Text Detection in Document Images by Machine Learning Algorithms

Darko Zelenika[1], Janez Povh[1], and Bernard Ženko[2]

[1] Faculty of Information Studies, Laboratory of Data Technologies,
Ulica talcev 3, SI-8000 Novo mesto, Slovenia
{darko.zelenika,janez.povh}@fis.unm.si
[2] Jožef Stefan Institute, Department of Knowledge Technologies,
Jamova cesta 39, SI-1000 Ljubljana, Slovenia
bernard.zenko@ijs.si

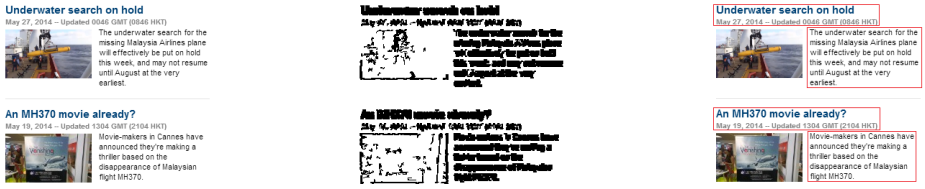
Abstract. In the proposed paper we consider a problem of text detection in document images. This problem plays an important role in OCR systems and is a challenging task. In the first step of our proposed text detection approach we use a self-adjusting bottom-up segmentation algorithm to segment a document image into a set of connected components (CCs). The segmentation algorithm is based on the Sobel edge detection method. In the second step CCs are described in terms of 27 features and a machine learning algorithm is then used to classify CCs as text or non-text. For testing the approach we have collected a dataset (ASTRoID), which contains 500 images of text blocks and 500 images of non-text blocks. We empirically compare performance of the proposed text detection method when using seven different machine learning algorithms.

Keywords: text detection, document segmentation, text/non-text classification, machine learning

1 Introduction

In today's digital age a lot of useful information is present as text in the form of digital document images such as invoices, business letters, web pages, etc. In order to effectively recognize this text with Optical Character Recognition (OCR) technology, location of the text must be detected first. The first step of text detection in document images involves segmentation, which is followed by classification. Document segmentation is a task which splits a document image into blocks of interest, as shown in Fig. 1b where each connected component (CC) of black pixels represents one block. Blocks of interest usually appear in two forms: text and non-text. In this paper we are mainly interested in text blocks, so our goal is to identify them and separate from non-text blocks (see Fig. 1c). It is important to understand that document segmentation and classification (identification) of segmented blocks can hardly be separated and are often treated together as "(physical) layout analysis" [1]. These tasks continue to

be very challenging, especially for documents which have multi colored and complex background. Text in such documents may be of different sizes, orientations, colors, etc. Most of the currently available document segmentation algorithms require some predefined parameters due to different font sizes, layouts and document image resolutions. Hence, robust and efficient techniques for document segmentation are required.



(a) (b) (c)

Fig. 1: a) Image document, b) Segmented image and c) Detected text

1.1 Text detection method

The purpose of this paper is to introduce and report results of a new text detection method. Our method consists of document segmentation and classification algorithms. The segmentation algorithm is based on bottom-up approach, which is obtained with edge detection methods. We created a custom dataset of text and non-text image blocks, called ASTRoID. From each image block we extracted 27 features, which are used by a machine learning algorithm to separate text from non-text image blocks; we test seven different machine learning algorithms. This work can be seen as an important step towards a parameter free document segmentation and accurate text detection in complex text documents.

1.2 Document segmentation techniques

Document segmentation techniques are traditionally classified into three categories: top-down, bottom-up and hybrid approaches. The top-down approach starts the segmentation from bigger blocks and repetitively segments the document image into smaller blocks until the document image is segmented into the smallest possible blocks. Top-down algorithms are usually fast but they tend to fail in segmenting documents with very complex layouts [2]; typical examples are X-Y Cuts [2–4], White streams [5], Run-length smearing [6] and other algorithms based on projection profile method [7, 8]. Kruatrachue et. al [4] used X-Y Cuts to build a fast segmentation method, but it is not well suited for complex documents because it's based on binarization.

