Offline Writer Identification using K-Adjacent Segments

Rajiv Jain and David Doermann
University of Maryland, College Park, USA
e-mail: rajivj@cs.umd.edu and doermann@umiacs.umd.edu

Abstract— **This paper presents a method for performing offline writer identification by using K-adjacent segment (KAS) features in a bag-of-features framework to model a user's handwriting. This approach achieves a top 1 recognition rate of 93% on the benchmark IAM English handwriting dataset, which outperforms current state of the art features. Results further demonstrate that identification performance improves as the number of training samples increase, and additionally, that the performance of the KAS features extend to Arabic handwriting found in the MADCAT dataset.**

Writer Identification; Handwriting; Codebook; Local Features; Document Forensics; K-Adjacent Segments

I. INTRODUCTION

Handwriting is a behavioral biometric, which can be used to uniquely identify or verify a document's author and is often admissible as evidence in legal proceedings. This paper addresses the research problem in offline writer identification of matching a handwritten sample from an unknown author to a set of handwriting samples with known authors. This is distinct from the related writer verification problem were one tries to authenticate the author of a handwritten sample.

There are many applications for offline writer identification in criminal forensics such as the analysis of ransom notes, where a document of interest is matched to a set of samples from a given suspect. For example, the FBI recently used handwriting analysis to identify and arrest an individual who mailed handwritten letters with threats along with white powder to political leaders in 2010 [1], demonstrating there continues to be a need for accurate writer identification. The main goal of this research is to design effective writer identification techniques to aid forensic document experts that would otherwise have to manually compare large numbers of documents.

This paper builds on previous work on writer identification, which is outlined in Section II. The approach of modeling character contours using K-adjacent segments (KAS) feature is described in Section III. In Section IV, we present how a codebook is constructed to represent a handwriting model as a vector of code words. Sections V and VI describe the experimental datasets and results.

II. RELATED WORK

Offline handwritten writer identification is a well-studied topic that has seen steady progress in the last ten years. Table 1 summarizes performance of some of the previous literature on this topic.

Author	Dataset Language	# of Writers	% Correct
Srihari [2]	English	900	87
Schlapbach [3]	English	50	94
Schlapbach [15]	English	100	98
Bulacu [5]	English	650	89
Schomaker[4]	Dutch	250	87
Bulacu [8]	Arabic	350	88
Abdi [7]	Arabic	82	90
Chen [14]	Arabic	60	75
He [6]	Chinese	20	80

Table 1: Performance of past writer identification approaches.

Previous research by Srihari [2] established the uniqueness of handwriting by showing that writer verification can be performed at a rate of 96%, and writer identification at a rate of 87%, for a dataset of over 1500 writers. They identify macro features that include intensity changes, slope, and contour features that operate at the paragraph, line and word levels. They also identify micro features, which include gradient, concavity, and structure at the character level. Micro features are shown to significantly outperform the macro features. While the results are impressive, the dataset set contains identical passages from all writers and the approach required manual segmentation, which may not be practical for real world scenarios.

In [3], Schlapbach uses a sliding window to extract nine simple geometric features from a line of text and builds a Hidden Markov Model (HMM) for each of the writers. The author uses the log likelihood output by the Viterbi algorithm to rank users and achieves an identification rate of 94% on 50 writers from the IAM dataset. This work is extended in [15] where Schlapbach uses a Gaussian Mixture Model and achieves an identification rate of 98.5% on 100 writers. Both of these techniques assume perfect line segmentation and require a substantial amount of training. The author uses a 4-fold cross validation on extracted lines during the experiments instead of entire pages, potentially mixing training and testing samples that occurred from the same page. Subsequent papers have used a leave-one-out methodology using between 300-650 writers in the experiments as has been done in this paper.

In [4], Schomaker models character allographs by creating a codebook of connected component contours (CO3) and matching using a bag of features model. In [5] Bulacu models the curvature of characters by introducing the edge hinge, which models the relative angle of two line segments on a character's contour. They combine this method with the CO3 slant features, and run lengths to achieve an identification accuracy of 89% on the IAM dataset for 650 writers. This approach is the current state of the art for writer identification.

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.159

Figure 1: This image illustrates how contours and edges are extracted from connected components in documents.

Recently the authors of [6], [7], [8] extended writer identification to Chinese and Arabic. In [6] the authors use Gabor wavelets for features and HMMs to classify Chinese with moderate success. In [7] and [8] the authors use features similar to those in [5] for Arabic datasets. [7] achieves a recognition rate of 90% on a dataset of 40 writers and [8] achieves an identification rate of 88% when using 5 training samples on a dataset of 350 writers.

While there has been a steady increase in writer identification performance, much of the recent progress has come from combining weaker features together. The edge-hinge feature developed in 2003 continues to be the best performing independent feature [5] and past research shows that similar features that can model properties of the underlying stroke patterns such as curvature and slant perform the best. With a maximum accuracy rate of 80% for a single feature, there is certainly room for improvement. This paper takes the approach of finding a feature that can more accurately model properties of the underlying stroke rather than identifying a novel characteristic of handwriting and combining it with previous features.

III. K-Adjacent Segments (KAS)

K-adjacent segments were introduced by Ferrari, 2008 [9] as a feature to represent the relationship between sets of neighboring edges in an image for object detection. It has since been successfully extended for a number of applications in handwritten text including language identification [10] and text zone detection and classification [11] based on the feature's ability to capture discriminative local stroke information in document images. This work aims to build upon the work in [5] by modeling the character contours using a codebook of KAS features.

In order to extract KAS features from a document image, a set of edges must be found. In color or gray scale images, Ferrari uses a Canny edge detector. Document images are typically binary, so contours that capture the shape and curvature are extracted. A line fitting algorithm is then used to decompose the smooth curves into a set of lines. This process is illustrated in Figure 1.

As the name K-adjacent segments implies, this feature describes any number of K neighboring line segments, but for this paper only 2, 3 and 4 adjacent segments (2AS, 3AS, 4AS) are tested. Any two lines are said to be adjacent if they share an endpoint. The lines that make up the KAS feature must be ordered in a consistent and repeatable manner so that KAS features can be directly compared against each other. The primary line segment is defined as the line with its midpoint closest to the center of the midpoints from all the lines. The remaining lines are ordered by their midpoints from left to right and then top to bottom. Each of the K lines can then be described by the following features:

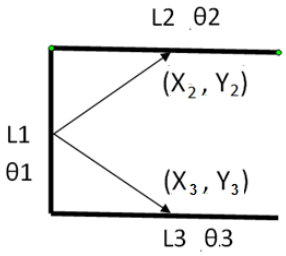


Figure 2: This figure illustrates the segment ordering and features captured for a 3AS, with the primary segment numbered 1.

$$\frac{r^{x_2}}{N_d}, \frac{r^{y_2}}{N_d}, \ldots, \frac{r^{x_k}}{N_d}, \frac{r^{y_k}}{N_d}, \theta_1, \ldots, \theta_k, \frac{l_1}{N_d}, \ldots, \frac{l_k}{N_d} \quad (1)$$

Here (r^x , r^y) define the vector that connects the midpoint of a given segment and the midpoint of the primary segment. Θ and l are the orientation and length of a given segment that makes up the KAS feature. N is the length of the largest segment and is used as a normalization factor to make the feature scale invariant. Features for a 3AS are illustrated in Figure 2. Two KAS features, A and B, can be compared using the distance function D(A,B):

$$D(a,b) = w^r \sum_{i=2}^{k} \left\| r_i^a - r_i^b \right\| + \quad (2)$$

$$w^\theta \sum_{i=1}^{k} \left| \theta_i^a - \theta r_i^b \right| \quad + w^l \sum_{i=1}^{k} \left| \log\left(l_i^a / l_i^b\right) \right|$$

The weights w_r, w_Θ, and w_l can be adjusted to assign more importance to particular features as needed. For this work we use weights of w_r=4, w_Θ=2, and w_l=1 as done in [9] because the segment size is the least stable portion of this feature.

IV. KAS Codebook

A bag of features (BOF) model is used to compare the writers from two documents by converting the KAS features extracted from a document into a vector of code words. We use a clustering technique known as affinity propagation [6] to cluster KAS features from a set of training data to construct a codebook for the BOF model. The input to the affinity propagation algorithm is a distance matrix between all features. Initially all points are considered exemplar clusters and each cluster is combined with neighboring clusters using a message passing algorithm. Two types of messages are passed that represent the responsibility and availability for a given exemplar. The responsibility message, sent from point *i* to point *k*, is defined by *r(i,k)* and represents accumulated evidence for how well suited a point *i* is to be an exemplar for point *k*. The availability message,

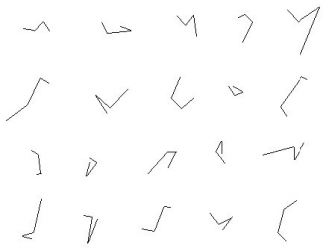


Figure 3: Example of TAS code words.

sent from point k to point i, is defined by $a(i,k)$ and represents how appropriate it would be for point i to represent the exemplar of point k. The equations for both can be seen below.

$$r(i,k) \leftarrow s(i,k) - \max_{k',k' \neq k} \{a(i,k') + s(i,k')\} \quad (3)$$

$$a(i,k) \leftarrow \min\{0, r(k,k) - \sum_{i',i' \neq \{k,i\}} max\{0, r(i',k)\}\} \quad (4)$$

These messages continue to pass until a "preference" threshold is met. It should be noted that unlike K-means this algorithm does not require the number of clusters ahead of time and that the number of clusters is instead controlled by the preference threshold. This approach is used instead of K-means mainly because it converges quicker and is less likely to get stuck in local minima [6].

Once a codebook is constructed, the source document is represented by a feature vector of KAS "code words" present in the document. This feature vector is normalized to sum up to 1 so that it is invariant to the size of the input. The two feature vectors can then be compared by their Euclidean distance. Figure 3 shows examples of the 20 most popular 3AS code words present in the IAM training dataset.

V. DATASETS

In our experiments described in the next section, we use two datasets to show that the performance of the KAS features is not dependent on a particular dataset and extends to multiple languages.

A. IAM Handwriting Dataset

The IAM dataset is made up of samples of handwritten English text from 650 different writers. This dataset has been used by a number of other authors and can be considered the primary benchmark dataset for writer identification [5]. The samples are each scanned as 300 DPI grayscale images. Each of the samples is made up of two or three sentences. The number of pages per writer varies from 1 to 59. 356 writers only provided a single sample, 301 writers provided at least two samples, 159 writers have provided at least three samples and 127 writers provided at least four samples. In order to process this data, each image is preprocessed by binarizing the data using a threshold of 70% and removing connected components with a mass less than 30 pixels in order to remove speckled noise from the binarization. Figure 4 illustrates samples from two different writers.

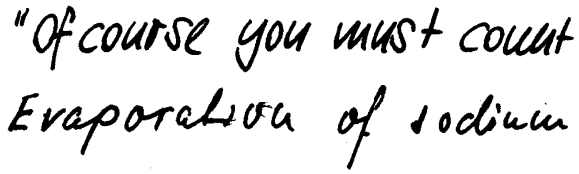


Figure 4: Two writer samples from the IAM dataset.

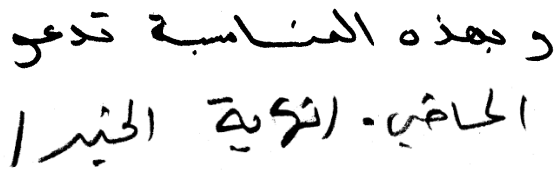


Figure 5: Two writer samples from the MADCAT dataset.

B. DARPA MADCAT/GALE Dataset

The DARPA MADCAT/ GALE dataset is made up of over 10,000 pages of scanned handwritten Arabic text from over 325 writers. A more detailed outlined of this dataset can be found in [10]. The images are sampled at 600 dpi, are already binarized, and are significantly noisier and less structured then the IAM dataset. For example, writers that contributed to this dataset were directed to write at various speeds using various writing instruments (pencils, pens and markers) and to add natural variation into the handwriting samples. Ten documents were chosen randomly from the collection for each of the 302 writers that had at least ten samples. As before, small components were removed to reduce the effect of speckled noise from the binarization process. Connected components with a mass less than 100 pixels were removed here instead of 30 due to the higher resolution of the images. Only one paper [14] has reported results for writer identification on the MADCAT data and they report an accuracy of 75% on similar data with 60 writers. Figure 5 shows samples from two different writers.

VI. EXPERIMENTS AND RESULTS

Four experiments are conducted on the IAM and MADCAT dataset to determine the effectiveness of the KAS feature for writer identification. The KAS feature is implemented in C++ and all experiments are run on 3.0 GHZ Quad Core PC. KAS features can be extracted from a 5 megapixel binary document image in approximately .14 seconds and all the nearest neighbor comparisons for the experiments were completed in less than one second for as many as 650 images.

A. Optimal number of K-adjacent line segments

The first experiment tested 2AS, 3AS, and 4AS for varying codebook sizes on the IAM dataset to determine which codebook size and segment size resulted in the best approach. Samples from the 350 writers who wrote a single page were used for creating the three codebooks and two

Rank	2AS	3AS	4AS
Top1	89.6%	**93.3%**	92.0%
Top2	92.5%	**94.1%**	93.6%
Top5	94.0%	**95.3%**	95.0%
Top10	94.6%	**96.0%**	95.8%

Table 2: This table shows the performance for 2AS, 3AS, and 4AS.

Rank	3 training samples	2 training samples	1 training sample
Top1	**99.8%**	99.2%	98.4%
Top2	**99.8%**	99.6%	99.3%
Top5	**100.0%**	99.9%	99.7%
Top10	**100.0%**	100.0%	99.8%

Table 3: Results on 127 writers from IAM dataset.

samples from the remaining 300 writers were randomly chosen for testing. Recognition was performed using a K-nearest neighbor (KNN) approach in a leave one out configuration similar to [7], meaning each query image was run against 599 other documents with only 1 possible positive match. The results are shown Figure 6 and Table 2.

Figure 6 shows that once the codebook reaches a size of 300 clusters, the identification performance does not increase significantly. The results in Table 2 indicate that 3AS is the best feature representation, with a Top 1 recognition rate of 93.3%. This could be because 2AS does not capture as much information and there were less repeatable 4AS features found in a given document. Still, each of the three approaches is within a couple percentage points and all outperform the state of the art features shown in [5]. While the features are similar to the edge hinge and slant features used previously, the improved performance is likely due to the extra segment found in the 3AS feature, the addition of segment size in the feature representation, and the use of a codebook of clusters rather than coarse quantization. Given the superior performance of the 3AS features, it is used for the remaining experiments.

A close examination of the errors revealed that five of the writers changed their writing style making it very difficult to identify them and resulting in an error rate of 1.7%. The remaining errors are primarily due to too few lines resulting in few KAS features or a large discrepancy between lowercase and uppercase characters between the writing samples. The top 10 score of 96-97% likely represents an upper-bound for this dataset and is consistent with other published results for this dataset.

B. *Improvement from Additional Training Samples*

A second experiment was performed on the IAM dataset to determine if more than one training sample benefits the writer identification performance. The 3AS feature and codebook are reused from the first experiment. Four samples were randomly chosen from 127 writers to create the experimental dataset. Between 1 and 3 of the samples were used for training and the remaining unused samples were used for testing using a KNN search as before. All possible combinations of training and testing were performed. When using more than one sample for training, the distances between the feature vectors found during the nearest neighbor search were averaged for the multiple "trained" samples. This process is repeated using only one dataset for testing and one for training for comparison purposes.

The results indicate that there is a 50% and 87.5% decrease in the error rate when using 2 and 3 training samples respectively. This experiment also shows that the 3AS feature is extremely effective for a population of around 127 writers. These results may be skewed due to the high precision when using just one training sample due to the limited numbers of writers available in the IAM dataset. For this reason, a further review of the relationship between the number of training samples and identification performance is continued on the MADCAT dataset in the third experiment.

C. *Performance on Arabic*

The third experiment again used the 3AS features on the Arabic MADCAT data containing 10 samples for 302 writers. A new codebook was trained from the unused portion of the dataset. This experiment followed the same procedure as experiment 2 in order to take advantage of the 10 pages per writer. A range between 1 to 7 pages were used for training to determine if having more training data was beneficial. Every possible combination of training and testing was run using a KNN search and the results are shown in Figure 7.

Figure 7 shows that increasing the number of training samples is certainly beneficial for identification performance. The top 1 accuracy rises from 80 percent with

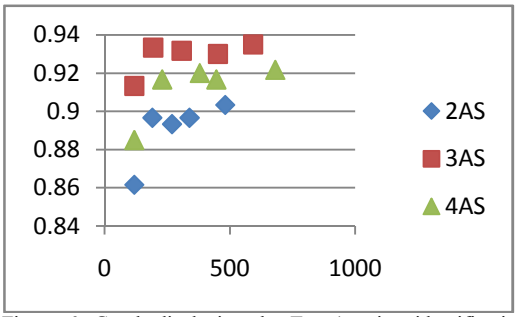


Figure 6: Graph displaying the Top 1 writer identification performance versus the number of clusters for 2AS, 3AS, and 4AS.

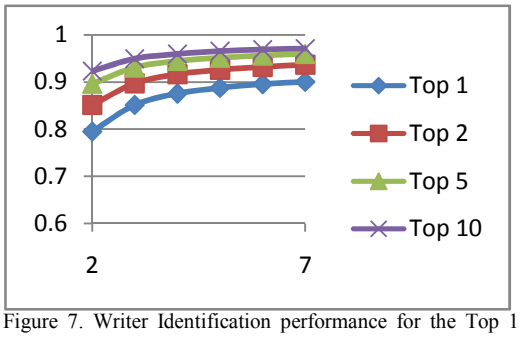


Figure 7. Writer Identification performance for the Top 1, 2, 5, and 10 score versus the number of training samples.

Rank	350 writers	650 writers
Top1	**92.3%**	92.1%
Top2	**93.8%**	93.6%
Top5	**94.8%**	94.5%
Top10	**95.8%**	95.8%

Table 4: Results on the IAM dataset using the Arabic codebook.

1 training sample, to 85% with 2 training samples and 90% with 7 training samples. This outperforms the previously reported rate of 75% in [14] and the 97% top 10 rate exceeded expectations given the amount of noise and variation present in this data.

D. Universality of the codebook

The last experiment uses the Arabic codebook on the IAM dataset. This was done in order to show that a codebook is not unique to a language as shown in [12]. This also allows all 650 writers from the IAM dataset to be tested on for a direct comparison to the state of the art results presented in [5]. Since the additional 300 writers have only published one page, they are retrieved against, but not queried for.

The results in Table 4 show that the codebook is indeed largely independent from language as there is only a 1% drop in accuracy between the English and Arabic codebooks for 350 writers on the IAM dataset. The addition of 300 more writers hardly impacts the accuracy showing that this approach should be robust to many more additional writers. The 92.1% precision on 650 writers represents a 27% reduction in error over the results presented in [5].

VII. CONCLUSION

This paper demonstrates the effectiveness of the KAS feature at modeling handwriting for writer identification. The KAS feature improves upon previous approaches where many features have to be combined and still do not reach the same performance. The identification rate of 93% on the IAM dataset and 90% on the MADCAT dataset outperforms the current state of the art. The experiments show that the codebook is generic between languages and writers so it does not have to be continually recreated and the technique is extremely robust. Additional improvements to the precision could also be made by combining the KAS feature with the approaches presented in past papers. Given the performance of KAS, other local features such as SIFT and shape context should be further examined as they could potentially provide an orthogonal approach to boost the overall performance.

REFERENCES

[1] "Aurora Man Sentenced for Mailing Threats Which Included White Powder to Government Officials", FBI Press Release, http://www.fbi.gov/denver/press-releases/2011/dn012811.htm. Jan. 28, 2011.

[2] S. Srihari, S. Cha, H. Arora, and S. Lee, "Individuality of Handwriting," J. Forensic Sciences, vol. 47, no. 4, pp. 1-17, July 2002

[3] A. Schlapbach and H. Bunke, "Using HMM-Based Recognizers for Writer Identification and Verification", ICFHR, pp. 167-172, Oct. 2004.

[4] L. Schomaker and M. Bulacu, "Automatic writer identification using connected-component contours and edge-based features of upper-case western script". *PAMI*, 26(6):787–798, 2004.

[5] M. Bulacu and L. Schomaker, "Text-independent writer identification and verification using textural and allographic features". *PAMI*, 29(4):701–717, April 2007.

[6] Z. He, X. You, Y. Tang, "Writer identification of Chinese handwriting documents using hidden Markov tree model, Pattern Recognition", Volume 41, Issue 4, April 2008, Pages 1295-1307

[7] M. N. Abdi, M. Khemakhem, and H. Ben-Abdallah, "A novel approach for off-line Arabic writer identification based on stroke feature combination" ISCIS 2009. 24th International Symposium on, Sep 2009, pp. 597 –600.

[8] M. Bulacu, L. Schomaker, A. Brink, "Text-Independent Writer Identification and Verification on Offline Arabic Handwriting," *ICDAR 2007*. pp.769-773, 23-26 Sept. 2007

[9] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection", IEEE Trans. PAMI, No. 30, 2008, pp. 36 – 51.

[10] S. Strassel, "Linguistic resources for Arabic handwriting recognition". In Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt, April 2009.

[11] J. Kumar et. al. "Shape Codebook based Handwritten and Machine Printed Text Zone Extraction." Document Recognition and Retrieval XVIII, pp. 1-8. January 2011.

[12] G. Zhu, X. Yu, Y. Li, D. Doermann, "Language Identification for Handwritten Document Images Using A Shape Codebook. Pattern Recognition", No. 42, pp 3184-3191, 2009.

[13] B. Frey and D. Dueck, "Clustering by Passing Messages Between Data Points," Science 315, 972-976.

[14] J. Chen, D. Lopresti, and E. Kavallieratou, "The Impact of Ruling Lines on Writer Identification". ICFHR. pp. 439-444, Nov, 2010.

[15] A. Schlapbach and H. Bunke. "Off-line Writer Identification Using Gaussian Mixture Models", ICPR. pp. 992-995. Sep. 2006.

.

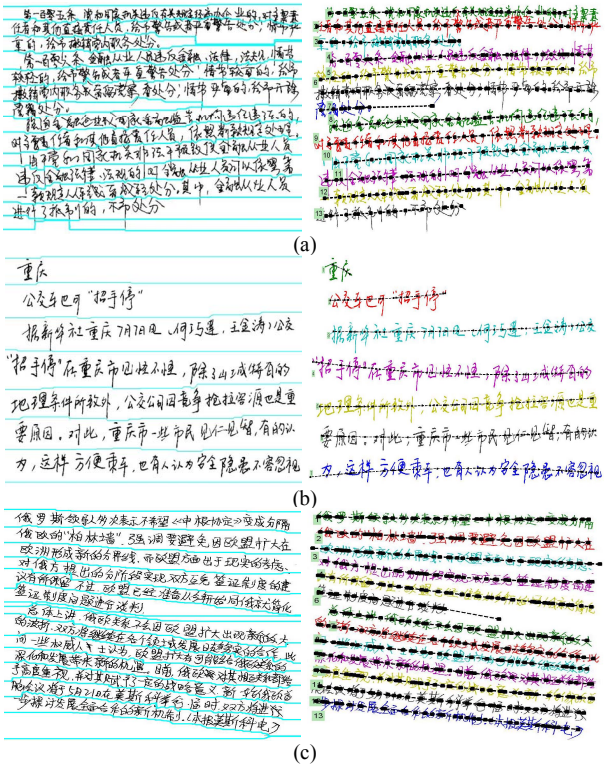


Figure 8. Several segmentation results on HIT-MW. (a) The heights of the text lines vary widely. Moreover the first and the second text lines overlap with each other. (b) The text lines have different lengths. (c) The text lines have various directions and skew angles.

