. Although many tools for efficient groundtruthing are available [10], and it is less expensive to obtain labels for text-line data, most existing approaches have hand-coded the knowledge obtained from "inspecting" the training data.

In this work, we develop a graph-based method for text-line segmentation which uses image-patches in the training data to obtain the contextual evidence needed for detecting text-lines in a new document images. Training images are transformed into summary maps, and context patches with local groundtruth masks are randomly sampled from the summary maps. Using K-medoids clustering, representative patches are selected to construct a codebook. A new document image is also transformed into a summary map and each non-zero bin is considered as a node in the graph. Context patches are sampled at each node. For each context patch the best match is found from the codebook, and the associated groundtruth mask is updated with the similarities between this node and others. After the similarity graph is constructed, multiclass normalized cuts is employed to partition the graph. A novel Sequential Gap Significance feature combined with a support vector machine predicts the number of clusters. The partitioning is performed recursively until no further split. Decisions on the graph are mapped back to the original image to provide the text-line segmentation. A postprocessing is employed to resolve fragmentation errors.

Our contributions are learning a graph based similarity among large variations of text-lines patterns, and solving the graph partitioning with accurate prediction of the graph structure.

The remainder of this paper is organized as follows. Section 2 overviews the related work on unsupervised and supervised text-line extraction. In Section 3 we present the proposed text-line segmentation approach. We give the details of a pixel-based evaluation mechanism and discuss our experimental results in Section 4 and conclude our paper

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.228

in Section 5.

II. Related work

Existing methods in an unsupervised setting for text-line segmentation can be broadly categorized into three classes: top-down projection based methods [6], bottom-up component grouping based methods [5], [11], and the hybrid methods [7]. While top-down methods partition the document image recursively in order to obtain text-lines, bottom-up methods group small units of the document image (pixels, CCs, characters, words) into text-lines. Bottom-up grouping can be implemented through clustering, which aggregates image components according to similarity and does not rely on the assumption of straight lines. Kumar et al. [11] proposed a local orientation detection based similarity for clustering primary CCs using Affinity propagation. In a post-processing step errors in the text-line are corrected iteratively using Expectation-Maximization (EM) [5]. Their method achieves high accuracy on a set of Arabic documents but since the method is based on grouping CCs, it is less likely to work on degraded document images where CCs are broken. Another method which achieves high accuracy on many data sets is based on steerable directional filter [12]. It finds the local orientation of a text-line by scanning in multiple directions for maximum response of a convolution of the filter with the image. In the final step, touching components are split at the contour level and the character images are reconstructed. This method too fails to group components when the image is degraded and spacing between character/words in text-lines has high variation.

Some recent work has incorporated supervised learning at different levels. Yin and Liu [13] learn a distance metric for pairs of CCs using a labeled dataset. The CCs are then grouped into a tree structure where text-lines are extracted by dynamically cutting the edges using a hyper-volume reduction criterion. By learning the distance metric, this algorithm is made robust to handle multi-skewed and curved text-lines. The method works only if spatial envelops of text-lines are not overlapping which limits the application to unconstrained handwritten lines of different scripts. Manohar et al. [7] proposed an ensemble system as a formulation of graph-clustering problem and applied it to combine outputs of multiple text-line segmentation methods. They construct a co-occurrence graph with nodes corresponding to CCs and edges connecting pairs of CCs with an associate cost of having the pair same label (line). The edge cost is determined by the cost of a false split and merge, and the likelihood that a pair of CCs has the same label conditioned on the ensemble output. These likelihoods are learned during training. Text-line segmentation is then formulated as the problem of minimum cost partitioning of the nodes in the graph. As expected, the method performs better than individual methods but the scaling and time performance of method is always lower-bounded by performance of individual methods.

III. Proposed approach

The proposed method learns the local spatial relationship between text-lines and applies an effective strategy to predict the graph structure for partitioning.

Visual codebooks constructed from invariant descriptors extracted from local image patches have been widely used in texture analysis and visual recognition [14]. In [15], an effective method for image quality assessment used raw image patches for codebook construction and achieved state-of-the-art performance. Inspired by these methods, we use image patches to encode the local evidence of text-lines.