The bottom-up approach is the opposite of the top-down approach. It starts from the smallest segments (characters) and then joins them into bigger and bigger blocks (words, paragraphs, etc.). Algorithms based on bottom-up approach

are flexible and robust but often slow because of time-consuming operations, some of these algorithms are based on the analysis of CCs [9–11] and some on morphological operations [12, 13].

An approach that does not fit into a top-down or bottom-up strategy or uses the combination of both is called a hybrid approach [14, 15]. Hybrid approaches often try to combine the speed of top-down approaches and the robustness of bottom-up approaches. In [15] authors used an adaptation of Scale Invariant Feature Transform (SIFT) approach for text character spotting in graphical documents. Their method uses a combination of bottom-up and top-down approaches to separate and locate text characters.

1.3 Classification

The task of classification algorithm is to classify the results of segmentation algorithm. Classification highly depends on the quality of segmentation algorithm. In [2] authors proposed a method to extract illustrations from digitized historical documents by using Support Vector Machine (SVM). Priyadharshini and Vijaya in [6] proposed a document block classification approach, which classifies the document blocks as text, image, drawing and table. In their approach a genetic programming based classifier is used to classify document blocks. In order to detect text and non-text blocks authors in [5, 9–11] extracted features from CCs in document images and classified them with machine learning algorithms. In this paper the proposed text detection method uses the similar approach.

1.4 Contribution

The main contributions of this paper are: (1) a self-adjusting segmentation algorithm for finding text CCs that is independent of the image resolution and font size, (2) a new set of features which describe differences between text and non-text image blocks based on information about their shape and context, (3) a custom benchmark dataset ASTRoID of text and non-text image blocks, and (4) demonstration of performance of seven machine learning algorithms for separation between text and non-text image blocks.

The rest of the paper is organized as follows. In section two, we introduce document segmentation algorithm, ASTRoID dataset and classification algorithm. The performance of our approach for five different machine learning algorithms is presented in section three. Obtained results are discussed in section four. Finally, section five concludes the paper.

2 Materials & Methods

Here we describe our text detection method, which performs two tasks: document image segmentation and classification. The text detection method extracts features from the segmented blocks (results of the segmentation algorithm), which are then classified with a machine learning algorithm as either text or non-text blocks.

2.1 Segmentation

The segmentation algorithm described in this paper follows the bottom up strategy. The algorithm segments document into small CCs, which are constructed with a combination of Sobel edge detection and dilation method [1, 16]. The proposed document image segmentation algorithm is composed of three parts: (1) finding an optimal rectangular kernel, (2) edge detection and (3) extraction of standalone document image objects. Before the segmentation process begins, the document image needs to be converted to grayscale (Fig. 2a). The segmentation algorithm receives the grayscale image and outputs the binary image.

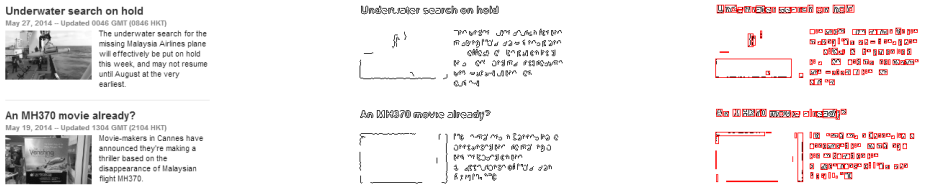
The first part of the segmentation algorithm tries to find an optimal rectangular kernel, which is then used by the other two parts of the algorithm. The optimal rectangular kernel highly depends on the height of the dominant text in the document image ($h_{mainText}$). In order to find the height of the main text we applied the Sobel edge detection algorithm over the grayscale document image (Fig. 2b). The height of the main text is obtained from the heights of CCs, which are extracted from the binary image. After the height is obtained we calculate the number of columns of optimal rectangular kernel based on the following *equation*:

$$N_{col} = \frac{height_{mainText}}{4} \tag{1}$$

The result of *equation* 1 is rounded and the optimal rectangular kernel is obtained (*equation* 2), which is a matrix of size $2XN_{col}$ containing only $1's$, which is going to be used to better emphasize textual objects on the document image.

