# Textual Content Retrieval from Filled-in Form Images

Soulib Ghosh[1], Rajdeep Bhattacharya[1](✉), Sandipan Majhi[1],
Showmik Bhowmik[1], Samir Malakar[2], and Ram Sarkar[1]

[1] Department of Computer Science and Engineering, Jadavpur University,
Kolkata, India
ghoshsoulib@gmail.com, rajdeep.cse17@gmail.com,
sandipan.majhi.email@gmail.com, showmik.cse@gmail.com, raamsarkar@gmail.com
[2] Department of Computer Science, Asutosh College, Kolkata, India
malakarsamir@gmail.com

**Abstract.** Form processing refers to the process of extraction of information from filled-in forms. In this work, we have addressed three very crucial challenges of a form processing system, namely touching component separation, text non-text separation and handwritten-printed text separation. The proposed method is evaluated on a database having 50 filled-in forms written in Bangla, collected during an essay competition in a school. The experimental results are promising.

**Keywords:** Form processing · Text non-text separation · Handwritten-printed text separation · Touching component separation · Bangla text

## 1 Introduction

Form processing refers to the process of extracting the textual information present in a filled-in form. Forms have been preferred for information collection in various departments such as railway, bank, educational organization, administrative office, etc. Forms can be divided into various categories such as orders, applications, claims and survey forms. Huge volume of such forms generated in every department makes manual processing tedious. Thus, development of an automated form processing system becomes a pressing need.

Processing of filled-in forms, however, has many challenges such as the diversity that exists in the type of the data (e.g. numerals, alphabet, etc.), occurrence of various non-text elements (e.g. tables, lines of different types, logs, etc.) and presence of both printed and handwritten text. In addition to these, two more critical problems are there, one of which appears at the text extraction level which is occurrence of touching components, and the other appears at the text recognition level due to the complexity of handwritten text. In this work, a system is developed for the extraction of text present in a filled-in form, where we have addressed three major issues namely, touching component separation, text non-text separation, and handwritten-printed text separation.

S. Sundaram and G. Harit (Eds.): DAR 2018, CCIS 1020, pp. 27–37, 2019.
https://doi.org/10.1007/978-981-13-9361-7_3

## 2 Related Work

Methods used for text non-text separation in filled-in form images are broadly classified into three groups – region based, connected component (CC) based and pixel based classification. In region based classification, the entire image is divided into regions and then each region is identified as a text or non-text region. In this category, mostly texture based features are used such as white tiles based features [1], run length based features [2], gray level co-occurrence matrix (GLCM) based features [3], etc. However, region based classification falters in cases where textual regions are highly scattered which become very challenging during segmentation. In CC level classification, each component is identified as text or non-text. In [4], Bhowmik et al. have applied Rotation Invariant Local Binary Pattern (RILBP) based features to characterize the component as text or non-text. In [5], Le et al. have studied shapes and sizes of CCs to extract effective set of features based on size, shape, stroke width and position to characterize each CC. Though component level classification provides good results in general, it fails in case of documents where components overlap each other due to poor binarization. To overcome these limitations, recently researchers have proposed pixel based classification [6] to classify each pixel as text or non-text pixel. But the main drawback of pixel based classification is its time consumption. A detailed study of text non-text separation in document image can be found in the survey paper by Bhowmik et al. [7]. In [8, 9], researchers have proposed line and table detection methods from document images. These methods require preprocessing steps and fail to provide good results for complex cases where lines or tables are not exactly vertical/horizontal.

In [10], Pal et al. have described machine-printed and handwritten text classification scheme based on statistical features for Bangla and Devanagari scripts. Chanda et al. [11] have proposed a method for separating handwritten and printed text in the perspective of sparse data and random alignment. They have used Support Vector Machine (SVM) classifier and chain-code feature vector for this purpose. The work presented in [12] consists of patch level separation and pixel level separation. Three different classes - machine printed text, handwritten text and overlapped text are initially identified using G-means (modified version of K-means) based classification followed by an MRF based relabeling procedure. In [13], Malakar et al. have extracted a 6-element feature set from each image first and then a decision tree classifier has been designed to perform the classification.

The most accurate methods are time consuming whereas the faster methods fail to provide good results in the presence of critical challenges. Textual regions in document forms are distributed throughout the entire image due to the handwritten text fields. Thus, the approach of isolating polygonal regions for textual parts will not serve the purpose due to which we avoid typical region based classification. Further pixel level classification is time consuming and hence, we restrict our method to component level classification.