III. PERFORMANCE EVALUATION METHOD

Some researchers perform the evaluation of text line segmentation algorithm based on visual criteria. Such evaluation method is not adopted due to its tediousness and strong subjectivity in this paper. To avoid the shortages, we use an automatic performance evaluation that has been followed in many document segmentation competitions [5] and published papers. Based on counting the number of the matching pixels between ground-truth text lines and detected text lines, the evaluation method [6] below provides evaluation results at both pixel level and line level.

Let the number of ground-truth lines be N_{gt}, the number of detected lines be N_d and $N_m=max(N_{gt}, N_d)$. A $N_m \times N_m$ square matrix *MatchScore* is constructed whose element *MS(i, j)* represents the number of matching pixels of the i[th] ground-truth line and the j[th] detected line. A line is allowed to be matched to a dummy one and the corresponding *MS(i, j)=0*. The one-to-one correspondence can be expressed as an array *C*:

index of ground-truth lines	1	2	…	*i*	…
	↕	↕		↕	
index of detected lines	*C*(1)	*C*(2)	…	*C*(*i*)	…

Our purpose here is to find a C_m by which the sum of matching pixels *S(C)* reaches maximum:

$$S(C)=\sum_{k}^{N_m} MS(k,C(k)) \tag{6}$$

$$C_m=\arg\sum_{k}^{N_m} MS(k,C(k)) \tag{7}$$

In theory, the enumeration of one-to-one correspondences is as many as the permutation of N_m. Thanks to the Hungarian algorithm [7], the global optimal solution can be obtained in an acceptable time.

The S_{gt} represents the sum pixels of all the ground-truth text lines. Then the overall correct rate at pixel level is defined as

$$PL=\frac{S(C_m)}{S_{gt}} \tag{8}$$

At the line level, the i[th] ground-truth text line is claimed to be segmented correctly only if the i[th] ground-truth text line and the corresponding detected one share at least 90 percent of the pixels with respect to both of them [6]:

$$\frac{MS(i,C_m(i))}{\sum_{j=1}^{N_m} MS(i,j)}\geq 0.9 \tag{9}$$

and

$$\frac{MS(i,C_m(i))}{\sum_{k=1}^{N_m} MS(k,C_m(i))}\geq 0.9 \tag{10}$$

The overall detection rate *DR* and recognition accuracy *RA* at line level are defined as:

$$DR=\frac{N_{match}}{N_{gt}} \tag{11}$$

$$RA=\frac{N_{match}}{N_d} \tag{12}$$

Where N_{match} is the number of text lines which satisfy (9) and (10).

IV. EXPERIMENTAL RESULTS

The proposed text line segmentation is tested mainly on HIT-MW [8], a freestyle, unconstrained Chinese handwritten text image database written by multiple writers. In this database, we can find some cases such as skewed, overlapping and touching text line that are very common in typical handwritten documents. Totally, it contains 853 handwritten samples containing 8664 text lines with ground truth at pixel level for each image, and no image includes any non-text content.

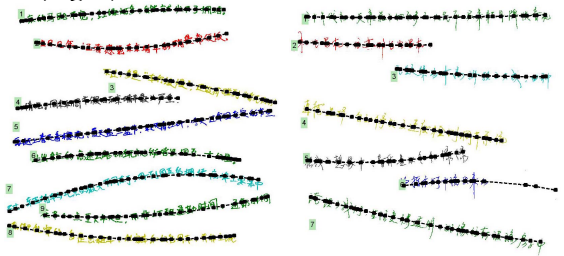
Based on the evaluation method described above, our algorithm has gained a high accuracy. The overall correct rate is 99.71% at pixel level while at line level the detection rate and recognition accuracy are 98.68% and 98.76%, respectively.

TABLE I. COMPARATIVE RESULTS OVER THE HIT-MW DATASET

Method	DR(%)
Docstrum	65.38
Piecewise projection based	92.07
MST clustering(with post processing)	98.02
Block-based	98.34
our method	98.68

A table of segmentation results by other method [9] and our method is given to illustrate the predominance of ours. In general, the proposed method outperforms the previously reported ones over the HIT-MW dataset. Besides, unlike other methods which need large number of training data to determine the parameters, in our method the key parameters (e.g. the size of character) are gained by the estimations. Moreover, the proposed method shows impressive performance in the cases that the text lines have overlapping components, arbitrary skew angles or various-width spaces between them.

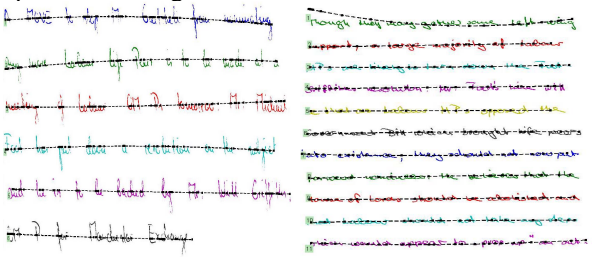
In order to demonstrate the robustness and adaptability of our algorithm, we collect a few samples with more writing freedom (Fig. 9).



(a) (b)

Figure 9. exapmles of documents with more writing freedom.

Additionally, we also test our method on the English handwritten document dataset to check its effectiveness on western-character handwritten documents. Unfortunately the detailed correct rate is not listed for there is no sufficient ground truth at pixel level for these dataset. A few examples are presented in Fig 10.



(a) (b)

Figure 10. exapmles of documents written in English.

We use the same parameter settings in all of these experiments mentioned in this paper.

V. CONCLUSION

In this paper an effective text line segmentation method for handwritten documents is proposed. It can solve the problems of text lines having various skew angles, touching or overlapping with each other. The algorithms in different scale are adaptively chosen to provide the segmentation results. Requiring no training, our method has proved its strong adaptability to various line conditions. We achieve satisfying segmentation results on plenty of samples using the same parameter settings. Experimental results show that the proposed method has obvious superiority over traditional methods.

ACKNOWLEDGMENT

This work was supported by the National Basic Research Program of China (973 program) under Grant No. 2007CB311004 and the National Natural Science Foundation of China under Grant Nos. 60872086.

REFERENCES

[1] Bar-Yosef. I, Hagbi. N, Kedem. K, Dinstein. I, "Line segmentation for degraded handwritten historical documents," Proc. 10th ICDAR, pp.1161-1165, 2009.

[2] G. Louloudis, B. Gatos, I. Pratikakis, K. Halatsis, "A Block-Based Hough Transform Mapping for Text Line Detection in HandWritten Documents" Proc. 10th International Workshop on Frontiers in Handwriting Recognition, Oct. 2006.pp.515–520.

[3] F. Yin, Cheng-Lin Liu, "Handwritten Chinese text line segmentation by clustering with distance metric learning," Pattern Recognition 42: 3169-383, 2009.

[4] A. Nicolaou, B. Gatos, "Handwritten Text Line Segmentation by Shredding Text into lines," Proc. 10th ICDAR, pp.626-630, 2009.

[5] B.Gatos, N. Stamatopoulos, G. Louloudis, "ICDAR2009 Handwriting Segmentation Contest," Proc. 10th ICDAR, pp.1393-1397, 2009.

[6] Y. Li, Y. Zheng, D. Doermann, S. Jaeger, "Script-Independent text line segmentation in freestyle handwritten documents," IEEE Trans. Pattern Analysis and Machine Intelligence. 30(8): 1313-1329, 2008.

[7] G. Liu, R. M,Haralick, "Optimal Matching Problem in Detection and Recognition Performance Evaluation," Pattern Recognition, vol. 4, no. 3, pp. 205-217, 2002.

[8] Tonghua. Su, Tianwen. Zhang, Dejun. Guan, "HIT-MW Dataset for Offline Chinese Handwritten Text Recognition", Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France, Oct. 2006.pp.515–520.

[9] Majid. Ziaratban, Karim. Faez, "An Adaptive Script-Independent Block-Based Text Line Extraction", 20th International Conference on Pattern Recognition, pp.249-252, Istanbul, Aug. 2010.

2011 International Conference on Document Analysis and Recognition

Document Image Indexing using Edit Distance based Hashing

Ehtesham Hassan, Santanu Chaudhury, M Gopal
Department of Electrical Engineering
Indian Institute of Technology Delhi, New Delhi
hassan.ehtesham@gmail.com, santanuc@ee.iitd.ac.in, mgopal@ee.iitd.ac.in

***Abstract*—We present a novel word image based document indexing scheme by combination of string matching and hashing. The word image representation is defined by string codes obtained by unsupervised learning over graphical primitives. The indexing framework is defined by distance based hashing function which does the object projection to hash space by preserving their distances. We have used edit distance based string matching for defining the hashing function and for approximate nearest neighbor based retrieval. The application of the proposed indexing framework is presented for two document image collections belonging to Devanagari and Bengali script.**

***Keywords*-Document image indexing, Distance based hashing, Edit distance, Shape descriptor**

I. Introduction

Image based document indexing provides an effective solution for storage and retrieval of document collections belonging to scripts having underdeveloped OCR technology. The objective of the present work is to define a novel word based document indexing scheme using edit distance based hashing. A word based indexing scheme fundamentally converts the word images segmented from documents to arithmetic features, and groups the similar words in index space based on defined similarity measure. The framework assigns unique indices to each word image group. The retrieval is done by projecting the query image to a word group and return documents corresponding to words in the group. Two primary challenges involved here are defining unique feature representation for word images and efficient grouping framework for similar word images.

In this direction, the paper presents a novel string based word image representation. The conventional feature based representations capture the global characteristics of the word shape [1][2][3]. In this case, the word spotting based retrieval schemes does not include the substrings based matches existing in the documents. Our work in this direction presents novel clustering based approach to define the word representation. We identify the graphical primitives in the word formulation and apply adaptive clustering for grouping these primitives. A graphical primitive represents the structural part of word image obtained after segmentation. The clustering identifies the equivalence groups of graphical primitives exiting in the document collection. Each group of graphical primitives is assigned a unique code. The word image representation is defined by identifying the graphical primitives and assigning the code based on its nearness to a group of primitives. In this sense, our approach presents a script invariant methodology for word image representation. The nearest neighbor based retrieval provide robust retrieval method. However the linear time search complexity is practically unacceptable for searching large datasets. The hashing based approximate nearest search performs retrieval in sub-linear time complexity with trade-off in accuracy. Our work presents a novel application of edit distance based hashing for document indexing. The effectiveness of the proposed framework has been demonstrated on document collections belonging to Devanagari and Bengali script.

The paper organization is as following: Section II presents related works. Proposed indexing and retrieval framework is presented in section III. Section IV presents discussion on Edit distance based hashing. Experimental evaluation of the proposed framework is presented in section V. The section VI presents conclusion and perspective of our work.

II. Related Works

Large amount of research has been done in the area of word image representation, and document indexing [4][5][6][7]. We present a brief and concise review of the related works in this direction. The string based word representation is generated by extracting character objects from the word image. Objects are assigned codes based on their similarity to example character objects. The concatenation of object codes in respective order defines the word string. In recent works, Shijian et al. [4] proposed a word shape coding without requiring character segmentation. The word image is represented by set of topological character shape features including ascenders/descenders, character holes and reservoirs. Simone et al. [5] have defined collection specific character prototypes from the character objects. The set of character prototypes are further used for word image representation. Nakayama [6] defined word shape tokens for printed words by sequence of character shape codes defined by the set of graphical features. In [7], word shape code is defined using conventional features as ascenders, descenders, character holes, deep eastward and westward concavity and horizontal-line intersections. The existing works on string based word representation is script dependent. Additionally,

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.242

For each function F a pair of objects (x_1, x_2) are chosen, a line is drawn between them that serves as coordinate axis, and the coordinate value along this axis for each object is determined by equation (1). If $\mathcal{X}$ is a general non-Euclidean space, then $F^{x_1,x_2}(x)$ does not have a geometric interpretation. However as long as $\mathcal{D}$ is available for $\mathcal{X}$, F^{x_1,x_2} can be defined and provides a simple way to project x on the line defined by (x_1, x_2). The mapping F is independent of the dimensionality of object representation as the inter object distances is the only requirement. The equation (1) defines a rich family of functions. For a collection of N objects in $\mathcal{X}$, $N(N-1)/2$ unique functions can be defined by applying equation (1) to each pair of objects. It is always convenient to have a hashing function that maps objects to $\{0, 1\}$. The functions defined using equation (1) are real valued, whereas we desire discrete-valued hashing functions. The binary hashing functions can be obtained from F^{x_1,x_2} using thresholds $t_l, t_2 \in R$ as:

$$F^{x_1,x_2}_{t_1,t_2}(x) = \begin{cases} 1 & \text{if} \quad F^{x_1,x_2}(x) \in [t_1, t_2] \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

In practice, (t_1, t_2) should be such that $F^{x_1,x_2}_{t_1,t_2}(x)$ maps approximately half the data points in $\mathcal{X}$ to 0 and half to 1, i.e. F generates balanced hash tables. We formalize this notion by defining for each pair $(x_1, x_2) \in \mathcal{X}$, the set $V(x_1, x_2)$ of intervals $[t_1, t_2]$ such that $F^{x_1,x_2}_{t_1,t_2}(x)$ splits the space in half as

$$V(x_1, x_2) = [t_1, t_2] | \Pr_{x \in \mathcal{X}}(F^{x_1,x_2}_{t_1,t_2}(x) = 0) = 0.5 \tag{3}$$

The hash function family $\mathbb{H}_{\mathbb{DBH}}$ for an arbitrary space $(\mathcal{X}, \mathcal{D})$ is defined as

$$\mathbb{H}_{\mathbb{DBH}} = F^{x_1,x_2}_{t_1,t_2}(x) | x_1, x_2 \in \mathcal{X}, [t_1, t_2] \in V(x_1, x_2) \tag{4}$$

An indexing framework is defined by hash function g as concatenation of k functions selected randomly from $\mathbb{H}_{\mathbb{DBH}}$ i.e. $g = [h_1, ..., h_k]$. The hash values generated by g for an object defines its bucket indices in the hash table. With increase in k, the collision probability $\Pr\{g(x) = g(y)\}$ = $(\Pr\{h(x) = h(y))^k$ decreases exponentially. Therefore L independent hash tables are generated by defining L independent hash functions. The retrieval is performed by computing query hash value $g_i(q)$ for i = 1, ..., L. The value $g_i(q)$ represents the query bucket indices in i^{th} hash table. The similar items are retrieved by performing similarity search over the collection of objects retrieved from buckets $g_i(q)$ in L hash tables. The hashing parameters, function length k and number of tables L are performance parameters. In general, large L will increase the recall with reduction in precision and increase in retrieval cost. Larger function length k increases precision with reduction in recall, therefore requires sufficient number of hash tables for satisfactory recall level.

V. Experimental Results

In this section, experimental results of the proposed document image indexing and retrieval framework is presented. The experiments have been performed on document image collection belonging to Devanagari and Bengali scripts. The Devanagari document collection contains 327 pages scanned from 4 books. The Bengali document collection contains 178 pages scanned from 3 books. The document collection has been prepared as a part of consortium project funded by Government of India [17]. The book pages have been collected from old books and scanning is performed at 300dpi. The scanned images in the collection are of low quality, primarily because of degradation in original document pages. The preprocessing step for Devanagari and Bengali documents included smoothening and deskewing. The conversion of original gray scale images to binary images is performed by Otsu's method. The word segmentation from document image is done by horizontal and vertical profile based technique. After initial filtering, Devanagari word dataset contains 14888 words, Bengali word dataset consists 11231 words. The filtering process removes stop words, punctuation marks, and words having length less than threshold.

The word image representation is generated following the discussion in section III-A. After the initial segmentation, each graphical primitive is represented by shape descriptor presented in [18]. The shape descriptors for graphical primitives have been computed for two set of parameters (distance bins:m, angular bins:n). The document indexing framework for Devanagari collection is tested for 271 queries, for Bengali documents the framework is tested for 211 queries. Conventional performance measures i.e. Precision and Recall are computed over unordered result set. However, the ranking of retrieved results is an important retrieval measure. Therefore we computed Mean Average Precision (MAP) for measuring the performance of proposed indexing scheme. The MAP for a query set X_v is computed as the mean of average precision values [19].

$$\text{MAP}(X_v) = \frac{1}{|X_v|} \sum_{j=1}^{|X_v|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \tag{5}$$

Average precision for query $q \in X_v$ is defined as mean of precision at each relevant recall point in the retrieved results. In equation (5), m_j represents the number of relevant retrieved results for query q_j and R_{jk} represents the ranked retrieval results from the top to k^{th} relevant result. If a retrieved document in R_{jk} is non-relevant, the precision value at this recall point is not considered. In the present indexing scheme, average precision for q is computed over the collection of neighbors obtained from L hash tables having indices $g_i(q)$ for $i = 1, ..., L$. The neighbors are ranked based on the Edit distance from the query string q. The Edit distance for word matching is computed as

Levenshtein distance between two strings. The MAP and average comparisons in the retrieval experiment for different hashing and descriptor parameters $\{(L,\ k)$ and $(m,\ n)\}$ is presented in the table I.

Table I
RETRIEVAL RESULTS

Results with Devanagari documents				
Descriptor paras.	m = 35, n = 36		m = 40, n = 40	
Hashing paras.	MAP	Comps.	MAP	Comps.
L = 18, k = 12	0.854	1413	0.861	1387
L = 18, k = 15	0.813	1028	0.816	984
Results with Bengali documents				
Descriptor paras.	m = 35, n = 36		m = 40, n = 40	
Hashing paras.	MAP	Comps.	MAP	Comps.
L = 18, k = 12	0.861	1127	0.864	1087
L = 18, k = 15	0.837	889	0.842	865

VI. CONCLUSION

We have presented a novel word image based document image indexing scheme. Our approach presents a novel script independent word image representation based on clustering which can used for defining indexing and retrieval framework for various class of document collections. We have showed novel application of edit distance based hashing for document indexing. The efficacy of the proposed framework have been shown experimentally on two printed document collection belonging to Indian scripts. The future direction of the work is to test the framework for other class of document collections.

ACKNOWLEDGMENT

This work is funded by MCIT, Government of India as a part of project *Development of Robust Document Analysis and Recognition System for Printed Indian Scripts*.

REFERENCES

[1] R. Bajaj and S. Chaudhury, "Signature verification using multiple neural classifiers," *Pattern Recognition*, vol. 30, no. 1, pp. 1–7, 1997.

[2] T. M. Rath and R. Manmatha, "Word image matching using dynamic time warping," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, vol. 2, pp. 521–527, June 2003.

[3] F. R. Chen, L. D. Wilcox, and D. Bloomberg, "Word spotting in scanned images using hidden markov models," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp. 1–4, 1993.

[4] S. Lu, L. Li, and C. L. Tan, "Document image retrieval through word shape coding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1913–1918, 2008.

[5] S. Marinai, E. Marino, and G. Soda, "Font adaptive word indexing of modern printed documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1187–1199, August 2006.

[6] T. Nakayama, "Content-oriented categorization of document images," *Proceedings of the* 16^{th} *International Conference on Computational Linguistics*, vol. 2, pp. 818–823, 1996.

[7] S. Bai, L. Li, and C. L. Tan, "Keyword spotting in document images through word shape coding," *Proceedings of the* 10^{th} *International Conference on Document Analysis and Recognition*, pp. 331–335, 2009.

[8] T. S. Mehmod, "Indexing of handwritten document images," *Proceedings of the 1997 Workshop on Document Image Analysis*, pp. 66–73, 1997.

[9] P. Indyk and R. Motwani, "Approximate nearest neighbor - towards removing the curse of dimensionality," *Proceedings of the* 30^{th} *ACM Symposium on Theory of Computing*, pp. 604–613, 1998.

[10] P. Haghani, S. Michel, and K. Aberer, "Distributed similarity search in high dimensions using locality sensitive hashing," *Proceedings of the* 12^{th} *International Conference on Extending Database Technology*, pp. 744–755, 2009.

[11] H. Shen, T. Li, and T. Schweiger, "An efficient similarity searching scheme in massive databases," *Proceedings of the* 3^{th} *International Conference on Digital Telecommunications*, pp. 47–52, 2008.

[12] W. Weihong and W. Song, "A scalable content-based image retrieval scheme using locality-sensitive hashing," *Proceedings of the International Conference on Computational Intelligence and Natural Computing*, vol. 1, pp. 151–154, 2009.

[13] B. Matei, Y. Shan, H. S. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert, "Rapid object indexing using locality sensitive hashing and joint 3d-signature space estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1111 – 1126, 2006.

[14] M. Ester, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proceedings of the* 2^{nd} *International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

[15] A. Vassilis, P. Michalis, P. Panagiotis, and K. George, "Nearest neighbor retrieval using distance based hashing," *Proceedings of the* 24^{th} *International Conference on Data Engineering*, pp. 327–336, April 2008.

[16] C. Faloutsos and K. I. Lin, "Fastmap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," *Proceedings of the ACM International Conference on Management of Data*, pp. 163 – 174, 1995.

[17] [Online]. Available: http://ocr.cdacnoida.in/

[18] E. Hassan, S. Chaudhury, M. Gopal, and J. Dholakia, "Use of mkl as symbol classifier for gujarati character recognition," *In the Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pp. 255–262, 2010.

[19] C. D. Manning, P. Raghavan, and H. Schtze, *An Introduction to Information Retrieval*. Cambridge University Press, Cambridge, 2009.

Scientific Table Type Classification in Digital Library

Seongchan Kim, Keejun Han
Dept. of Knowledge Service Engineering
KAIST
Daejeon, Korea
sckim, keejun.han@kaist.ac.kr

Soon Young Kim
Dept. of Overseas Information
KISTI
Daejeon, Korea
maya@kisti.re.kr

Ying Liu
Dept. of Knowledge Service Engineering
KAIST
Daejeon, Korea
yingliu@kaist.edu

ABSTRACT
Tables are ubiquitous in digital libraries and on the Web, utilized to satisfy various types of data delivery and document formatting goals. For example, tables are widely used to present experimental results or statistical data in a condensed fashion in scientific documents. Identifying and organizing tables of different types is an absolutely necessary task for better table understanding, and data sharing and reusing. This paper has a three-fold contribution: 1) We propose Introduction, Methods, Results, and Discussion (IMRAD)-based table functional classification for scientific documents; 2) A fine-grained table taxonomy is introduced based on an extensive observation and investigation of tables in digital libraries; and 3) We investigate table characteristics and classify tables automatically based on the defined taxonomy. The preliminary experimental results show that our table taxonomy with salient features can significantly improve scientific table classification performance.