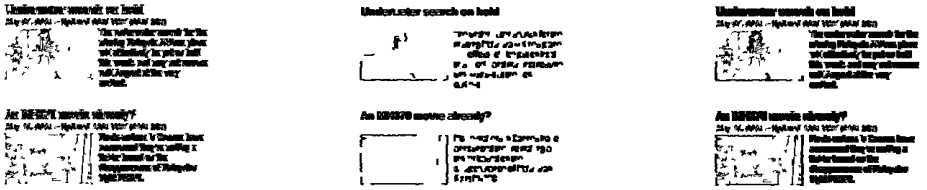
$$K_{opt} = \begin{bmatrix} 1\ 1 \ldots 1 \\ 1\ 1 \ldots 1 \end{bmatrix} \tag{2}$$

In the second part of the segmentation algorithm we apply different Sobel kernels (vertical, horizontal and diagonal) on the grayscale document image. Text blocks are composed of vertical, horizontal and diagonal edges, and accordingly, in this part of the segmentation algorithm we used only a combination of vertical and diagonal Sobel kernels of different orientations to better emphasize text blocks. The result binary images of different Sobel kernels are dilated by the optimal rectangular kernel and combined into one image by logical AND and OR operations, as shown in Fig. 3a. In the third part of the segmentation algorithm we again use the image with detected Sobel edges from Fig. 2b. We localize all CCs (red rectangles in Fig. 2c) on this image and keep only those CCs that intersect with two or less than two other CCs (due to characters such as "B" and "8") and we call them standalone CCs. In such a way most of CCs which do not belong to text blocks are removed. The result image of the third part of segmentation algorithm is binary image with standalone CCs, which is dilated by the optimal rectangular kernel (see Fig. 3b). The final part of the segmentation algorithm is to combine the obtained binary images from the second and third parts of the algorithm with logical OR operation (see Fig. 3c). In such a way we segmented the document image into blocks that can either be text blocks or non-text blocks.



(a) (b) (c)

Fig. 2: a) Grayscale image, b) Sobel edges and c) Localized CCs



(a) (b) (c)

Fig. 3: Segmentation steps, a) 2nd part, b) 3rd part and c) Final part

2.2 Text vs. non-text classification

ASTRoID dataset. We created our custom dataset of text and non-text image blocks, which we called ASTRoID, in order to evaluate our method. We were choosing at random articles from ten web portals (one article per portal), took screenshots of full web pages and saved them as PNG (Portable Network Graphics) image files. These ten web portals were: abcnews.go.com, cnn.com, nationalgeographic.com, pcmag.com, telegraph.co.uk, racunalniske-novice.com, radio1.si, slovenskenovice.si, bljesak.info and dnevnik.hr, they include five portals in English language, three in Slovenian language and two in Croatian language. We used the proposed segmentation algorithm over all ten article images to extract all CCs and save them as PNG image files. For our purposes we manually chose 500 of image blocks which contain plain text of different size, length, color, font style, etc., and 500 image blocks of different size which do not contain text. It is important to state that our dataset has 150 image blocks of text that contain only one or two characters, which some avoid to use in order to get better classification results. The ASTRoID dataset is available to download at: `http://dk.fis.unm.si/ASTRoID.zip`.

Feature extraction. In document images most of the text blocks are uniformly structured and have a regular shape. On the other hand, non-text blocks have a lot of variability in shape, i.e., mostly they have an irregular shape. But only shape information is not enough to classify text from non-text blocks, we also need to take into account the information on the context of these blocks. Therefore, we need to create a set of features that describe the context and are able to

improve the accuracy of distinguishing text from non-text blocks. In this paper we used features similar to the ones proposed in [5, 6, 9, 10]. We took different approach to calculate some of the proposed features and introduced a new feature "color density". In our approach we extract features from multiple images of the same segmented block (Fig. 5), which are obtained by different methods, unlike to the approaches found in the literature where features are extracted from usually one binary image. Before the actual feature extraction each segmented block is resized to 100 pixels in height while maintaining the width to height aspect ratio.