# 3 Proposed Work

Entire work presented here has some key modules which are shown in Fig. 1.

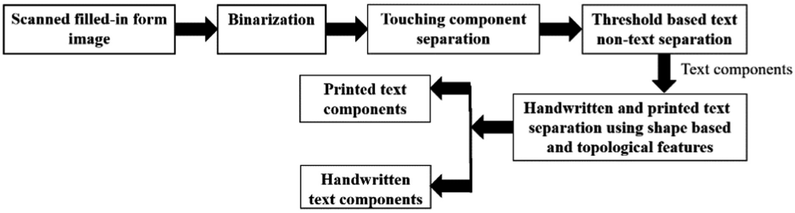

Scanned filled-in form image
Binarization
Touching component separation
Threshold based text non-text separation
Text components
Handwritten and printed text separation using shape based and topological features
Printed text components
Handwritten text components

**Fig. 1.** Flowchart of the proposed filled-in form processing method showing all the key modules.

## 3.1 Detection and Separation of Touching Components

Touching text/non-text components are very common in filled-in forms (see Fig. 2) and hence detection and thereby separation of touching components is a pressing need. A method is proposed here for the said purpose which consists of two steps. The components touching the substantially large horizontal lines, mainly occurring as parts of tables, are detected in the first step and then detached in the second step.

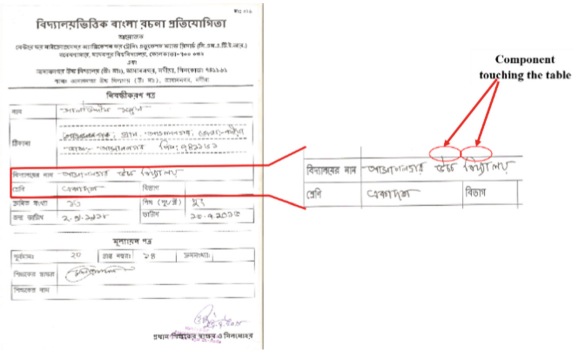

Component touching the table

**Fig. 2.** Examples of touching components in a table.

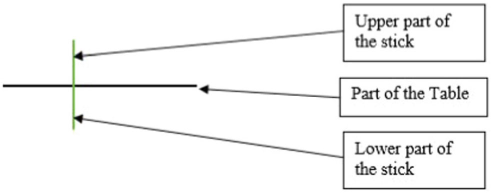
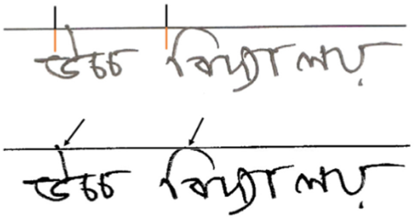


**Fig. 3.** Orientation of the stick on a line. The middle of the stick is attached to the line.



Low mean and high standard deviation of distances occurred in the lower part. This indicates the presence of touching component in the lower part of the table.

After detaching the touched component from the table. The detached portions are indicated using arrows.

**Fig. 4.** Example of detaching a touching component present in the lower portion of a table by the proposed method.

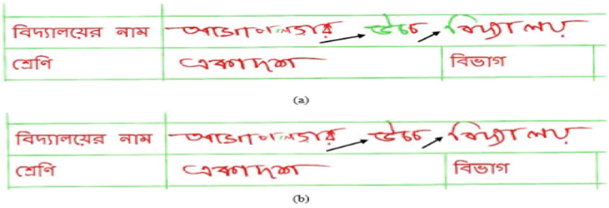


**Fig. 5.** Text non-text classification results. Denotes the classification result: (a) before applying our method, (b) after applying our method. Red denotes text and green color denotes tables. Only the zoomed portion is shown for clarity. The modified portions are indicated using arrows. (Color figure online)

For the purpose of detection, a vertical stick consisting of two parts – upper and lower, is introduced whose middle portion is always clinched to the horizontal line (see Fig. 3). The main motive of using such a stick is to place it on a horizontal line to determine the accurate position where the touching component occurs. The stick remains vertical throughout the process and is traversed on large horizontal line to calculate the nearest data pixel distance from each pixel of the stick horizontally in the forward direction. Therefore, for each position of the stick we obtain a set of distances whose mean and standard deviation (SD) are calculated. Intuitively, it can be concluded that if the mean of the distances is lower than a particular threshold, then there is a component which touches the horizontal line near the stick. We further infer that if the low distances occur on the upper part of the stick, it indicates that the component is hooked on the upper side of the horizontal line and similarly for the lower part as well. SD is additionally used to check for the cursive nature of the touching component. The proper presence of any touching component can be assured if mean is less and SD is more than the appropriate threshold values (see Fig. 4). After the detection of the touching component, that component is detached from the horizontal line (see Fig. 4).