A. Context patch extraction

An $M \times N$ binary document image is divided into square cells of size $p \times p$. A summary map of $(M/p) \times (N/p)$ bins is constructed, where each bin records the number of foreground pixels in the corresponding cell of the document image. Thus the summary map resembles a downsampled version of the original image. In the summary map, we define the context patch of a bin as an $H \times W$ region centered at this bin, where H and W are odd numbers so that there is an exact geometric center. In practice, each context patch is constructed as a vector by concatenating its columns, and is normalized to unit norm. Figure 1(a) and 1(b) show part of a document image and its summary map with a sampled context patch.

B. Codebook construction

Each training image is transformed into a summary map, and context patches are randomly sampled at non-zero bins. The ground truth of each training image is at pixel level, i.e. each foreground pixel has a label indicating which text-line it belongs to. Thus we can obtain the bin level ground-truth by a majority voting among the pixels a bin corresponds to. We say one bin is a "fellow" of another if the two bins have the same label, i.e. they are in the same text-line. Each context patch is accompanied by a mask of the same size, called the fellow mask, which records fellows of the center bin in this context patch. Considering the storage of the codebook and the efficiency of comparison between a query patch and all codewords, it is desirable to use fewer patches to represent the space of training data. We simply use K-medoids to select representative patches. Figure 1(c) shows a codebook containing context patches and fellow masks. The best matching codeword for a query patch is obtained by finding the nearest neighbor in Euclidean space, which is equivalent to finding the maximum inner product, given that the points sit on the unit sphere. It is also possible to store all the sample patches and apply an efficient (approximate) nearest neighbor search.

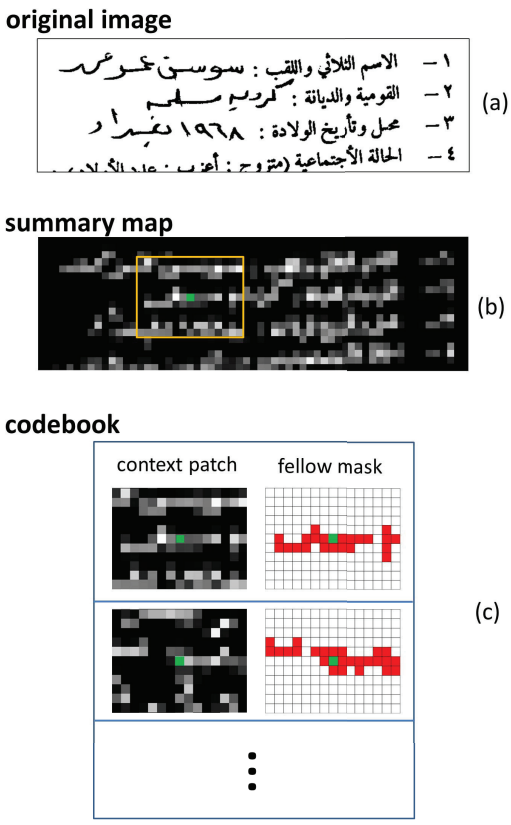

Figure 1. (a) Part of a sample image (b) Summary map and a context patch (in yellow box) centered at the bin marked green. (c) Codebook consisting of context patches and fellow masks. Fellows are marked in red.

C. Estimating the number of clusters and graph partitioning

We employ multiclass normalized cuts [16] to partition a similarity graph. Multiclass normalized cuts is designed to provide a near optimal solution for partitioning a graph where only local similarities are obtained for data points, which fits our situation well.