In our approach we used the following features: number of CCs [5], aspect ratio [5, 6, 9], foreground density [5, 6, 9, 10], color density, standard deviation of the heights and widths of CCs [6, 9, 10], and standard deviation of the lengths of horizontal and vertical runs [6]. Most of the features are extracted from the different binary images (Fig. 5b - 5h), which are obtained with the following methods: skeletonization (Fig. 5d) [18], horizontal Sobel kernel (Fig. 5e), vertical Sobel kernel (Fig. 5f), diagonal Sobel kernels (Fig. 5g) and Canny edge detection method (Fig. 5h) [16, 19].

Number of CCs is computed after binarization of grayscale image by using Otsu's [17] thresholding algorithm (Fig. 5b). Aspect ratio (A_r in *equation* 3) is defined as the ratio of a block's width-to-height if height is greater than width or height-to-width if width is greater than height.

$$A_r = \frac{w}{h} or A_r = \frac{h}{w} \tag{3}$$

The feature foreground density (D_f in *equation* 4) is defined as the ratio of the number of the foreground (black) pixels to the total number of pixels in the binary image, and is calculated from two binary images (Fig. 5b and 5h).

$$D_f = \frac{N_f}{N} \tag{4}$$

In order to determine the feature color density we first extract the most frequent colors in the color image (Fig. 5a). The color density (D_c in *equation* 5) is defined as the ratio of the number of extracted colors to the total number of colors in the color image. We created binary image based on the location (coordinates) of extracted colors (Fig. 5c), which will be used for extraction of other features, by filling coordinates of extracted colors in new binary image with white pixel (background) and all the remaining coordinates with black pixel (foreground).

$$D_c = \frac{N_{extractedC}}{N_{totalC}} \tag{5}$$

Based on the extracted CCs from the binary images we use heights and widths of each CC to calculate the features standard deviation of the heights and widths of CCs. Standard deviation of the heights of CCs is calculated from the Fig. 5b, 5c, 5d, 5f, 5g and 5h, and from the Sobel vertical lines which are extracted from the Fig. 5c and 5h. Standard deviation of the widths of CCs is caluclated

from the Fig. 5e and from the Sobel horizontal lines which are extracted from the Fig. 5h. Also the binary images are used to calculate the features standard deviation of the lengths of the vertical and horizontal runs of black (foreground) pixels. The extraction of lengths of horizontal runs can be explained by using the following matrix [00110011110011100] where 0 is white and 1 black pixel. The lengths of horizontal runs of black pixels are 2, 4 and 3. Standard deviation of the vertical runs is calculated from the Fig. 5b, 5c, 5d, 5f, 5g and 5h and from the Sobel vertical lines which are extracted from the Fig. 5c, 5g and 5h. Standard deviation of the horizontal runs is calculated from the Fig. 5b and 5e, and from the Sobel horizontal lines which are extracted from the Fig. 5h.

And finally, by using features described above each image block (text or non-text) is represented with the feature vector which consists of 27 features in total.

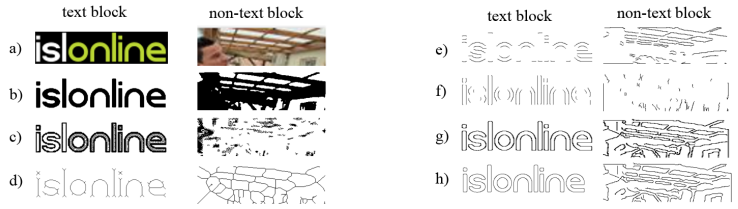


Fig. 4: a) Color image, b) Binary image, c) Image of extracted colors, d) Skeleton image, e) Horizontal Sobel, f) Vertical Sobel, g) Diagonal Sobel and h) Canny

Classification with machine learning. As stated above, based on the proposed segmentation algorithm we created ASTRoID dataset which contains 500 images of text blocks and 500 images of non-text blocks. We extracted 27 features (described in previous section) from each image block of the dataset and appointed class label (text or non-text) to them. In this way we created a dataset, which we used for classification with seven popular machine learning algorithms in order to evaluate our choice of our features, and to find out which machine learning algorithm works best with our text detection method. We used: Naïve Bayes, C4.5 (decision tree), k-Nearest Neighbors (k-NN), Random Forest, Linear Support Vector Machine (SVM), Polynomial SVM and Radial SVM. The accuracy of each of the above mentioned algorithms is estimated by 10-fold cross-validation. We tuned the parameters of some of the machine learning algorithms with an internal cross-validation. We used the implementations of machine learning algorithms in the WEKA data mining suite [20]. Naïve Bayes and C4.5 (decision tree) algorithms are used with default parameters while parameter k for the k-NN algorithm is tuned to 1 and parameter number of trees for the Random forest algorithm is tuned to 100. For SVMs we normalized data by tuning normalize parameter. The parameter C for Linear SVM is tuned to

