Pattern Anal Applic (2015) 18:651–665
DOI 10.1007/s10044-014-0412-8

INDUSTRIAL AND COMMERCIAL APPLICATION

Image-based logical document structure recognition

Grzegorz Kamola · Michal Spytkowski ·
Mariusz Paradowski · Urszula Markowska-Kaczmar

Received: 7 July 2013 / Accepted: 7 September 2014 / Published online: 25 September 2014

Abstract The paper presents a complete solution for recognition of textual and graphic structures in various types of documents acquired from the Internet. In the proposed approach, the document structure recognition problem is divided into sub-problems. The first one is localizing logical structure elements within the document. The second one is recognizing segmented logical structure elements. The input to the method is an image of document page, the output is the XML file containing all graphic and textual elements included in the document, preserving the reading order of document blocks. This file contains information about the identity and position of all logical elements in the document image. The paper describes all details of the proposed method and shows the results of the experiments validating its effectiveness. The results of the proposed method for paragraph structure recognition are comparable to the referenced methods which offer segmentation only.

Keywords Logical document structure recognition · Graphics recognition · Table recognition · Text recognition

1 Introduction

In the last ten years, automatic document structure recognition has been intensively researched and developed. It is an essential component of automatic document processing, especially in document digitization. It is very important for transformation of old documents into electronic form, e.g., medical documents [17], scanned administrative documents [3] or legal articles [2]. In these cases, the proposed approaches are domain specific, i.e. a set of possible document layouts is not very huge and can be predicted. This is not the case in NEKST project[1].

1.1 Motivations

The research presented in this paper is a part of the NEKST project. Its final goal is to create a general question answering system, i.e. a system which gives appropriate information in response to the problem described by a user in a natural language. The response is built on the basis of some multimedia elements and content analysis of text documents using ontology and WordNet resources.[2] To achieve the presented goal a huge corpus of digital documents is needed. Logical document structure recognition is necessary before content processing. It solves the following problems:

- paragraph hierarchy can be restored,
- images and tables can be automatically annotated on the basis of their captions,
- extracted images can be used for similar document retrieval,
- paragraphs spanning to more than one column can be merged,
- logical flow of paragraphs can be restored (context of words is not lost).

G. Kamola · M. Spytkowski · M. Paradowski ·
U. Markowska-Kaczmar (✉)
Wroclaw University of Technology, Wyb. Wyspianskiego 27,
50-370 Wroclaw, Poland
e-mail: urszula.markowska-kaczmar@pwr.edu.pl

[1] NEKST—Adaptive System Supporting Solving Problems on the Basis of Content Analysis of Available Electronic Resources: http://www.ipipan.waw.pl/nekst/.

[2] http://plwordnet.pwr.wroc.pl.

The performed research is also useful for the ongoing construction of the *Polish WordNet* and for various applications in *Natural Language Processing*.

The Internet is a source of vast numbers of digital documents. Due to huge diversity of authoring tools, fully automatic extraction of structured text and image elements is not possible. A large variety of commercial (e.g. *Adobe Acrobat*) and non-commercial (e.g. *PDF To Text*) tools is available for extraction of text from these documents. Nevertheless, output of such tools is far from perfect because:

- there is no guarantee that the complete text is extracted;
- order of paragraphs or even single sentences may be disrupted;
- national and Latin characters can be omitted due to character coding;
- enumerations or itemizations structure can be flattened to simple text;
- tables and similar logical structures may not be extracted.

1.2 Contribution

In this paper we address the problem of *document structure recognition* which is not dedicated to any special domain nor to specific document layouts. Our aim was to create a general approach. As such it is more challenging task than the design of a method processing documents in a specific domain or with known layouts. A detailed description of all addressed problems is given in Sect. 3.

The presented solution of structure segmentation uses computer vision techniques. Document pages are visualized in high resolution and processed as separate images. In contrary to other methods, we consider digital documents, so we could miss all preprocessing phases which are essential for scanned documents. The input of the system is a single document page acquired from the Internet. The final output is an XML file which contains a set of rectangular outlines with assigned document structure classes.

There are similar studies in the literature but:

- they are limited to a specific structure recognition, e.g. mathematical expressions [35], tables [19], block of texts [22],
- they refer to the recognition of the whole document structure but in a specific domain, where the considered set of layouts is known in advance.

We would like to underline that in comparison to the existing methods our solution presents an aggregated approach recognizing 10 document structure classes and it does not depend on a document layout. Our original contribution in this paper is:

- the method of graphic component segmentation,
- the rule-based line segmentation method,
- the text line grouping method,
- layout independent text structure recognition rules,
- the gridded table detection method,
- the evaluation routine, based on non-white pixel analysis.

1.3 The paper overview

The paper is organized as follows. Section 2 describes the related research. Section 3 formulates the problem to solve. Section 4.1 presents the proposed method. Section 5 describes the experimental study and shows the results. The last section concludes the paper.

2 Related research

In typical document structure recognition system, the procedure starts from document segmentation. Research in this domain has its roots in *Optical Character Recognition* (OCR), e.g. [21]. Primary goal of these methods was to locate text. It may be marked using bounding boxes or more flexible, non-rectangular outlines.

2.1 Document segmentation

Three groups of methods can be distinguished considering how information is processed: *top-down*, *bottom-up* and *hybrid*. The first methods of structure separation, developed in the 90s, represent the bottom-up approach, e.g. [11, 23, 40]. These algorithms process local information about groups or single non-white pixels. Pixel groups are connected into words, lines and paragraphs. *Connected components* [13] is one of the commonly used methods. There are various upgrades of this approach, e.g. [40]. More recent methods utilize typographic principles and knowledge about document structure. A common approach to parametrization is based on statistical features [24], such as vertical and horizontal gaps between objects. It helps to handle font size, font type, its orientation, line spacing, etc. Many approaches, for example [12, 31], use manual heuristic rules and grammar. These methods are very time consuming and have problems with processing when the document layout changes significantly. Large diversity of document layouts is one of the largest problems of the methods from this group.

Top-down methods are based on global page image information. They decompose pages into columns, blocks, lines and words. The location of white stripes is crucial to successful decomposition. The most common example is

Recursive X–Y Cuts method [10, 28], which decomposes the document page image by recursive cuts into rectangular blocks. Document segmentation is represented by a tree structure. Each tree node represents a cut associated with a horizontal or vertical projection (histogram). A cut is made at each point where a defined number of neighbouring histogram bin values fall below a defined threshold. Sensitivity to rotation is the weakness of the algorithm. It is worth mentioning that a complex layout containing fonts of various sizes decreases the accuracy of the method. The modification of this method can be found in [27]. This method enables an assignment of order in which the paragraphs are read. Another example is *Maximal Whitespace Rectangles* method [4], which analyses the structure of the document's background. Empty rectangles, found using the Breuel's algorithm, are used to divide the document page. Sorting empty rectangles allows to find their covers. Covers acceptance is based on their assigned weights. Accepted covers define the page structure division. *Run-Length Smearing Algorithm* [30] is a well-known algorithm in this group based on segment growing.

It starts with the binary input image. It then converts a horizontal or vertical sequence of pixels (run) shorter than a given threshold to black pixels (this is called 'run length smearing'). This step is proceeded horizontally and vertically for various values of thresholds. The outcome image is a combination of the images for both directions combined by the logical operation AND. The primary components join into groups. Smearing techniques make an assumption regarding the size of components being the base to calculate thresholds. *Docstrum* [32] segmentation method, which also belongs to this group, can be applied for complex non-Manhattan layouts and rotated documents (with arbitrary skew angles of text lines). It relies on the text line segmentation and requires a lack of letter connections in a character sequence. There is also a problem in cases when characters are situated too far away. This situation can also happen after the text is aligned into columns.

In the paper [21] a method being a *hybrid* combining top-down and bottom-up approaches to document analysis is proposed. It consists of two steps: block extraction using 32×32 window of pixels (top–down) and multi-column block detection and segmentation (bottom–up). If the window covers 10 non-white pixels, it is considered to be associated with the non-white pixels. Then the contour of the 32×32 windows is traced. If the space between blocks or columns of text is larger than 32 pixels, the segmentation process is done. Further improvements of this method are described in [20].

