Big Data Research 24 (2021) 100195

Contents lists available at ScienceDirect

Big Data Research

www.elsevier.com/locate/bdr

Comparative Study of Layout Analysis of Tabulated Historical Documents ☆

Xusheng Liang [a], Abbas Cheddad [a,*], Johan Hall [b]

[a] Department of Computer Science, Blekinge Institute of Technology, SE-371 79, Karlskrona, Sweden
[b] ArkivDigital AB, Växjö, Sweden

ARTICLE INFO

Article history:
Received 2 February 2020
Received in revised form 6 September 2020
Accepted 2 January 2021
Available online 8 January 2021

Keywords:
Layout analysis
Image processing
Machine learning
Historical handwritten documents
Feature extraction

ABSTRACT

Nowadays, the field of multimedia retrieval system has earned a lot of attention as it helps retrieve information more efficiently and accelerates daily tasks. Within this context, image processing techniques such as layout analysis and word recognition play an important role in transcribing content in printed or handwritten documents into digital data that can be further processed. This transcription procedure is called document digitization. This work stems from an industrial need, namely, a Swedish company (ArkivDigital AB) has scanned more than 80 million pages of Swedish historical documents from all over the country and there is a high demand to transcribe the contents into digital data. Such process starts by figuring out text location which, seen from another angle, is merely table layout analysis. In this study, the aim is to reveal the most effective solution to extract document layout w.r.t Swedish handwritten historical documents that are featured by their tabular forms. In short, outcome of public tools (i.e., Breuel's OCRopus method), traditional image processing techniques (e.g., Hessian/Gabor filters, Hough transform, Histograms of oriented gradients -HOG- features), machine learning techniques (e.g., support vector machines, transfer learning) are studied and compared. Results show that the existing OCR tool cannot carry layout analysis task on our Swedish historical handwritten documents. Traditional image processing techniques are mildly capable of extracting the general table layout in these documents, but the accuracy is enhanced by introducing machine learning techniques. The best performing approach will be used in our future document mining research to allow for the development of scalable resource-efficient systems for big data analytics.

1. Introduction and background

In the computer vision field, accurate document image layout segmentation is still a challenging research issue. Such feature facilitates information retrieval which is an operation that enables users to retrieve requested information from a collection of information resources, which can be content-based indexing or full-text. With the current advanced image processing technology, contents in printed or handwritten documents can be recognized and transcribed into digital data. Retrieving the requested information from this type of system can save users a lot of time to manually search the information in documents. This whole procedure of transforming the physical document information into representative system data is called document digitization.

Nowadays, document digitization has been widely used in transcribing historical record into digital data. This work aims to find out the best suitable document layout analysis solutions in respect to the Swedish handwritten historical documents which are featured by their tabular forms containing handwritten text.

Among the document digitization procedures, document layout analysis needs to be deployed. Document layout analysis is a mean to identify different functional/logical content elements (e.g. sentences, titles, captions, author names, and addresses) on a given page. It is realized by segmenting physical contents (e.g. pixels, characters, words, lines, figures, tables, and background) on the page and classify them into predefined functional/logical categories, in other words, by assigning these classified entities labels. Document layout analysis plays a crucial role within the document digitization procedure because the correctness of layout analysis determines whether a subsequent text recognition procedure is operated on the correct text object. When implementing layout analysis, there are generally two approaches to carry out this procedure [1], top down approach and bottom up approach. The

☆ This article belongs to Special Issue: Big Data in Industry.

* Corresponding author.
E-mail address: abbas.cheddad@bth.se (A. Cheddad).

https://doi.org/10.1016/j.bdr.2021.100195

Table 1
A taxonomy of selected key techniques of layout analysis.

Method	Description	Drawback
Statistical Inference		
Top down (TD)/ bottom up (BU)/ Hybrid Algorithms (H)	(TD) X-Y Cut [2], shape-directed cover [3], white streams based segmentation [4] and whitespace cover [5]. (BU) Docstrum [6], Voronoi diagram [7], run-length smearing [8] and segmentation with generative Markov model [9]. (H) split-and-merge strategy was also included in the analysis in [10] [1].	These algorithms are proposed to segment the general content block instead of segmenting table content in our case.
Breuel's study [11] (OCRopus Algorithm)	This OCRopus system supports both layout analysis and text line recognition functionalities. It has build-in functions supporting pre-processing operations such as binary morphology [12], adaptive thresholding [13], RAST-based skew correction [14], page frame detection [15] and maximal whitespace rectangle processing [16].	Disconnected lines, see Section 2.2.
Run-length algorithm, projection profile and Hough line transform	In [17], Jahan and Ragel developed a document layout structure extraction approach by exploiting horizontal/vertical projection profiles in a top-down fashion. In [18], Gatos et al. carried out the line detection task by computing the horizontal and vertical black runs in the binarized image (using adaptive binarization, skew correction [19] and noise border removal [20].) and filtering the black runs with several rules applied to the detected lines. In [21], Tian et al. proposed the use of run length smoothing algorithm (RLSA) and Hough line transform to detect lines longer than a predefined threshold.	Even though these methods work well in their scenarios, both the run-length and the projection profile applications require the de-skewing process to be correctly performed. Unfortunately, the skewness exhibited in our dataset is more complex since most of the two pages scans have skew at different degrees (i.e., cannot have a uniform deskewness criterion).
Hough line transform and Fast Fourier Transform (FFT)	In Lee's study [22], the author also tried to exploit Hough line transform to detect lines and refined the search using lines periodicity with FFT.	This method does not work well on low quality scanned images featuring some broken solid lines.
Directional Single-Connected Chain (DSCC)	Zhang et al. [23] computed two DSCC co-line distance to estimate the probability that two DSCCs are on the same line.	Allegedly, this method performs well at the precision level; however, the time cost is expensive due to its complex computational process.
Machine Learning		
Contextual features and fixed-point model/neural network	In [24], the authors proposed a table detection solution by developing foreground and background features which enclose foreground block characteristics and contextual information. These features are fed into the fixed-point model to classify/label these blocks. They acquired the text blocks by segmenting the document into a homogeneous region (e.g. text, graphics) using Leptonica library[1] followed by a morphological closing operation to produce the text blocks. Rashid et al. [25] proposed a similar approach but used a neural network classifier.	These methods are generally deployed to differentiate table layout amid mixed content in a cluttered document page. Since in our Swedish historical handwritten documents there are several empty table cells in the document page, these approaches do not fit into our context.
Support vector machines (SVM)	In [26], run-length algorithm is used in vertical/horizontal directions to detect corresponding lines. Subsequently, they extracted features from connected components (CCs) such as enclosure, regularity of structural arrangement and convex deficiency; finally, they fed the extracted features into SVM for classification.	These methods' pre-processing phase (using CCs features) seems a well fit approach for our requirement of detecting table layout elements in our dataset.
Convolutional Neural Network (CNN)	In [27], the vertical and horizontal projections are fed to CNN to train the model to perform the multi-class classification of text, image and table in a given document image.	
Multilayer perceptron classifier (MPC)	In [28], the authors used MPC to classify CCs into text/non-text category by using features combining context features and aspect ratio, area ratio and density.	
Faster R-CNN (FCNN)	In [29], their solution benefits from transfer learning to fine tune FCNN pre-trained model that was proposed by Ren et al. [30] and Shelhamer et al. [31] to identify table elements such as rows, columns and cells.	