15. The parameters C, degree, gamma and coefficient for Polynomial SVM are tuned to 20, 3, 1 and 1, respectively. The parameters C and gamma for Radial SVM are tuned to 20 and 1, respectively.

3 Results

The classification results of all machine learning algorithms are shown in Table 1. The accuracy of all machine learning algorithms is higher than 90% which suggests that the choice of our features for text/non-text differentiation is appropriate. The classification results show that Random forest and SVMs perform best for our text detection method. The machine learning algorithm that has the highest accuracy 98.2% is Radial SVM.

Classifier	Naive Bayes	k-NN	C4.5 - decision tree	Random forest	Lin. SVM	Poly. SVM	Rad. SVM
Accuracy	90.1%	93.6%	94.3%	97%	97%	97.3%	98.2%
Precision of text blocks	0.860	0.911	0.937	0.976	0.964	0.961	0.978
Precision of non-text blocks	0.953	0.964	0.949	0.964	0.976	0.986	0.986
Recall of text blocks	0.958	0.966	0.950	0.964	0.976	0.986	0.986
Recall of non-text blocks	0.844	0.906	0.936	0.976	0.964	0.960	0.978

Table 1: Classification results

4 Discussion

The results obtained by the proposed text detection method are promising. The chosen set of features, by which machine learning algorithms can separate text from non-text image blocks with good accuracy seems appropriate. Other authors who worked on similar problems also obtained comparable classification results. In [11] authors used SVM and classified text from non-text blocks with the accuracy of 96.62%. In [6] authors obtained 97.5% classification accuracy by using genetic programming to classify the document blocks as text, image, drawing and table. In [9] authors used Multilayer Perceptron to classify text from non-text blocks and obtained 97.25% of accuracy. The only disadvantage of approaches in [6, 9, 11] is that they fail to detect text on documents with complex layout, due to their segmentation algorithm. The advantage of our segmentation algorithm is that it self-adjusts to the document image regardless the image resolution and font size, it does not need any input parameters and it is able to segment documents with complex layout. The only disadvantage is that it fails to detect (segment) some text blocks of very light text color, and also some text blocks with very complex (with a lot of details) background and very decorated text strings. In the future we plan to improve our method in order to avoid current disadvantages by using Canny together with the Sobel edge detection method. We also plan to increase the size of our ASTRoID dataset and test our text detection method on other datasets and compare it with other methods.

5 Conclusion

In this paper we presented a text detection method, which consists of document segmentation and feature extraction algorithms. The proposed segmentation algorithm is based on the bottom-up strategy of analysis and segmentation is done by using the Sobel edge detection method. We created ASTRoID dataset of images, which consists of 500 image blocks of text and 500 image blocks of non-text. It is important to state that the dataset has 150 text blocks that contain either one or two characters, which some avoid to use to get better classification results. In order to classify text from non-text blocks we used seven machine learning algorithms. Before classification we first extracted features from image blocks, which are based on information about their shape and context. We used 27 different features in order to differentiate text from non-text regions. The accuracy of all machine learning algorithms is higher than 90% which suggests that the choice of our features is appropriate. The classification results show that Random forest and SVMs are the best choices for our text detection method. SVM with radial kernel has the highest accuracy 98.2%.

Acknowledgments. The presented work was supported by Creative Core FISNM-3330-13-500033 'Simulations' project funded by the European Union, The European Regional Development Fund. The operation is carried out within the framework of the Operational Programme for Strengthening Regional Development Potentials for the period 2007-2013, Development Priority 1: Competitiveness and research excellence, Priority Guideline 1.1: Improving the competitive skills and research excellence.