Categories and Subject Descriptors
H.3.m [**Information Storage and Retrieval**]: Miscellaneous; I.2.6 [**Artificial Intelligence**]: Learning – knowledge acquisition.

General Terms
Algorithms, Measurement, Experimentation

Keywords
Scientific tables, IMRAD, fine-grained, taxonomy, classification

1. INTRODUCTION
Tables are ubiquitous in all types of documents such as scientific publications, web pages, financial reports, newspapers, magazine articles, etc. Understanding table type, function, and purpose is crucial for better table understanding and for more accurate table data sharing and reuse. Moreover, automatic identification of the functionality of each document table could be useful for many information-processing tasks, including advanced information retrieval, knowledge extraction [2], mobile access, and data integration.

Tables are differently used to present different types of information with various purposes. Scientists use various types of tables to display such things as experimental results or statistical data for multiple purposes. Scholars can easily obtain valuable insight by examining such type-specified tables. For example, a medical scientist may want to search for tables containing information about "cancer." He or she may want only tables that contain definitions, experimental results, or medical interview questions. However, none of the currently available table search engines (e.g., BioText Search Engine [5], *Tableseer* [7]) support table categorization by type. When issuing a query to these table-specialized search engines, end-users will get only a list of keyword-relevant tables, regardless of their types. The purpose of search by table type is, therefore, to help users to easily recognize the relevance of results by referring to the same type of tables.

Table type related research has recently been receiving considerable attention. Crestan and Pantel [2] report on a census of the types of HTML tables on the Web and propose a fine-grained classification taxonomy. However, their taxonomy is too limited to apply to scientific tables, since table types are heavily dependent on the nature of documents. Kim and Liu [6] first suggest functional-based table types for scientific tables; however, they do not provide fine-grained taxonomy but only two types of table: commentary and comparison. To the best of our knowledge, this is the first study of fine-grained table taxonomy for scientific tables in digital libraries.

In this paper, we observe tables in scientific papers, abstract the underlying table functional-based types, investigate the table characteristics, and demonstrate the distribution of different table types. We focus on scientific papers, since they are one of the most important media in digital libraries and contain many tables (1.28 tables per paper in our dataset), which are all genuine, unlike Web tables [2]. The preliminary experimental results show the effective performance of our system of automatic table type classification. The contributions of this paper are as follows: 1) We propose the finest (IMRAD-based) table functional classification system for scientific documents by considering the structural position of tables within a document; 2) A fine-grained table taxonomy is introduced first, based on an extensive period of table observation and investigation in digital libraries; 3) We investigate the scientific table characteristics and automatically classify tables based on the defined taxonomy; and 4) The whole system and methodology can be easily applied to tables in any fields and formats without much modification in order to achieve a fully automatic table classification and understanding.

The paper is organized as follows. In section 2 we present our table type taxonomy including definitions and descriptions for each category. Section 3 reports the design of our experiment and our results. Finally, we conclude in section 4.

2. Table Type Taxonomy
We propose a brand-new table type taxonomy that is the results of our extensive observation and investigation of 2,500 tables that were randomly collected from 25 randomly selected scientific journals [1] published by Springer from 2006 to 2010 in five

DocEng'12, September 4–7, 2012, Paris, France.

[1] All lists are available at http://issl.kaist.ac.kr/table

2011 International Conference on Document Analysis and Recognition

Segmentation of Handwritten Textlines in Presence of Touching Components

Jayant Kumar Le Kang David Doermann Wael Abd-Almageed
Institute of Advanced Computer Studies
University of Maryland College Park, USA
{*jayant, lekang, doermann, wamageed*}*@umiacs.umd.edu*

***Abstract*—This paper presents an approach to textline extraction in handwritten document images which combines local and global techniques. We propose a graph-based technique to detect touching and proximity errors that are common with handwritten text lines. In a refinement step, we use Expectation-Maximization (EM) to iteratively split the error segments to obtain correct text-lines. We show improvement in accuracies using our correction method on datasets of Arabic document images. Results on a set of artificially generated proximity images show that the method is effective for handling touching errors in handwritten document images.**

***Keywords*-Text-lines, Handwritten Documents, Arabic**

I. Introduction

Segmentation of text-lines can be a crucial step in many document processing tasks including skew detection, layout analysis and word/character recognition. For handwritten text, the problem is even more difficult due to its free style nature, character size variations and non-uniform spacings between components. Moreover, the touching of characters across lines and overlapping spatial envelopes of text-lines make the problem more challenging. Methods previously developed for segmentation of printed text-lines do not adapt well to these scenarios.

Methods based on connected-components are fast but they suffer from touching or close proximity of components. In Figure 1, we show the result of a connected-component based text-line extraction method [1] on a document image with many touching components (shown in dotted boxes). When the touching components are separated manually by cutting the characters using an image editing tool, the same method gives correct segmentation without any change of parameters. One obvious solution is to detect and correct such touching errors before applying text-line segmentation. But this may be computationally expensive as the number of components in a document image may be large. We take another approach in which the detection and correction of such errors are delayed until an initial estimate of text-line segmentation is obtained. Each line is then checked for touching and proximity errors. This is computationally more efficient as the number of lines detected (20-30) are far less than the number of components (600-800) in a typical document image.

In this work we present a graph based text-line segmentation method which combines both local and global approaches to first obtain an initial estimate of text-lines. In the next step, we use the distances along the nodes in the local-orientation graph to automatically detect touching and proximity errors. Expectation-maximization (EM) is then iteratively applied to split the touching lines. Finally, the error components are localized and cut to obtain a accurate estimate of text-lines. The main contribution of this work are a graph based error detection technique and an EM based correction technique which have shown promising improvements on our handwritten Arabic data set.

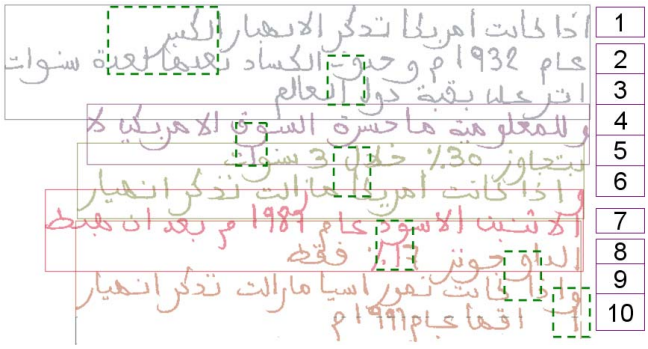


Figure 1. Text-line segmentation errors due to touching components across different lines. Line 1,2,3 are grouped as one segment due to touching components. Text-lines are color coded by the algorithm. Dotted boxes show the various touching component.

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

II. Related Work

Existing text-line extraction techniques can be broadly categorized as *projection based*, *component-grouping based* or *hybrid methods*. *Projection based* methods typically divide the document image into vertical strips, compute horizontal projection profiles to extract components and group them based on few heuristics to extract text-lines. In [3], components are grouped by modeling the text-lines as bivariate Gaussian densities. A recent method initially

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.31

Table I
COMPARISON OF THE PROPOSED METHOD WITH THE TOP EIGHT METHODS IN ICDAR 2009 SEGMENTATION COMPETITION.

Methods	CUBS	ILSP-LWSeg-09	PAIS	CMM	CASIA-MSTSeg	PortUniv	PPSL	LRDE	Proposed Method
F-Measure	99.5	99	98.5	98.4	95.6	94.5	93.4	92	97.8

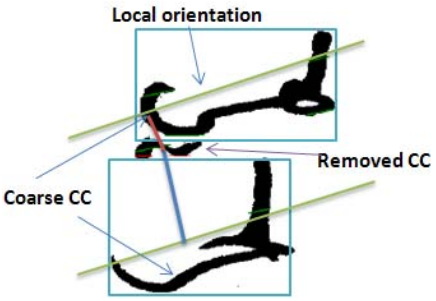


Figure 6. Ambiguity in assignment of diacritic and accent component is resolved based on distance to orientation line. CC refers to connected-component.

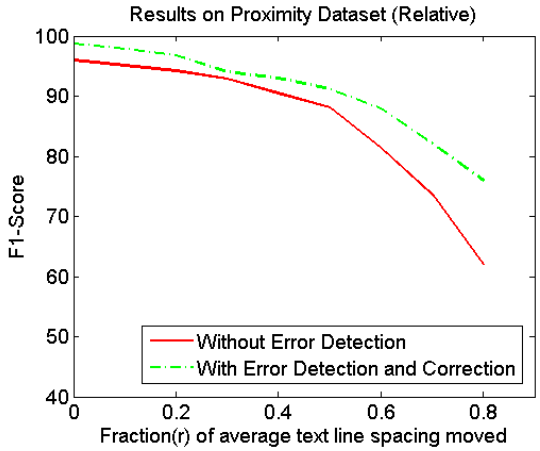


Figure 7. Plots of F-1 scores obtained using our method and a previous method which do not apply any touching error detection and correction method.

D. Assignment of Diacritic and Accent Components

In this final step we assign all the components which were removed as probable diacritic and accent components to the text-lines. Each such component is given the label of best matched coarse component closest to it. We give first preference to the coarse components whose bounding-box fully encloses the bounding-box of the removed component. Second preference is given to those components whose bounding-box overlaps with the removed component. In ambiguous cases, we resolve the ambiguity by computing the distances from the centroid of removed component to the orientation line detected at the coarse components and the component with the minimum distance is chosen (Figure 6).

IV. EXPERIMENTS AND EVALUATION

Our dataset consists of a set of 125 Arabic document images with 1974 handwritten text-lines. We generated a set of proximity datasets using these images to test the robustness of our approach. We moved each line closer to the line above it, in steps of some fixed fraction of average distance between the lines, to generate a series of datasets. We call this the *Relative* proximity dataset [11] which has 19740 text-lines.

We evaluated our results using a pixel-based matching-score(MS) criterion which is computed as follows:

$$MS(r_i, g_j) = \frac{T(P(r_i) \cap P(g_j))}{T(P(r_i) \cup P(g_j))} \tag{4}$$

where $MS(r_i, g_j)$ is a real number between 0 and 1 and represents the matching score between the result zone r_i and the ground truth zone g_j. P represents the foreground and T is an operator that counts the number of pixels in the zone. We obtain the matching-scores between all the result zones and the ground-truth zones. If the score is found above a pre-defined threshold then the result zone is counted as a True positive (TP). Result zones which are not matched to any ground truth zones are False positives (FP) and the ground-truth zones which are left unmatched are False-negatives (FN). We compute precision, recall and the F1-score as follows:

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1_{Score} = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

Figure 7 shows the F1 scores obtained using our method on the relative-proximity data at MS threshold of 90. As shown, we obtained a F1 score of 98.76% on the original data (Fraction r = 0). We compare our results with the results in [1] which did not apply touching error detection and correction. As text-lines are moved closer, the performance of the proposed approach does not degrade as rapidly as the other method. We observe an improvement of 2.76% in accuracy on the original data and 14% on the proximity data

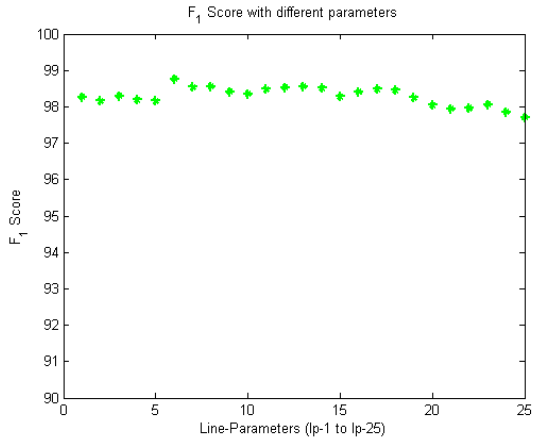


Figure 8. Plot of F-1 scores with different values of parameters.

at r = 0.8. Total number of touching components in the the proximity data at r = 0.8 is 604. If the two lines overlap with each other completely then our method breaks down and the accuracy falls substantially. But in real documents we rarely see all the components of two text-lines touching.

Figure 8, shows the F_1 scores for different sets of parameters. We varied the parameters $HBand_{thres}$ and $VBand_{thres}$ defining the dimension of rectangular region for local orientation computation. Values from 5 to 3 in steps of 0.5 for $HBand_{thres}$ and 0.8 to 0.4 in steps of 0.1 for $VBand_{thres}$ are used to create 25 sets of parameters (lp-1 to lp-25). As shown, our method is robust to these two parameters. We also evaluated our method on ICDAR 2009 segmentation competition dataset (200 images) [12] and F-Measures(FM) of top eight methods along with the proposed method is given in the Table I. Although the proposed method was developed for Arabic, it adapts well to other scripts like English, French, German and Greek used in the dataset. The average time taken for the processing of a single image is 2.2 seconds for the proximity data and 3.2 seconds for the ICDAR competition data on a P4 machine with 3BG RAM.

V. Conclusion

We have presented a graph-based approach for the extraction of handwritten text-lines in monochromatic document images. Using the local orientation graph we detect touching and proximity errors in our results. We then apply EM algorithm to correct errors. The proposed detection and correction scheme relies on the same graph used for clustering and does not add any computational overhead. Proposed method is fast due to the removal of small components in the first step. We demonstrated the effectiveness of our method on different datasets. We also showed the improvement in accuracies on a previous method [1] which did not use any touching detection and correction strategy. In general, our method can be used as a post-processing step in any CC-based method which gives an initial estimate of text-lines.

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] J. Kumar, W. Abd-Almageed, L. Kang and D. Doermann, *Handwritten Arabic Text Line Segmentation using Affinity Propagation*, Document Analysis Systems, pp. 135–142, 2010.

[2] U. Pal and S. Datta, *Segmentation of Bangla Unconstrained Handwritten Text,* Intl. Conf. on Document Analysis and Recognition, vol.2, pp. 1128-1132, 2003.

[3] M. Arivazhagan, H. Srinivasan, and S. Srihari, *A Statistical Approach to Line Segmentation in Handwritten Documents,* Volume 6500. SPIE, 2007.

[4] V. Papavassiliou, T. Stafylakis, V. Katsouros, G. Carayannis, *Handwritten Document Image Segmentation into Textlines and Words*, Pattern Recognition, Vol. 43, Issue 1, pp. 369–377, 2010.

[5] G. Louloudis, B. Gatos, C. Halatsis, *Text Line Detection in Unconstrained Handwritten Documents Using a Block-Based Hough Transform Approach,* Intl. Conf. on Document Analysis and Recognition, vol. 2, pp.599–603, 2007.

[6] Y. Li, Y. Zheng and D. Doermann, *Detecting Text Line in Handwritten Documents,* Intl. Conf. on Pattern Recognition, pp. 1030-1033, 2006.

[7] A. Zahour, B. Taconet, P. Mercy, S. Ramdane, *Arabic Handwritten Text-line Extraction,* Intl. Conf. on Document Analysis and Recognition, pp.281–285, 2001.

[8] B. J. Frey and D. Dueck, *Clustering by Passing Messages Between Data Points,* Science 315, pp. 972–976, 2007.

[9] A. P. Dempster, N. M. Laird and D. B. Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm.*, J. R. Statist. Soc. B, 39:1–38, 1977

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. *Introduction to Algorithms*, MIT Press and McGraw-Hill.

[11] DATASET: Handwritten Arabic Textline Segmentation and Proximity Datasets, Language and Media Processing Laboratory, Univeristy of Maryland College park, http://lampsrv02.umiacs.umd.edu/projdb/project.php?id=65

[12] B. Gatos, N. Stamatopoulos and G. Louloudis, *ICDAR2009 Handwriting Segmentation Contest*, Intl. Conf. on Document Analysis and Recognition, pp. 1393–1397, Barcelona, Spain, 2009

2012 10th IAPR International Workshop on Document Analysis Systems

Automatic Room Detection and Room Labeling from Architectural Floor Plans

Sheraz Ahmed*†, Marcus Liwicki*, Markus Weber*†, Andreas Dengel*†
* *German Research Center for AI (DFKI)*
Knowledge Management Department,
Kaiserslautern, Germany
{*firstname.lastname*}*@dfki.de*
† *Knowledge-Based Systems Group,*
Department of Computer Science, University of Kaiserslautern,
P.O. Box 3049, 67653 Kaiserslautern,Germany

***Abstract*—This paper presents an automatic system for analyzing and labeling architectural floor plans. In order to detect the locations of the rooms, the proposed systems extracts both, structural and semantic information from given floor plans. Furthermore, OCR is applied on the text layer to retrieve the meaningful room labeling. Finally, a novel post-processing is proposed to split rooms into several sub-regions if several semantic rooms share the same physical room. Our fully-automatic system is evaluated on a publicly available dataset of architectural floor plans. In our experiments, we could clearly outperform other state-of-the-art approaches for room detection.**

***Keywords*-floor plan analysis; structure analysis; architecture; wall detection; symbol spotting; room detection;**

I. Introduction

Image analysis and image understanding are considered as important areas of research in the pattern recognition community. Floor plan analysis can be viewed as a special case of image analysis and image understanding. In floor plan analysis the goal is to extract different structural and semantic aspects of a building by analyzing the 2D image of the floor plan. In the past, various efforts have been made to analyze a given floor plan for different purposes, e.g., [1], [2], [3] analyze the floor plans for generation of 3D models, [4] focused generation of corresponding CAD format for a given floor plan. In [5], [6] floor plan analysis is performed to detect rooms and their connectivity topology. Similarly, in [7] the aim is to enable the search in a large repository of floor plans.

This paper presents an extension of the work presented in [8] with an emphasis on semantic analysis. First, the semantics are extracted taking also the room labels into account in order to find the functions of the detected rooms. This information is finally used for correction of room detection errors.

The remainder of this paper is organized as follows. First, Section II briefly summarizes other work related to this paper. Second, Section III gives an overview of the proposed method and describes the specific processing steps in more detail. Subsequently, experimental results are described in Section IV. Finally, Section V concludes the paper and gives an outlook to future work.

II. Related Work

The specific task of floor plan analysis has been addressed already for more than 20 years. [4] proposed a method of interpreting a hand-sketched floor plan. This method focuses on understanding the hand sketched floor plan and converting it into a CAD representation. Similarly, [9] proposed a method for understanding hand drawn floor plans using subgraph isomorphism and Hough transform. [10] presented a complete system for the analysis of architectural diagrams. Numerous automated graphics recognition processes are applied for recognizing the basic primitives. Also human feedback is used throughout the analysis phase.

[5] proposed a method to detect rooms in the architectural floor plan images. This method is adopted and expanded in this paper. We introduce new processing steps like wall edges extraction, and boundary detection. The main application area of our approach is the retrieval of similar floor plans as described in [7], where only a simple room detection method has been applied. However, the methods can be applied to any application area in the context of architectural floor plans. [11] focused on detection of walls from floor plan image. These detected walls can be used for different purposes during the complete floor plan analysis like 3D reconstruction or building boundary construction.

III. Floor Plan Analysis System

The input data of our system is available in binary format.[1] Figure 1 depicts the complete flow of our floor plan analysis system which will be described in the following. The analysis process starts with fine segmentation which separate the various types of information from one another (see Section III-A). Leading to information segmentation is structural analysis which aims to retrieve the structure of the rooms (see Section III-B). Finally, a semantic analysis is applied to enhance the results of structural analysis

[1]The actual image size is 2479×3508. For making the analysis process more efficient, isotropic down scaling to 1413×2000 has been applied.

978-0-7695-4661-2/12 $26.00 © 2012 IEEE
DOI 10.1109/DAS.2012.22

doors, windows, or sometime at gates. To extract these edges convex/concave hypothesis [8] is used.

As a next step the gaps between the extracted edges are closed. Note, that here focus is to close only those gaps where windows or doors are likely to be found based on empirically defined thresholds T_{merge}. Boundary image is then generated using the thick lines image from information segmentation. Combining boundary image with the gaps closed walls image give us an overall structure of the complete building, where almost all the gaps on the outer walls as well as most of the gaps due to doors and windows inside the building are closed. Still the gaps greater then T_{merge} were not closed. To get the actual bounds of rooms it is necessary to close all the gaps, especially due to doors. The focus of this paper is on semantic analysis, for more details on structural analysis see [8]. In Section III-C semantic analysis is used to close these remaining gaps.

C. Semantic Analysis

The aim of semantic analysis is to extract the semantic information of the floor plan. Semantic analysis spots different building elements in the floor plan and interprets them with respect to their context. To close the remaining gaps from structural analysis, we apply a symbol spotting technique in order to detect the doors in the floor plan. The speeded up robust features (SURF) [14], which is a robust, translation, rotation, and scale invariant representation method is used to locate the door symbols in the floor plans.

Figure 1.11 shows the extracted positions of windows and doors. Note that some erroneous symbols have been extracted by our approach. At a later step these symbol positions are matched with the gaps found during wall edge detection. Only those results which overlap with edges are taken into account as actual doors. Figure 1.12 shows the image where the gaps at the doors are closed.

To detect the actual bounds of rooms, the image with the closed gaps is inverted and connected component analysis is performed on it. Each of the connected component is referred as room. The detected rooms can be found in Figure 1.12.

After detection of rooms the next step is to define there functions like WC, Living room etc. In order to find the function of each room, the text layer from the information segmentation as well as the connected component of the room is used. In particular, all text components which lie in the boundary of a room are taken into account. After extraction of the room text, horizontal and vertical smearing is performed on the extracted text to merge the neighboring characters, resulting in the bounds for words. Using the bounding boxes all the words are rotated to a horizontal direction and OCR is performed on them. The OCR[2] result is then compared to rooms title dictionary and the closest title according to the Levenshtein distance is assigned to the room. Note that before applying dictionary, all the digits and special characters are removed from the OCR result.

After assigning label to rooms, novel post-processing is performed. This post-processing splits rooms into several sub-regions based on detected labels. The rooms which do not have any physical partition may contain more than one label. Some of these labels represent the function of rooms whereas other refer to items in rooms, e.g., cupboard/lockers etc. Rooms which contain more than one function label are selected for further splitting, e.g., the detected room in Figure 2b has two function labels. To further split rooms into several regions we select one label and look for a label in its neighborhood. Both horizontal and vertical distances between the selected labels are calculated. If the horizontal distance between labels is greater, horizontal splitting is performed at the middle of both labels, otherwise vertical splitting is performed. This process of partitioning is repeated until all rooms have one label.

Splitting of room with more than one function labels into several regions is very subjective. The room in Figure 3 has more than one function labels, but still in ground truth it is marked as single room, whereas Figure 2 shows the case where the ground truth is divided into several rooms.

After splitting rooms into several sub-regions, the next step is to merge those regions which do not have any room label. Each region which do not have any label is merged with neighboring room which is aligned with it. Figure 1.15 shows the rooms after splitting and merging.