2.2 Segment recognition

Document segment recognition is the subsequent step after segmentation. Its primary goal is to assign all detected structures into meaningful groups. The assignment is done using block content analysis, graphic features and spatial relations between regions. Various recognition approaches are used to classify segments. Neural network-based classification is a well-explored area, e.g. [6, 16, 41]. The first method uses a neural network to classify a set of masks into the three texture classes in the page segmentation problem: halftone, background, and text and line-drawing regions. The text and line-drawing regions are further discriminated based on connectivity analysis. Recognition of textual and graphic blocks has been done using self organizing maps [41]. Radial basis function and probabilistic neural networks are applied to region classification, [16]. Recurrent fuzzy adaptive system ART, [6], recognizes text and graphic blocks. Statistical features such as: region size, its position, number of lines and words are mainly used as an input of neural network, [36]. A deep survey of neural network application in segment analysis and recognition can be found in [25].

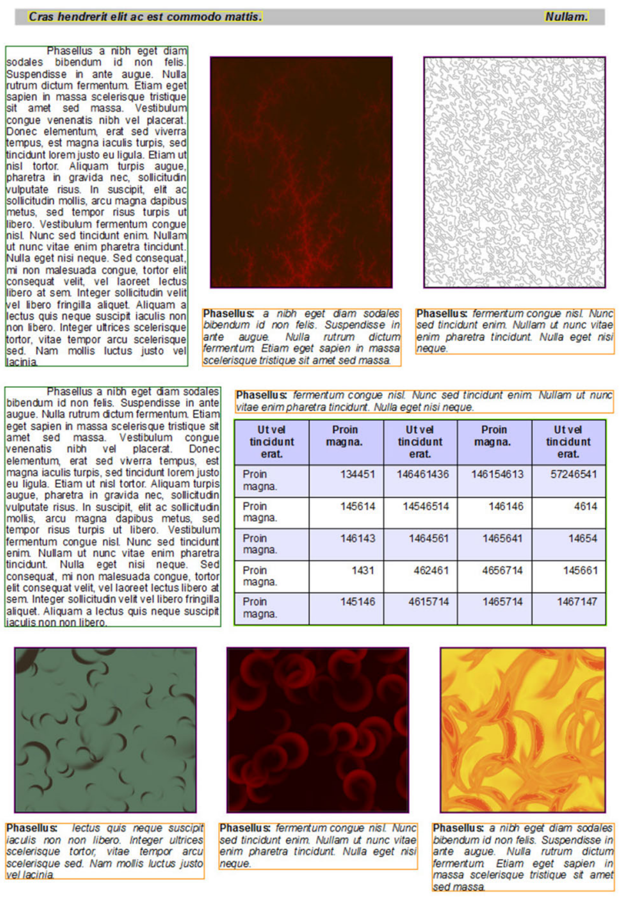
Classification of graphics based on statistical features is presented in [42]. The proposed method works with the non-Manhattan layouts where standard segmentation methods fail. Smearing techniques and assignment of regions are applied in strictly defined order. The region is classified as a graphics if the histogram does not show regularities. The paper [8] also considers picture detection. The method uses OCR to separate out the text. It applies the *Normalized Cuts* algorithm [38] to cluster the non-text pixels into the picture regions. The authors correct under- and over-segmentation based on deducing the number of pictures. The work [29] presents a summary of segmentation methods and their ability to recognize various document structures.

A lot of research in document image analysis is devoted to methods focused on a specific element of the document structure. An example of such structure is tables. The paper [14] presents a horizontal and vertical line detection method and applies it to table recognition. The method does not use dedicated heuristics and thus is much more universal. The work [19] describes *T-Recs*, a system that recognizes tables based on the bounding boxes of text contained in the table's cells. The paper [43] presents a statistical approach and line detection to find tables. The authors in [7, 15, 19, 37, 39] present other solutions of this problem.

Many early works in document structure recognition presented their results in a graphic form without a standardized quality measurement. These experiments did not take into consideration a large variety of existing document layouts. The lack of the publicly recognized benchmark data sets with annotated document page images causes difficulties in comparison of various algorithms. According to [24] all experiments are made with various document sets. The next problem is the absence of a standard testing procedure enabling such comparison. The authors of the paper [33] have proposed an evaluation method based on pixels instead of contours which shares similarity with the method presented here. Thanks to this approach it is possible to compare the results independently on the method by which the segmentation regions were obtained. This strategy was proposed to compare the result of segmentation with manually annotated set of regions containing each possible correct segmentation. Each segmented region was compared with the region from the ground truth set if their intersection was large enough.

3 The problem formulation

In our system the document structure recognition problem is divided into two sub-problems: segmentation (grouping) and recognition. The input is a document page image, the output is a set of recognized bounding boxes of document structures. Each page is processed separately. The exemplary input and output are presented in Fig. 1.



Cras hendrerit elit ac est commodo mattis. *Nullam.*

Phasellus a nibh eget diam sodales bibendum id non felis. Suspendisse in ante augue. Nulla rutrum dictum fermentum. Etiam eget sapien in massa scelerisque tristique sit amet sed massa. Vestibulum congue venenatis nibh vel placerat. Donec elementum, erat sed viverra tempus, est magna iaculis turpis, sed tincidunt lorem justo eu ligula. Etiam ut nisl tortor. Aliquam turpis augue, pharetra in gravida nec, sollicitudin vulputate risus. In suscipit, elit ac sollicitudin mollis, arcu magna dapibus metus, sed tempor risus turpis ut libero. Vestibulum fermentum congue nisl. Nunc sed tincidunt enim. Nullam ut nunc vitae enim pharetra tincidunt. Nulla eget nisi neque. Sed consequat, mi non malesuada congue, tortor elit consequat velit, vel laoreet lectus libero at sem. Integer sollicitudin velit vel libero fringilla aliquet. Aliquam a lectus quis neque suscipit iaculis non non libero. Integer ultrices scelerisque tortor, vitae tempor arcu scelerisque sed. Nam mollis luctus justo vel lacinia.

Phasellus: *a nibh eget diam sodales bibendum id non felis. Suspendisse in ante augue. Nulla rutrum dictum fermentum. Etiam eget sapien in massa scelerisque tristique sit amet sed massa.*

Phasellus: *fermentum congue nisl. Nunc sed tincidunt enim. Nullam ut nunc vitae enim pharetra tincidunt. Nulla eget nisi neque.*

Phasellus a nibh eget diam sodales bibendum id non felis. Suspendisse in ante augue. Nulla rutrum dictum fermentum. Etiam eget sapien in massa scelerisque tristique sit amet sed massa. Vestibulum congue venenatis nibh vel placerat. Donec elementum, erat sed viverra tempus, est magna iaculis turpis, sed tincidunt lorem justo eu ligula. Etiam ut nisl tortor. Aliquam turpis augue, pharetra in gravida nec, sollicitudin vulputate risus. In suscipit, elit ac sollicitudin mollis, arcu magna dapibus metus, sed tempor risus turpis ut libero. Vestibulum fermentum congue nisl. Nunc sed tincidunt enim. Nullam ut nunc vitae enim pharetra tincidunt. Nulla eget nisi neque. Sed consequat, mi non malesuada congue, tortor elit consequat velit, vel laoreet lectus libero at sem. Integer sollicitudin velit vel libero fringilla aliquet. Aliquam a lectus quis neque suscipit iaculis non non libero.

Phasellus: *fermentum congue nisl. Nunc sed tincidunt enim. Nullam ut nunc vitae enim pharetra tincidunt. Nulla eget nisi neque.*

Ut vel tincidunt erat.	Proin magna.	Ut vel tincidunt erat.	Proin magna.	Ut vel tincidunt erat.
Proin magna.	134451	146461436	146154613	57246541
Proin magna.	145614	14546514	146146	4614
Proin magna.	146143	1464561	1465641	14654
Proin magna.	1431	462461	4656714	145661
Proin magna.	145146	4615714	1465714	1467147

Phasellus: *lectus quis neque suscipit iaculis non non libero. Integer ultrices scelerisque tortor, vitae tempor arcu scelerisque sed. Nam mollis luctus justo vel lacinia.*

Phasellus: *fermentum congue nisl. Nunc sed tincidunt enim. Nullam ut nunc vitae enim pharetra tincidunt. Nulla eget nisi neque.*

Phasellus: *a nibh eget diam sodales bibendum id non felis. Suspendisse in ante augue. Nulla rutrum dictum fermentum. Etiam eget sapien in massa scelerisque tristique sit amet sed massa.*

Fig. 1 The example of a document image with marked bounding *boxes* (a typical layout content of which, for the publication purpose, was randomly generated on the basis of real one existing in the corpus. In all experiments the real documents from the Internet were used)

3.1 Segmentation problem

The key goal of segmentation is to divide the page image into regions containing *coherent content*. A great difficulty of segmentation comes from the need of processing of various document layouts.