top-down approaches segment a page as a whole into one or more content blocks and recursively segment the segmented blocks into paragraphs, lines, words and character. In contrast, the bottom up approach works in the opposite way, the first group of connected components produced by the black and white pixel into characters, then words, then text lines. However, these two approaches are occasionally combined.

Currently, there exist several algorithms that can be used for layout analysis tasks. In what follows, several studies have been extracted, reviewed and summarized in Table 1.

As for the datasets, studies [24] and [25] have tested their solutions on the UW-III (University of Washington) dataset, UNLV dataset and their own additional dataset; studies [26] and [28] used the MAURDOR campaign dataset [32].

In respect to the Swedish historical handwritten document's dataset, each scanned image is a two pages content of historical handwritten textbooks in a table form. Table layout in these documents may exhibit distortions in terms of skew and curl during the scanning process. Even though there are many existing document layout analysis techniques discussed, most of these techniques/tools are unfortunately not generic and therefore cannot easily extend to our data set. The main problem here is that some table lines are skewed in different degrees on each page within the same scanned image which cannot be handled by applying traditional deskew techniques. Moreover, the table lines in some scanned images are not continuous (i.e., show some cut offs). This work focuses on investigating the performance of 2D filters with and without other techniques/tools' support such as machine learning and Optical Character Recognition (OCR).

In a nutshell, the contribution of this work is twofold.

- First, its comparative study of the different existing methods for document layout analysis and extraction.
- Second, we attempt to enhance a machine learning based pipeline by introducing 2D filtering (i.e., oriented Hessian filter) to mask out the text and help boost the classification process.

[1] An open source library: http://www.leptonica.com/.

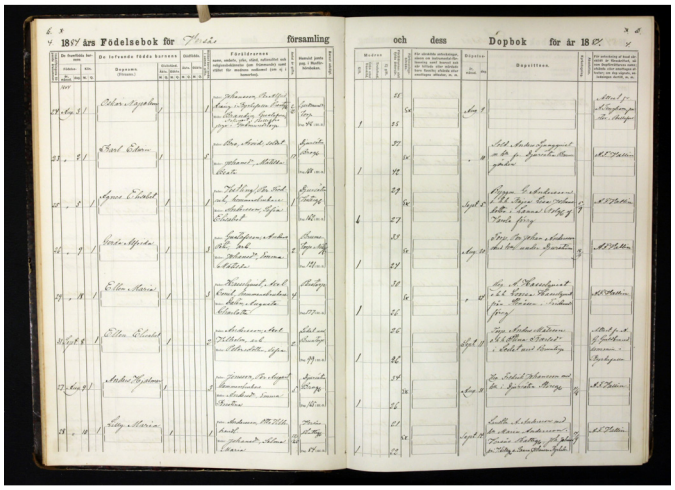

Fig. 1. The original image to be examined in the next sections' analysis.

2. Methodology

Since this work was done progressively, we used different criteria in different phases of our study. Initially, we tried to identity open source tools that have layout analysis functionality enabling us to run some quick experiments on the scanned documents. Our choice landed on a tool called OCRopus whose underlying algorithm is fully described in Breuel's publication [11]. The results, unfortunately, show poor performance of the OCRopus. We opt to still discuss these negative results to provide an evaluation of the tool for the concerned research community. This inadequate performance of the tool motivated us to try identifying other plausible techniques. Hence, the literature review reveals that the methods can be divided into two categories, direct methods (i.e., Hough transform, run-length encoding) and machine-learning based methods. Our data set encompasses tables with heavily skewed lines which the direct methods are not designed for. From our prior studies, we know that kernel-based filters (i.e., Hessian and Gabor filters) are used for detecting curved faded lines (i.e., retinal vascular structure extraction, a.k.a, vesselness, in fundus images). Henceforth, we decided to scrutinise three major venues, Breuel's method (OCRopus) [11], the two kernel-based filters, and the machine-based approaches.

2.1. 2D kernel filter applications

With the purpose of extracting horizontal and vertical table lines in our Swedish historical handwritten dataset, two filters (Hessian and Gabor filters) are adopted. Note that the Hough line detection is a quite different strategy than filtering. It detects lines based on pixel quantity in the same line instead of using the kernel to perform filtering. In the following sections, a brief description of the Hessian and Gabor filters is given. Their applications are discussed in section 3.3. Fig. 1 depicts the document which we exemplify in our analysis in the next sub-sections.