IV. Evaluation

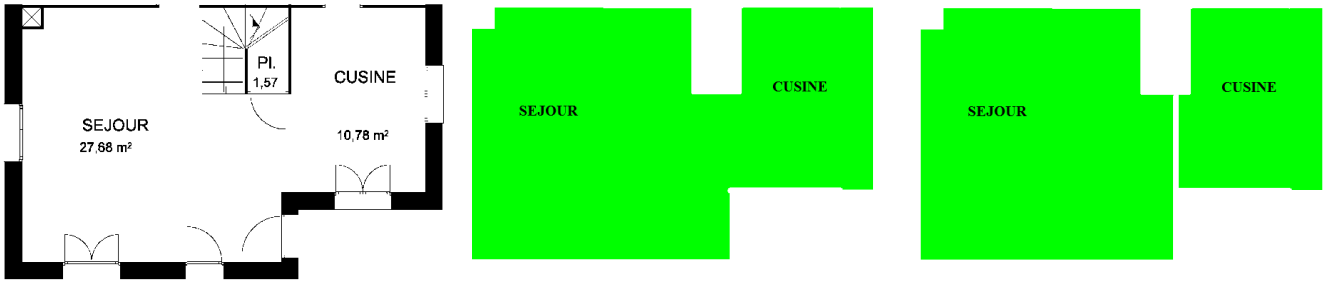
Our system is evaluated using a data set containing original floor plan images. This data set was introduced in [5] and contains the floor plan images from the period of more than ten years. The size of each floor plan image in the data set is $2,479 \times 3,508$. All floor plans are binarized to ensure that only structural information of the floor plans is used for the analysis (and not the color information).

In order to report the accuracy of our system, we use the protocol introduced by [15]. It allows reporting *exact match* (one to one) as well as *partial matches* (one to many and many to one). For further details refer to [15].

Table I shows the results of rooms detection over the series of 80 floor plan images dataset. The overall detection rate when no semantic division is performed is $89\,\%$ which is $4\,\%$ higher than the $85\,\%$ achieved in the reference system by [5]. More remarkably, the recognition accuracy has been improved by $10\,\%$. For around $20\,\%$ of the images we received the recognition accuracy and detection rate both greater then $90\,\%$. In the worst case, the recognition accuracy and detection rate of our system were still $50\,\%$ and $61.53\,\%$ respectively.

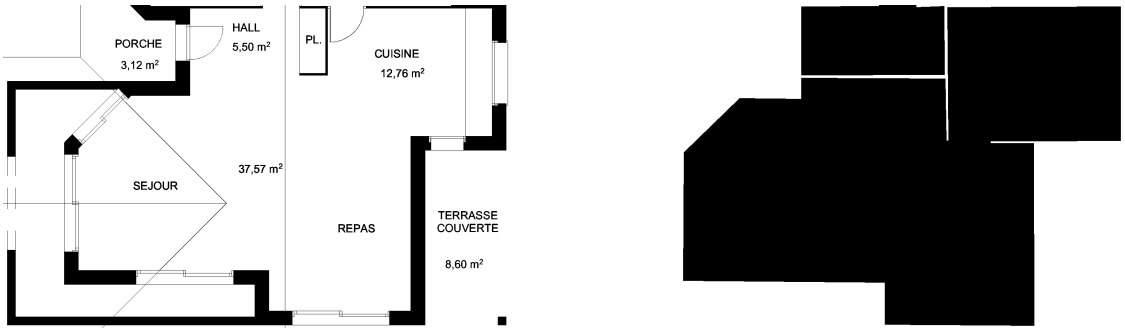
In case of semantic division the overall detection rate is the same as in [5], whereas recognition accuracy is improved

[2]Tesseract has been used to perform OCR



(a) (b) (c)

Figure 2: Room Spliting: Room part from original floor plan image(a) Room detection and labeling (b) Room Spliting using labeling results(c)



(a) (b)

Figure 3: Room Spliting: Ambigous case. Original image with more than one label(a) Ground truth with no partition(b)

by 13 %. In this case our detection rate is decreased from 89 % to 85 % because of the subjective nature of rooms which do not have any partition. Figure 3 shows the case where more function labels are present but still in ground truth it is marked as single room.

Further analysis of results in Table I reveals that our system has a good recognition accuracy and detection rate, along with less one to many count on average. This is because, a region is split in to sub region wherever a door or physical partition is found. Our one to many count is further reduced by performing the semantic division, because now the rooms which have more than one function are split in to several sub regions. Our many to one count is higher because of ambiguous nature of rooms with more than one label. To further reduce it floor plan ontologies can be used.

Table II shows the results of room labeling over the series of 80 floor plan images dataset. Manual evaluation is performed for room labeling, as no ground truth was available for it. The results show that more than 80 % of the rooms were correctly labeled. The mislabeling are some times due to the erronus results for the text which is touching some graphic components. To reduce this mislabeling some preprocessing is required before performing OCR.

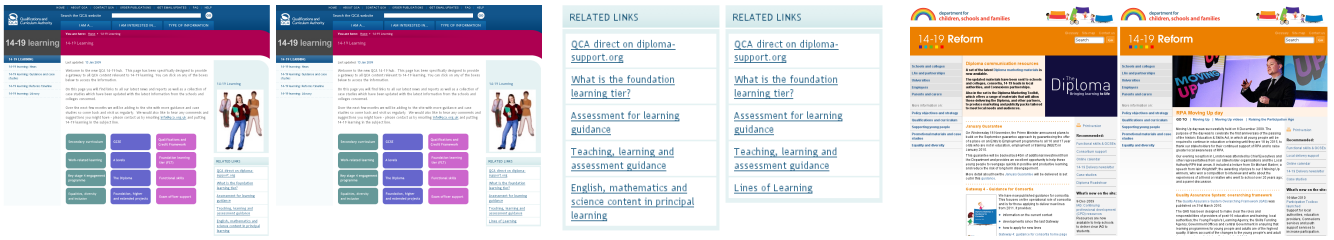
Room Labels	Numbers	Percentage(%)
Correct	736	82.33%
Wrong	158	17.67%
Total	894	100%

Table II: Room Labeling results

V. Conclusion and Future Work

A complete system for automatic room detection and room labeling from architectural floor plans was presented in this paper. The system applies several structural and semantic analysis steps in order to retrieve the room information. Furthermore, the system extracts the room labels to identify the functions of the rooms. The label information is also used to further split the room into functional rooms even if no physical segmentation exists.

Our system has been evaluated on a database from the literature. We outperform previous state-of-the-art methods and achieve a perfect recognition rate on several floor plans. Our experiments have shown that the proposed method works very well on a large corpus of 80 floor plans. In addition to the structural information, the text information is used to define the functions of room. Room labeling is done by using OCR results and rooms title dictionary. To improve



(a) Similar versions (b) zoom over the **only** difference between the versions of (a) (c) Dissimilar versions

Figure 1: Similar and dissimilar versions of Web pages. The versions of (a) share the same information, they are exactly similar except the links in (b), they do not need to be crawled twice. The versions of (c) have the same banner and menus but the main information of the page is changed, a second crawling is then necessary.

for content extraction [8], they use the relative positions between elements of pages but no visual appearance features. Additionally, we propose a machine learning framework to set all the similarity parameters and combination weights. We claim to get in this manner a semantic similarity close to archivists' attempts. Our contribution is three-fold: (1) a complete hybrid Web page comparison framework combining computer vision and structural comparison methods, (2) a new measure dedicated to Web archiving that only considers the visible part of pages without scrolling, (3) a machine learning based approach for supervised feature selection to increase prediction accuracy by eliminating noisy features.

2. WEB PAGE COMPARISON SCHEME

Two versions of a given Web page are considered similar if the changes that occurred between them are not important enough to archive both of them. They are dissimilar otherwise (see Figure 1). To compare versions of Web pages, we first extract features from them as described below.

2.1 Visual descriptors

Important changes between page versions will often produce differences between the visual rendering of those versions. We propose to quantify these differences by computing and comparing the visual features in each page version. Each version is described as an image of its rendering capture (snapshot). We compute a visual signature on this captured image for each page. Images are first described by color descriptors, because they seem appropriate for Web page changes and are already used in Phishing Web page detection [6]. We also incorporate powerful edge-based descriptors with SIFT descriptors [9] because they give state-of-the-art performances in real image classification tasks.

For image representation, we follow the well-known Bag of Words (BoW) representation [10, 11]. The vector representation of the rendered Web page is computed based on a sampling of local descriptors, coding and pooling over a visual dictionary. Recent comparisons for image classification point out the outstanding performances of a regular dense sampling [12, 13]. We apply a first strategy called *whole Web page* feature, that samples regularly the visual representation of the whole page. However, the most significant information is certainly not equally distributed over the whole captured Web page. As noted in [14], the most important information is generally located in the visible part of pages without scrolling. A second strategy called *Top of Web page* feature, provides a visual vector using only the features located in the visible part of the page without scrolling.

Since the visible part of a Web page without scrolling depends on the browser window size, we take a generic window height of 1000 pixels, greater than 90% of users' browser resolutions to ensure we do not miss information directly visible by most users. In the next sections, we will denote *the visible part of Web pages without scrolling* by *top of Web pages*.

2.2 Structural descriptors

We extract various features directly from the code of Web pages. For instance, we extract Jaccard indices [2], a similarity value that indicates the preservation between versions of hyperlinks and of URL addresses of images. We assume that similar pages tend to keep the same hyperlinks and images.

We also extract some features from the difference tree returned by the VI-DIFF algorithm [4] that detects some operations between the VIPS structures of versions, e.g. insertions, deletions or updates of VIPS blocks, or even a boolean value returning whether two versions have the same VIPS structure. The more operations are detected, the less similar versions are assumed to be. We denote the features extracted from the VI-DIFF algorithm by VI-DIFF features.

2.3 Similarity between versions

Let V^A be the last archived version of a Web page and V^N the new version of the same Web page. We extract several visual and structural descriptors (see sections 2.1 and 2.2), and use different metrics (Euclidian, χ^2 distances, *etc*) to compare them. Heuristics may be used to set them individually and to select the best similarity function with a manually-tuned threshold to discriminate dissimilar pairs of Web pages from the similar ones.

We propose here an alternate scheme embedding all the similarity functions into a learning framework. Let the M visual feature/metric associations and the N structural similarities be aggregated in a vector $\mathbf{x}$. We can write $\mathbf{x}^T$ as: $[s_v^1(V^A, V^N) \dots s_v^M(V^A, V^N), s_s^1(V^A, V^N) \dots s_s^N(V^A, V^N)]$.

We observed that none of the similarities we experimentally extracted presented a trivial individual decision boundary. However, all of them did seem to follow certain expected patterns, some of them working better than others. Instead of using them individually, we propose to combine those different similarities in a binary classification scheme that returns whether a couple of versions are similar or not by using $\mathbf{x}$, the vector of their similarity scores. Combining both approaches then seems appropriate to have a better understanding of the changes as perceived by human users. Learning combinations of complementary descriptors also makes the categorization task more efficient [15]. We investigate in the next section a statistical learning strategy based on a labeled dataset to classify the vectors $\mathbf{x}$.

3. CLASSIFICATION FRAMEWORK

We are interested in learning distances [16] between versions in a supervised framework to determine whether two versions are similar or not. However, it is not a version classification problem as in many distance learning problems [17]. Indeed, we do not want to classify samples (versions) but similarities. Moreover, our similarities are based on human judgement and allow subtilities as shown in Figure 1.

We then propose to express the learning of the combination of similarities as a binary classification in similarity space: for any couple of versions $(V^A, V^N)_i$, let their class $y_i = 1$ iff V^A and V^N are similar, -1 otherwise. Let $\mathbf{x}_i$ be a vector derived from heterogeneous similarities between V^A and V^N (as defined in subsection 2.3). We train a linear Support Vector Machine (SVM) to determine $\mathbf{w} = \sum_j \alpha_j y_j \mathbf{x}_j$ such that $\langle \mathbf{w}, \mathbf{x}_i \rangle = \sum_j \alpha_j y_j \langle \mathbf{x}_j, \mathbf{x}_i \rangle$ gives us the class of $(V^A, V^N)_i$. The similarity vectors $\mathbf{x}_j$ of training couples $(V^A, V^N)_j$ are used to train an SVM. For any test couple $(V^A, V^N)_i$, the trained SVM returns (1) whether $y_i = 1$ or $y_i = -1$, (2) whether V^A and V^N are similar or dissimilar, (3) whether V^N needs to be archived or not, with V^A already archived. Those three propositions are equivalent.

To study the contributions of the different types of features in the discrimination task, we first train a linear SVM with all the features. Each element w_k of $\mathbf{w}$ corresponds to the weight associated to the k-th similarity feature of $\mathbf{x}$. Therefore, if the learned w_k are close or equal to 0, the k-th similarity features of $\mathbf{x}$ are not determinant for categorization. Such similarities are considered noisy, irrelevant (not discriminant) in determining whether two versions are similar or not. To go one step further, we also propose a more explicit feature selection method based on the automatic *normal based feature selection* [18] that uses the fact that a feature k with the weight w_k close to 0 has a smaller effect on the prediction than features with large absolute values of w_k. Then features with small $|w_k|$ are good candidates for removal. The number of selected features may be set based on data storage and calculation constraints, or iteratively reduced using a validation set.

4. EXPERIMENT RESULTS

4.1 Dataset and settings

We work on a dataset of about 1000 pairs of Web pages manually annotated "similar" or "dissimilar" provided by *The Internet Memory Foundation*[1]. The pages are captured from many different governmental Web sites from the United Kingdom about education, health, sport, justice, industry, security... The identical couples of versions are removed and not taken into account in the evaluation. Finally, 202 pairs of Web pages were extracted: 147 and 55 (72.8% and 27.2%) couples of similar and dissimilar versions, respectively.

To compute visual similarities, we use SIFT and HSV (Hue Saturation Value) color descriptors with visual codebooks of sizes 100 and 200. These are relatively small compared to the sizes used on large image databases but consistent with the size of our base. Bigger codebook sizes did not improve our classification task. The BoWs of page versions are computed using the two strategies described in section 2.1: (1) over the rendering of whole Web pages and (2) the top of Web pages. Euclidian and χ^2 distances are then computed between the BoWs of successive page versions normalized using L^2-norm and L^1-norm, respectively. We also compute for each couple of page versions, the VIPS structures [3] and the VI-DIFF difference trees [4] from which we extract structural similarity values, e.g. the (symmetrized) ratio of identical nodes, boolean values on some criteria such as an identical VIPS structure. In the end, we have 16 visual and 25 structural features.

[1]http://internetmemory.org/

4.2 Binary classification

We use leave-one-out cross-validation (on the 202 pairs) to evaluate our model. We compare our results to the random classifier which automatically predicts the most represented class in the dataset, yielding a baseline accuracy of 72.8%.

Evaluation of visual features.

Selected Visual Features		Accuracy (%)
Whole Web page	Top of Web page	
None	SIFT	84.2
None	color	82.7
None	SIFT + color	**87.1**
SIFT	None	79.7
color	None	80.7
SIFT + color	None	83.2
SIFT + color	SIFT + color	85.1

Table 1: Visual feature classification performances.

We first use only the visual information of pages. Structural similarities of $\mathbf{x}$ are ignored. The accuracies when selecting different subsets of local descriptors (SIFT and color) sampled on whole pages or top of pages are presented in Table 1. SIFT and color descriptors achieve good performances for Web page change detection. Using the top of pages (87.1%) is also a lot more discriminant than using whole pages (83.2%). Combining both of them gives even worse results (85.1%) than using only the top of pages (87.1%). Important changes are more likely to be directly observable whereas changes at the bottom of Web pages, often advertisements, are more likely to be less important and noisy. The accuracies obtained validate our approach.

Evaluation of structural features.

Selected Structural Features		Accuracy (%)
Jaccard Indices	VI-DIFF	
Yes	No	85.1
No	Yes	76.7
Yes	Yes	**87.6**

Table 2: Structural feature classif. performances.

We study in Table 2 the accuracies when different subsets of structural similarities only are used. Jaccard Indices of links are the most discriminant structural features (85.1%) but the other structural features extracted from VI-DIFF are still informative, 4% better than the random classifier.

Structural and visual feature combination evaluation.

We investigate the combination of structural and visual features in Table 3. The accuracy when combining all of them (90.1%) is better than when using only structural (87.6%) or visual (87.1%) features. Visual and structural features are then complementary.

Segmentation and Normalisation in Grapheme Codebooks

Tara Gilliam, Richard C. Wilson, John A. Clark
Department of Computer Science
University of York
York, United Kingdom
{tg, wilson, jac}@cs.york.ac.uk

***Abstract*—The grapheme codebook is a high-performing technique for offline writer identification. This paper considers whether the de facto standards for initial grapheme extraction are optimal for both modern and historical datasets. We examine the construction and representation of the graphemes that comprise the codebook, testing three segmentation methods and two grapheme size normalisation methods on two datasets: a 93-writer IAM dataset, and a 43-writer medieval English dataset. The standard minima-split segmentation is compared to a complementary segmentation method that preserves ligature shapes, as well as the union of both these methods. Classification performance for each method is compared on a range of codebook sizes. We demonstrate that grapheme aspect-ratio is not always a writer-specific feature, and that preserving the character body shape in segmentation is more informative than preserving cursive text ligatures.**

***Index Terms*—Writer identification; Grapheme; Segmentation; Codebook**

I. Offline Writer identification

Given a set of documents written by a group of authors, the writer identification task attempts to label unseen documents with the correct writer. Offline writer identification is a refinement of this task that uses only static images of handwritten text as input and can be used in situations where no movement information is available, such as the analysis of historical documents.

Common feature extraction methods for offline writer identification include slant and edge-hinge distributions [1], [2], [3], run-length distributions [4], [1], Gaussian Mixture Models [5], texture features such as Gabor-based wavelets and filters [6], [7], [8], character-specific structural features [9], [10], and text-fragment shape distributions.

This last method operates by splitting the ink trace into fragments, and selecting a reference set of these to use in generating features for each image. Several variants exist: the most widespread is the grapheme codebook as described in [11], but similar ideas appear in the writer invariants method [12], and at the level of stroke fragments in [13], [14]. Advantages of this approach include high identification performance, text-independence, and automatic adaptation to the script used [2], [15].

The grapheme codebook method is an instance of the bag-of-words strategy for general image matching [16]. A major advantage of this specialisation is a natural and meaningful image segmentation which takes into account the writing

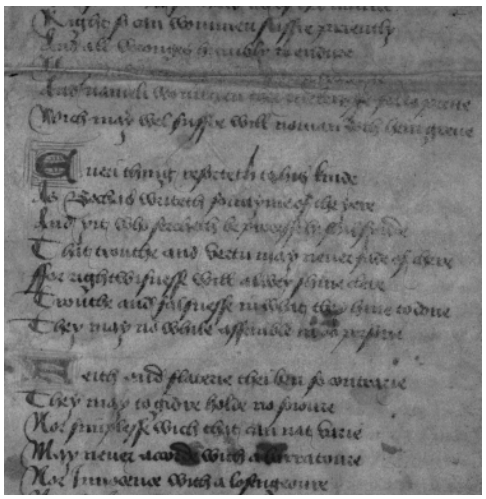


Figure 1. Sample image from the medieval scribes dataset

structure. The typical segmentation method used assumes a binary input image (black text on a white background), and breaks cursive writing heuristically on the vertical minima of the ink trace. This method was originally given in [17] for Optical Character Recognition and aims to produce the most character-like segments possible, but this occurs at the expense of breaking up the joins between them. Ghiasi and Safabakhsh observe that these joins, or ligatures, between characters contain writer-specific information which can be lost using standard segmentation. They propose an alternate method that combines different sizes of fixed-width segmentation, with good results [15].

After segmentation, graphemes can be represented as contours or bitmaps, with little impact on the algorithm performance [18]. Using bitmaps, the grapheme images are usually normalised in size to a uniform 50 x 50 pixels, preserving the aspect ratio. However, Schlapbach and Bunke find that some types of text size normalisation reduce identification accuracy [19]. Fornes et al. find that fixed-ratio normalisation in musical notation consistently gives a higher identification accuracy than normalisation that preserves aspect-ratio [20].

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.129

These results suggest that the typical approach to grapheme extraction may not always be optimal. This work therefore tests alternative methods for the two main aspects of grapheme extraction: segmentation from the cursive ink trace, and size normalisation for grapheme similarity matching. The identification accuracy of the codebook method with these modifications is tested on both a clean benchmark dataset (IAM) and a noisy historical dataset. This allows us to compare whether realistic data responds to these modifications in the same way as a benchmark dataset.

Two grapheme size normalisation methods are tested: square-ratio and aspect-ratio. The square-ratio method scales all graphemes to fill a 50 x 50 pixel square, while aspect-ratio scales by only the largest dimension, preserving the original height:width ratio of the grapheme. We also propose a variable-width segmentation method that complements the minima-split approach by preserving ligatures, and compare them against each other and the combination of both.

The following section describes the datasets used, the general grapheme codebook process, and the particular implementations and methodology used in these experiments.

II. Method and Data

The grapheme codebook method first splits the ink trace of an image into approximately character-level fragments, using some segmentation method. These fragments are called graphemes, and can be stored as either bitmap images or contour representations without loss of performance [18]. A reference set of graphemes is produced by selecting a subset of these – the codebook. Selection can be by Kohonen Self-organising Map [11], k-means clustering [18], or random selection [3]; overall identification accuracy is essentially independent of selection method [3].

The features for each image are formed by measuring the similarity of each of its graphemes to each of the codebook graphemes, and binning it against the closest match. The resulting probability distribution is the sample's feature vector; codebook size determines the dimensionality.

The work in this paper considers the initial process of segmenting and storing the graphemes, and the effect that this has on classification accuracy. In these experiments, codebook sizes of 50, 100, 150, 200, 250, and 500 were chosen for each experiment, as the results in [2] suggest performance peaks at codebooks of around 100 - 400.

Codebook graphemes were selected randomly from the total pool of graphemes generated for a dataset for the given normalisation/segmentation method combination. Grapheme similarity is measured using simple pixel-wise image correlation to generate the feature vectors and the Euclidean-distance nearest-neighbour algorithm is used for classification, which is performed on a leave-one-out basis.

Each experiment was run eight times, with a new set of random codebooks generated for each run, and the mean and standard error of the Top-1 classification accuracy are reported (plotted in Figures 3, 5, 7 and 8).

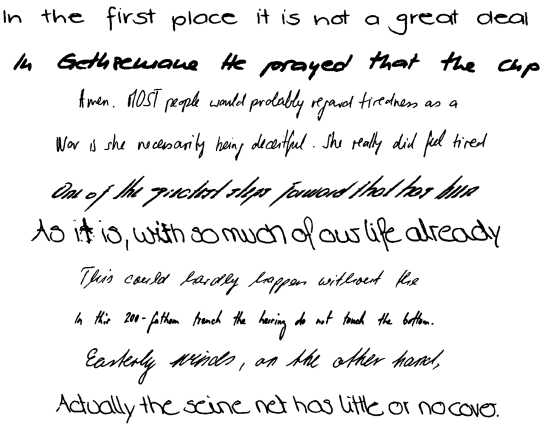


Figure 2. Sample IAM dataset text lines from [21]

To see how the proposed methods respond to varying sample size and noise levels, all experiments are run on two datasets (total 384 runs). The first is an IAM dataset of the 93 writers made available from the 100-writer identification set [22]. The images are greyscale, containing a single line of text segmented from a copied varied-text paragraph. Image noise is virtually non-existent – standardised recording forms were used, scans are uniform and high-quality, and text lines are cleanly separated, making it an excellent baseline for comparison (Figure 2). The IAM images were processed whole and binarised at a constant threshold using the ImageMagick library[1].