Document layouts are very diverse, starting from a fixed composition with clear horizontal and vertical blocks (*Manhattan*), up to a very loose, non-rectangular one (*Non-Manhattan*). *Non-Manhattan* layouts mainly exist in popular journals, tabloids and advertisements.

One of the most common features differentiating layouts is *spacing*. In typography *spacing* is defined as an empty space between elements of the composition, i.e. between page parts, margins, texts, words, letters and spaces inside letters. Spacing can vary and can be assigned according to the subjective feelings of documents' authors. Text alignment should also be taken into consideration. Left and right-aligned paragraphs are characterized by the changeable line length. Centred texts (often used in tables) are more difficult to handle, because of changeable outline coordinates. Justified texts do not share these problems, but introduce *kerning* (a correction of spacing between letters and words to fit the column).

In our research we focus on Manhattan layouts. We assume the abidance of basic typesetting rules, for example:

- Positions of page headers and footers;
- Closeness of an image and its caption;
- Uniformity of spacing between lines and paragraphs.

The mentioned assumptions cause a set of problems to be faced. Existence of *rivers* (spaces between words situated one under another) and *kerning* may cause segmentation artefacts and subsequent recognition failure.

3.2 Recognition problem

The key goal of recognition is to utilize features to distinguish between region classes:

- Abstract,
- Author,
- Caption,
- Header,
- Page Footer,
- Page Header,
- Paragraph,
- Table,

- Title,
- Graphic elements: line, picture, diagram, scheme, chart.

Each generated region should have coherent visual and spatial features. For a specific region class the features may be constructed based on specific properties of page structure elements, for example:

- A photo is usually a spatially coherent collection of non-white pixels;
- A block of text has a distinctive outline along its borders;
- A drawing consists of interconnected groups of lines and curves mixed with fragments of text;
- A caption is a block of text spatially related to a graphic element and may begin with a keyword.

3.3 Formal problem definition

The task can be formally described as follows. The goal of segmentation of page P is to find bounding boxes $B_i : i = \{1, \ldots, n\}$ that include coherent content:

$$P = \{B_1, B_2, \ldots, B_n\}. \tag{1}$$

Each bounding box B_i is a set of pixels between its top–left (x_i^1, y_i^1) and bottom–right (x_i^2, y_i^2) coordinates. Bounding boxes do not overlap:

$$\forall_{B_i \in P} \forall_{B_j \in P - \{B_i\}} B_i \cap B_j = \emptyset. \tag{2}$$

The goal of recognition of bounding boxes B is to assign a label $L_i \in \mathcal{W} : i = \{1, \ldots, n\}$ to each bounding box $B_i \in B$:

$$\Psi(\{B_1, B_2, \ldots, B_n\}) = \{L_1, L_2, \ldots, L_n\}. \tag{3}$$

The dictionary $\mathcal{W}$ of class labels consists of labels identifying possible logical structure elements (graphics, header, caption, paragraph, etc).

4 The proposed method

The proposed method is called *Document Structure Recognizer* (DSR). Because the collected documents stem from the Internet, the method assumes that the document is properly preprocessed: free of scanning artefacts, orthogonally aligned, de-noised and uniformly saturated. The following, additional requirements should be met:

- High resolution of the document image is necessary, at least 300 DPI. The limitation stems from the applied OCR tool (*Tesseract*). Its recognition quality decreases when lower resolutions are used;
- Document image contains uniform background colour and distinct contrast between background and foreground;

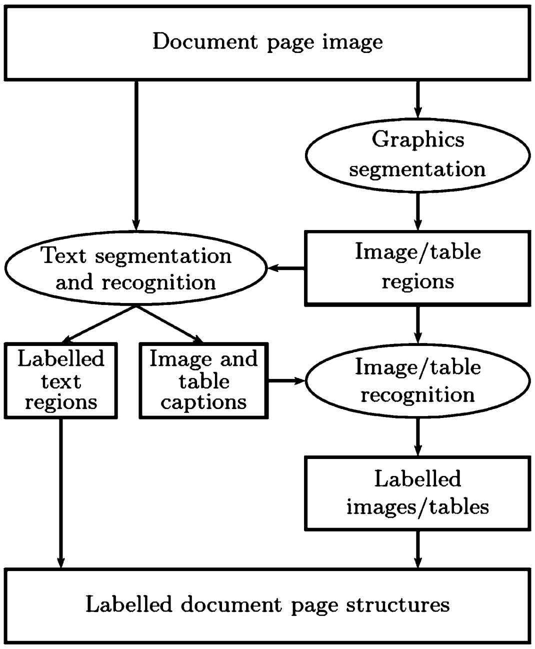


Fig. 2 The scheme of the proposed DSR method of the document structure recognition

- Text path has to be a straight, orthogonal line;
- Text font size within a single paragraph has to be constant.

4.1 The idea of the method

The general scheme of the method is presented in Fig. 2. Document structure recognition is composed of several methods specifically designed to handle distinct types of document structures. Textual and graphic blocks have different processing pipelines, thus need to be separated in the very beginning. Graphics type recognition is performed in later stages. Text regions are omitted at this stage.

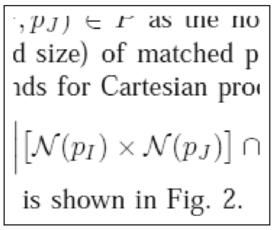
The general idea of the DSR method is presented in Fig. 2. As an input the document page image is taken. First, the page image is processed by the module of graphics segmentation. Because text segmentation is very sensitive to the presence of graphic elements, therefore all graphic components have to be masked beforehand. The text segmentation method finds groups of letters using a well-known routine called *connected components labelling* [5, 9]. It finds obstacles represented by space between columns, paragraphs, text lines and words. These obstacles are employed for the segmentation of text regions. After text segmentation, the text structure class recognition is performed. OCR and rule based text analysis are helpful in recognition of specific text blocks, such as: captions, abstracts, document authors, etc. Tables without grid lines (plain text) are recognized by the *T-Recs algorithm* [18]. The labelled text regions are obtained as an output from this module. Captions are then delivered to the image/table recognition module. It also comes in hand to recognize types of masked graphic elements. Graphic regions are classified into disjoint groups: tables with grid lines, photos, drawings, charts and diagrams. This classification is based on the approach presented in [26].

4.2 Detailed description

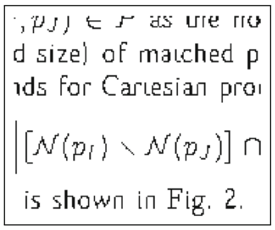
In this section, the following components of DSR (Fig. 2) are presented: *graphics segmentation*, *table detection text segmentation and recognition*.

4.2.1 Graphics segmentation

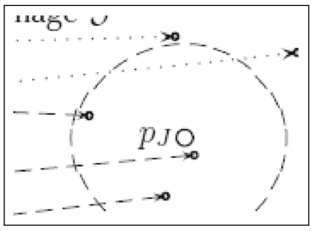
The method simultaneously detects various graphic elements: pictures, charts, diagrams, schema and tables with lines. The input image is transformed in several stages to distinguish all these components from plain text and non-graphic structures.