2.1.1. Hessian filter

As a square matrix with second-order partial derivatives of a function, the Hessian matrix can be used to approximate the output of a second derivative test on a grayscale image to detect local maximum, local minimum or a saddle point by considering the image as a two-variable function. For instance, Frangi et al. [33] developed 2D Hessian filter to enhance the vesselness structure in X-ray vessel visualization and quantification. The main theory of their study is using a second derivative Gaussian kernel with scale s to measure the contrast between the region inside and outside the range $(-s, s)$. Inspired by these characteristics of vascular structures, we opt to explore the Hessian filter on the extraction of the line structure in our Swedish historical handwritten document dataset. The outcome of the application of this Hessian filter is shown below in Fig. 2.

2.1.2. Gabor filter

Gabor filter is a linear filter that is composed of Gaussian kernel function modulated by a sinusoidal plane wave [34]. It is generally used to conduct texture analysis. In this case, the table lines lying in our Swedish historical handwritten document image dataset can be detected with Gabor filter by conducting the texture analysis in the horizontal and vertical directions. Different effects of Gabor filter can be intuitively perceived by tweaking different parameters [35]. In our detection test, two parameters (i.e. wavelength λ and orientation θ) are used. By tuning these two parameters, texture with different direction and different density can be detected. Here, texture density can be interpreted as frequency. The wavelength λ of the Gabor filter is the wavelength of the sinusoidal wave carrier, it associates the texture density (frequency) to be detected. The orientation θ of the Gabor filter concerns the texture direction to be detected. In our test, different wavelengths have been tested on our dataset. Examples of the test outcome are presented below in Fig. 3 and Fig. 4.

As seen from the results above, all lines in the original image can be detected by using either of the three wavelengths. As the wavelength gets longer, the detected lines get thicker. Comparing this to the detected lines by the Hessian filter, Gabor detected lines are smoother (less coarse).

2.2. Open source tool application

As we mentioned in the previous related works, *Breuel's* OCRopus [11] is a popular open source OCR tool with layout analysis capabilities. In this section, OCRopus is tested and studied regarding its layout analysis functionality. Based on the online description of OCRopus,[2] the tool is capable of recognizing sections of text lines, ruling lines, sidebars, captions, tables, line drawing, continuous tone image, page number, headers and footers.

Parameters have been experimentally tuned. It is found that the best setup for OCRopus to be applied to our dataset is the following:

```
ocropus-gpageseg -d --minscale 1 --maxseps 500
--maxcolseps 0 --csminheight 5
--maxlines 600 example.jpg
```

Command parameter interpretations are listed in Table 2.

By inspecting the output, we found that OCRopus does not detect the horizontal lines, which negates our goal of detecting the table layout. To exploit OCRopus regarding our goal, we have tried rotating the image and removing the text component pixels followed by applying OCRopus column separator detection function.

Table 2
OCRopus command parameter interpretations.

-d:	debug option
–maxseps:	the maximum black column separator
–maxcolseps:	maximum white column separators
–maxlines:	maximum text lines
–minscale	minimum scale permitted in image size check
–csminheight	minimum height of detecting white column separator

[2] https://github.com/tmbdev/ocropy/wiki/OCRopus-File-Formats#physical-layout.

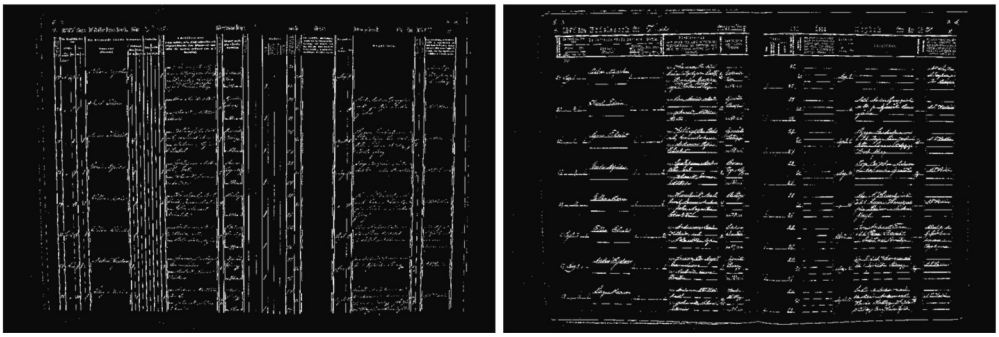

Fig. 2. Outcome of the Hessian filter on the image shown in Figure1 imposing vertical filtration (left) and horizontal filtration (right).