This method takes the entire image as input and accordingly detaches all the touching components from any large horizontal line, predominantly occurring in the horizontal part of a table. The main advantage of this module is that it successfully detaches the touching component from the horizontal lines and tables to protect them from getting classified as non-text along with a table and makes information retrieval more appropriate (see Fig. 5). This, in turn, makes the text non-text classification less complex.

### 3.2 Text Non-text Separation

In this module, we aim to separate the text components from the non-text components. We have identified an exhaustive list of non-text components that may occur in a form and handled each of those separately. First, we have extracted all the CCs from an image and excluded the components whose area is very small (say less than $\alpha_1$). It is found that such components usually represent stray marks and hence are ignorable. Then for the components with area more than the said threshold value, we have performed the following operations.

**Line Separation:** For text, the lines usually appear as 'Matra' or strikethrough which occur in conjunction with a word as one single CC and, hence, are handled along with them. Here, we deal with standalone lines, which are generally used to separate one part of the document from another or act as a base for the user to write on. For separating these lines, we use the following feature.

$$\frac{\max\{height, width\}}{\min\{height, width\}} \tag{1}$$

We note that in case of a line, either the width is very large compared to the height (in case of horizontal line) or the height is very large compared to its width. We classify it as a line, and hence non-text, if this ratio is greater than a certain threshold value (say $\beta_1$) in either case.

**Table Separation:** We note that a table would cover a significant portion of the area of the form and at least one out of the height and width will be of large magnitude. Hence, according to our algorithm, a component is said to be a table if it satisfies the following condition.

$$\begin{gathered}(\text{component height} > \gamma_1 \times \text{height of the image})\\ \text{or } (\text{component width} > \gamma_2 \times \text{height of the image})\end{gathered} \tag{2}$$

Here, $0 \le \gamma_1 < 1 \, and \, 0 \le \gamma_2 < 1$

Here, the values of $\gamma_1$ and $\gamma_2$ are chosen experimentally.

**Dot Separation:** In this section, we begin by checking if the area of the component is in a certain range that most dots satisfy. If so, we check if both the height and width are greater than a certain number of pixels (say $\in$), which is chosen to be sufficiently large, as otherwise, even a small line might satisfy the area threshold. If it satisfies the above conditions, then the component can be classified as a dot. Now, a dot can be text or non-text. For example, in Fig. 6, the Bangla letter contains a dot as descender. For this, we have considered the context information of an identified dot. We have measured the average CC height $\sigma$ after table and line separation, and scanned $\sigma$ number of pixels above and below the dot. If we find sufficient number of data pixels (greater than a certain threshold $\tau$), it is classified as a text dot, otherwise non-text dot.

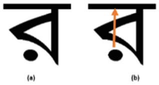


**Fig. 6.** (a) Dot as a part of text (b) scanning of upper region for context information.

**Strikethrough Separation:** If the component is not small enough to be a dot as well, then we check for strikethrough. The feature used for this is mainly Euler number as we note that for a skeleton of a strikethrough, the Euler number becomes low. For this, we first swell the component up so that any negligible gap in the handwriting is filled up. The parameters for swell are so given that only the ink gaps which are left in the course of writing are filled and not the larger ones. Then we take the morphological skeleton of the component. We note that for a strikethrough on a word or character, it intersects the same at quite a few points and hence these intersections create many holes. This decreases the Euler number significantly for the strikethrough components. An example is provided in Fig. 7 for reference.

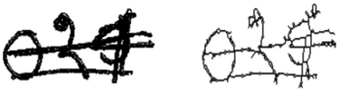


**Fig. 7.** A strikethrough component and its skeleton after it is swelled up.

As it can be understood from the image, the Euler number of the second component is less as the number of holes in it is very high. Thus, we set a threshold (say, $\alpha_2$) where if the Euler number of this skeleton is less than $\alpha_2$, then the component is classified as strikethrough and hence non-text, else text.