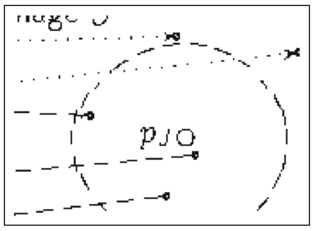
(a) A fragment of text



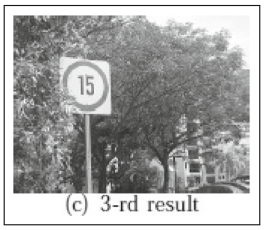
(b) Skeleton of the text



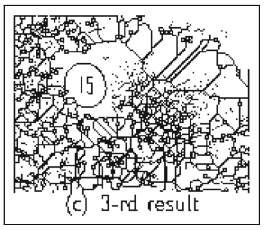
(c) A part of a diagram



(d) Skeleton of the diagram



(e) A picture



(f) Skeleton of the picture

Fig. 3 The examples of skeletons of typical document components

Image skeletonization This transformation is based on the classic algorithm presented in [34]. For the binary page image, the operation filters all black pixels that are not equidistant to its boundaries. The skeleton usually emphasizes geometrical and topological properties of the shape. Skeleton examples of typical document page components are presented in Fig. 3. Image skeletonization enables extraction of content independent features. The processed image can be represented by a set of curves and connections between them. Each disjoint skeleton part is considered separately. Figure 3 shows that a typical text element is noticeably smaller than a graphic element. This observation allows to build a feature vector used to discriminate graphic and non-graphic elements. Two features are considered:

- the number of pixels in the segment skeleton,
- the skeleton height.

The above idea is formalized in Algorithm 1.

Initial segment recognition The aim of this stage is to classify a segment into one of the two classes: a graphic or text element. First, the parameter values of the method are automatically assigned. It is necessary because various font sizes and different resolutions of a document image exist. The values of features are calculated (height and number of pixels) for each disjoint skeleton. For the whole image statistics are calculated and 0.99-quantile of these values is taken.

Algorithm 1: Graphics segmentation

```
Require: $i$ – input image
Require: $T$ – skeletonization thresholds set
  $i \leftarrow$ equalize-histogram(greyscale($i$)) {Prepare image}
  $s \leftarrow \emptyset$ {Empty binary image}
  for all $t \in T$ do
    $p \leftarrow$ skeletonize(threshold(invert($i$), $t$))
    $p \leftarrow$ remove-vertical-edges($p$)
    $p \leftarrow$ remove-horizontal-edges($p$)
    $s \leftarrow s \cup p$ {Add skeleton pixels}
  end for
  $(V, E) \leftarrow (s,$ 8-neigborhood $(s))$
  $\{v_1, v_2, ..., v_n\} \leftarrow$ graph-connected-components $(V, E)$
  $f_s \leftarrow \text{quantile}_{0.99}\ \{|v_1|, |v_2|, ..., |v_n|\}$
  $f_h \leftarrow \text{quantile}_{0.99}\{max(v_1^y) - min(v_1^y),$
      $max(v_2^y) - min(v_2^y),\ ..., max(v_n^y) - min(v_n^y)\}$
  $C \leftarrow \{v_1, v_2, ..., v_n\}:$
      $|v_i| \geq t_s \wedge (max(v_i^y) - min(v_i^y) \geq t_h)$
  $s \leftarrow$ dilate(flatten($C$)) $\cup$ sobel($i$)
  $(V, E) \leftarrow (s,$ 8-neigborhood $(s))$
  $U \leftarrow$ graph-connected-components $(V, E)$
  $B \leftarrow \emptyset$ {Set of bounding boxes}
  for all $u \in U$ do
    if $u \in$ bounding-boxes(C) then
      $B \leftarrow B \cup$ bounding-box($u$)
    end if
  end for
  $B \leftarrow$ merge-overlapping-boxes($B$)
  return $B$
```

It is not possible to unambiguously assign the quantile so that it always distinguishes between text and graphic elements. As previously mentioned, the skeletons of text elements are generally smaller than those of graphic elements. Therefore, the quantile values are multiplied by experimentally determined factors. The skeleton height quantile value is multiplied by a factor of two and the skeleton size quantile value is multiplied by a factor of three. The final values create threshold used during classification to distinguish between text and graphic elements.

Assembling of graphic elements At this point, the skeletons classified as belonging to graphic elements correspond to fragments of the actual graphic elements. The aim of this step is to expand them to fully encompass the image and to combine fragmented graphic elements. To do so the segments are dilated and a Sobel operator is applied to the input image. Then the dilated skeleton is combined with the result of the Sobel operator using the OR operator. The resulting data are once again separated into disjoint segments. All segments that do not share at least one pixel with the initially accepted skeleton segments are rejected. The remaining segments are reduced to their bounding boxes. Overlapping bounding boxes are merged. The collection of bounding boxes containing graphics is the output of this stage. They are sent to the text segmentation and graphics recognition modules.

4.2.2 Text segmentation and recognition

The textual region segmentation is performed using concepts found in mathematical morphology, skeleton representation, connected component processing and typography. The graphic regions are masked with bounding boxes before further processing at this stage. Letters, the basic element of text, are detected by a method called connected components labelling [9]. As an input the algorithm obtains binary image and as an output it gives the set of connected components $\mathrm{CCS} = \mathrm{cc}_1; \mathrm{cc}_2, \ldots, \mathrm{cc}_n$. Each component is described by a set of pixels.

Next, the proposed methodology is divided into four consecutive steps:

- detection of blocking connected components,
- text line segmentation,
- text line grouping—paragraph segmentation,
- text region classification.

Detection of blocking connected components Many *bottom–up* techniques decide whether to group or separate connected components, relying on the estimation of the distances between objects. This kind of approach gives successful results for documents with almost identical spaces between the characters and words in the same and in different text lines. However, the distances between words are unstable because of the process of adjusting the spacings (*kerning*). In consequence, some words in lines with greater spacing become isolated. It is worth mentioning that there is an approach based on the use of the spacing ratio measure to solve this problem [1].

Over-segmentation errors are especially common in multi-column documents, where text often happens to be justified and standard deviation of the distances tends to be higher. When thresholds are overestimated, words from separated columns might be merged. Additional techniques are required to find objects within the document that clearly determine columns.

To assure unconnected component stayed disjoint, obstacles (blocking components) are used, which prevent merging these groups. Merging two components is not allowed if after merging the resulted bounding box crosses blocking object. Essential blocking elements are letters, part of words or whole sentences located in extreme places of separate columns. Groups of connected components aligned to the left or to the right are an indicator whether these components are blocking ones. Two objects are aligned left, if the difference between their left coordinates is less than a threshold ϵ. In case of right alignment the difference between right coordinates is taken. This procedure allows to find extreme left or right character in each column.

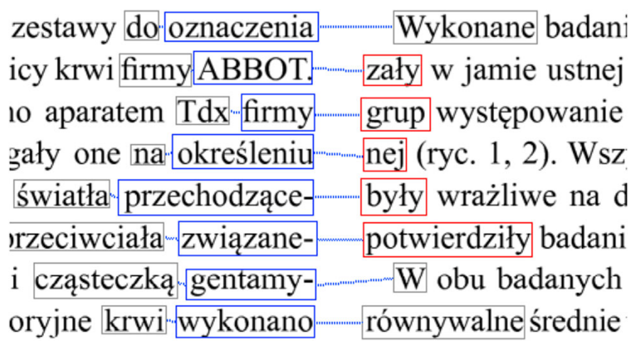
Searching extreme located elements (blocking) may cause indication of false components inside paragraphs, therefore the following processing steps are executed.

For each group, the average distance of neighbouring connected components on the left (avg_l) and right side (avg_r) are computed. In Fig. 4 break lines assign distances used in the left average distance avg_l and the right average distance avg_r calculation. They are used to indicate blocking components. The Eq. 4 formally presents how the left average distance (avg_l) is calculated.

$$\mathrm{avg}_\mathrm{l} = \frac{1}{|\mathrm{B}|}\sum_{\mathrm{b}\in\mathrm{B}}\begin{cases} \min(|\mathrm{c}^\mathrm{r} - \mathrm{b}^\mathrm{l}|), \mathrm{c} \in \mathrm{C}_\mathrm{b} & \text{if} \quad \mathrm{C}_\mathrm{b} \neq \emptyset \\ \text{const} & \text{if} \quad \mathrm{C}_\mathrm{b} = \emptyset \end{cases}, \tag{4}$$

where, B is the set of bounding boxes b belonging to the group, C_b is the set of boxes from which we will choose the box closest to b. It can be said that C_b contains all bounding boxes to the left of the bounding box b. The c^l, b^l, c^r, b^r, c^t, b^t, c^b and b^b values for bounding boxes are their corresponding limits in pixels (e.g. c^l is the x-coordinate of c box's left-side edge on the image). For a bounding box c to be included in C_b it must hold that $b^l > c^r$, $b^t < c^b$ and $b^b > c^t$. The right average distance avg_r is calculated in an analogous manner.