References

1. Kise, K.: Page Segmentation Techniques in Document Analysis. Handbook of Document Image Processing and Recognition, 135–175 (2014)
2. Coppi , D., Grana, C., Cucchiara , R.: Illustrations Segmentation in Digitized Documents Using Local Correlation Features. In: 10th Italian Research Conference on Digital Libraries, pp. 76–83. Procedia Computer Science, Vol. 38, Padua, Italy (2014)
3. Shafait, F., Keysers, D., Breuel, T.: Performance evaluation and benchmarking of six-page segmentation algorithms. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 941–954. IEEE Press (2008)
4. Kruatrachue, B., Moongfangklang, N., Siriboon, K.: Fast Document Segmentation Using Contour and X-Y Cut Technique. In: The Third World Enformatika Conference. WEC (5), pp. 27–29. Turkey (2005)
5. Barlas, P., Kasar, T., Adams, S., Chatelain, C., Paquet, T.: A typed and handwritten text block segmentation system for heterogeneous and complex documents. In: 11th IAPR International Workshop on Document Analysis Systems, pp. 46–50, IEEE Press, Tours, France (2014)
6. Priyadharshini, N., Vijaya, M.S.: Genetic Programming for Document Segmentation and Region Classification Using Discipulus. International Journal of Advanced Research in Artificial Intelligence, Vol. 2, 15–22 (2013)

7. Priyanka, N., Pal, S., Mandal, R.: Line and Word Segmentation Approach for Printed Documents. International Journal of Computers and Applications 1, 30–36 (2010)
8. Vikas , J.D., Vijay , H.M.: Devnagari Document Segmentation Using Histogram Approach. International Journal of Computer Science, Engineering and Information Technology 1, 46–53 (2011)
9. Bukhari, S.S., Azawi, M.A., Shafait, F., Breuel, T.M.: Document image segmentation using discriminative learning over connected components. In: 9th IAPR International Workshop on Document Analysis Systems, pp. 183–190. Boston, MA, USA (2010)
10. Bukhari, S.S., Asi, A., Breuel, T.M., El-Sana, J.: Layout analysis for arabic historical document images using machine learning. In: International Conference on Frontiers in Handwriting Recognition, pp. 639–644. (2012)
11. Zagoris, K., Chatzichristofis, S.A., Papamarkos,N.: Text Localization using Standard Deviation Analysis of Structure Elements and Support Vector Machines. EURASIP Journal on Advances in Signal Processing 47, 1–2 (2011)
12. Bukhari, S.S., Shafait, F., Breuel, T.M.: . Improved Document Image Segmentation Algorithm using Multiresolution Morphology. In: 18th Document Recognition and Retrieval Conference, pp. 1–10., San Jose, CA, USA (2011)
13. Sumathi , C.P., Priya , N.: A Combined Edge-Based Text Region Extraction from Document Images. International Journal of Advanced Research in Computer Science and Software Engineering, 827–835 (2013)
14. Kundu, M.K., Dhar, S., Banerjee, M.: A New Approach for Segmentation of Image and Text in Natural and Commercial Color Document. In: Proc. of International Conference on Communication, Devices and Intelligent Systems, pp. 85–88. IEEE Press, India (2012)
15. Roy, P.P., Pal, U., Lladós, J.: Touching Text Character Localization in Graphical Documents Using SIFT. In: Proceedings of the 8th International Conference on Graphics Recognition: Achievements, Challenges, and Evolution, pp. 199–211. Springer-Verlag, France (2010)
16. Vasuki, S., Ganesan , L.:) Performance Measure for Edge Based Color Image Segmentation in Color Spaces. In: Proceedings of the International Conference on Emerging Technologies in Intelligent System and Control: Exploring, Exposing, and Experiencing the Emerging Technologies, pp. 621–626. Allied Publishers, Coimbatore (2005)
17. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man and Cybernetics 9, 62–66 (1979)
18. Basilis , G.G.: Imaging Techniques in Document Analysis Processes. Handbook of Document Image Processing and Recognition, 73–131 (2014)
19. Burger, W., Burge, M.J.: Principles of Digital Image Processing. Springer, London (2009)
20. WEKA (Open source, Data Mining software in Java), University of Waikato, New Zealand, `http://www.cs.waikato.ac.nz/ml/weka`