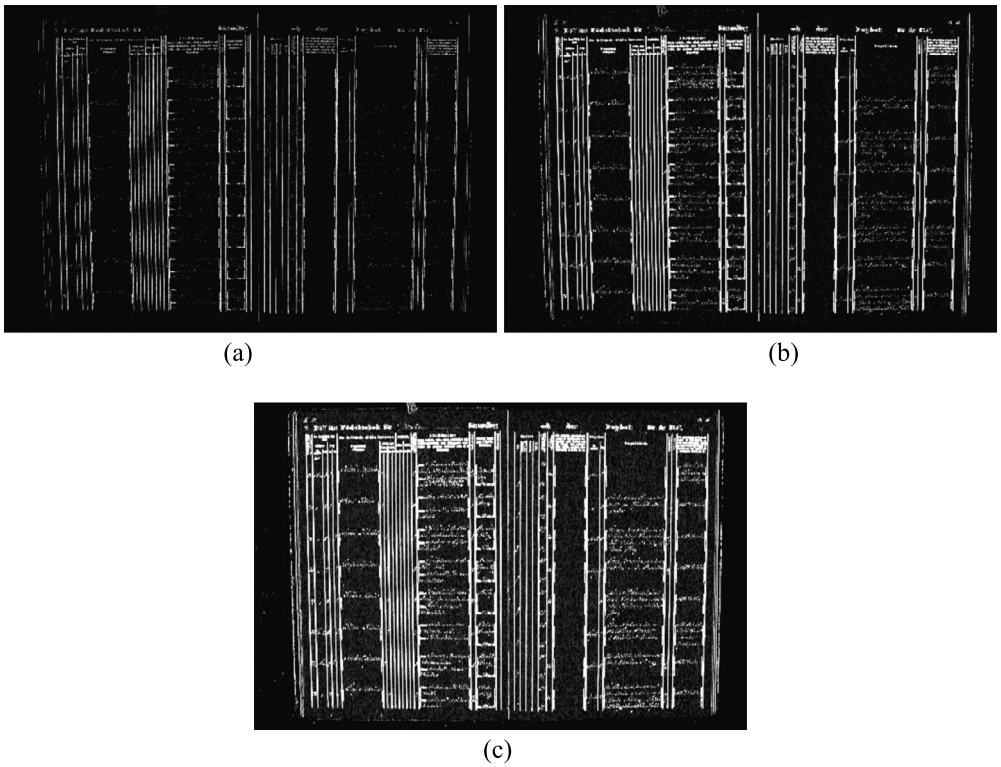

Fig. 3. Gabor filtration result with orientation 0 degrees and three different wavelengths, results with wavelengths 3, 4 and 5 for (a) (b) and (c), respectively.

The horizontal lines detected by OCRopus in this way are unfortunately disconnected. The mask image corresponding to the example image in Fig. 1 is depicted in Fig. 5. Since OCRopus did not provide what we hoped for, we decided to abandon the use of OCRopus in subsequent experiments.

2.3. Investigated solutions

Before implementing each solution, we first pre-processed each image by applying contrast limited Adaptive histogram equalization (CLAHE) [36], [37] technique. This technique enhances contrast while preventing the amplification of noise. It helps in our case to separate the foreground and the background when applying Hessian and Gabor filters.

2.3.1. HessMask - Hessian filter masking

This solution mainly utilizes the filtered images by Hessian filter to produce a binarized image that reflects the table layout. The main idea is to dilate the filtered image of the second derivative along the diagonal direction and to use it as a mask to remove the residual text pixels in the filtered image along the vertical and horizontal filter output, see Fig. 6.

As shown in Fig. 6, image *Dxx* embodies the 2nd derivative along the *x*-axis to detect horizontal lines, similarly, *Dxy* embodies the diagonal line pixels and *Dyy* embodies the vertical lines pixels. Next, since the handwritten text is cursive, we extract it by dilating the image *Dxy* then we use it to mask out the text from the aggregated signal of *Dxx* and *Dyy*. This operation removes, to a certain extent, the text and produces the result image with only detected horizontal and vertical lines.

The figures, Fig. 7(a) and (b), show outcome examples of two typical layout documents when applying solution I. This solution's performance is adequate in most document types. However, some horizontal line segments are not detected due to either low-contrast and/or the interference of text residue from the previous stage (formally known in the document analysis field as bleed-through).

2.3.2. GaborMask - morphological opening with Gabor filter

As mentioned in the previous section, a Gabor filter is a linear filter that detects frequency components along specific directions. As shown earlier, applying Gabor filter on our Swedish historical handwritten documents gives quite acceptable result on detecting

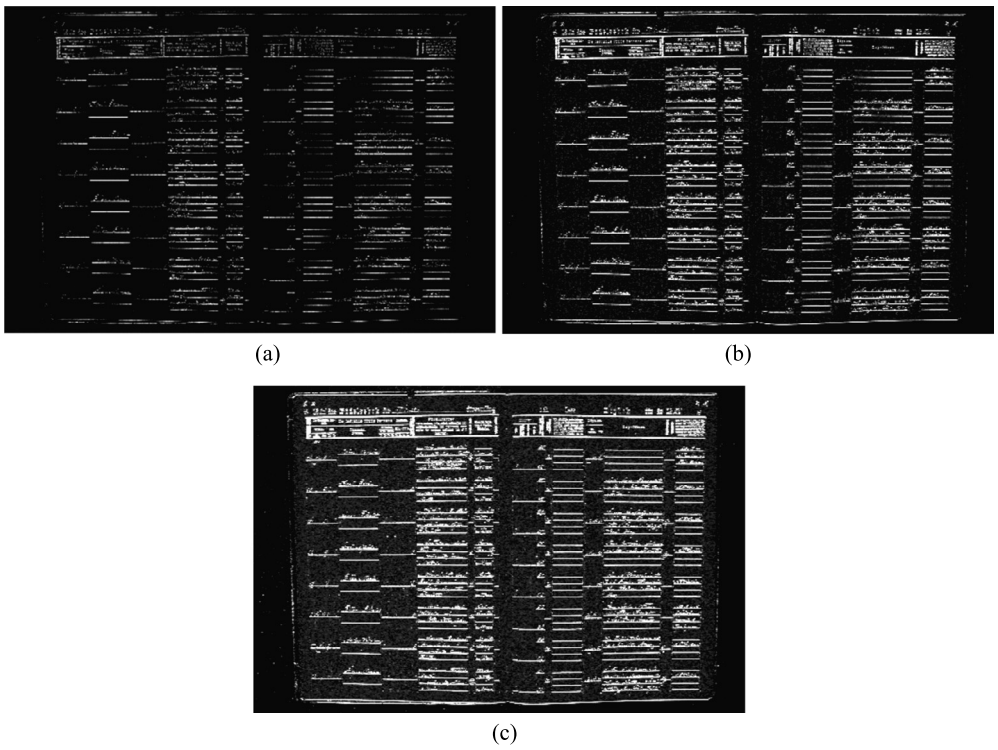

Fig. 4. Gabor filtration result with orientation 90 degrees and three different wavelengths, results with wavelengths 3, 4 and 5 for (a) (b) and (c), respectively.