Then the following conditions are checked.



zestawy do oznaczenia Wykonane badani
icy krwi firmy ABBOT zały w jamie ustnej
ıo aparatem Tdx firmy grup występowanie
gały one na określeniu nej (ryc. 1, 2). Wsz
światła przechodzące- były wrażliwe na d
ɪrzeciwciała związane- potwierdziły badani
i cząsteczką gentamy- W obu badanych
oryjne krwi wykonano równywalne średnie

Fig. 4 The connected components (words or word fragments) are surrounded by *bounding boxes*. In this figure for simplicity, only the words used to calculate the left average distance avg_l and the right average distance avg_r are shown in *bounding rectangles*. Indication of blocking components is based on searching groups of connected components aligned to the *left* (*blue*) and to the *right* (*red*) (colour figure online)

- If a group is left-aligned and $avg_l < avg_r$ then the group is ignored because it is assumed that the average distance from the neighbouring components on the left should be greater than the average distance from components on the right side;
- If a group is right-aligned and $avg_r < avg_l$ then the group is ignored on the basis of the similar rule: the average distance from the neighbouring components on the right should be greater than the average distance from components on the left side;
- If the average distances are similar ($|avg_r - avg_l| < \epsilon$) then the group is ignored. This case often happens for the groups of words inside paragraphs due to a coincidental alignment of spaces, called *rivers*.

As a result, all remaining groups are considered to consist of left (OBS_L) and right (OBS_R) limit words of columns. Each connected component is marked accordingly to the group it belongs to. Information whether a component is left or right blocking is used in the next step.

Text line segmentation There are three conditions used in the text line segmentation process. Firstly, the connected components' *alignment condition* is examined. The process of grouping is based on basic rules of alignment of characters and words within the same text line. In typography, four horizontal imaginary lines are identified: ascent line, mean line, base line and descender line. The ascent line is set by ascenders—portions of a minuscule letter in a Latin-derived alphabet that extends above the mean line of a font. The mean line is the line that determines where non-ascending lower-case letters terminate. The baseline is a line upon which most letters rest and below which descenders extend. The descender line is set by a portion of a letter that extends below the baseline. Relationships between those lines and all possibilities of combinations between two characters let us introduce four simple rules. Based on these rules the decision is made whether two components are within the same text line.

Two adjoining components cc_i and cc_j are going to be recognized as candidates for characters or words in the same text line according to the following rules:

- Components are possibly both capital or small letters if both the top and bottom edges of the bounding boxes lie on the ascents line and the base line;
- If components are small and capital letters then horizontal overlapping should have at least 33 % of the highest component (small letters in main text in majority of documents have at least 50 % of the heights of capital letters);
- If one of the components is a letter and the second one is an ascent or punctuation mark then they cover significantly different areas and are placed on appropriate alignment lines.

Secondly, the candidates have to satisfy the *x distance constraint*. Based on the histogram of the distances between the adjacent bounding boxes, for which the alignment condition is met, a proper threshold value (γ) is calculated. To link the components which are farther from each other than the most often occurring value, the threshold is multiplied by the factor of three.

The blocking components are used to prevent including words from adjacent columns into text lines. The components cc_i, cc_j are satisfying the *connectivity condition* if any of the given statements are true:

- Both components are not marked as blocking:

$$cc_i \notin (OBS_L \cup OBS_R) \wedge cc_j \notin (OBS_L \cup OBS_R); \quad (5)$$

- The component on the left side (cc_i) is marked as left-blocking and the component on the right side (cc_j) is not left-blocking:

$$cc_i \in OBS_L \wedge cc_j \notin OBS_L; \quad (6)$$

- The component on the left side (cc_i) is not marked as blocking and the component on the right side (cc_j) is marked as right-blocking:

$$cc_i \notin (OBS_L \cup OBS_R) \wedge cc_j \in OBS_R. \quad (7)$$

Finally, only those connected components are allowed to form a text line, for which each adjoining pair of components fulfils the *alignment condition*, *x distance constraint* and *connectivity condition*.

Text line grouping The purpose of this step is to merge text lines into the left, right-aligned and centred paragraphs. One of the attributes used for grouping is font thickness. To estimate thickness the morphological operation *edge out*, also called *external gradient*, is used. This method finds the difference between the original image and a dilation of that image. As a result, the background pixels immediately next to the shape are returned. The value pin is defined as the number of pixels inside contours. Using a skeleton image, the number of pixels of the skeleton (ps) is calculated. Then, font thickness th is estimated as follows:

$$\mathrm{th} = \frac{\mathrm{pin} - \mathrm{ps}}{\mathrm{ps}}. \tag{8}$$

Using *edge out* to find outlines, in contrast to counting all non-white pixels, gives a more accurate number due to the smoothing properties of dilatation.

Another such attribute is font colour. On the basis of the assumption that homogeneous elements in documents have the same colour the representation of each non-white and non-transparent pixel inside bounding boxes of text lines is computed. Those values can be compared for any possible colour representation (RGB by default). In this paper ΔE measure is used according to *CIE2000* standard, where two colours are considered to be similar if their difference is smaller than a *just noticeable difference* (jnd = 2.3).

The above metric lets us introduce the *similar graphic feature condition*. The text lines l_i and l_j are similar in respect to the graphic features if:

- The variance of font thickness is smaller than the threshold $\Gamma = 0.6$;
- The dominant colour inside the bounding boxes of the text lines is similar;
- Both text lines have a similar height (exact to ϵ) or the shorter text line has at least 66 % height of the longer one.

Text lines are grouped into paragraph if the following conditions are met:

- Both text lines have similar graphic features: font thickness and colours with thresholds: $\Gamma = 0.6$, jnd = 2.3;
- All text lines are aligned according to the paragraph type -left, -right or centre-aligned;
- Both text lines are no further from each other than ϕ—maximum value from the set of y distance values blocking object between them.

In the next stage classification is performed to assign a class to each of these regions.

Text region classification Text block classification is done using the following attributes:

- dominant colour,
- font thickness,
- location,
- location in relation to other objects,
- text recognized by OCR tools (Tesseract),
- relation with blocking objects.

The purpose of this stage is to assign classes describing roles of these structures in the document. Seven types of classes are distinguished: paragraphs, titles, captions, tables, images, page headers and page footers. Let us remind that images and some tables have already been recognized. A rule-based recognition algorithm is applied.

An object is recognized as a *caption* if:

- the object is located in the neighbourhood of a recognized image or table region;
- the text begins with one of the following keywords: “image”, “img.”, “figure”, “fig.”, “photo”, “ph.”, “table”, “tab.”, “diagram”. National keywords are also included.

An object is recognized as a *header* if:

- the object is located just over the paragraph or a single text line;
- there is no blocking objects between the object and the associated paragraph;
- horizontal projections of the object and the paragraph are overlapping;
- the object has a font thickness greater than the average thickness of the paragraph or is written with capital letters.

An object is recognized as a *page header* if:

- apart from images, lines and other objects within the same text line (alignment condition), the object is located at the top of the document;
- the recognized text starts with keyword: “p.”, “page”, “no.” or a digit.

An object is recognized as a *page footer* if:

- apart from images, lines and other objects within the same text line (alignment condition), the object is located at the bottom of the document;
- the recognized text starts with keyword: “p.”, “page”, “no.” or digit.

An object is recognized as a *table* if:

- it is initially recognized as an image;
- there is a caption in the neighbourhood that implies that the image is a table.

If none of these rules are met, the element is recognized as a *paragraph*.

Recognition of tables without grid lines These table structures are recognized using the *T-Recs table recognition and analysis system* [18]. T-Recs realises the bottom-up clustering of word segments and does not apply any other top-down specific techniques (separator detection). The table detection is based on search for the neighbouring words in the previous and next line (relative to the currently inspected item) which bounding boxes horizontally overlap and they are linked together. With this simple symmetrical relation, independence from delineations or conspicuous white spaces is achieved. The constructed segmentation graph includes several characteristics which are distinctive for tabular structures. Next, several post-processing steps are applied to fix inherent segmentation errors. Three error classes are considered: columns merged together by a common header which consistently overlaps with the first word of each column, blocks that are over-segmented because of the occasional gap (river), words without a lower nor upper neighbour. Errors are assigned to one of the above classes and processed accordingly.