The second is a historical dataset containing approximately 400 full- and part-page images from Middle-English manuscripts, written by 43 scribes. There are between one and 52 images attributed to each scribe; identification of each image was provided by University of York Professor of Medieval English Palaeography, Linne Mooney. The dataset is very irregular, and image noise levels are high. The ink trace is often broken and faded, text lines can be curved or overlapping, and usually both ink and background vary in colour due to ageing or staining (Figure 1). Even where the document is well-preserved, the script within a page can change size, layout and font. The images also vary in size and resolution, from archival quality to samples from a handheld digital camera. In contrast to the IAM set, the medieval dataset is representative of the problems encountered in analysing real-world historical datasets, and required a greater level of processing. Selection and binarisation of the text areas was carried out manually on a per-image basis, with some images requiring additional noise removal. Where necessary, the binarised image was median-filtered to reduce holes in the ink trace. Unfortunately in some cases the original images are low-resolution, leading to some graphemes that are unavoidably jagged.

The remaining sections of this paper describe each experi-

[1] http://www.imagemagick.org/script/index.php

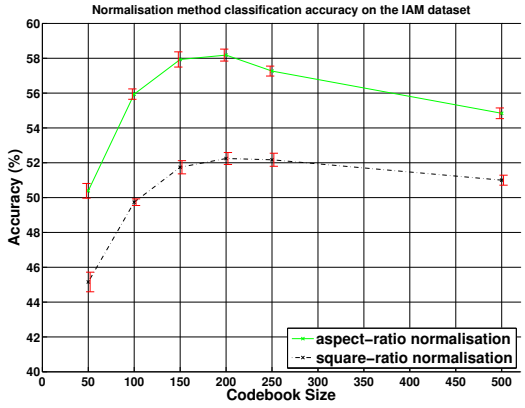


Figure 3. Normalisation results on the IAM dataset

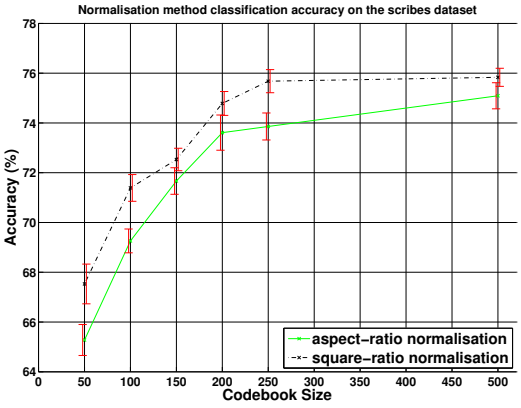


Figure 5. Normalisation results on the medieval dataset

ment and its results in some detail, before concluding.

III. Normalisation

Normalisation methods are specific to the representation of the graphemes, and are essential to allow comparison between writings which vary in size. This experiment considers two possible size normalisation options for grapheme bitmaps: fitting either a single dimension, or both dimensions, to 50 pixels. Figure 4 illustrates the horizontal (columns 1 & 2) and vertical (columns 3 & 4) stretching effect that square normalisation has over the natural aspect ratio of four graphemes from a medieval manuscript image. Preserving the aspect-ratio by scaling in only one dimension retains ratio information that may be writer-characteristic, and (by inspection) appears to be the de facto standard for bitmap normalisation in grapheme codebook experiments [23], [3]. However, some forms of constant scaling in both dimensions has been shown to be beneficial to the writer identification rate, e.g. increasing classification accuracy from 76% to 97% in [19], and providing the best results across all three feature extraction options in [20].

In this experiment, the standard minima segmentation was used to generate two sets of graphemes for each of the scribes and IAM datasets, one aspect-scaled and the other square-scaled. As described in Section II, reference codebooks were generated for each run by randomly drawing from the graphemes generated from the relevant dataset/normalisation combination only. The feature vector generation and classification was identical across experiments in all other respects.

Figure 4. Comparison of graphemes produced by the ratio (top) and square (bottom) grapheme size normalisation methods

Results: Figures 3 and 5 show the variation in Top-1 classification accuracy on each dataset, with error bars of +/- 1 standard error (plotted with some horizontal jitter for clarity). The normalisation experiments on the IAM dataset clearly show that aspect-ratio preservation performs better than fixed-dimension scaling. It produces a highly significant improvement in identification accuracy of 5–6 percentage-points, a substantial effect size equivalent to a boost of 7–11% over the square-scaled accuracy . This effect is fairly constant across all codebook sizes, and confirms that aspect-ratio in freehand Latin scripts carries writer-specific information.

On the medieval scribes dataset, the results are more inconclusive, but suggest that the square normalisation may offer a small (1–2 percentage-point) boost, i.e. aspect-ratio does not carry writer-specific information in this dataset. The reason for this may be that aspect-ratio in character formation is font-specific, rather than directly writer-specific.

Scribes did not typically write in a personal freehand style, but adopted fonts appropriate to the manuscript. These fonts are typical of particular periods and geographic areas, and have a largely fixed aspect-ratio. This implies that aspect is likely to be more strongly correlated with font than with writer in the scribes dataset, as it is limited to scripts produced during the medieval period in England.

The difference in baseline classification accuracy between the datasets is due to the differences in sample size: the scribes dataset images contain up to a page of text (an average of approximately 1000 graphemes), whereas the IAM dataset consists of text-line samples of around 35 graphemes.

Following these results, aspect-ratio normalisation was chosen for the segmentation experiments, as it gives the best overall performance across datasets.

IV. Segmentation

The second experiment compares segmentation heuristics, which determine how the cursive ink trace is split into usable

fragments. The standard method to this point aims to approximately divide it into characters, but as we are concerned only with the shape distributions generated by the writer and not the semantic content of the text, this is not a requirement.

In this experiment, the minima method is compared with its complement, which breaks in the centre of characters wherever possible in order to preserve the ligatures instead. Figure 6 shows the difference in splitting points on a connected section of ink trace for each of these methods.

The minima method has been implemented by inserting a vertical break through the minimum inflection points on the lower contour of the ink trace, if it additionally holds that:

- The ink trace height at that point is approximately one stroke-width
- The segmentation will produce a grapheme with a sensible minimum width (set at 5 pixels)

The stroke-width is estimated automatically per-document from the vertical and horizontal run-length distributions.

The assumption implicit in the minima splitting method is that the character body contains the writer-specific information. An alternative hypothesis is that the between-character ligatures contain writer-specific information, and should be preserved.

The implementation of the ligature method initially employs the same minima detection process, but splits instead at the midpoint between adjacent minima (and the connected-component boundaries where necessary). A notable effect of this process is that graphemes are no longer guaranteed to be connected-components themselves.

As these segmentation techniques are complementary, their combination is also tested. To do this, each image in the dataset is represented by the union of the bags of graphemes output by both methods: the raw image data is essentially duplicated, but each copy emphasises a different characteristic. Graphemes identical under both methods are included only once to avoid skewing the feature vector distributions in favour of single characters and small connected-components.

This method distinguishes between the cases where the two splitting methods produce redundant or complementary information: if there is an exact overlap in the information provided, the classification accuracy of the combination should approximately equal whichever single method (minima or ligature) is best. If the two methods are extracting different information, combining should give a classification accuracy greater than either method individually.

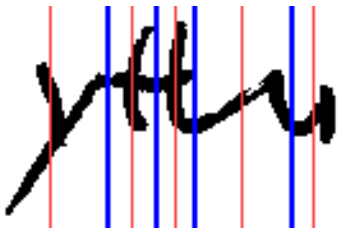


Figure 6. Comparison of splitting points produced by the minima (bold/blue line) and ligature (light/red line) segmentation methods

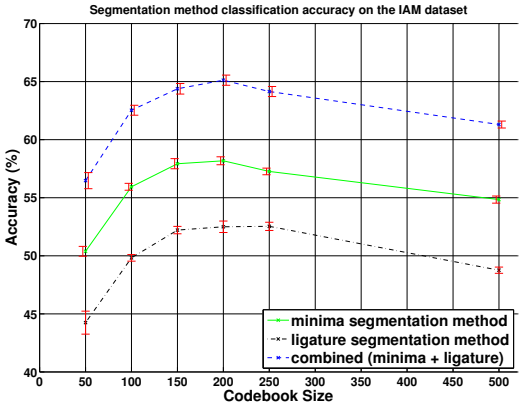


Figure 7. Segmentation results on the IAM dataset

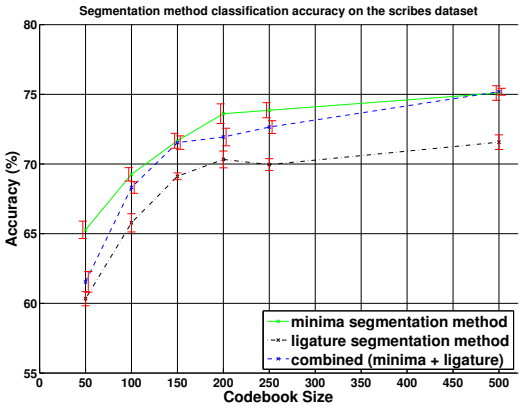


Figure 8. Segmentation results on the medieval dataset

Results: As before, Figures 7 and 8 show the Top-1 classification accuracy on each dataset, with error bars of +/- 1 standard error (plotted with some horizontal jitter for clarity). The segmentation results show that graphemes constructed preserving character ligatures do provide substantial writer-specific information, but the minima segmentation method performs significantly better on both datasets. Again, this result is much less clear on the scribes dataset due to noise effects.

On the IAM dataset, combining the output of both methods gives a significant performance boost, suggesting that the writer information extracted from character body and ligatures is independent to some degree. Identification accuracy for the combined methods increases by 5–6 percentage-points over the minima-segmentation method, and by 12–13 percentage-points over the ligature method. This reflects a substantial proportional accuracy increase of 12% and 25% respectively.

On the scribes dataset, the minima-split method significantly increases accuracy by 3–4 percentage-points, or 5–7% compared to the ligature method. This confirms that the body of the character preserves more writer-specific information

than a focus on the between-character ligatures. However in contrast to the IAM dataset, the combined method does not perform significantly differently to the best single-strategy approach. This may be due to the much larger number of graphemes already available per-image, as well as a greater natural variability in results.

V. Conclusions

This work has examined bitmap normalisation and segmentation methods for grapheme codebooks on two very different datasets. Preserving the aspect-ratio of freehand text was found to significantly improve classification accuracy by 7–11% compared to a grapheme size normalisation that discards this information. These results suggest that at the grapheme level, aspect ratio is a writer-specific feature in contemporary freehand writing. However, likely due to the geographic and period influences of font on historical manuscripts, this does not necessarily hold true of historical data: there is at best no increase in performance from aspect-ratio preservation.

In grapheme segmentation for both datasets, preserving solely the character body provides significantly more writer-specific information than preserving solely the between-character ligatures. This effect is greatest on the IAM dataset, with a performance difference of approximately 10%, compared to a difference of approximately 6% for the historical data. Combining multiple splitting methods produces a significant boost in accuracy on the small, clean IAM samples, but the high image noise levels typical of historical datasets may offset any practical gain.

Overall, the standard minima segmentation and aspect-ratio normalisation methods appear to perform well on clean benchmark datasets, but an improvement in identification accuracy can be made for small image samples by combining multiple segmentation methods. However on noisy historical data, the standard aspect-ratio normalisation may have a negative impact, and combining segmentation methods offers no improvement. We conclude that extraction methods appropriate for modern freehand benchmark datasets may not be optimal when applied directly to the increasing numbers of historical datasets in this area.

Future work will include examining the effects of varying sample sizes when combining segmentation methods.

Acknowledgement

The authors would like to thank Professor Linne Mooney of the Department of Medieval Studies, University of York, UK, for providing the data and scribal classifications for this work.

References

[1] M. Bulacu, L. Schomaker, and L. Vuurpijl, "Writer identification using edge-based directional features," in *7th International Conference on Document Analysis and Recognition*, 2003.

[2] M. Bulacu and L. Schomaker, "Text-Independent Writer Identification and Verification Using Textural and Allographic Features," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 701–717, 2007.

[3] L. J. P. van der Maaten, "Improving Automatic Writer Identification," Master's thesis, Maastricht University, 2005.

[4] B. Arazi, "Handwriting identification by means of run-length measurements," *IEEE Trans. Syst., Man and Cybernetics*, vol. 7, no. 12, pp. 878–881, 1977.

[5] A. Schlapbach and H. Bunke, "Off-lineWriter Identification Using Gaussian Mixture Models," in *18th International Conference on Pattern Recognition*, vol. 3, pp. 992–995, 2006.

[6] F. Shahabi and M. Rahmati, "A New Method for Writer Identification of Handwritten Farsi Documents," in *10th International Conference on Document Analysis and Recognition*, pp. 426–430, 2009.

[7] Z. He, B. Fang, J. Du, Y. Y. Tang, and X. You, "A novel method for offline handwriting-based writer identification," in *8th International Conference on Document Analysis and Recognition*, vol. 1, pp. 242–246, 2005.

[8] H. E. S. Said, K. D. Baker, and T. N. Tan, "Personal identification based on handwriting," in *Fourteenth International Conference on Pattern Recognition*, vol. 2, pp. 1761–1764, 1998.

[9] V. Pervouchine and G. Leedham, "Extraction and analysis of forensic document examiner features used for writer identification," *Pattern Recognition*, vol. 40, no. 3, pp. 1004–1013, 2007.

[10] B. Zhang, S. N. Srihari, and S. Lee, "Individuality of handwritten characters," in *7th International Conference on Document Analysis and Recognition*, pp. 1086–1090, 2003.

[11] L. Schomaker and M. Bulacu, "Automatic Writer Identification Using Connected-Component Contours and Edge-Based Features of Upper-Case Western Script," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 787–798, 2004.

[12] A. Nosary, L. Heutte, T. Paquet, and Y. Lecourtier, "Defining writer's invariants to adapt the recognition task," in *Document Analysis and Recognition, 1999. ICDAR '99. Proceedings of the Fifth International Conference on*, pp. 765–768, 1999.

[13] A. Seropian, M. Grimaldi, and N. Vincent, "Writer identification based on the fractal construction of a reference base," in *7th International Conference on Document Analysis and Recognition*, pp. 1163–1167, 2003.

[14] I. Siddiqi and N. Vincent, "Writer Identification in Handwritten Documents," in *9th International Conference on Document Analysis and Recognition*, vol. 1, pp. 108–112, 2007.

[15] G. Ghiasi and R. Safabakhsh, "An Efficient Method for Offline Text Independent Writer Identification," pp. 1245–1248, 2010.

[16] F.-F. Li and P. Perona, "A Bayesian Hierarchical Model for Learning Natural Scene Categories," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 524–531, 2005.

[17] R. G. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 7, pp. 690–706, 1996.

[18] M. Bulacu and L. Schomaker, "A Comparison of Clustering Methods for Writer Identification and Verification," in *8th International Conference on Document Analysis and Recognition*, (Washington, DC, USA), pp. 1275–1279, IEEE Computer Society, 2005.

[19] A. Schlapbach and H. Bunke, "Writer Identification Using an HMM-Based Handwriting Recognition System: To Normalize the Input or Not?," in *Proc. 12th Conf. of the Int. Graphonomics Society*, pp. 138–142, 2005.

[20] A. Fornés, J. Lladós, G. Sánchez, and H. Bunke, "On the Use of Textural Features for Writer Identification in Old Handwritten Music Scores," in *10th International Conference on Document Analysis and Recognition*, pp. 996–1000, 2009.

[21] U. V. Marti and H. Bunke, "Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition systems," *International Journal of Pattern Recognition and Artificial Intelligence*, pp. 65–90, 2002.

[22] A. Schlapbach and H. Bunke, "A writer identification and verification system using HMM based recognizers," *Pattern Analysis and Applications (PAA)*, vol. 10, no. 1, pp. 33–43, 2007.

[23] M. Bulacu and L. Schomaker, "Automatic handwriting identification on medieval documents," in *Proc. of 14th Int. Conf. on Image Analysis and Processing (ICIAP 2007)*, pp. 279–284, 2007.

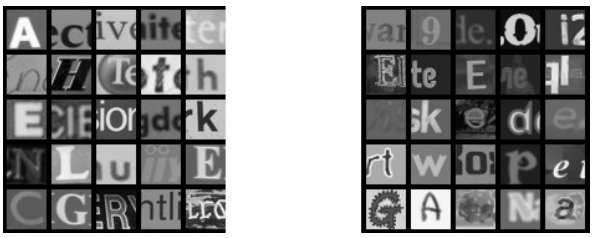


Figure 2. Examples from our training set. Left: from ICDAR. Right: synthetic data

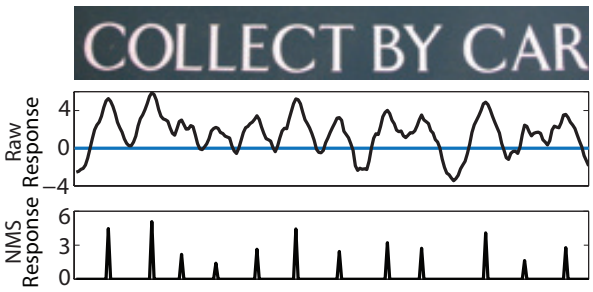


Figure 3. Detector responses in a line.

detector across a full scene image to identify candidate lines of text, on which we perform word-level segmentation and recognition to obtain the end-to-end results.

For both detection and recognition, we use a multi-layer, convolutional neural network (CNN) similar to [8, 16]. Our networks have two convolutional layers with n_1 and n_2 filters respectively. The network we use for detection with $n_1 = 96$ and $n_2 = 256$ is shown in Figure 1, while a larger, but structurally identical one ($n_1 = 115$ and $n_2 = 720$) is used for recognition.

We train the first layer of the network with an unsupervised learning algorithm similar to [2, 3]. In particular, given a set of 32-by-32 grayscale training images[1] as illustrated in Figure 2, we randomly extract m 8-by-8 patches, which are contrast normalized and ZCA whitened [6] to form input vectors $x^{(i)} \in \mathbb{R}^{64}, i \in \{1, ..., m\}$. We then use the variant of K-means described in [2] to learn a set of low-level filters $D \in \mathbb{R}^{64 \times n_1}$. For a single normalized and whitened 8-by-8 patch x, we compute its first layer responses z by performing inner product with the filter bank followed by a scalar activation function: $z = \max\{0, |D^{\mathsf{T}}x| - \alpha\}$, where $\alpha = 0.5$ is a hyperparameter.

Given a 32-by-32 input image, we compute z for every 8-by-8 sub-window to obtain a 25-by-25-by-n_1 first layer response map. As is common in CNNs, we average pool over the first layer response map to bring its dimensions to 5-by-5-by-n_1. We stack another convolution and average pooling layer on top of the first layer to obtain a 2-by-2-by-n_2 second layer response map. These outputs are fully connected to the classification layer. We discriminatively train the network by back-propagating the L_2-SVM classification error,[2] but we fix the filters in the first convolution layer (learned from K-means). Given the size of the networks, fine-tuning is performed using multiple GPUs.

[1] Our dataset consists of examples from the ICDAR 2003 training images [10], the English subset of the Chars74k dataset [4], and synthetically generated examples.

[2] In the form of a squared hinge loss: $\max\{0, 1 - \theta^{\mathsf{T}}x\}^2$.

3 End-to-End Pipeline Integration

Our full end-to-end system combines a *lexicon* with our detection/recognition modules using post-processing techniques including NMS and beam search. Here we assume that we are given a lexicon (a list of tens to hundreds of candidate words) for a particular image. As argued in [18], this is often a valid assumption as we can use prior knowledge to constrain the search to just certain words in many applications. The pipeline mainly involves the following two stages:

(i) We run sliding window detection over high resolution input images to obtain a set of candidate lines of text. Using these detector responses, we also estimate locations for the spaces in the line.

(ii) We integrate the character responses with the candidate spacings using beam search [15] to obtain full end-to-end results.

First, given an input image, we identify horizontal lines of text using multiscale, sliding window detection. At each scale s, we evaluate the detector response $R_s[x, y]$ at each point (x, y) in the scaled image. As shown in Figure 3, windows centered on single characters at the right scale produce positive $R_s[x, y]$. We apply NMS [13] to $R_s[x, r]$ in each individual row r to estimate the character locations on a horizontal line. In particular, we define the NMS response

$$\tilde{R}_s[x,r] = \begin{cases} R_s[x,r] & \text{if } R_s[x,r] \geq R_s[x',r], \\ & \quad \forall x' \; s.t. \; |x'-x| < \delta \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where δ is some width parameter. For a row r with non-zero $\tilde{R}_s[x, r]$, we form a line-level bounding box L_s^r with the same height as the sliding window at scale s. The left and right boundaries of L_s^r are defined as $\min(x)$ and $\max(x)$, s.t. $\tilde{R}_s[x, r] > 0$. This yields a set of possibly overlapping line-level bounding boxes. We score each box by averaging the nonzero values of $\tilde{R}_s[x, r]$. We then apply standard NMS to remove all

L's that overlaps by more than 50% with another box of a higher score, and obtain the final set of line-level bounding boxes $\tilde{\mathbf{L}}$. Since gaps between words produce sharply negative responses, we also estimate possible space locations within each L_s^r by applying the same NMS technique as above to the negative responses.

After identifying the horizontal lines of text, we jointly segment the lines of text into words and recognize each word in the line. Given a line-level bounding box L and its candidate space locations, we evaluate a number of possible word-level bounding boxes using a Viterbi-style algorithm and find the best segmentation scheme using a beam search technique similar to [9]. To evaluate a word-level bounding box B, we slide the character recognizer across it and obtain a $62 \times N$ score matrix M, where N is the number of sliding windows within the bounding box. Intuitively, a more positive $M(i, j)$ suggests a higher chance that the character with index i is centered on the location of the j^{th} window. Similar to the detection phase, we perform NMS over M to select the columns where a character is most likely to be present. The other columns of M are set to $-\infty$. We then find the lexicon word w^* that best matches a score matrix M as follows: given a lexicon word w, compute the alignment score

$$S_M^w = \max_{l^w \in L^w} \left(\sum_k^{|w|} M(w_k, l_k^w) \right) \tag{2}$$

where l^w is the alignment vector[3] between the characters in w and the columns of M. S_M^w can be computed efficiently using a Viterbi-style alignment algorithm similar to [17].[4] We compute S_M^w for all lexicon words and label the word-level bounding-box B with the highest scoring word w^*. We take $S_B = S_M^{w^*}$ to be the *recognition score* of B.

Having defined the recognition score for a single bounding box, we can now systematically evaluate possible word-level segmentations using beam search [15], a variant of breadth first search that explores the top N possible partial segmentations according to some heuristic score. In our case, the heuristic score of a candidate segmentation is the sum of the S_B's over all the resulting bounding boxes in a line of text L. In order to deal with possible false positives from the text detection stage, we threshold individual segments based on their recognition scores. In that way, segments with low recognition scores are pruned out as being "non-text."

[3]For example, $l_4^w = 6$ means the 4^{th} character in w aligns with the 6^{th} column of M, or the 6^{th} sliding window in a line of text.