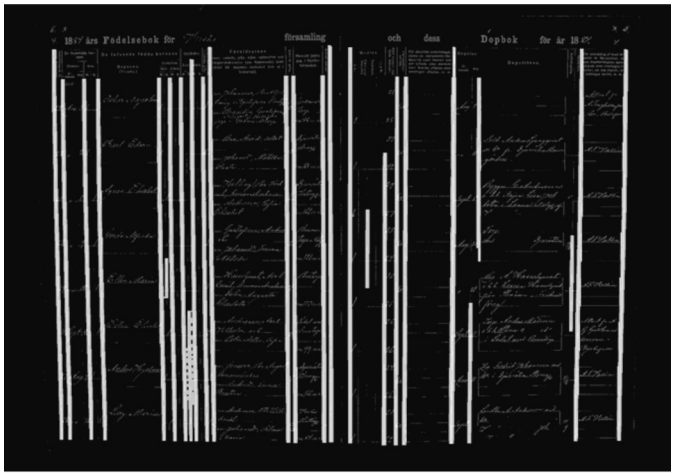

Fig. 5. Column separator in mask image produced by OCRopus [11].

vertical and horizontal lines. Comparing the outcome of the Hessian filter and the Gabor filter, it can be seen that text components residuals in the filtered images produced by Gabor filter are much less than the ones produced by the Hessian filter. Moreover, the quality of detected lines produced by Gabor filter is reasonably superior to that produced by the Hessian filter. Hence, we apply morphological opening on the filtered image produced by the Gabor filter where vertical and horizontal rectangles are used as structuring elements (e.g. kernels) to remove the remainder of text in the horizontal and vertical filtered images, respectively. Fig. 8 shows the workflow of this solution.

The wavelength of the Gabor filter and the shape of the structuring element used in morphological opening operation greatly affect the solution outcome, therefore, they need careful tuning. In respect to the wavelength parameter, a longer wavelength leads to more intact lines to be produced but introduces more noise. In respect to the structuring element's shape, we experimented on different possibilities and we found out that the slender rectangular structure fits well our demands. We also found that over elongated structure elements remove text pixels at the expense of the destruction of lines integrity. Hence, we resorted to using the wavelength of 4 and to having a regular structuring element shape with length of 41 and width of 5. Fig. 9 shows the result of this solution.

From these experiments, we can see that *GaborMask* gives apparently better result than *HessMask*, especially when it comes to detecting horizontal lines. However, some horizontal lines with severe low contrast still pose a problem.

2.3.3. Connected component and machine learning classifiers

In this section, an alternative solution is explored by incorporating machine learning techniques. In the work reported in [26] and [28], CCs are extracted and their features are analyzed and fed into machine learning classifiers to predict table/non-table lines classes. As we mentioned before, this strategy cannot handle CCs which have text and line elements connected. However, the results presented earlier show that Gabor filter can aid along this line to a certain extent; moreover, Hough line transform can also help in dissipating the text pixels if applied on each CC. With the aforementioned set-up, we can advocate for an adapted version of solution similar to [26] and [28] to handle the Swedish historical handwritten document dataset. In general, the whole process can be broken down into four stages: preprocessing, CCs production, feature extraction and classification of CCs. The workflow of the solution is presented in Fig. 10.

In the training process, and because of lack of ground truth data, we had no option but to manually annotate a sample set of CC instances as a training set. In what follows, our solution procedures are detailed.

Pre-processing: In this stage, the main purpose is enhancing contrast to distinguish text and line element from the background. CLAHE is applied to achieve this as stated earlier. Thereafter, Gabor