However, it is necessary to determine the number of clusters, which remains a difficult problem. In [17], an eigen gap heuristic is suggested - a relatively large eigen gap usually indicates the number of clusters if the dataset contains very well pronounced clusters. We tried this eigenvalue heuristic by choosing the number k if the kth eigen gap is the largest one. But the performance of this strategy is far from satisfactory, with an accuracy below 20% on a semi-synthetic dataset explained in Section 4. We observed that the real kth gap was usually significant but there were often larger gaps at somewhere larger than the k. We conjectured that the number should be determined by both the sequential order and the significance of the eigengaps. We designed a feature called the Sequential Gap Significance (SGS). Assume we have m eigen values $\lambda_1, \lambda_2, \cdots, \lambda_m$ in ascending order and their eigengaps g_i computed as follows

$$g_i = |\lambda_{i+1} - \lambda_i|, i = 1, 2, \ldots, m-1 \tag{1}$$

The SGS feature is expressed as follows:

$$SGS(i) = \begin{cases} \dfrac{g_i}{T}, & i = 1, \\ \dfrac{g_i}{max_{j=1,,i-1} g_j}, & i = 2, \ldots, m-1. \end{cases} \tag{2}$$

where T is a small positive offset. The intuition of SGS is that it measures how "significant so far" each eigengap is. Figure 2 compares eigen values, eigen gaps and SGS feature in characterizing the cluster structure. If we let k' be the number of clusters such that

$$k' = \arg \max_{i=1,\ldots,m-1} SGS(i) \tag{3}$$

then a reasonable estimation accuracy around 90% was obtained on the same semi-synthetic dataset as we mentioned earlier. This is still not satisfactory. Since training data is available, we solve this problem in a supervised learning fashion. We trained a multiclass SVM classifier that took the SGS features as input feature vectors and predicted the number of clusters. The accuracy achieved in this way was about 97%.

One issue that remains is that the method described above can only estimate the cluster number between 1 and $m-1$. In practice there may not be a guarantee that the real cluster number k is less than m. To address this issue we iteratively partition the subgraphs until no further split is predicted.

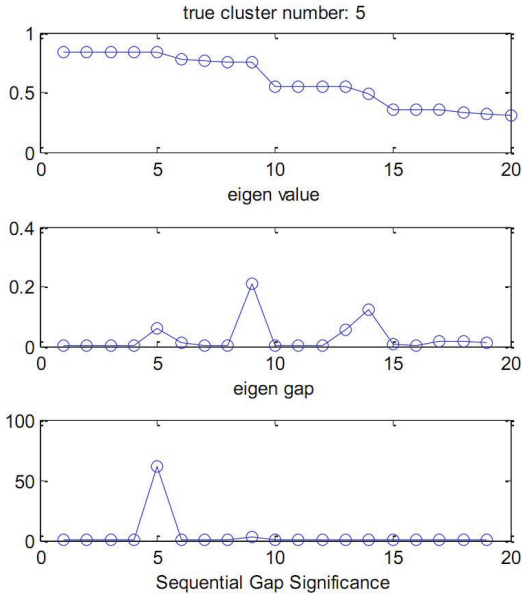

Figure 2. Comparison of eigen values (negated and shifted, least becomes greatest), eigen gaps and SGS feature in characterizing the cluster structure.

D. Text-line segmentation and postprocessing

After graph partitioning, each bin gets a label indicating its cluster membership, and the mainframes of text-lines are clear. Then each bin's label is mapped back to the foreground pixels on the original image, and a coarse text-line segmentation is obtained.

We observed that characters touching across text-lines are reasonably segmented while some non-touching strokes close to neighboring text-lines were incorrectly segmented into two or more pieces. This fragmentation is understandable since the proposed approach is based on bins rather than connected components. We developed a defragmentation technique to repair the errors. For each fragment in a connected component, we find the support text that has the same label and locates in the neighborhood of context patch size. Robust multilinear regression [18] is used to fit a line to the support text, treating each pixel as a point. Relative to this line, we compute the residual of the mass center of the fragment, the average residual of the support text and the standard deviation, denoted as R_c, R_{avg} and R_{std} respectively. We define the fragment quality Q as follows:

$$Q = \frac{R_c - R_{avg}}{R_{std}} \tag{4}$$

If $Q < 1$, the fragment is considered a major fragment, and its label remains unchanged. Otherwise it is a minor fragment and ready to be merged into the closest major fragment, or the fragment with maximum Q if no major fragment exists in this connected component. With this simple postprocessing, a lot of fragmentation errors are resolved while most touching segmentations are kept.