**Bracket Separation:** If the component satisfies none of the above checks, then we examine whether it is a bracket or not. We again swell up the component and take the morphological skeleton of it. This step reduces the stroke width significantly and ensures that the thickness of the bracket is not an issue for further processing. Then we analyze the component row-wise. If in any row, we find that the number of data pixels crosses a certain threshold (say $\beta_2$), then we classify it as a non-bracket, else bracket. This is illustrated in Fig. 8.

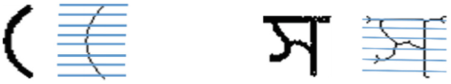


**Fig. 8.** Demonstration of row-wise scanning of a skeletonized bracket and non-bracket component. Here, for the bracket, no row encounters a significant number of data pixels. This is because a bracket is generally written in a single stroke as shown.

### 3.3 Handwritten and Printed Text Separation

In this module of our work, an effort has been made to successfully segregate the handwritten and printed text present in the form images. For this, following features are extracted from the components.

**Otsu Threshold:** Printed text has a fixed typeset and uniform pixel intensity, so Otsu threshold is supposed to be almost similar for all of them, and these values for all printed text should converge to a fixed value. The same is not true for handwritten text because of non-uniformity in ink flow, hand pressure etc.

**SD of Pixel Intensities:** The gray scale image of a CC of printed text is assumed to be having mostly similar pixel intensity which is not true for handwritten text. Hence, SD of the pixel intensities of printed text becomes less than the SD of handwritten text.

**Entropy:** Entropy is a statistical measure which is viewed as the amount of information available in a data distribution. Handwritten text contains more variations in gray level values than printed text and hence possess less entropy value. It is so because printed text attains almost same pixel intensity patterns of the source, which is the typeset of the document processor. Entropy (H) of a gray scale image can be computed from its histogram counts $p_n$ by the Eq. (3).

$$H = -\sum_{n=0}^{255} p_n \log_2 p_n \tag{3}$$

**Stroke Width:** As the format of printed text are set beforehand in a document processor, the stroke width remains fairly fixed with respect to the set value. However, this is not true for handwritten text. Hence, stroke width is considered in four different directions for the said analysis. The four directions that are considered are horizontal, vertical, left diagonal and right diagonal (see Fig. 9). In this context, the mean and SD of the stroke widths of all directions are taken.

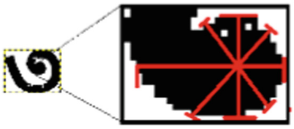


**Fig. 9.** Determination of stroke width along four directions. The red lines correspond to the continuity in pixels along a particular direction. (Color figure online)

**Distribution of Contour to Boundary Pixel Maps:** Some variations in the curvatures and orientations of the strokes are found while writing same symbol (see Fig. 10). Therefore, in order to include these variations, the said features are considered. From each pixel in the contour of a CC, the minimum distance from each of the boundaries of the component is measured. The closest boundary from a particular pixel gets a contribution of 1 while other get a contribution of 0. In this way, the pixels are distributed among the four boundaries (see Fig. 11). At last, the SDs of these distributions along each boundary are taken as features.

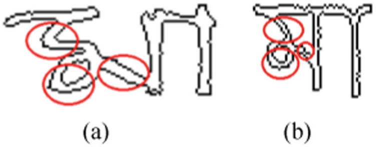


**Fig. 10.** Same word written in (a) handwritten form. (b) printed form. Red circular regions denote the differences in shapes and curvature, between (a) & (b) in similar regions. (Color figure online)

## 4 Experimental Results

As there is no publicly available dataset related to filled-in form in *Bangla* script, hence a dataset is prepared for the evaluation of the proposed method which consists of 50 forms. A sample form image taken from our database is shown in Fig. 12a and the corresponding ground truth (GT) image is shown in Fig. 12b. The GT for each scanned form is created manually. To assess the performance of text non-text separation module, the final outputs after applying our methods are evaluated with the previously produced GTs. For evaluating handwritten-printed text classification, CCs from the last 30 forms are considered to train a *Random Forest* classifier, whereas the CCs of the first 20 forms are used for testing. A total of 42,324 CCs are present in these collected forms, out of which non-text, printed and handwritten CCs are 4784, 18437 and 19103 in number respectively.

After exhaustive experimentation over a variety of data, the following thresholds have been set optimally: $\alpha_1 = 6$, $\beta_1 = 50$, $\gamma_1 = 0.16$, $\gamma_2 = 0.4$, $\in = 7$, $\sigma = 40$, $\tau = 3$, $\alpha_2 = -9$ and $\beta_2 = 5$. Also, the range of area referred to in dot separation is 50 to 100.