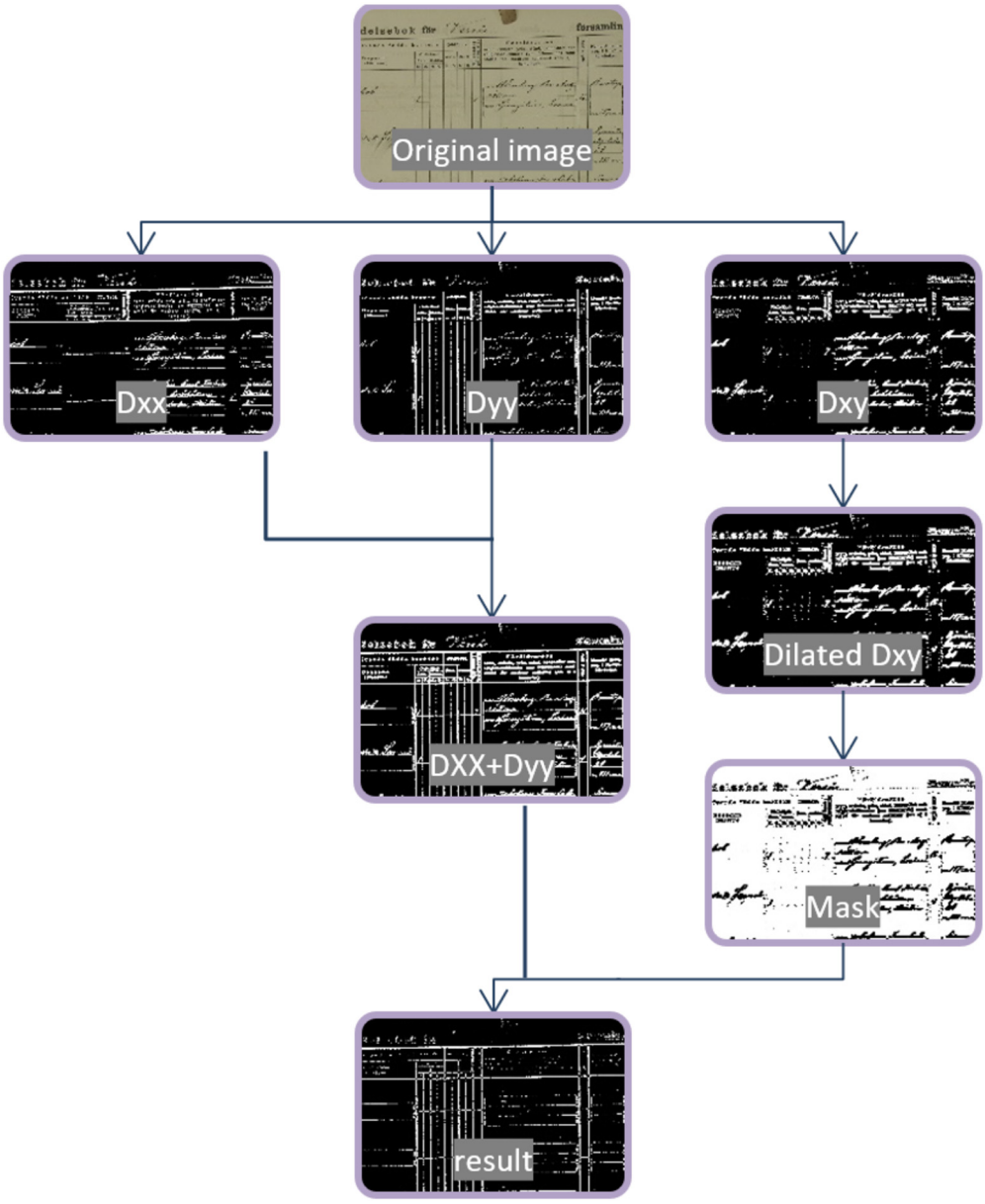

Fig. 6. Process flow of mask image production using 2D Hessian filters.

filter was tuned to extract the roughly horizontal and vertical foreground elements respectively as before where we did show also the merit over choosing the Hessian filter. Next, Otsu binarization process was carried out. Very small blobs which we consider to be noise are discarded.

In respect to CCs with connected text and line segments, Hough line transform is applied to detect straight lines as shown in Fig. 11.

Training Set (*CC Extraction and Annotation*): As for our annotated training set, we annotated CCs from 111 images. This yielded a total of 668,078 non-line CCs, 33,850 horizontal line CCs and 16,213 vertical line CCs in this dataset.

Feature extraction from CCs: At this stage, we choose two approaches to conduct the feature extraction task. One is using transfer learning, and the second approach uses custom features inspired by [26] and [28].

A. *Transfer learning features (TL_SVM):* In deep learning domain, transfer learning is getting popular to carry out image processing tasks in the recent years, in biomedical imaging field [39] and in plant species recognition [40], to name a few. The most common use case of transfer learning in image processing nowadays is making use of pretrained network model that is able to produce features to recognize instances similar to the original recognized object(s) of the network. With transfer learning, it saves the end-user a substantial time and hurdle to train and tune a model on a target dataset. Thus, we adopt this mechanism, which is similar to that used in [29].

Convolutional neural network deep features are extracted from a pre-trained model, AlexNet, using MATLAB [41]. The AlexNet model [42] is a pretrained model on 1.2 million images from ImageNet LSVRC-2010 [43] contest and is able to classify images into 1000 different classes. To extract AlexNet features, we first produced segmented images of CCs, resized them to a uniform size (e.g., a network architecture requirement) and then fed them to the network to produce the needed deep features.

B. *Custom features (CF_SVM)*: The second alternative, is to use the traditional approach to analyze image characteristics and ex-

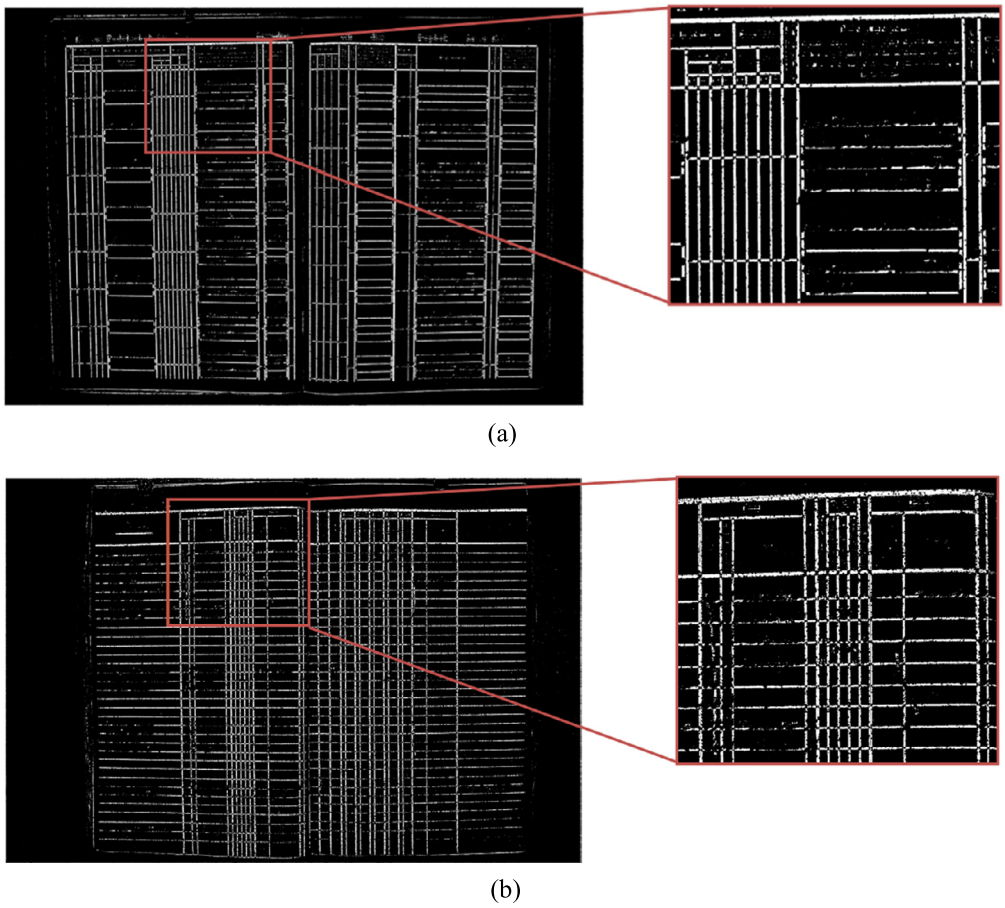

Fig. 7. Examples of output masks using *HessMask*.