[4]In practice, we also augment S_M^w with additional terms that encourage geometric consistency. For example, we penalize character spacings that are either too narrow or vary a lot within a single word.

Table 1. Cropped word recognition accuracies on ICDAR 2003 and SVT

Benchmark	I-WD-50	I-WD	SVT-WD
Our approach	**90%**	**84%**	70%
Wang, *et al*. [18]	76%	62%	57%
Mishra, *et al*. [11]	82%	-	**73%**

4 Experimental Results

In this section we present a detailed evaluation of our text recognition pipeline. We measure cropped character and word recognition accuracies, as well as end-to-end text recognition performance of our system on the ICDAR 2003 [10] and the Street View Text (SVT) [18] datasets. Apart from that, we also perform additional analysis to evaluate the importance of model size on different stages of the pipeline.

First we evaluate our character recognizer module on the ICDAR 2003 dataset. Our 62-way character classifier achieves state-of-the-art accuracy of 83.9% on cropped characters from the ICDAR 2003 test set. The best known previous result on the same benchmark is 81.7% reported by [2]

Our word recognition sub-system is evaluated on images of perfectly cropped words from the ICDAR 2003 and SVT datasets. We use the exact same test setup as [18]. More concretely, we measure word-level accuracy with a lexicon containing all the words from the ICDAR test set (called I-WD), and with lexicons consisting of the ground truth words for that image plus 50 random "distractor" words added from the test set (called I-WD-50). For the SVT dataset, we used the provided lexicons to evaluate the accuracy (called SVT-WD). Table 1 compares our results with [18] and the very recent work of [11].

We evaluate our final end-to-end system on both the ICDAR 2003 and SVT datasets, where we locate and recognize words in full scene images given a lexicon. For the SVT dataset, we use the provided lexicons; for the ICDAR 2003 dataset, we used lexicons of 5, 20 and 50 distractor words provided by the authors of [18], as well as the "FULL" lexicon consisting of all words in the test set. We call these benchmarks I-5, I-20, I-50 and I-FULL respectively. Like [18], we only consider alphanumeric words with at least 3 characters. Figure 5 shows some sample outputs of our system. We follow the standard evaluation criterion described in [10] to compute the precision and recall. Figure 4 shows precision and recall plots for the different benchmarks on the ICDAR 2003 dataset.

As a standard way of summarizing results, we also

Figure 5. Example output bounding boxes of our end-to-end system on I-FULL and SVT benchmarks. Green: correct detections. Red: false positives. Blue: misses.

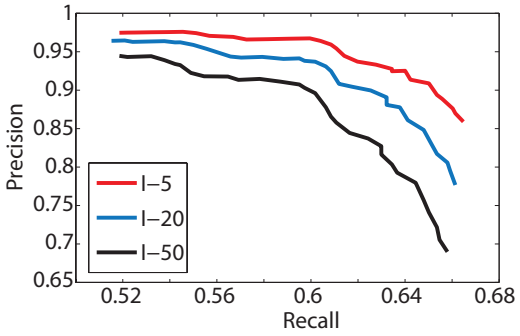


Figure 4. End-to-end PR curves on ICDAR 2003 dataset using lexicons with 5, 20, and 50 distractor words.

report the highest F-scores over the PR curves and compare with [18] in Table 2. Our system achieves higher F-scores in every case. Moreover, the margin of improvement is much higher on the harder benchmarks (0.16 for I-FULL and 0.08 for SVT), suggesting that our system is robust in more general settings.

In addition to settings with a known lexicon, we also extend our system to the more general setting by using a large lexicon $\mathbb{L}$ of common words. Since it is infeasible to search over all words in this case, we limit our search to a small subset $P \in \mathbb{L}$ of "visually plausible" words. We first perform NMS on the score matrix M across positions and character classes, and then threshold it with different values to obtain a set of raw strings. The raw strings are fed into Hunspell[5] to yield a set of suggested words as our smaller lexicon P, Using this simple setup, we achieve scores of 0.54/0.30/0.38 (precision/recall/F-score) on the ICDAR dataset. This

[5]Hunspell is an open source spell checking software available at http://hunspell.sourceforge.net/. We augment its default lexicon with a corpus of English proper names to better handle text in scenes.

Table 2. F-scores from end-to-end evaluation on ICDAR 2003 and SVT datasets.

Benchmark	I-5	I-20	I-50	I-FULL	SVT
Our approach	**.76**	**.74**	**.72**	**.67**	**.46**
Wang, *et al.* [18]	.72	.70	.68	.51	.38

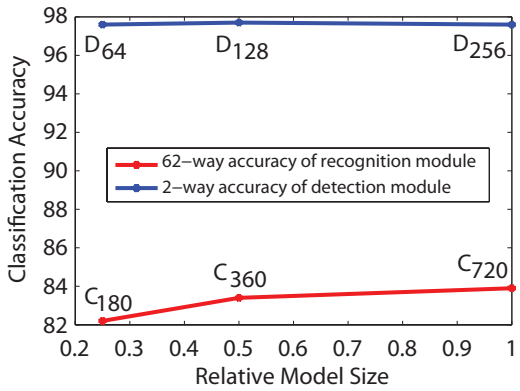


Figure 6. Accuracies of the detection and recognition modules on cropped patches

is comparable to the best known result 0.42/0.39/0.40 obtained with a general lexicon by [14].

In order to analyze the impact of model size on different stages of the pipeline, we also train detection and recognition modules with fewer second layer convolutional filters. The detection modules have $n_2 = 64$ and 128 compared to 256 in our full model. We call the detection modules D_{64}, D_{128} and D_{256} respectively. Similarly, we call the recognition modules C_{180}, C_{360} and C_{720}, which corresponds to $n_2 = 180$, 360 and 720. The smaller models have about $^1/_4$ and $^1/_2$ number of learnable parameters compared to the full models.

To evaluate the performance of the detection mod-

Table 3. Classification and end-to-end results of different recognition modules

Recognition module	C_{180}	C_{360}	C_{720}
Classification accuracy	82.2%	83.4%	83.9%
End-to-end F-score	.6330	.6333	.6723

ules, we construct a 2-way (character vs. non-character) classification dataset by cropping patches from the ICDAR test images. The recognition modules are evaluated on cropped characters only. As shown in Figure 6, the 62-way classification accuracy increases as model size gets larger, while the 2-way classification results remain unchanged. This suggests that larger model sizes yield better recognition modules, but not necessarily better detection modules.

Finally, we evaluate the the 3 different recognition modules on the I-FULL benchmark, with D_{256} as the detector for all 3 cases. The end-to-end F-scores are listed against the respective classification accuracies in Table 3. The results suggests that higher character classification accuracy does give rise to better end-to-end results. This trend is consistent with the findings of [12] on house number recognition in natural images.

5 Conclusion

In this paper, we have considered a novel approach for end-to-end text recognition. By leveraging large, multi-layer CNNs, we train powerful and robust text detection and recognition modules. Because of this increase in representational power, we are able to use simple non-maximal suppression and beam search techniques to construct a complete system. This represents a departure from previous systems which have generally relied on intricate graphical models or elaborately hand-engineered systems. As evidence of the power of this approach, we have demonstrated state-of-the-art results in character recognition as well as lexicon-driven cropped word recognition and end-to-end recognition. Even more, we can easily extend our model to the general-purpose setting by leveraging conventional open-source spell checkers and in doing so, achieve performance comparable to state-of-the-art.

References

[1] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. High performance neural networks for visual object classification. Technical Report IDSIA-01-11, Dalle Molle Institute for Artificial Intelligence, 2011.

[2] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *ICDAR*, 2011.

[3] A. Coates, H. Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011.

[4] T. E. de Campos, B. R. Babu, and M. Varma. Character recognition in natural images. In *VISAPP*, 2009.

[5] B. Epshtein, E. Oyek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *CVPR*, 2010.

[6] A. Hyvarinen and E. Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[7] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

[8] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

[9] C.-L. Liu, M. Koga, and H. Fujisawa. Lexicon-driven segmentation and recognition of handwritten character strings for japanese address reading. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(11):1425–1437, Nov. 2002.

[10] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. ICDAR 2003 robust reading competitions. *ICDAR*, 2003.

[11] A. Mishra, K. Alahari, and C. V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *CVPR*, 2012.

[12] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[13] A. Neubeck and L. Gool. Efficient non-maximum suppression. In *ICPR*, 2006.

[14] L. Neumann and J. Matas. A method for text localization and recognition in real-world images. In *ACCV*, 2010.

[15] S. J. Russell, P. Norvig, J. F. Candy, J. M. Malik, and D. D. Edwards. *Artificial intelligence: a modern approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[16] Z. Saidane and C. Garcia. Automatic scene text recognition using a convolutional neural network. In *Workshop on Camera-Based Document Analysis and Recognition*, 2007.

[17] S. Sarawagi and W. W. Cohen. Semi-markov conditional random fields for information extraction. In *NIPS*, pages 1185–1192, 2004.

[18] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *ICCV*, 2011.

[19] J. J. Weinman, E. Learned-Miller, and A. R. Hanson. A discriminative semi-markov model for robust scene text recognition. In *ICPR*, Dec. 2008.

Skeleton Comparisons

The Junction Neighbourhood Histogram

Jannis Stoppe
University of Bremen
Am Fallturm 1
D-28359 Bremen
jannis@informatik.uni-bremen.de

Björn Gottfried
University of Bremen
Am Fallturm 1
D-28359 Bremen
bg@informatik.uni-bremen.de

ABSTRACT

For analysing and comparing characters, using skeletons is a promising approach due to their topology-preserving nature and the resemblance of the skeleton to the original writing movement. We suggest a novel qualitative approach to skeleton comparison that is based on the adjacency of junctions and end points and the steps of a preceding skeleton simplification. By using a multi-dimensional histogram that contains information about the adjacency and the degree of joints, we gain high comparison speeds which, when combined with the multi-step approach, can be used for a generic topology distance metric.

Categories and Subject Descriptors

I.4.7 [**Image Processing and Computer Vision**]: [Feature Measurement - Size and shape]

General Terms

Algorithms

Keywords

Skeletons, Characters, Comparison, Similarity

1. INTRODUCTION

Skeletons are well-suited for the analysis of characters and text, as they focus on representing topological features, which are the major type of feature to classify characters. However, there are no straightforward ways to process both, raster based skeletons and graph based representations, concerning the topology and their comparison with other skeletons, which are suitable for fast processing of vast datasets of characters. In this paper, we present a method to efficiently compare skeleton topologies and make this comparison more robust to small changes in the topology based on the chosen skeletonisation algorithm or slight shape differences.

2. SKELETON SIMPLIFICATION

Skeletons consist of those points in a shape that are equidistant to two or more borders. As such, they represent a shape's topology. We do believe that such a topology based information is well suited for the analysis of text of all kinds, especially handwritings. Several factors (such as pen tip shape) that are not part of the actual information to be extracted are inherently removed during the skeletonisation process.

However, as raster based skeletonisation approaches usually result in a too detailed skeleton, we use the approach suggested in [11] to simplify the given raster skeletons first. As the suggested approach simplifies the skeleton iteratively, the question remains when to stop the process and use the current skeleton for further analysis or comparison. As each simplification step also results in an error level that indicates the appearance's difference between the simplified skeleton and the original one, this value can be used to specify a threshold at which the simplification process should be stopped. However, two similar shapes do not neccessarily have the same simplified skeletons when using the same simplification threshold (see fig. 1).

Our approach to solve this problem is to use *all* resulting simplification stages instead of just one predefined final stage.

In addition, we added a second error metric to the calculation of skeleton errors. In [11], the only measure suggested was the distance from the original pixels to the skeleton bones representing them. In some cases, this leads to very long bones which run parallel to each other, distorting the topology. Our solution is to add a penalty for bones that are longer or shorter than the pixels they represent. In general, the "length" of a pixel could be anything between 1 and $\sqrt{2}$. If a bone exceeds or falls below this range (with its lower and upper bound being multiplied by the amount of pixels the bone represents), the difference between the bone's length and this range is added to the skeleton's error level (see eq. 1). Consider P the set of points $p_0 \dots p_n$ that are assigned to a bone. The error level of a bone then receives an additional penalty of p (see eq. 1). The $weight$ is an additional factor that allows coordinates to be assigned to several bones: as our algorithm creates a graph that has its nodes places directly on the coordinates of pixels, the original coordinates usually cannot be attached to a bone unambiguously. Instead of picking one bone, we assigned the coordinate to all adjacent bones, with the sum of all weights being 1, resulting in a more predictable and consistend behaviour of the simplification.

Figure 1: Two similar shapes, simplified with given threshold 1, resulting in two different skeleton topologies.

DocEng'11 September 19-22, 2011, Mountain View, California, USA.

A Table Detection Method for Multipage PDF Documents via Visual Seperators and Tabular Structures

Jing Fang, Liangcai Gao[a]
Institute of Computer Science & Technology,
Peking University, Beijing, China
fangjing, gaoliangcai @icst.pku.edu.cn

Kun Bai
IBM Research T.J. Watson Research Center
19 Skyline Dr., Hawthorne NY, USA 10532
kunbai@us.ibm.com

Ruiheng Qiu
State Key Laboratory of Digital Publishing Technology,
Beijing, China
qiurh@founder.com.cn

Xin Tao, Zhi Tang
Institute of Computer Science & Technology,
Peking University, Beijing, China
taoxin, tangzhi@icst.pku.edu.cn

***Abstract*—Table detection is always an important task of document analysis and recognition. In this paper, we propose a novel and effective table detection method via visual separators and geometric content layout information, targeting at PDF documents. The visual separators refer to not only the graphic ruling lines but also the white spaces to handle tables with or without ruling lines. Furthermore, we detect page columns in order to assist table region delimitation in complex layout pages. Evaluations of our algorithm on an e-Book dataset and a scientific document dataset show competitive performance. It is noteworthy that the proposed method has been successfully incorporated into a commercial software package for large-scale Chinese e-Book production.**

Keywords- table detection; table spotting; PDF documents; separators; ruling lines

I. INTRODUCTION

Table, as an efficient and compact means to present data and two-dimensional relationship information, has been widely used in different kinds of documents. Table recognition has become an important task in the field of document structure recognition, and has attracted a good number of researches in the past two decades.

The most straightforward motivation of our work stems from the requirement of electronic book (e-Book) reading on handheld devices. The ubiquitous e-Books are part of the next generation web, which has promoted development of web publishing industry. Along with the advancement of wireless network and mobile technology, handheld devices (e.g. smartphones, Kindle, iPad) have rapidly gained popularity and become a common platform for rendering e-Books. Due to the limited screen sizes of these devices, e-Book documents usually need to be re-flowed and re-composed to provide readers a friendly reading experience.

Firstly, the table regions should be detected and separated from other elements within a page. Then they could be isolated as independent objects to be rendered on handheld device screens. In a further step, the detailed structure of tables, i.e. columns, rows and cells should be recognized. In this way, an oversize table can be laid out in several continuous screen pages, or users can hide certain columns or rows according to their reading preference. In this paper, we focus on the first and the most crucial step--table boundary detection. Structure recognition will be discussed in future work.

As an open document format, PDF has been exploited extensively in web publishing. However, existing approaches for table detection in image-based documents do not work reliably on PDF files, whose internal and low-level information can be mined deeply. This paper proposed a table detection method targeting at PDF documents. The method has been incorporated into a commercial software of Founder Corporation[b], and has been used to detect and re-flow tables in about two millions of e-Books in the last several months.

The rest of the paper is organized as follows. Section II reviews several relevant studies in table detection, especially those on PDF files. Section III firstly gives an overview of the proposed solution and then presents each step in detail. Section IV demonstrates the experimental results. Conclusion and future work are included in Section V.

II. RELATED WORKS

A good number of research efforts have been made on table detection so far. Among them surveys provided by Zanibbi et al. [1] and Silva et al.[2] both summarized table detection approaches in detail.

However, the majority of researches on table detection concentrated on image-based documents, such as one of the pioneering works T-Rect and T-Rect++ systems proposed by Kieninger et al.[3, 4]. Although PDF format, as a higher level of document representation, becomes increasingly important, there is much less prior works toward PDF documents (e.g. [5-8]).

pdf2table system, proposed by Yildiz et al.[5], is the first relative research carried out on PDF documents, performing two tasks: table detection and table decomposition. The table regions were spotted by detecting and merging multi-lines with more than one text segments. Similarly, Oro et al.[7] classified the lines into three classes: text lines, table lines and unknown lines, according to the number of

[a] Liangcai Gao is the corresponding author
[b] http://www.apabi.cn/English/index_en.html

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.304

2) Table Detection Solution

a) Ruled tables

According to people's reading experience, if only horizontal ruling lines are detected, the table columns are often delimitated by obvious vertical white spaces. Therefore, we rank the horizontal graphic lines and begin with the longest one to examine whether it crosses multiple vertical whitespace separators. If so, the longest vertical white space will be treated as the table height, and the width of this horizontal line represents the table width. All the other lines and vertical white spaces inside this table region are removed from the candidate sources to avoid forming overlap tables. The termination condition is that there is no source graphic lines exist. Otherwise, if both horizontal and vertical graphic lines exist, the same method is executed as outlined above, using vertical lines instead of vertical whitespace delimiters.

b) Unruled tables

In real-world documents, completely unruled tables are rare but still exist (see example in Fig. 4). Those tabular data are usually typesetted regularly with clear whitespace delimiters, not only in vertical direction but also in horizontal direction. Whitespace delimiters not only distinguish different tables in multi-table pages but also separate tables from other logical components. Please note that we only keep the horizontal white spaces wider than the dominant interline spacing. Then the similar step can be carried out as we did for the ruled tables. In addition, since the white spaces recognized are not as accurate and creditable as graphic ruling lines to determine the wide range of tables, we create a constraint that if the spotted table crosses the page columns, it should be segmented along the column space.

3) Post processing

Due to the versatility of table formats in real-world documents, false positive tables are inevitable in the existing detection approaches. In our method, they are mainly caused by graphic figures and matrix formulas.

After examining the false positive tables, we find that although figures contain graphic lines and small text segments as notations, they seldom have the grid feature like tables. Besides, most tables do not contain curve line objects as graphic figures do. As for the matrixes, the graphic lines of them are usually two vertical lines, while tables seldom follow this pattern.

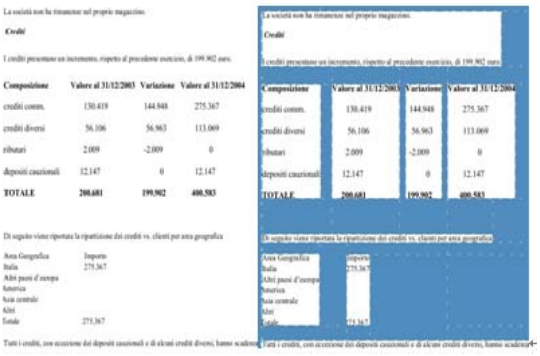


Figure 4. Multiple unruled tables on one page

Therefore, the following conditions are checked to verify whether the extracted tables are false positive ones or not:

1). There should be at least two rows and two columns in a table.

2). There should not be curve line objects in a table.

3). The separators and columns inside a table should be spaced at intervals.

4). The table should not contain only vertical ruling lines.

IV. EXPERIMENTAL RESULTS

We test our algorithm on two datasets. The first dataset (D1) includes 70 PDF e-Books provided by Founder Corporation. Those books were selected randomly from the two million e-Book library and manually examined to ensure each of them contains tables with various layouts. The second dataset (D2) is scientific documents provided by Liu, which is used in her previous work [6, 11]. We manually selected 70 documents with the same requirement as D1. In total, 3802 tables in D1 and 197 tables in D2 are evaluated. Since there is no existing ground truth, all the correctness comparisons are obtained from human understanding. We evaluate two performance metrics: recall (the percentage of the true objects that the method finds) and precision (the percentage of the objects that are in fact true).

Table I compares our evaluation results with that of [6]. The results are analyzed to explore the advantages as well as main causes for errors. Fig.5 provides some examples to illustrate the effectiveness of our methods in different cases, which cannot be handled well in existing methods.

A. Experimental Results and Analysis

From the data presented in the Table I, we conclude that our method has desirable improvement on recall rate over that of method [6], because they define captions as necessary attribute of tables. In real-world documents, captions may be absent or labeled with keywords different from their predefined list. Besides, in the e-Book dataset, we also get a higher precision, because sparse tables with a few cells are hard to be detected due to lacking of the sparse line feature proposed by Liu. However, ruling lines of these tables are more likely to be explored as boundaries. As a result, our method shows competitive performance.

However, errors are still inevitable in the following aspects: *i)* if the page column space is overlap coincidently with the table column spaces, the wrongly detected page column may cause over-segmentation of tables; *ii)* the whitespace separators are not as accurate as ruling line separators, which could result in tables being spotted partially; *iii)* some figures with tabular structure and only straight lines are still mistakenly detected as tables.

TABLE I. EXPERIMENTAL DATA

Mehods	D1		D2	
	Precision	Recall	Precision	Recall
Method in this paper	96.13%	92.07%	94.42%	93.71%
Method in Liu[6]	93.56%	83.20%	96.28%	92.50%

2011 International Conference on Document Analysis and Recognition

Character enhancement for historical newspapers printed using hot metal typesetting

Iuliu Konya Stefan Eickeler Christoph Seibert
Fraunhofer IAIS
Sankt Augustin, Germany
{*e-mail: iuliu.konya, stefan.eickeler, christoph.seibert*}*@iais.fraunhofer.de*

***Abstract*—We propose a new method for an effective removal of the printing artifacts occurring in historical newspapers which are caused by problems in the hot metal typesetting, a widely used printing technique in the late 19^{th} and early 20^{th} century. Such artifacts typically appear as thin lines between single characters or glyphs and are in most cases connected to one of the neighboring characters. The quality of the optical character recognition (OCR) is heavily influenced by this type of printing artifacts. The proposed method is based on the detection of (near) vertical segments by means of directional single-connected chains (DSCC). In order to allow the robust processing of complex decorative fonts such as Fraktur, a set of rules is introduced. This allows us to successfully process prints exhibiting artifacts with a stroke width even higher than that of most thin characters stems. We evaluate our approach on a dataset consisting of old newspaper excerpts printed using Fraktur fonts. The recognition results on the enhanced images using two independent OCR engines (ABBYY FineReader and Tesseract) show significant improvements over the originals.**

***Keywords*-character enhancement; OCR; retro-digitization; historical documents; hot metal typesetting**

I. Introduction

The quality of the results of the optical character recognition (OCR) is directly influenced on one side by the quality of the scanning process and by the printing process on the other side. In a digitization workflow the human operator can control the scanning process of the documents and directly take the appropriate measures to deal with scanning errors in the digitization process. By contrast, printing is normally completed a long time before the scanning procedure and is out of the control of the scanning operator performing the retro-digitization. Therefore, the development of algorithms capable of alleviating the problems occurring in the printing process is a highly desirable endeavor.