4.2.3 Recognition of tables with grid lines

As a result of graphic segmentation some of the tables in the document are recognized as images. Such regions need to be filtered out and properly labelled separately from other tables. It was assumed, that the tables had grid lines separating their cells because only such tables should find themselves among detected images. The method works on data acquired from vectorizing the image and then the largest graph is chosen. It represents the table grid. The gridded table recognition is performed by applying a set of three rules to the resulting data:

- If the largest graph has less than 85 % vertexes with orthogonal lines then the image is not a table;
- If the resulting grid cannot be used to reconstruct at least one cell then the image is not a table:
- If less than 50 % of the cells do not have content within them then the image is not a table, otherwise the image is a table.

The first rule uses the fact that in a table most of the lines will be orthogonal to the coordinate system. The approximation of this grid will be contained within the largest graph of the vector representation. Of course it is possible that the grid will not be full due to either joined cells or noise. Still the number of such lines in vector representations of tables is much larger than in vector representations of images.

The second rule assumes that the vertexes in the vector representation will coincide with the points at which the grid lines intersect. Based on these points a grid is approximated.

The third rule requires the grid to be detected. It is used to formulate features which differentiate tables from charts or diagrams. A table has content in most of its cells while the two other structures are much sparser. If most cells have some content (in the form of other graphs contained within) the image is recognized as a table.

5 Experimental study

The aim of the performed research is evaluation of the quality of the developed method and comparison to state-of-the-art methods. The section starts by describing the evaluation methodology and the corpus (assumed as ground truth) of document page images collected for this project. Database of document page images with manually annotated bounding boxes is used in all experiments.

5.1 Evaluation methodology

It is worth noticing that it is possible to compare the results on the basis of recognized objects or on the basis of recognized region of objects. In the first case there is the binary evaluation—the compared object has or has not been found. Small and large regions have the same impact on the result. In the second case the comparison relies on measurement of how precisely the ground truth object is projected by the method. The measure is calculated on the basis of correctly classified pixels.

Different text segmentation methods can return fundamentally different segments. In relation to the ground truth the compared segments can be smaller (lines, single words, chunks of letters or single letters), similar size (paragraphs and other coherent text block) or larger (columns, all text on page). Directly comparing the ground truth to the returned regions is correct only for the methods returning segments of similar size, but it is not true for the referential methods. For this reason a method is proposed that would allow comparing results with any other method while handicapping only our own text segmentation method.

The evaluation method changes its focus from returned regions to groups of non-white pixels (cut-off at 0.1 brightness on a scale from 0 to 1) within the regions according to Algorithm 2.

```
Algorithm 2: Basic scoring algorithm
  Require: $G$ – ground truth regions for a single page
  Require: $R$ – generated regions for a single page
  Require: $t$ – pixel brightness threshold
    $TP \leftarrow 0, TN \leftarrow 0, FP \leftarrow 0, FN \leftarrow 0$
    $P \leftarrow$ all pixels on the processed page
    for all $p \in P : brightness(p) < t$ do
      $(p \in G) \wedge (p \in R) \rightarrow (TP \leftarrow TP + 1)$
      $(p \in G) \wedge (p \notin R) \rightarrow (FN \leftarrow FN + 1)$
      $(p \notin G) \wedge (p \in R) \rightarrow (FP \leftarrow FP + 1)$
      $(p \notin G) \wedge (p \notin R) \rightarrow (TN \leftarrow TN + 1)$
    end for
    return $TP$, $TN$, $FP$, $FN$
```

In the evaluation common metrics like *precision* (prec) and *recall* (rec) are applied:

$$\text{prec} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{rec} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{9}$$

where: TP—True Positive, FP—False Positive and FN—False Negative.

Direct comparison of smaller regions to the ground truth leads to higher precision and lower recall. When only non-white pixels are compared the high precision remains while recall rises. It is a consequence of the method not being penalized for not returning white space between letters or lines. For larger regions on the other hand the precision is low and recall is high. Again, if only non-white pixels are compared the high recall remains while precision rises. In this case, the method is not penalized for returning white space between text blocks.

However, this type of comparison does not take into account whether the pixels are within the same or different regions (segments). A more thorough comparison would include selecting the pixels from only the region with the largest overlap with the current ground truth region. This variant is used for methods that return segments of similar to the ground truth according to Algorithm 3.

Algorithm 3: Largest region overlap scoring algorithm

```
Require: $G$ – ground truth regions for a single page
Require: $R$ – generated regions for a single page
Require: $t$ – pixel brightness threshold
  $TP \leftarrow 0, TN \leftarrow 0, FP \leftarrow 0, FN \leftarrow 0$
  for all $g \in G$ do
    $r^* \leftarrow \arg\max_{r \in R} r \cap g$
    for all $p \in (g \cup r^*) : brightness(p) < t$ do
      $(p \in g) \wedge (p \in r^*) \rightarrow (TP \leftarrow TP + 1)$
      $(p \in g) \wedge (p \notin r^*) \rightarrow (FN \leftarrow FN + 1)$
      $(p \notin g) \wedge (p \in r^*) \rightarrow (FP \leftarrow FP + 1)$
      $(p \notin g) \wedge (p \notin r^*) \rightarrow (TN \leftarrow TN + 1)$
    end for
  end for
  return $TP$, $TN$, $FP$, $FN$
```

5.2 Document image corpus

For this research, a collection of documents was gathered because the existing corpora in English are not publicly available or are dedicated to testing the recognition of a specific document structure, for instance to recognize tables. Other collections contain handwritten documents, copies of letters or scanned books. Such content is not coincident with the aim of the NEKST project.

The collected corpus of documents was selected from various Internet sources to ensure a variety of layouts to satisfy the aim of the project.

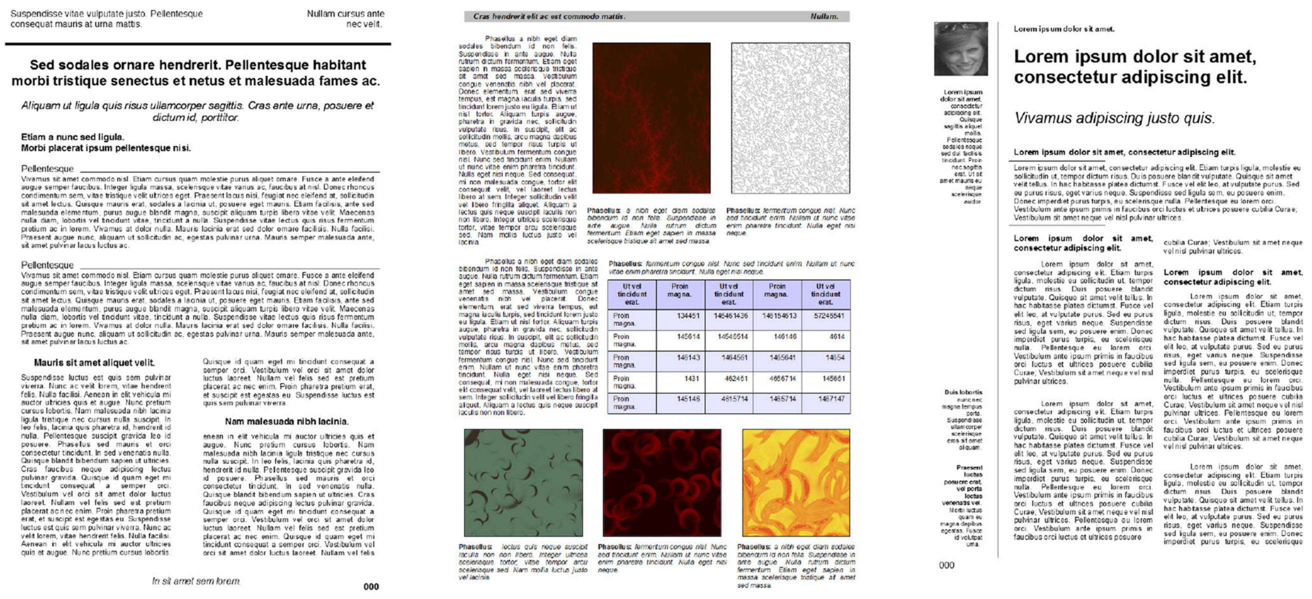
Table 1 The total number of elements for each document structure class in the whole document corpus

Class	The number of elements
Paragraph	5,839
Header	1,377
Page footer	1,159
Page header	1,262
Caption	1,168
Graphic element	728
Table	415
Title	232
Author	218

The collected documents were in , .doc, .docx formats, then they were printed to 300 DPI resolution in format and finally each document page was printed to an image in format. New documents were added to the collection if they diversified the corpus. To increase the corpus quality, it has been annotated by three people. All conflicts have been resolved by voting. The document corpus contains 1,218 page images. The number of elements in each document structure class is presented in Table 1. Some examples of document pages are shown in Fig. 5. They are specially generated for this publication on the basis of chosen layouts from the corpus.