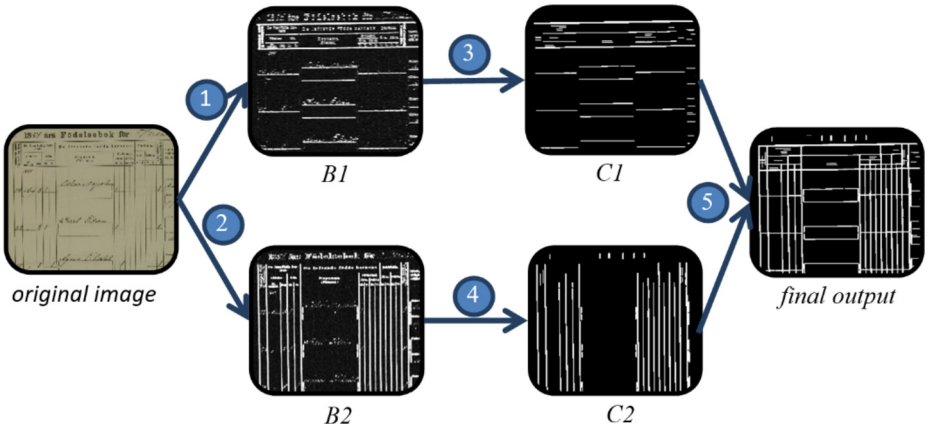

Fig. 8. Workflow of *GaborMask*, (1) apply the Gabor filter along the horizontal direction; (2) apply the Gabor filter along the vertical direction; (3) apply morphological opening with horizontal rectangle as the structuring element to remove text component, binarize the image afterwards with the Otsu algorithm [38] to produce image C1; (4) the same procedure as in (3) but using vertical rectangle as structuring element; (5) fuse C1 and C2 to produce the final detected mask image indicating the expected document table layout.

tract handcrafted features which describe the table lines characteristics. By studying all the extracted features used in [26] and [28], we employed those which are suitable for this purpose. Moreover, we extended the feature set with additional ones which encode useful information about table lines. In Table 3, the custom features are listed.

GLCM captures image texture features [44], [45]. It uses a matrix to indicate the distribution of co-occurring grey scale value pairs at a specified offset in the image. In scikit-learn library, GLCM [45] is represented as a 4-dimensional array. The two additional dimensions are used for multiple offsets and offset orientations. In our study, GLCM statistics is used to encode CC contextual information and to uncover the relationship of current CC and neighbouring CCs. Two GLCMs are computed on two patches extracted from both ends of the CC.

HOG [46] is an image feature descriptor using histograms to count the occurrence of oriented gradients of a localized portion of the image. It is commonly used in object detection. In our study,

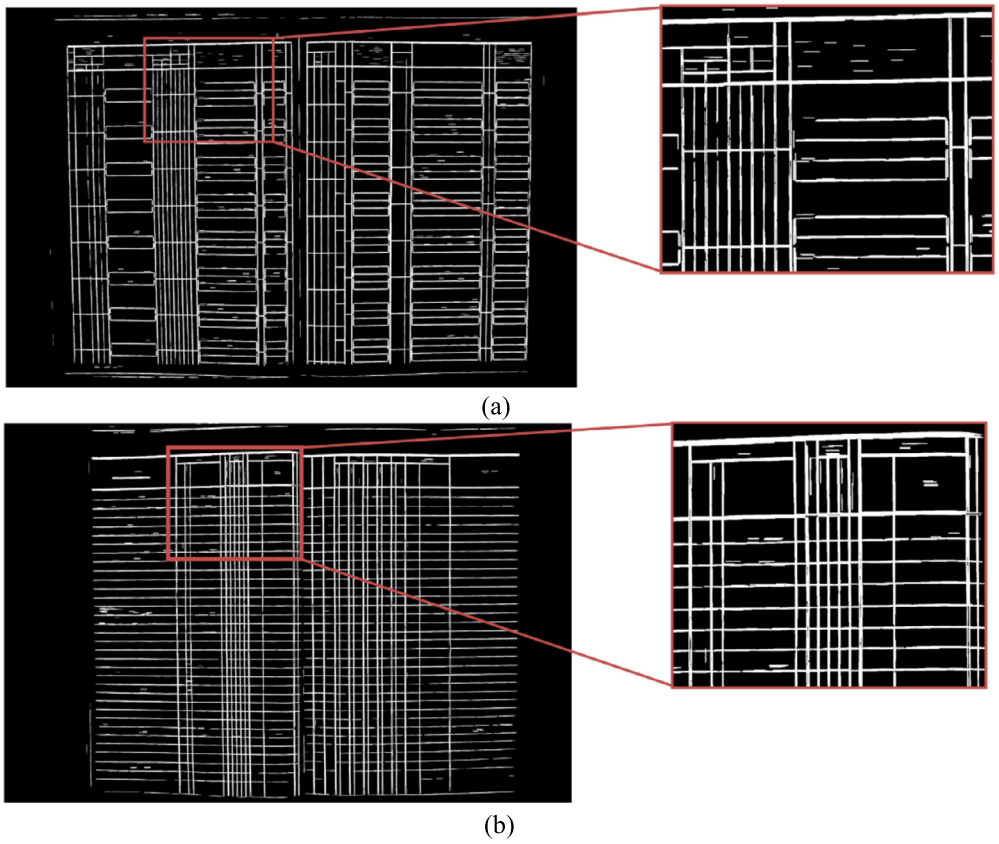

Fig. 9. Examples of two output masks by using *GaborMask*.