In this paper we describe such a method for improving the quality of digitized text documents initially produced using letterpress printing. Specifically, we focus on documents printed during the time period spanning from the beginning until the middle of the 20^{th} century. In this period, a very widespread typesetting technique was hot metal typesetting (also known as hot lead typesetting). This technique represented one of the earliest attempts at mechanizing the printing process, which in turn facilitated its use on an industrial scale. The process consisted of injecting molten type metal (with lead as its main constituent) into molds, each having the shape of a single character or a ligature. The obtained sorts were subsequently used to press ink onto paper and produce the print. A common problem with the procedure was the fact that ink tended to infiltrate between the individual sorts upon imbuement, thus producing specific near-vertical artifacts upon pressing them against the paper. As can be seen in figures 1, 4 and 3, such artifacts are quite prominent and have a comparable stroke width as well as the exact same gray level as regular characters in the digitized image. This fact makes it virtually impossible for the artifacts to be removed effectively by using any state-of-the-art global or local thresholding algorithms. A wide selection of thresholding algorithms, alongside with a performance comparison on a pixel-accurate ground truth can be found in the paper of Gatos et al. [1].

As such, we follow a different approach, presented in more detail in section III. Before that however, we give a short overview of related approaches for document and character enhancement. The proposed approach is evaluated on a dataset consisting of old German-language newspaper excerpts via the OCR results obtained from two well-known OCR engines, namely ABBYY FineReader [2] and Google's Tesseract [3]. Conclusions and directions for future work (partly arisen from the practical issues encountered during the evaluation) are present at the end of the paper.

II. Related Work

Much research work has been done in the different areas of degraded document processing. Algorithms for character enhancement are continuously being proposed for coping with all types of printing techniques as well as for hand-written documents. Despite the continued research effort, until now no unified algorithm applicable for all printing techniques exists. Because of the sheer variety of artifacts it is indeed doubtful that such a generic improvement method is actually possible.

One of the most well-researched related areas is the family of bleed-through/show-through removal techniques for double-sided documents. Most similar in intent to the current paper are techniques belonging to the blind family, such as blind source separation [4]. These methods attempt

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.190

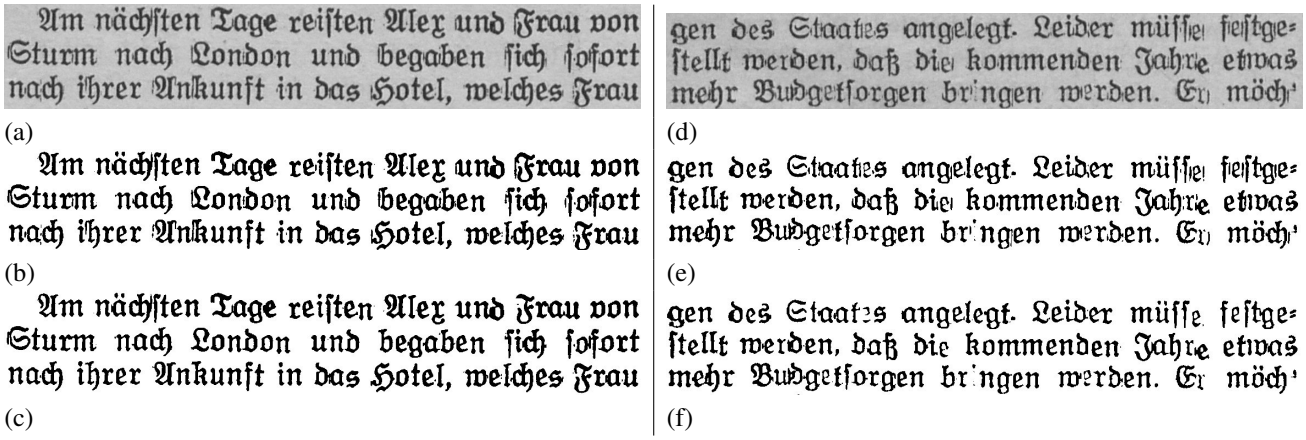


Figure 4. (a), (d) - portion of original grayscale image; (b), (e) - binarization result using Otsu's method [12]; (c), (f) - result obtained using the proposed method

this approach will obviously not converge to the actual x-height, but in such cases we found it actually desirable to adapt and allow for taller, respectively shorter artifacts. Subsequently, we simulate the deletion of each candidate from its containing connected component. If the height of the resulting component height does not change significantly the candidate is kept, as it likely represents a superfluous protrusion. The other situation when the candidate is kept in the list is when its deletion has cause the connected component to (almost) completely disappear. In the latter case we are very likely dealing with an isolated printing artifact.

At this point the filtering procedure already produces satisfactory results with the notable exception of two cases: the small letter "i" on one side and the set of letters containing counters (e.g. "e", "d", "a", "o", "g", "p"). In both situations the algorithm would identify portions of the respective characters as likely candidates (i.e. either the stem of the "i" or the bowls of the other characters), thus leading to many false positives. The first case can be readily dealt with by searching for a small connected component resembling a dot located right above the candidate. The second case can be identified just as easily, with the sole difference that it additionally involves the extraction of all background connected components from the bitmap of the text line and a straightforward adjacency test.

IV. Evaluation

The evaluation data set consists of 52 single-column text-only excerpts from grayscale newspaper pages printed between 1920 and 1950 and totaling more than 63 000 characters. The newspaper pages originate from the German-language newspaper "Liechtensteiner Volksblatt" and feature a Fraktur script. We have chosen document images printed using this highly decorative script on purpose so as to assess the robustness of the proposed method. As can be seen in figure 3, the proposed technique can readily handle more traditional typefaces, such as Antiqua, as well as non-Romanic scripts. The data set was split about evenly into two groups of images: one containing only text regions clearly affected by printing artifacts and the other containing completely uncorrupted regions. The distinction was done in order to be able to obtain more meaningful evaluation results. This is because a typical newspaper image contains both kinds of regions, irregularly mixed and widely differing in size (number of characters), thus potentially skewing the results greatly in one direction or the other. Also, by having separate evaluation results for affected and unaffected regions one may readily compute a weighted average as an approximation of the expected quality for any image featuring mixed content.

The proposed artifact removal procedure has as primary goal the qualitative improvement of OCR results. Therefore we have chosen as evaluation measure the Levenshtein

Method	**Affected dataset** **33460 chars** 5034 words 27 images	**Unaffected dataset** **30036 chars** 4730 words 25 images
Tesseract	4793	743
Tess + Enhance	2288	756
Tess + Enhance relative diff.	**52.3%**	**-1.7%**
FineReader	1720	424
FR + Enhance	1074	441
FR + Enhance relative diff.	**37.5%**	**-4%**
Overall relative diff.	**44.9%**	**-2.8%**

Table I
LEVENSHTEIN DISTANCE [14] FROM THE GROUND TRUTH WITHOUT AND WITH THE PROPOSED FONT ENHANCEMENT METHOD, USING TESSERACT [3] AND ABBYY FINEREADER [2] AS OCR ENGINES

distance [14] in conjunction with manually corrected OCR ground truth. In order to ensure that the obtained enhancements are indeed generic and not specific to a particular OCR method, we perform the evaluation using two different, well-known OCR engines: the open source Tesseract v3 [3] and the commercial product ABBYY Finereader 7 [2]. The exact same manually corrected rectangular regions (each corresponding to a single text line) were fed into both OCR engines and the Levenshtein distance was computed on a line-by-line basis. All newspaper images were binarized using Otsu's global thresholding method [12] prior to the application of the enhancement and/or the OCR procedure. Note that the primary purpose of our evaluation is to assess the quality of the proposed enhancement method in the absence of any other external factors. Because of this, document images were specifically chosen so that the scan quality was good enough to allow the application of a global binarization method.

As can be seen from table I our method manages to consistently improve the OCR results of both engines on affected regions. At the same time, the quality loss of the OCR results on uncorrupted text regions is minimal. As such, this is a very encouraging result. In addition, the false positive rate can potentially be reduced even further by observing and appropriately handling the specific situations in which the algorithm still fails. However, we believe that the addition of a (binary) classifier capable of separating corrupted from uncorrupted regions would potentially result in a much more substantial improvement in OCR quality. The prior classification on a region-by-region or line-by-line basis would allow for a more aggressive artifact filtering in heavily affected areas. In both newspaper images from figure 4, one may clearly see that there is still room for improvement especially for thick artifacts intersecting characters.

V. Conclusion

We described a novel method for the automatic removal of letterpress printing artifacts produced as a direct result of problems in the hot metal typesetting. The approach was tested on a dataset consisting of historical newspaper excerpts printed using a Gothic (Fraktur) font and totaling over 63 000 characters. OCR results obtained from two independent state-of-the-art OCR engines (ABBYY FineReader and Google Tesseract) show substantial improvements on document images enhanced via the proposed algorithm. Further research will focus on the automatic detection of affected regions/text lines in order to allow a selective (and consequently much more effective) application of the algorithm.

Acknowledgement

This work was supported by the German Federal Ministry of Economics and Technology (BMWi) funded program *Theseus* in the project *Contentus*.

References

[1] B. Gatos, K. Ntirogiannis, and I. Pratikakis, "ICDAR 2009 document image binarization contest (DIBCO 2009)," in *Proc. Int'l Conf. Document Analysis and Recognition*, 2009, pp. 1375–1382.

[2] ABBYY, "ABBYY FineReader," http://finereader.abbyy.com/, accessed 19 Mar 2011.

[3] Google, "tesseract-ocr," http://code.google.com/p/tesseract-ocr, accessed 19 Mar 2011.

[4] A. Tonazzini, E. Salerno, and L. Bedini, "Fast correction of bleed-through distortion in grayscale documents by a blind source separation technique," *Int'l J. Document Analysis and Recognition*, vol. 10, pp. 17–25, 2007, 10.1007/s10032-006-0015-z. [Online]. Available: http://dx.doi.org/10.1007/s10032-006-0015-z

[5] B. Allier, N. Bali, and H. Emptoz, "Automatic accurate broken character restoration for patrimonial documents," *Document Analysis and Recognition*, vol. 8, no. 4, pp. 246–261, 2006.

[6] J. D. Hobby and T. K. Ho, "Enhancing degraded document images via bitmap clustering and averaging," in *Proc. Int'l Conf. Document Analysis and Recognition*, 1997, pp. 394–400.

[7] A. Antonacopoulos and C. C. Castilla, "Flexible text recovery from degraded typewritten historical documents," in *Proc. IEEE Int'l Conf. on Pattern Recognition*, 2006, pp. 1062–1065.

[8] B. Gatos, S. Mantzaris, S. Perantonis, and A. Tsigris, "Automatic page analysis for the creation of a digital library from newspaper archives," *Digital Libraries*, vol. 3, pp. 77–84, 2000.

[9] T. Breuel, "Two algorithms for geometric layout analysis," in *Proc. Workshop on Document Analysis Systems*, vol. 3697, 2002, pp. 188–199.

[10] Y. Zheng, C. Liu, X. Ding, and S. Pan, "Form frame line detection with directional single-connected chain," in *Proc. Int'l Conf. on Document Analysis and Recognition*. IEEE Computer Society, 2001, pp. 699–703.

[11] C. R. Maurer, R. Qi, and V. Raghavan, "A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions," *Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, 2003.

[12] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[13] J. Hartigan, *Clustering algorithms*. New York: John Wiley and Sons, Inc., 1975.

[14] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966.

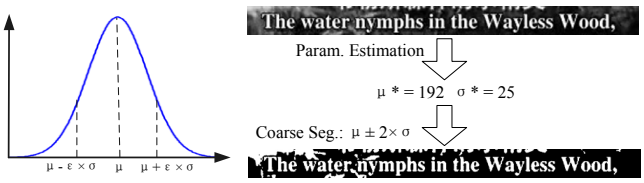


Figure 4. Coarse segmentation by gaussian model

4) Coarse segmentation: For the normalized input image I, once the model parameters μ and σ are obtained, the coarse segmentation can be carried out by:

$$B_c(p) = \begin{cases} 255 & \text{if } |I(p) - \mu| < \varepsilon\sigma \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

Here B_c refers to the result image of coarse segmentation, and ε is the segmentation threshold. In this paper, $\varepsilon = 2.0$.

B. Color-based noise elimination

Most of the text pixels can be extracted by the stroke-based coarse segmentation. However, there also would be non-text noises in the coarse segment results, as the color of non-text noises might fall into the color band of text. Thus the noise elimination step is needed.

For the coarse segmentation result image B_c, the noise elimination process is carried out as follows:

1) Border seed filling: The method of border seed fill is proposed by Lienhart et al. [6]. This process aims at removing the non-text regions connected to the border.

2) Connected component analysis(CCA): After the process of border seed filling, the CCA is adopted to extract the connected components.

Let $C = \{C_0, C_1, \cdots, C_N\}$ denote the connected component (CC) set, for each component C_i, we calculate the local mean μ_i and standard variance σ_i as follows:

$$\mu_i = \frac{1}{N_i}\sum_{j \in C_i} p(j) \quad \sigma_i = \sqrt{\frac{1}{N_i}\sum_{j \in C_i}(p(j) - \mu_i)^2} \quad (17)$$

where j is the point in C_i and $p(j)$ denotes the gray value of the j-th pixel in the normalized image. N_i is the number of pixels in C_i.

3) Color ditribution homogeneity based noise elimination: Remove the CCs which don't satisfy color distribution homogeneity constraint. For the i-th CC C_i, it should be eliminated if C_i doesn't obey the conditions in Equation 12.

IV. Experiments and analysis

To evaluate the effectiveness of our method, we grab 1200 text blocks including 11789 characters from images and video frames. The characters in the dataset involve Chinese, English, and digits. All the experiments are done on the computer with a CPU of Pentium IV 2.8GHZ.

The proposed approach is evaluated by the recognition results and the process time cost per image. In this paper,

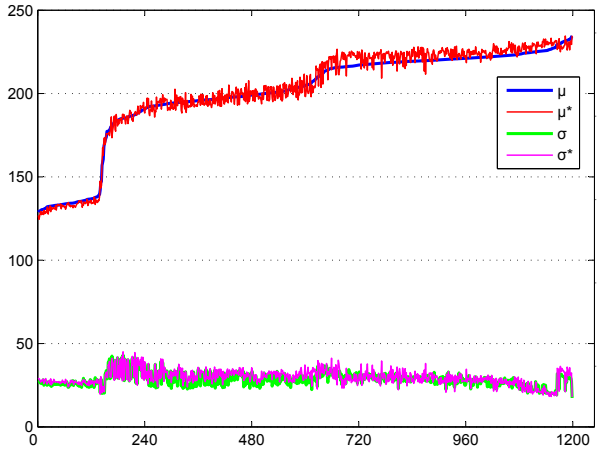


Figure 5. Parameters estimation results on the test dataset: μ and σ are calculated by labeled ground truth image; μ^* and σ^* are calculated by the stroke-based evaluation method.

Table II
The performance of the noise elimination process

Alg.	RPR	RRC	LRR
Coarse segmentation	71.45%	70.53%	39.97%
After noise elimination	97.27%	97.88%	85.22%

the recognition precision rate (RPR), recognition recall rate (RRC) and line recognition rate (LRR) are adopted as evaluation criteria. RPR, RRC and LRR are calculated as follows:

$$RPR = \frac{N_C}{N_R} \quad RRC = \frac{N_C}{N_G} \quad LRR = \frac{L_C}{L_G} \quad (18)$$

Here, N_R and N_C denote the number of totally recognized and correctly recognized characters. N_G is the ground truth number of characters. L_C is the number of correctly recognized text lines. L_G is the ground truth number of text lines. The text recognition in this paper is implemented by commercial OCR engine from Hanvon Tech., Co. Ltd..

In this paper, we utilize a heuristic method to evaluate the color distribution parameters μ and σ by the stroke map (see Table I). To prove the effectiveness of our approach, we compare the evaluated parameters with the ground truth value. The results are showed in Figure 6, where μ and σ are calculated by labeled ground truth binary image, μ^* and σ^* are obtained by the stroke-based evaluation method. It can be seen that the evaluated parameters match the ground truth well.

The proposed method includes a coarse segmentation and noise elimination process. To test the performance of this two process, we do comparison experiments on the test data. The results are showed in Table II. It can be seen that the performance is highly increased by the noise elimination process. The RPR and RRC have been increased by nearly

where L is the number of levels of the pyramid. Since the amount of visual words assigned to each bin is lower at higher levels of the pyramid, due to the fact that the spatial bins are smaller, the visual words contribution is weighted by $w_l = P_x^l P_y^l$, where l is the corresponding pyramid level. In our experiments, we have used a two levels SPM, with $P_x = 2$ and $P_y = 1$, resulting in 3 spatial bins and therefore a descriptor of 4500 dimensions for each patch. Using this configuration, the descriptors encode separately the left and right side of the words. This spatial information increases the whole performance of the method.

Summarizing, for each document page D_i we obtain a number of overlapping local patches p_j^i with $j \in \{0, ..., N\}$. Each patch p_j^i is characterized by a BoVW model over densely extracted SIFT descriptors quantized with a k-dimensional codebook and a SPM of two levels. Each p_j^i is then described by an M-dimensional descriptor $\mathbf{f}_j^i$.

III. Latent Semantic Indexing

Each p_j^i in our collection of patches is represented by the descriptor $\mathbf{f}_j^i$ obtained by the BoVW model presented in the previous Section. Hence, the patch corpus is represented by a feature-by-patch matrix $\mathbf{A}^i \in \mathbb{R}^{M \times N}$, where M is the descriptor dimensionality and N is the number of patches. The matrix $\mathbf{A}^i$ is then weighted by applying the *tf-idf* model. This normalization emphasizes the features that are frequent in a particular patch and infrequent in the complete patch corpus. After this normalization, we apply the LSI technique first introduced by Deerwester et al. in [11]. The motivation of using LSI is that this technique is able, given a text retrieval framework, to return results that are conceptually similar in meaning to the query even if the results do not share an important set of words with the query. In our particular methodology, the use of LSI allows us to retrieve relevant patches even if they do not contain the same exact features than the query sub-image.

The LSI model assumes that there exists some underlying semantic structure in the descriptor space. This semantic structure is defined by assigning to each patch descriptor a set of topics, which can be estimated in an unsupervised way using standard statistical techniques. The goal is to obtain a transformed space where patches having similar topics but with different descriptors will lie close. This transformed space is obtained by decomposing the feature-by-patch matrix in three matrices by a truncated Singular Value Decomposition (SVD). In order to reduce the descriptor space to K topics we proceed as follows:

$$\mathbf{A}^i \simeq \hat{\mathbf{A}}^i = \mathbf{U}_K^i \mathbf{S}_K^i \left(\mathbf{V}_K^i\right)^\top, \tag{2}$$

where $\mathbf{U}_K^i \in \mathbb{R}^{M \times K}$, $\mathbf{S}_K^i \in \mathbb{R}^{K \times K}$ and $\mathbf{V}_K^i \in \mathbb{R}^{N \times K}$. The super-index i indicates that a different LSI transformation is generated for each document separately. In this way, each of our topics model the words in a document page, allowing to work with heterogeneous document sets. In our experimental setup, we use a value of $K = 200$ topics. Note that LSI does not result in a reduction of the number of features, but a transformation from the patch descriptor space to a topic space.

IV. Retrieval Stage

At the retrieval stage, the user provides an example of the word he wants to find. This sub-image is taken as if it corresponded to a single patch within a document. Dense SIFT descriptors are thus extracted from the query image and quantized by using the codebook. Then, applying the same SPM configuration than in the document corpus, the final query descriptor $\mathbf{f}_q$ is obtained.

The first step of the retrieval is to obtain a list of patches sorted by the similarity to the query for each document page in the collection. This is accomplished by first projecting the descriptor $\mathbf{f}_q$ to each document topic space by

$$\hat{\mathbf{f}}_q^i = \mathbf{f}_q^\top \mathbf{U}_K^i \left(\mathbf{S}_K^i\right)^{-1}. \tag{3}$$

Then, we obtain the similarity list by using the cosine distance between $\hat{\mathbf{A}}^i$ and $\hat{\mathbf{f}}_q^i$ for each document in the corpus. By just considering the 200 topmost patches, we build a voting space in order to find the zones of the image having more accumulation of evidences that the queried word is likely to be found. The final retrieved zones are determined by searching for local maxima in the voting space.

V. Experimental Results

A. Datasets and Performance Evaluation

To perform the experiments, we worked with three datasets of different nature. On the one hand the George Washington (GW) dataset described in [1]. This dataset consists of 20 handwritten pages with a total of 4860 words. On the other hand, the Lord Byron (LB) dataset consists on 20 typewritten pages from a 1825 book[1] with a total of 4988 words. The ground-truth for both collections contains the word transcriptions and their bounding-boxes. Finally, the Persian (PE) dataset consists of 20 typewritten pages from a 1848 book written in Persian. Unfortunately, we do not have any ground-truth for this dataset, and will only be used as a proof-of-concept that the method is able to work with non-Latin scripts by showing qualitative results.

Concerning the performance evaluation of the method, we will show the precision and recall curves and the mean average precision indicator. In order to compute these measures, we need to define a notion of relevance from the returned results. For a given query, a returned zone will be labeled as relevant if it overlaps more than a 50% of one of the bounding-boxes in the ground-truth containing the same queried word.

[1]A binary and cleaned version of the book can be downloaded from http://books.google.com/books?id=u6poWVzCIWsC

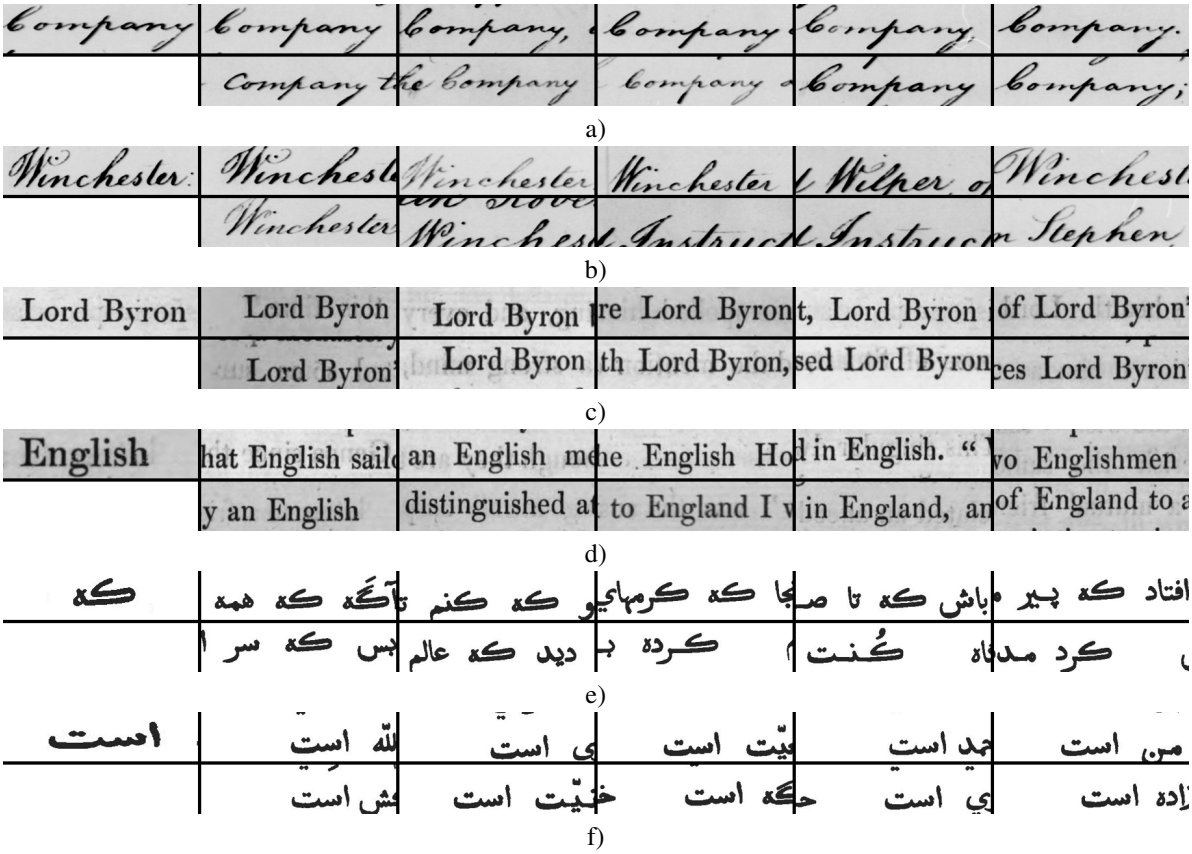


Figure 1. Qualitative results. Query and top ten retrieved words. a) and b) GW dataset. c) and d) LB dataset. e) and f) PE dataset.