5.3 Evaluation of text structure recognition

Using the evaluation method described in the Sect. 5.1 the results of DSR text segmentation were evaluated and compared with three referenced methods. All methods processed data from the same corpus. The parameter values of the referenced method were adjusted to guarantee effectiveness for the processed document set. According to the suggestions of their authors the threshold values were assigned on the basis of a 100 page images from our corpus. For the *Docstrum* method [32], its default parameter values $K = 5$ (the number of nearest neighbours), $f_t = 2.578$ (threshold of connecting components into line), $f_d = 9$ (proportion of letters), $f_{pe} = 1.3$ (threshold of connecting components into text blocks for perpendicular direction), $f_{pa} = 1.5$ (threshold of connecting components into text blocks for parallel direction) were substituted with $K = 8$, $f_t = 2.578$, $f_{pe} = 1.1$, $f_{pa} = 2.112$. For *ARLSA* algorithm [30] the values of parameters were as follows $T_h = 3.5$ (threshold referring to the proportion of component height), $c = 0.4$ (constant), $a = 5$ (constant). T_{max} (threshold referring to the number of congruent background pixels) was set as an average of all components. The remaining


Suspendisse vitae vulputate justo. Pellentesque consequat mauris at urna mattis.

Nullam cursus ante nec velit.

Sed sodales ornare hendrerit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac.

Aliquam ut ligula quis risus ullamcorper sagittis. Cras ante urna, posuere et dictum id, porttitor.

Etiam a nunc sed ligula.
Morbi placerat ipsum pellentesque nisi.

Pellentesque

Pellentesque

Mauris sit amet aliquet velit.

Nam malesuada nibh lacinia.

In sit amet sem lorem.

000

Cras hendrerit elit ac est commodo mattis. *Nullam.*

Ut vel tincidunt erat.	Proin magna.	Ut vel tincidunt erat.	Proin magna.	Ut vel tincidunt erat.
Proin magna.	134451	145461436	146154613	57245541
Proin magna.	145614	14546514	146146	4614
Proin magna.	146143	1464561	1465641	14654
Proin magna.	1431	462451	4656714	145661
Proin magna.	145146	4615714	1465714	1467147

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Vivamus adipiscing justo quis.

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

000

Fig. 5 The examples of the document pages generated for the publication on the basis of the document layouts from the collected corpus

parameters were unchanged and were set as described in the paper [29]).

The algorithm *RXYC* [21] was run with a window size of 16 × 32. Its other parameters only change computation time. The window width, $w = 16$, used to find cutting points was unchanged because it does not influence on the received results.

Course of the experiment All reference methods (Docstrum, ARLSA, RXYC) and the proposed DSR were applied in the process of text structure recognition (i.e. paragraphs, headers, titles, notes, captions, page headers and page footers). Their results were compared with the ground truth using Algorithm 2.

Results On the basis of the results presented in Table 2, one can conclude that the text segmentation achieved by the DSR method is comparable with the best results of the best method. ARLSA achieves a bit better results but the evaluation method is negatively biased towards our method.

In addition, it is worth underlying that the DSR method accomplishes two tasks: text segmentation and recognition, while the other methods can be only used to segment data.

5.4 Evaluation of paragraph recognition

The goal of the next experiment was to test DSR's paragraph recognition quality in relation of various layout types.

Course of the experiments The method was applied to the corpus containing: 743 document pages with one-column layout, 475 document pages with two columns and 415 document pages with more than two columns.

Table 2 Comparison of text structure recognition with referenced methods

Method	Prec (%)	Rec (%)	*F*-Score
Docstrum	92.94	93.14	93.03
ARLSA	93.61	93.23	93.41
RXYC	74.04	93.45	82.62
DSR	92.60	93.13	92.86

Results The results are presented in Table 3. They show that DSR is not sensitive to the number of columns in the page image. Under manual inspection there was no evident typical errors. Most of them were a consequence of specific component page composition.

5.5 Quality of the whole document structure recognition method for all recognized document structures

This experiment was performed to evaluate the method quality for each document structure separately.

Course of the experiment In this case, the testing procedures were run only on a filtered set of regions. In each of the experiments this set contained only regions from the ground truth and the system output that belonged to one specific class. The evaluation procedure described by Algorithm 3 was used.

Results In Table 4, the results for this experiment are presented. The obtained results in term of precision and recall are acceptable for distinguished classes. Only class

Table 3 The paragraph recognition in relation to the different types of documents

Prec (%)	Rec (%)	*F*-Score
One-column document		
87.14	86.77	86.95
Many columns document		
85.87	84.34	85.09
Documents with many fonts		
84.12	88.09	86.05

Table 4 The result of comparison for the structure classes (collocation of segments done using Algorithm 3)

Class	Prec (%)	Rec (%)	*F*-Score
Paragraph	87.40	85.12	86.24
Header	88.28	80.15	84.01
Page footer	84.18	79.16	81.59
Page header	88.16	86.41	87.27
Table	96.31	65.23	77.78
Caption	88.04	88.12	88.07
Graphic element	92.01	80.22	85.71
Title	69.15	44.98	54.50

Title has unsatisfying scores. The reasons are discussed in Sect. 5.7.

5.6 Evaluation of gridded table recognition

The method was performed on the 712 non-labelled segmented objects obtained from the text segmentation routine. They contained 49 gridded tables. Tests gave a precision of 93 % and a recall of 25 %. Despite the low recall, the method still manages to find 25 % of the tables missed by the previous method.

5.7 Discussion of the results

The selected reference segmentation methods used in the comparison are representatives of three main structure recognition approaches: *top–down*, *bottom–up* and *hybrid*. All of the tested methods were compared using the described metric and the evaluation procedure specifically design for this purpose. As shown in Table 2, for all tested methods the obtained results were comparable but it is worth mentioning that DSR, performing two tasks—segmentation and classification, achieves a precision of 87 % and a recall of 86 %.

Referring to classification results, relatively low results, in comparison to the other structures, are achieved for the *Title* class. This is caused by a lack of knowledge whether the page contains title elements or not and what is the number of the processed page. In addition, some pages contain multiple titles (in several languages or title with lengthy subtitles). Incorrect classification occurs when the title was not surrounded by any other objects but paragraphs. For these cases, it was difficult to formulate graphic features distinguishing titles from paragraphs. An essential problem causing lower results for title and header recognition is deciding whether a text line (distinguished in relation to dominant document content) is really a title or header. Because information about specific location of a text line (in the whole document not just the current page) is not used in the recognition process, it is not possible to determine whether the page should or should not contain a title.

When considering the results it is worth mentioning that classification of some objects depends on OCR tools. The applied system, *Tesseract* developed by Google, is one of the best freeware systems. Commercial OCR systems should achieve better results.

6 Summary

Our goal was to propose an efficient and universal method of segmentation and recognition of logical document structures to support collecting and processing a huge corpus of documents in the NEKST project. The proposed method uses knowledge about the document structure in a small degree therefore, it is independent of the document layout. The method processes page images so that it remains independent of specific electronic document formats. It is based on simple features distinguishing classes of structures. Its results for paragraph structure recognition (precision 87 % and recall 85 %) are comparable to those of methods offering only segmentation.

During research a corpus of documents with more than 1,200 document pages was collected. Each of them was analysed and manually annotated. It is worth emphasizing that the corpus was collected with care for the variety of layouts. Though the collected documents, used for the experiments, were in Polish the presented method is not limited to this language. It represents a general approach, but the document should fulfil all the described requirements. Currently, the corpus does not explicitly list page numbers or the order of pages within a single document. The inclusion of this information is expected to have a positive effect on title recognition.