To evaluate the proposed method, we have considered three popular metrics *Precision, Recall* and *Accuracy* [7]. Detailed results for each of these modules are described in Table 1. From this table, it can be observed that our proposed method performs satisfactorily. It is to be noted that all these measurements are done at CC level. A sample image and its corresponding text non-text separated and handwritten-printed text separated images are shown in Fig. 13.

Also, some of the erroneous results are shown in Fig. 14. The touching component module fails when a text component touches a vertical line of any table. Text non-text separation module falters in the situation of a text overlapping with a stamp which causes a sufficient decrease of Euler number and results in the component being misclassified as a strikethrough. In case of very less cursive and non-complex handwritten text, it is also misclassified as printed text.

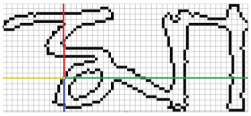


**Fig. 11.** The intersection point of the four colored lines denote the concerned pixel. Red, green, blue and yellow lines denote the distances of that pixel from the top, right, bottom, and left boundaries respectively. As it can be seen, the blue line is of smallest length and hence the contribution of that point for the bottom boundary is counted as 1 and for the rest as 0. (Color figure online)

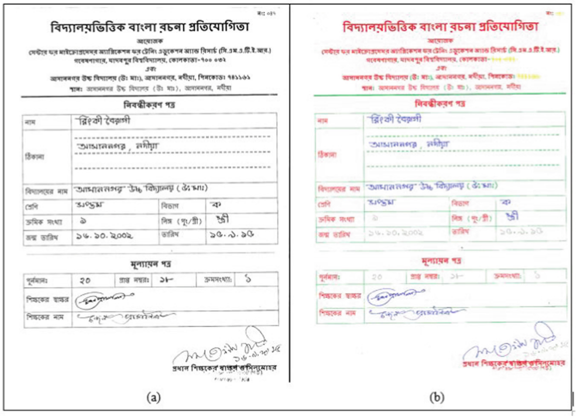
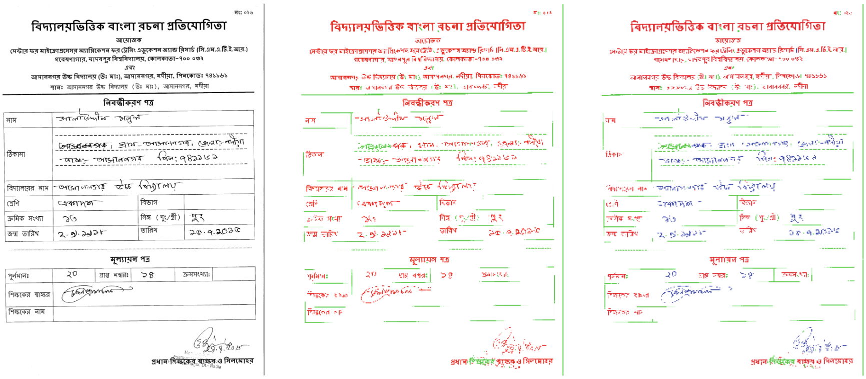


**Fig. 12.** (a) Sample image taken from our dataset, (b) corresponding GT. Here red represents printed text, blue represents handwritten text, green represents non-text, yellow represents printed numerals and gray represents handwritten numerals. (Color figure online)

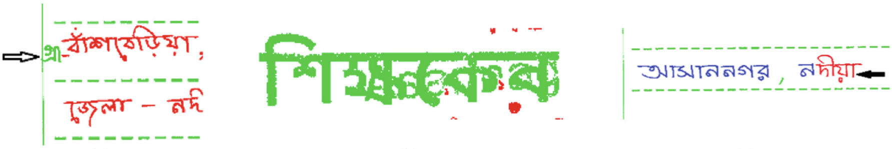
**Table 1.** Detailed experimental outcomes obtained by the different modules of the proposed method

| Module | Precision | Recall | Accuracy |
|---|---|---|---|
| Text/Non-text separation | 92.37% | 90.47% | 86.03% |
| Text/Non-text separation + Touching separation | 93.27% | 91.55% | 87.65% |
| Handwritten-printed separation | 95.8% | 98.7% | 96.2% |



(a) (b) (c)

**Fig. 13.** Example of final output obtained by our proposed method. (a) sample input image, (b) corresponding text non-text separated image (c) corresponding handwritten-printed text separated image. Green color denotes non-text. Red color denotes text part in text non-text separated image and printed text in the final output whereas blue color denotes handwritten text in the final output. (Color figure online)