IV. Experiments

Two datasets were used in our experments, an Arabic field dataset and the ICDAR2009 handwriting segmentation contest dataset [8]. The traditional evaluation criterion was employed as in [8]. For a result region R and a ground truth Region G, a matchscore is computed as

$$matchscore(R, G) = \frac{\|R \cap G\|}{\|R \cup G\|} \tag{5}$$

where $\|\cdot\|$ denotes the cardinality of a set. The result region R is accepted if its matchscore is above the threshold T.

The Arabic field dataset contains 487 images with a total of 13904 text-lines. Images in this dataset present complex layouts and different levels of noise and degradation, which are similar to the data used in [7]. Figure 3 shows an example from the field dataset. Text-line segmentation on this kind of data has been a very challenging task [7]. The whole dataset was randomly divided into three parts: 150 for training, 100 for validation and 237 for test. The codebook was constructed on the training set, and the SVM parameters are optimized on the validation set. Due to the noisy nature

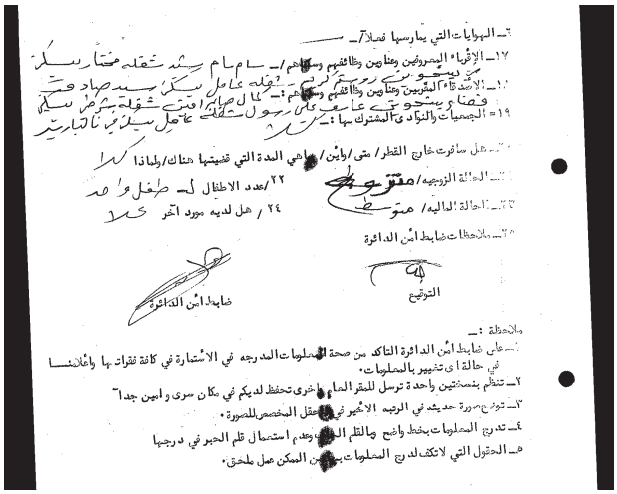

Figure 3. An example from the field dataset.

Figure 4. Samples of segmentation results.

of this dataset, a loose threshold $T = 0.75$ was used for evaluation to allow minor matching errors. Table I shows that the proposed approach outperformed [5] on the test set. Figure 4 shows a few samples of segmentation results. We can see the proposed approach is able to segment complex text-lines.

Table I
F1 ON THE FIELD DATASET (%)

[1]	proposed
56.5	64.9

Table II
F1 ON THE ICDAR2009 CONTEST DATASET (%)

T	[1]	proposed
0.95	97.8	98.3
0.75	–	99.4

The ICDAR2009 contest dataset contains 200 images with a total of 4034 text-lines. We randomly split this dataset into two parts, training and test, each with 100 images. The codebook was not directly constructed on the training set, instead, we first created two semi-synthesized datasets S_1 and S_2 from the training set. S_1 and S_2 are created in the same way as follows : each training image produced 5 copies $I_1, I_2, \ldots, I_5$ and j text-lines are randomly erased on $I_j, j = 1, 2, \ldots, 5$. Then we obtained 500 semi-synthesized images for S_1 and S_2 each. The purpose of the semi-synthesized datasets is to provide graphs with different number of clusters for training. If we directly train our model on the original data, connections exist between bins of neighboring text-lines, thus most images will produce a single graph containing all text-lines, and training samples of small cluster number are scarce. When text-lines are randomly erased, separate graphs with different cluster numbers emerge in each semi-synthesized image, thus balanced training samples are obtained. The codebooks and RBF kernel SVM models under different sizes of context patch and bins were constructed on S_1. Parameters of SVM models were optimized on S_2. Since we simply applied the proposed method without much post-processing and optimization targeted at this dataset, we have not achieved state-of-the-art accuracy (99.53% reported in the contest [8]) yet. Table I shows the performance on test set with different acceptance threshold T. Although a full comparison is lacked due to the current unavailability of [5]'s performance at $T = 0.75$, we can see the accuracy approaches 100% at the looser threshold. This indicates that main bodies of text-lines are mostly correctly detected and character and stroke level errors primarily affect performance, and this is confirmed by a careful visual checking.