Word count statistics extracted from the documents can be disturbed if page headers and footers are not accounted for. Due to their repetitive nature the text contained in such structures has a negative impact on keyword extraction and disturbs word count. From the perspective of semantic analysis, multiple text columns create another problem.

They are solved by the proposed document structure recognition method.

Acknowledgments This work was partially supported by the Innovative Economy Programme project POIG.01.01.02-14-013/09-00.

References

1. Agrawal M (2009) Voronoi++: a dynamic page segmentation approach based on voronoi and docstrum features. In: Document analysis and recognition. 10th international conference on document analysis and recognition, ICDAR '09, pp 1011–1015
2. Bach NX, Minh N, Oanh T, Shimazu A (2013) A two-phase framework for learning logical structures of paragraphs in legal articles. ACM Trans Asian Lang Inf Process 12(1):1–32 (Article No. 3)
3. Belaid A, D'Andecy VP, Hamza H, Belaid Y (2008) Administrative document analysis and structure. In: Marenglen B, Xhafa F (eds) Learning structure and schemas from documents. Studies in computational intelligence, vol. 375, pp 51–72
4. Breuel T (2002) Two geometric algorithms for layout analysis. In: International workshop on document analysis systems, pp 188–199
5. Cao H, Prasad R, Natarajan P, MacRostie E (2007) Robust page segmentation based on smearing and error correction unifying top-down and bottom-up approaches. In: International conference on document analysis and recognition, vol. 1, pp 392–396
6. Carpenter G, Grossberg S, Markuzon N, Reynolds JH, Rosen DB (1992) Fuzzy artmap: a neural network architecture for incremental learning of analog multidimensional maps. IEEE TNN 3(5):698–713
7. Cesarini F, Marinai S, Sarti L, Soda G (2002) Trainable table location in document images. In: Proceedings of international conference on pattern recognition, pp 236–240
8. Chiu P, Chen F, Denoue L (2010) Picture detection in document page images. In: Proceedings of the 10th ACM symposium on document engineering, pp 211–214
9. Dillencourt M, Samet H, Tamminen M (1992) A general approach to connected-component labeling for arbitrary image representations. J ACM 39(2):25–280
10. Dori D, Doermann D, Shin C, Haralick R, Phillips I, Buchman M, Ross D (1997) Handbook of character recognition and document image analysis, chap. The representation of document structure: a generic object-process analysis, pp 421–456. World Scientific, Singapore
11. Drivas D, Amin A (1995) Page segmentation and classification utilizing bottom-up approach. In: Proceedings of the third international conference, document analysis and recognition, pp 610–614
12. Fankhuser P, Xu Y (1993) Mark it up! an incremental approach to document structure recognition. Electron Publ 6(4):447–456
13. Fletcher LA, Kasturi R (1988) A robust algorithm for text string separation from mixed text/graphics images. IEEE Trans PAMI 10(6):910–918
14. Gatos B, Danatsas D, Pratikakis I, Perantonis SJ (2005) Automatic table detection in document images. In: Proceedings of the third international conference on advances in pattern recognition (ICAPR'05), lecture notes in computer science, vol. 3686, pp 609–618
15. Hu J, Kashi R, Lopresti D, Wilfong G (2000) Medium-independent table detection. In: Proceedings of the SPIE document recognition and retrieval VII, pp 291–302
16. Jain AK, Zhong Y (1996) Page segmentation using texture analysis. Pattern Recognit 29(5):743–770
17. Kawanaka H, Shiroyama Y, Tsuruoka S, Shinogi T, Yamamoto K (2008) A study on document structure recognition of discharge summaries for analogous case search system. In: The eighth IAPR workshop on document analysis systems, pp 423–430
18. Kieninger T, Dengel A (1999) The t-recs table recognition and analysis system. In: Selected papers from the third IAPR workshop on document analysis systems: theory and practice, pp 255–269
19. Kieninger T, Dengel A (2005) An approach towards benchmarking of table structure recognition results. In: Proceedings of the eighth international conference on document analysis and recognition. ICDAR '05IEEE Computer Society, Washington, DC, pp 1232–1236
20. Kruatrachue B, Moongfangklang B, Siriboon K (2007) Fast document segmentation using contour and x-y cut technique. Int J Comput Inf Syst Control Eng 1(5):1425–1427
21. Kruatrachue B, Suthaphan P (2001) A fast and efficient method for document segmentation for ocr. In: Proceeding of IEEE region 10 international conference on electrical and electronic technology, vol. 1, pp 381–383
22. Liang J, Phillips IT, Haralick RM (2000) Consistent partition and labelling of text blocks. Pattern Anal Appl 3(2):196–208
23. Legourgiois F, Bublinski Z, Emptoz H (1992) A fast and efficient method for extracting text paragraphs and graphics from unconstrained documents. In: Proceedings of 11th IAPR international conference B: pattern recognition methodology and systems, pp 272–276
24. Mao S, Rosenfeld A, Kanungo T (2003) Document structure analysis algorithm: a literature survey. In: DDR SPIE, vol. 5010, pp 197–207
25. Marinai S, Gori M, Soda G (2005) Artificial neural networks for document analysis and recognition. IEEE Trans Pattern Anal Mach Intell 27(1):23–35
26. Markowska-Kaczmar U, Minda P, Ociepa K, Olszowy D, Pawlikowski R (2011) Towards automatic image annotation supporting document understanding. In: International conference hybrid artificial intelligence systems, lecture notes in computer science, vol. 1, pp 420–427. Springer, Berlin
27. Meunier JL (2005) Optimized xy-cut for determining a page reading order. In: ICDAR, international conference on document analysis and recognition, pp 347–351
28. Nagy G, Seth S (1984) Hierarchical representation of optically scanned documents. In: Proceedings of the 17th conference on pattern recognition, pp 347–349
29. Nikolaou N, Makridis M, Gatos B, Stamatopoulos N, Papamarkos N (2010) Segmentation of historical machine-printed documents using adaptive run length smoothing and skeleton segmentation paths. Image Vis Comput Arch 28:590–604
30. Nikolaou N, Makridis M, Gatos B, Stamatopoulos N, Papamarkos N (2010) Segmentation of historical machine-printed documents using adaptive run length smoothing and skeleton segmentation paths. Image Vis Comput 28(4):590–604
31. Niyogi D, Srihari S (1995) Knowledge-based derivation of document logical structure. In: Proceedings of the international conference on document analysis and recognition, pp 472–475
32. O'Gorman L (1993) The document spectrum for page layout analysis. IEEE Trans Pattern Anal Mach Intell 15(11):1162–1173
33. Randriamasy S, Vincent L (1994) Benchmarking page segmentation algorithms. In: Proceedings IEEE computer society conference on computer vision and pattern recognition, pp 411–416

34. Rangayyan R (2005) Biomedical image analysis (Biomedical Engineering). CRC Press, Boca Raton
35. Sain K, Dasgupta A, Garain U (2011) Emers: a tree matching based performance evaluation of mathematical expression recognition systems. Int J Doc Anal Recognit (IJDAR) 14(1):75–85
36. Sainz Palmero G, Dimitriadis Y, Sanz Guadarrama R, Cano Izquierdo J (2002) Neuro-fuzzy art-based document management system: application to mail distribution and digital libraries. Eng Appl Artif Intell 15:17–29
37. Shafait F, Smith R (2010) Table detection in heterogeneous document. In: Proceedings of international workshop document analysis systems, pp 65–72
38. Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE TPAMI 22:431–439. Normalized Cuts software. http://www.cis.upenn.edu/jshi/software/
39. e Silva A (2009) Learning rich hidden Markov models in document analysis: Table location. In: Proceedings of international conference on document analysis and recognition, pp 843–847
40. Simon A, Pret J, Johnson A (1997) A fast algorithm for bottom-up document layout analysis. IEEE Trans Pattern Anal Mach Intell 19:273–276
41. Strouthopoulos C, Papamarkos N (1998) Text identification for document image analysis using a neural network. Image Vis Comput 16:879–896
42. Wang Y, Haralick R, Phillips I (2006) Document zone content classification and its performance evaluation. Pattern Recognit 39(1):57–73
43. Wang Y, Phillips IT, Haralick R (2001) Automatic table ground truth generation and a background-analysis-based table structure extraction. In: Proceedings of the sixth international conference, document analysis and recognition, pp 528–532