(a) (b) (c)

**Fig. 14.** Examples of several errors encountered while using our proposed method. Error encountered in (a) Touching component separation (b) Text non-text separation (c) Handwritten-printed separation module. Green color denotes non-text. In (c), Red color denotes printed component and Blue color denotes handwritten component. (Color figure online)

## 5 Conclusion

In this paper, we propose a text extraction technique from filled-in form images written in *Bangla* script. It comprises a novel touching component elimination method and a hybrid approach to decide trade-off between threshold and machine learning based approaches. Along with that some useful shape based and topological features are used for the separation of handwritten-printed text separation. All the used features are calculated without using any language heuristic. Hence, the method can also be applied to filled-in forms written in other languages. We plan to make use of dynamic values of certain parameters in the next scope of our work. The method for eliminating touching components can also be modified further to handle both horizontally and vertically touched components.

**Acknowledgement.** S. Bhowmik is thankful to Ministry of Electronics and Information Technology, Govt. of India, for providing PhD-Fellowship under Visvesvaraya PhD scheme.

## References

1. Antonacopoulos, A., Ritchings, T.R., De Tran, C.: Representation and classification of complex-shaped printed regions using white tiles. In Proceedings of the Third International Conference on Document Analysis and Recognition, Montreal, QC, Canada, 14–16 August 1995, vol. 2, pp. 1132–1135 (1995)
2. Park, H.C., Ok, S.Y., Cho, H.: Word extraction in text/graphic mixed image using 3-dimensional graph model. In: Proceedings of the ICCPOL, Tokushima, Japan, 24–26 March 1999, vol. 99, pp. 171–176 (1999)
3. Oyedotun, O.K., Khashman, A.: Document segmentation using textural features summarization and feedforward neural network. Appl. Intell. **45**, 198–212 (2016)
4. Bhowmik, S., Sarkar, R., Nasipuri, M.: Text and non-text separation in handwritten document images using local binary pattern operator. In: Mandal, J., Satapathy, S., Sanyal, M., Bhateja, V. (eds.) International Conference on Intelligent Computing and Communication, pp. 507–515. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-2035-3_52
5. Le, V.P., Nayef, N., Visani, M., Ogier, J.-M., De Tran, C.: Text and non-text segmentation based on connected component features. In: Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, Tunisia, 23–26 August 2015, vol. 45, pp. 1096–1100 (2015)
6. Garz, A., Sablatnig, R., Diem, M.: Layout analysis for historical manuscripts using sift features Document. In: Proceedings of the 2011 International Conference on Document Analysis and Recognition (ICDAR), Beijing, China, 18–21 September 2011 (2011)
7. Bhowmik, S., Sarkar, R., Nasipuri, M., Doermann, D.: Text and non-text separation in offline document images: a survey. Int. J. Doc. Anal. Recognit. (IJDAR) **21**(1–2), 1–20 (2018)
8. Gatos, B., Danatsas, D., Pratikakis, I., Perantonis, S.J.: Automatic table detection in document images. In: Singh, S., Singh, M., Apte, C., Perner, P. (eds.) ICAPR 2005. LNCS, vol. 3686, pp. 609–618. Springer, Heidelberg (2005). https://doi.org/10.1007/11551188_67
9. Arvind, K.R., Kumar, J., Ramakrishnan, A.G.: Line removal and restoration of handwritten strokes. In: International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007), Sivakasi, Tamil Nadu, India. IEEE (2007)
10. Pal, U., Chaudhuri, B.B.: Machine-printed and handwritten text lines identification. Pattern Recogn. Lett. **22**(3–4), 431–441 (2001)
11. Chanda, S., Franke, K., Pal, U.: Structural handwritten and machine print classification for sparse content and arbitrary oriented document fragments. In: Conference: Proceedings of the 2010 ACM Symposium on Applied Computing (SAC), Switzerland, 22–26 March 2010 (2010)
12. Peng, X., Setlur, A., Govindaraju, V., Sitaram, R.: Handwritten text separation from annotated machine printed documents using Markov random fields. Int. J. Doc. Anal. Recogn. (IJDAR) **16**(1), 1–16 (2013)
13. Malakar, S., Das, R.K., Sarkar, R., Basu, S., Nasipuri, M.: Handwritten and printed word identification using gray-scale feature vector and decision tree classifier. Procedia Technol. **10**, 831–839 (2013)