V. CONCLUSION

We proposed a codebook based method for handwritten text-line segmentation. Variations of text-line patterns and graph-based similarity are learned from image patches. A novel feature was presented to accurately predict cluster numbers in partitioning the graph. Our method achieves a good accuracy on a set of degraded and un-constrained handwritten Arabic document images. Results on ICDAR2009 segmentation contest dataset also show that the method is competitive to previous approaches.

ACKNOWLEDGMENT

The partial support of this research by DARPA through BBN/DARPA Award HR0011-08-C-0004 under subcontract 9500009235, the US Government through NSF Award IIS-0812111 is gratefully acknowledged.

REFERENCES

[1] Z. Lu, R. Schwartz, P. Natarajan, I. Bazzi, and J. Makhoul, "Advances in the bbn byblos ocr system," in *ICDAR*, Sept. 1999, pp. 337 –340.

[2] R. Haralick, "Document image understanding: geometric and logical layout," in *CVPR*, June 1994, pp. 385 –390.

[3] S. Messelodi and C. Modena, "Automatic identification and skew estimation of text lines in real scene images," *Pattern Recognition*, vol. 32, no. 5, pp. 791 – 810, 1999.

[4] Y. Li, Y. Zheng, D. Doermann, S. Jaeger, and Y. Li, "Script-independent text line segmentation in freestyle handwritten documents," *PAMI*, vol. 30, no. 8, pp. 1313 –1329, Aug. 2008.

[5] J. Kumar, L. Kang, D. Doermann, and W. Abd-Almageed, "Segmentation of handwritten textlines in presence of touching components," in *ICDAR*, Sept. 2011, pp. 109 –113.

[6] U. Pal and S. Datta, "Segmentation of bangla unconstrained handwritten text," in *ICDAR*, Aug. 2003, pp. 1128 – 1132.

[7] V. Manohar, S. Vitaladevuni, H. Cao, R. Prasad, and P. Natarajan, "Graph clustering-based ensemble method for handwritten text line segmentation," in *ICDAR*, Sept. 2011, pp. 574 –578.

[8] B. Gatos, N. Stamatopoulos, and G. Louloudis, "ICDAR 2009 handwriting segmentation contest," in *ICDAR*, July 2009, pp. 1393 –1397.

[9] N. Sebe, I. Cohen, A. Garg, and T. Huang, *Machine Learning in Computer Vision*. Springer, 2005.

[10] D. Doermann, E. Zotkina, and H. Li, "GEDI - a groundtruthing environment for document images," in *Document Analysis Systems*, 2010.

[11] J. Kumar, W. Abd-Almageed, L. Kang, and D. Doermann, "Handwritten arabic text line segmentation using affinity propagation," in *Document Analysis Systems*, 2010, pp. 135–142.

[12] Z. Shi, S. Setlur, and V. Govindaraju, "A steerable directional local profile technique for extraction of handwritten arabic text lines," in *ICDAR*, July 2009, pp. 176 –180.

[13] F. Yin and C.-L. Liu, "Handwritten chinese text line segmentation by clustering with distance metric learning," *Pattern Recognition*, vol. 42, no. 12, pp. 3146 – 3157, 2009.

[14] F. Jurie and B. Triggs, "Creating efficient codebooks for visual recognition," in *ICCV*, vol. 1, Oct. 2005, pp. 604 – 610.

[15] P. Ye, J. Kumar, L. Kang, and D. Doermann, "Unsupervised feature learning framework for no-reference image quality assessment," in *CVPR*, June 2012.(To Appear)

[16] S. Yu and J. Shi, "Multiclass spectral clustering," in *ICCV*, Oct. 2003, pp. 313 –319 vol.1.

[17] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec. 2007.

[18] W. Dumouchel and F. O'Brien, "Computing and graphics in statistics," A. Buja and P. A. Tukey, Eds. New York, NY, USA: Springer-Verlag New York, Inc., 1991, ch. Integrating a robust option into a multiple regression computing environment, pp. 41–48.