B. Results

We present in Figure 1 some qualitative results of the method in the three different datasets for a couple of queries. Note that the proposed method is able to work with queries formed by multiple words. Note also that the false positives are still visually similar to the queries.

Concerning the quantitative evaluation, we used all the words in GW and LB datasets as queries. For the GW dataset the mean average precision was 30.42% and the mean recall was 71.1% whereas for the LB dataset the mean average precision was 42.83% and the mean recall was 85.86%. For the proposed spotting method, we can see that most of the relevant words are retrieved, even if the ranking is not that good due to false positives. Analyzing the results, we noticed that the performance of the system was highly related to the length of the queried words. We can see in Figure 2 the precision and recall curves for different word lengths. The mean average precision and mean recall results depending on the query length are shown in Figure 3. As we can appreciate, if we do not consider small words as queries (usually corresponding to stop-words) the system performance presents an important increase. For example, considering words larger than 5 characters, for the GW dataset the mean average precision is increased until reaching a 53.76% and the mean recall results in a 93.39% whereas for the LB dataset the mean average precision results in a 70.23% and the mean recall is increased until reaching a 98.32%. Typewritten documents perform a little bit better than the handwritten ones basically due to the variability of word shapes in the handwritten context.

Regarding the time complexity, the process of querying a word against a single page (that is, indexing more than 9200 patches) takes in average 340ms. in our prototype implementation using Matlab and Python.

VI. Conclusions and Future Work

In this paper we have presented a word spotting method that does not rely on any previous segmentation step. We have shown that the proposed method can be used in heterogeneous collections since it yields good results in both handwritten and typewritten documents. Our method can also be used with non-Latin scripts and does not require any preprocessing step of noise removal or normalization. The presented method combines the use of a bag-of-visual-words model based on SIFT descriptors and the later refinement of the descriptors by using the latent semantic indexing technique. A final voting scheme aims to locate the zones within document images where the queried word is likely to

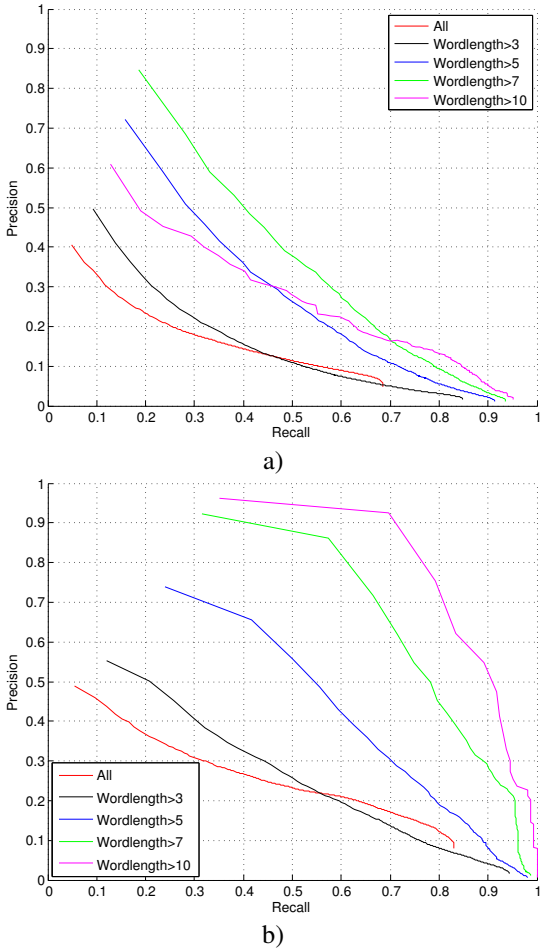


Figure 2. Precision and recall curves for different word lengths. a) GW dataset. b) LB dataset.

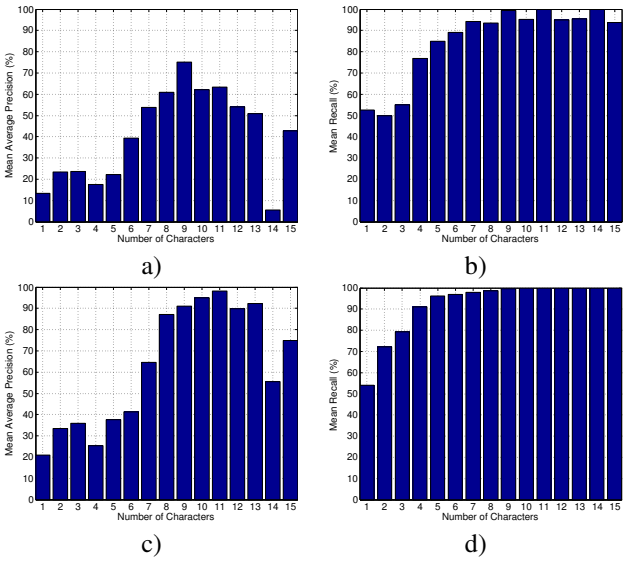


Figure 3. Quantitative results for different word lengths. a) and b) mean average precision and mean recall for GW dataset respectively. c) and d) mean average precision and mean recall for LB dataset respectively.

appear. As future research lines we would like to combine the use of the proposed method with some approximate nearest neighbor technique in order to avoid the one-to-one distance computation.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Education and Science under projects TIN2008-04998, TIN2009-14633-C03-03, TRA2010-21371-C03-01, Consolider Ingenio 2010: MIPRCV (CSD200700018) and the grant 2009-SGR-1434 of the Generalitat de Catalunya. Special thanks go to M. Rouhani for his assistance with the Persian documents.

References

[1] T. Rath and R. Manmatha, "Word spotting for historical documents," *Int. J. Doc. Anal. Recogn.*, vol. 9, no. 2–4, pp. 139–152, 2007.

[2] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "HMM-based word spotting in handwritten documents using subword models," in *Proc. of the Int. Conf. on Pattern Recognition*, 2010, pp. 3416–3419.

[3] V. Frinken, A. Fischer, and H. Bunke, "A novel word spotting algorithm using bidirectional long short-term memory neural networks," in *Artifitial Neural Networks in Pattern Recognition*, ser. LNCS, 2010, vol. 5998, pp. 185–196.

[4] Y. Leydier, A. Ouji, F. LeBourgeois, and H. Emptoz, "Towards an omnilingual word retrieval system for ancient manuscripts," *Pattern Recognit.*, vol. 42, no. 9, pp. 2089–2105, 2009.

[5] B. Gatos and I. Pratikakis, "Segmentation-free word spotting in historical printed documents," in *Proc. of the Int. Conf. on Document Analysis and Recognition*, 2009, pp. 271–275.

[6] S. Bai, L. Li, and C. Tan, "Keyword spotting in document images through word shape coding," in *Proc. of the Int. Conf. on Document Analysis and Recognition*, 2009, pp. 331–335.

[7] E. Şaykol, A. Sinop, U. Güdükbay, O. Ulusoy, and A. Çetin, "Content-based retrieval of historical ottoman documents stored as textual images," *IEEE Trans. on Im. Proc.*, vol. 13, no. 3, pp. 314–325, 2004.

[8] B. Fulkerson, A. Vedaldi, and S. Soatto, "Localizing objects with smart dictionaries," in *Computer Vision - ECCV*, ser. LNCS, 2008, vol. 5302, pp. 179–192.

[9] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *Int. J. Comput. Vision*, vol. 73, no. 2, pp. 213–238, 2007.

[10] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. of the Conf. on Computer Vision and Pattern Recognition*, 2006, pp. 2169–2178.

[11] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Am. Soc. Inform. Sci.*, vol. 41, no. 6, pp. 391–407, 1990.

sequential correction style. The accuracy results are described in Fig. 2.

In our dataset, there are even some street signs and non-document images. The fraction of these non-document images is about 20%. The "REJ" rates of our methods are 23.68% and 26.32%, which is mainly caused by too large distortions. For a mobile phone with some proper interactive GUIs, users may accept the results of HIT, HHIT, VHIT, and REJ because the resulting image from these has a much better quality than (or a same quality as) the originally captured image. In this way, the acceptance rates of our methods are 98.80% and 98.33%.

Compared with M3, our methods (M1 and M2) improve the "HIT" groups by 16.26% and 11.48% respectively. This shows that character vertical strokes are very useful to detect the vertical vanishing point for documents without vertical boundaries. Compared with M4, our methods improve 11.72% and 6.94% for the "HIT" accuracy. Our hybrid approach is more robust for vanishing point detection. Compared with M5, our methods improve the "HIT" accuracy by 7.17% and 2.39% and decrease the processing time by 11ms and 12ms, which shows that our clustering strategy is robust and fast compared to the traditional model fitting. Our methods have a similar performance with M6, but M6 uses a sequential style with partial rectification.

The processing speed is shown in Table 1, where "Time" represents the average processing time for each image without including the time for the grayscale image conversion and the final perspective transformation. Experiments are run on a DELL PC with 3GHz CPU, 2G Memory on a Windows XP OS.

Table 1. Results of the average processing time.

	M1	**M2**	**M3**	**M4**	**M5**	**M6**
Time (ms)	108	103	72	90	115	226

As shown in Table 1, the average processing time of our methods is largely less than M6, and the reduced time is more than 100ms. We also test the direct approach by a hierarchical search for horizontal vanishing point detection in [2], which is more time consuming. For one image in test samples, the detection time is more than one second.

Compared to M2, our new approach (M1) has a higher accurate rate. The "HIT" accuracy is improved from 53.11% to 57.89%. This shows that vanishing point candidate clustering on the Gaussian sphere is effective. Moreover, the additional processing time is only 5ms.

Our rectification technology for perspective document images has been implemented and applied into real mobile phones. Real applications show that our method has an acceptable performance for both accuracy and speed. For a mobile phone camera-based document image (with a 1280*960 resolution), the average processing time is about 1s~2s. Some real samples and corresponding rectified images are shown in Fig. 3.

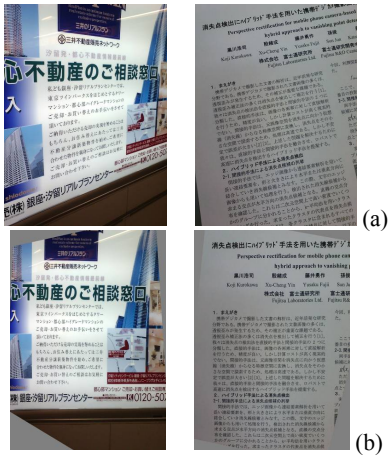


Fig. 3. Some real samples captured by mobiles: (a) original images, (b) rectified images.

V. CONCLUSIONS

Perspective rectification of MobileCam-based documents faces several challenges, such as speed, robustness, non-boundary documents, etc. In this paper, we present a fast and robust technology to deal with these problems. In our methods, the hybrid approach for vanishing point detection combines direct and indirect approaches with high precision and fast speed. The experiments on different document images captured by mobile phone cameras show that our method has a good performance with an average speed of about 100ms on a regular PC. Moreover, our perspective rectification system has been applied into real mobile phones.

ACKNOWLEDGMENTS

The work of the authors Xu-Cheng Yin and Hong-Wei Hao is partly supported by the R&D Special Fund for Public Welfare Industry (Meteorology) of China under Grant No. GYHY201106039 and the Fundamental Research Funds for the Central Universities under Grant No. FRF-BR-10-034B.

REFERENCES

[1] J.F. Canny, "A computational approach to edge detection," *IEEE Trans. on PAMI*, vol. 8, no. 6, pp. 679-698, 1986.

[2] P. Clark, and M. Mirmehdi, "Rectifying perspective views of text in 3D scenes using vanishing points," *Pattern Recognition*, vol. 36, no. 11, pp. 2673-2686, 2003.

[3] C.R. Dance, "Perspective estimation for document images," *Proceedings of SPIE Conference on Document Recognition and Retrieval IX*, pp. 244-254, 2002.

[4] W.L. Hwang, C.-S. Lu, and P.-C. Chung, "Shape from texture: estimation of planar surface orientation through the ridge surfaces of continuous Wavelet transform," *IEEE Trans. on Image Processing*, vol. 7, no. 5, pp. 773-780, 1998.

[5] D.X. Le, G.R. Thoma, and H. Wechsler, "Automated borders detection and adaptive segmentation for binary document images," *Proceedings of ICPR*, pp. 737-741, 1996.

[6] J. Liang, D. Doermann, and H.P. Li, "Camera-based analysis of text and documents: a survey," *International Journal on Document Analysis and Recognition*, vol. 7, no. 2-3, pp. 84-104, 2005.

the similarities $p(\varphi_{k\alpha}, x)$ between the character x and each of the P the prototypes $\varphi_{k\alpha}$ of letter α, as:

$$p(\varphi_{k\alpha}, x) = \frac{exp(-\beta * dist_{(k\alpha)}(f_x))}{\sum_{i=1}^{P} exp(-\beta * dist_{(i\alpha)}(f_x))} \quad (4)$$

where $dist_{(k\alpha)}(f_x)$ is the mean Euclidean distance between f_x and the reduced feature vectors of the samples belonging to the prototype $\varphi_{k\alpha}$. Parameter $\beta > 0$ is a fixed scalar which determines the selectivity of exponential function. In our system, we use $\beta = 0.1$.

The term frequency $TF_{\varphi_{k\alpha}}(D)$ of prototype $\varphi_{k\alpha}$ in the document D is defined as follow:

$$TF_{\varphi_{k\alpha}}(D) = \frac{1}{\sum\limits_{x \in D} \delta_{letter(x)=\alpha}} \sum_{x \in D} p(\varphi_{k\alpha}, x) \delta_{letter(x)=\alpha} \quad (5)$$

where $\delta_{letter(x)=\alpha} = 1$ if x is recognized as the letter α, and 0 otherwise. The value $IDF_{\varphi_{k\alpha}}(\Omega)$ is calculated as in Equation (3), with $\varphi_{k\alpha}$ instead of φ_i. We consider that $\varphi_{k\alpha} \in D$ when at least a character x of D is such that:

$$\delta_{letter(x)=\alpha} \quad \text{and} \quad p(\varphi_{k\alpha}, x) = \underset{j=1...P}{Argmax}(p(\varphi_{j\alpha}, x))$$

Further, each document D is represented by a weight matrix

$$W = [TF_{\varphi_{k\alpha}}(D) * IDF_{\varphi_{k\alpha}}(\Omega)] \quad (6)$$

of size $26 \times P$, describing the writing style of this document.

During the recognition stage, we can use any distance (Euclidean, normalized cosine, χ^2) or dissimilarity measure between the weight matrices of the documents to compare.

The main differences with the methods in [14], [15] lie in the use of a different character segmentation/recognition method, in the use of both on-line and off-line features, of PCA and of a different similarity measure $p(\varphi_{k\alpha}, x)$.

B. Method based on graphemes

The method proposed in the previous section is very effective in general. However, when the character recognizer fails to correctly recognize a given character x, then x is assigned to an incorrect letter α and the similarities $p(\varphi_{k\alpha}, x)$ on which is based the writer recognition process are non relevant, inducing a systematic bias in the writer recognizer. In order to reduce this bias, we conceive a method based on graphemes instead of characters. This method is very similar to the method based on characters that is described in the previous section and in Figure 1, except that instead of segmenting and recognizing characters, we extract graphemes by using our segmentation method introduced in [20]. Then, the graphemes x are characterized by using the features in [20]. In order to reduce variability of the clustering input dataset as well as the weight matrices computation time, graphemes are pre-classified into 4 groups α depending on the initial writing direction θ ($\theta \in \{[0; 90[, [90; 180[, [180; 270[, [270; 360[$). Then, PCA is applied. Inside each group α, graphemes are clustered into P different cluster prototypes using a variant of the k-means algorithm. Here, based on experiments, we use $P = 20$. Then, the similarity $p(\varphi_{k\alpha}, x)$ between the grapheme x belonging to group α and the k^{th} prototype $p_{k\alpha}$ of group α is computed using equation (4) with $\beta = 0.1$ and the weight matrices are computed as in the previous method.

During the recognition stage, any two documents (and therefore two writers) are compared using any distance between the corresponding weight matrices.

IV. Experimental results

In order to assess the effectiveness of the proposed approaches, we perform a series of experiments of increasing complexity, first with the character-based approach and second with the grapheme-based approach. In both cases, the results presented in this section are obtained using χ^2 distance, as it gave the best results.

A. Results of the character-based method

First experiment: We use our own handwriting database including 32 documents written by 16 writers on our electronic tablet (2 documents per writer). Each document contains, for each of the 26 letters of the latin alphabet, 10 occurences in predefined fields. Both the reference and the test databases include 1 document per writer, and therefore $16 \times 10 \times 26 = 4160$ characters. If we use the ground-truth and skip the character segmentation/recognition step, the writer recognition rate is 100%. If we use our isolated character segmentation/recognition module described in [20], the writer recognition rate is still 100% even though the character recognition rate has dropped to 94%. Therefore we can conclude that this approach is relatively robust towards character segmentation/recognition errors.

Second experiment: We use cursive words extracted from the IRONOFF database [21] to construct both the reference and test databases. We consider words written by 10 to 300 different writers, with 30 words per writer (among which 20 words are included in the reference database, the 10 remaining words in the test database). Figure 3 (dotted lines) shows the writer recognition rates when using the ground-truth and when using our character segmentation/recognition module. The comparison of these two curves highlights the bias introduced by the automatic character segmentation/recognition errors. Indeed, the difference between the recognition rates of these two experiments in on average 12%. We can also see that the recognition rate drops faster when the character segmentation/recognition step is performed automatically, than when the ground-truth is used. This shows the limits of the robustness towards character segmentation/recognition errors (robustness illustrated by the previous experiment).

We implemented a method which is as similar to the one presented in [15] as we could implement. Instead of My

2011 International Conference on Document Analysis and Recognition

Document recto-verso registration using a dynamic time warping algorithm.

Rabeux Vincent
University of Bordeaux
LaBRi
Bordeaux
rabeux@labri.fr
shema

Journet Nicholas
University of Bordeaux
LaBRi
Bordeaux
journet@labri.fr

Domenger Jean Philippe
University of Bordeaux
LaBRi
Bordeaux
domenger@labri.fr

***Abstract*—Recto verso registration is an important step allowing detection of missing digitized pages, or location of the bleed-through defect over a page. An efficient way to restore or evaluate the bleed-through of a digitized document consists in analyzing at the same time both the recto side and the verso side. This method requires the two images to be aligned, registered.**

Without particular knowledge about document, recto verso registration is complex. Indeed, the only information that we can use to register the two is the bleed-through. Recto verso registration is complex because the recto's bleed-through is a highly degraded version of verso's ink pixels. Therefore, in this particular context, usual image comparison methods [1] are not very relevant.

Nevertheless, document recto verso registration algorithms has been proposed [2], [3] [4], but these methods have important time computation costs, are noise sensitive and even fail in some cases where bleed-through is too light. The previous techniques are based on a pixel to pixel approach where the bleed-through is considered to be just a set of grey pixels.

In this article, we consider the structure of the ink pixels on the verso page. The recto verso registration method presented here is based on the fact that bleed-through has the same structure that the ink on the verso side. The method registers the recto's bleed-through layout and the verso's ink layout, in two main steps, first a de-skewing algorithm is applied to both pages then, horizontal and vertical profiles are extracted and aligned with a dynamic time warping. The time complexity of our method is linear according to the image size. Moreover, experiments detailed at the end show the accuracy of our method.

I. Introduction

Document recto verso registration is an important step allowing detection of missing digitized pages, or location of the bleed-through defect over a page. The bleed-through defect is a well known type of degradation. It appears on mainly very old documents and is due to an important quantity of verso's ink that can be seen from the recto side [5]. This defect is of great interest. Indeed, documents suffering from bleed-through have their readability greatly decreased. Moreover, it results in a high OCR error rate [6]. For these reasons, research has been done to measure this defect in order to predict OCR error rates [7], and other works are able to restore degraded documents [8], [2], [9]. The main issue with bleed-through is that it cannot be identified by a global thresholding method because gray levels are often too similar to the ink. Nevertheless, there exists two kinds of restoration methods, those that use both side of the document's page (the recto and the verso) [8], [2] and the ones who do not [9]. Latters restore not only bleed-through but also any present noises. Bleed-through identification techniques that use the verso side of the page, select pixels on the recto corresponding to ink pixels on the verso. But, since both sides of the page are scanned separately, the verso side may be shifted or rotated of a few millimeters from the recto side. Since the optical parameters and configuration of the scanner is fixed, scale transformations are not applied. The registration process aims to find a transformation that will align the recto and the verso.

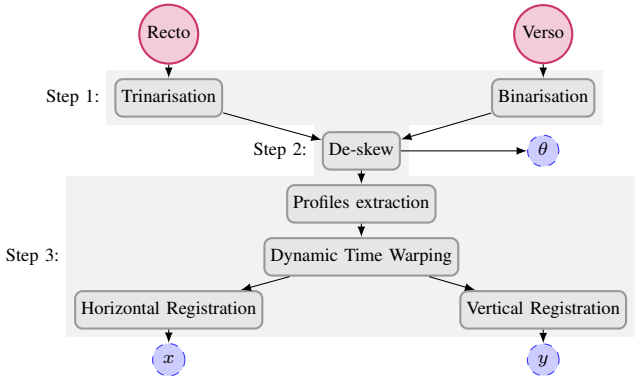


Figure 1. The overall registration method : dashed circles are the transformation parameters, θ for the rotation, x for the horizontal shift and y for the vertical shift .

Image registration methods aim at bringing two or more dataset in the same coordinate system [1]. Most of these methods are not suitable for the registration of a recto-verso pair since recto and verso have different image intensity and topography. The only relevant information that can be used in a recto verso registration is the bleed-through, but this information is a highly degraded version of verso's ink. Nevertheless, several recto-verso registration are proposed. In [2], [3] a parameter optimization method aims to find the appropriate transformation matrix that minimizes the

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.248