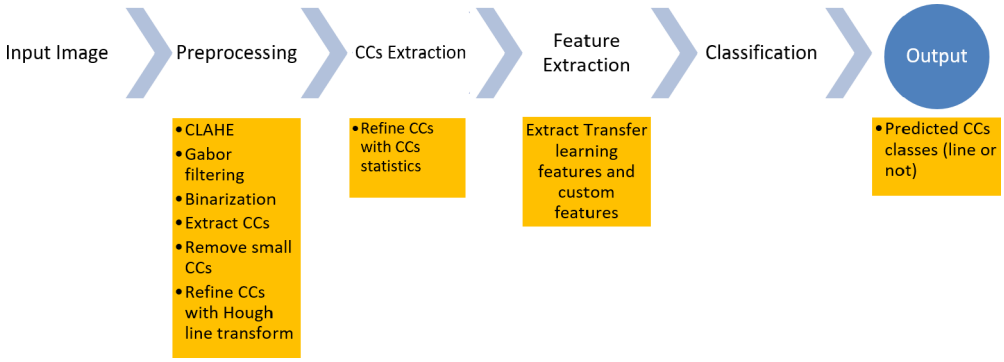

Fig. 10. A generic workflow of this approach.

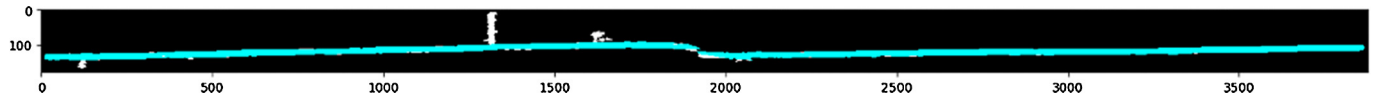

Fig. 11. Line-text separation, the regions shown in cyan colour are the detected line segments using Hough line transform. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

we use this feature to encode the shape of CCs to further enrich information about CC for the classifier.

Classification of CCs: In the training phase, SVM and Random forest (RF) classifiers were trained respectively on the two types of features, the custom features and the transfer learning features. Before feeding data into the classifiers, it is a common practice to apply data pre-processing techniques (e.g. imputation, scaling) [47]. Three imputation strategies are tested on custom feature data, namely, constant, mean and median of each feature across instances. As for scaling, standardization and normalization are tested. Standardization scales data belonging to the same feature so that the overall distribution has zero mean and unit variance. Normalization scales data belonging to the same instance to have a unit norm. Principal component analysis (PCA) [48] is also tested in our experiment. PCA is used to reduce the feature dimensions and keep the top N best features characterizing instances of the dif-

Table 3
Custom features used for training the model.

Feature	description
Category	Horizontal/vertical, indicate which filtering process produces current CC.
Length	Width of CC if CC in horizontal category, otherwise height.
Aspect ratio	$\frac{width_{cc}}{height_{cc}}$ if CC in horizontal category, otherwise $\frac{height_{cc}}{width_{cc}}$
Orientation	Orientation of CC which is extracted by taking the orientation of fitted ellipse on the current CC. If the width/height is too small, the orientation of fitted line on the current CC is used instead.
Extent	$\frac{area_{cc}}{width_{bd} \times height_{bd}}$ if both $width_{CC}$ and $height_{CC}$ are greater than 2 px, otherwise $\frac{area_{cc}}{width_{CC} \times height_{CC}}$. Where $area_{cc}$ is the area of CC, $width_{bd}$ and $height_{bd}$ are width and height of the bounding box of CC, respectively.
Solidity	$\frac{area_{cc}}{area_{hull}}$ if both $width_{CC}$ and $height_{CC}$ are greater than 2 px, otherwise *Solidity* equals to *Extent*. The $area_{hull}$ is the area of the convex hull of CC.
Eccentricity	$\sqrt{1-\left(\frac{axis_{minor}}{axis_{major}}\right)^2}$ if fitted ellipse of CC can be detected; otherwise $\sqrt{1-\left(\frac{height_{cc}}{width_{cc}}\right)^2}$ for CC in horizontal category, and $\sqrt{1-\left(\frac{width_{cc}}{height_{cc}}\right)^2}$ for CC in vertical category. A flag of -1 if they are all inapplicable. $axis_{minor}$ and $axis_{major}$ are the minor and major axis of fitted ellipse of CC, respectively.
Perimeters ratio	$\frac{perimeter_{hull}}{perimeter_{cc}}$, $perimeter_{hull}$ is the perimeter of convex hull of CC.
Number of intersecting lines of CC	This number indicates how many CCs, in the filtered image, the current CC region is intersecting in the other category. It is supposed to detect the horizontal and vertical lines' intersection.
Grey level co-occurrence matrix (GLCM) statistics	Contrast, correlation, energy, homogeneity of GLCM of the CC. As for the parameters to produce GLCM, the offset is set to 5, the orientation parameter is set to the orientation of CC and orthogonal angle of CC.
HOG	Parameters used to compute HOG features: (a) window size (60,60), (b) cell size (20, 20), (c) block size (40, 40), (d) block stride (20, 20), (e) number of bins 6

Table 4
Basic setup of experiment I.

Dataset	A small set of features extracted from several randomly selected images. • Number of samples in *the horizontal category* (59086): True: 3648, False: 55438. • Number of samples in *the vertical category* (22463): True: 1353, False: 21110.
Validation strategy	Cross validation with 3 folds stratified split of training data

ferent classes. In the prediction phase, the two best classification setups found in the training experiments using transfer features and custom features are used to classify CCs produced in the test images.

3. Results and discussions

In this section, Experiments I and II are conducted to find proper classification setup (e.g. imputation, scaling, classifier, etc.); Experiment III is conducted to evaluate the performance of the three best classification setup found in Experiments I and II; Experiment IV is conducted to train the final classifiers.

3.1. Experiments

3.1.1. Experiment I - custom features

In this experiment, the goal is to compare different setup performance when the classification is conducted on custom features. Different imputation, scaling strategies and different parameters tuning for RF and SVM classifiers were tested. The incorporation of PCA was also tested in this experiment. To optimize the fine-tuning of the classifiers' parameters in each alternative, exhaustive grid search [49] is adopted to explore all the different combinations of input parameters. Classification of CCs in horizontal category is separated with CCs in vertical category. Basic experiment setup is listed in Table 4.

Classification results of horizontal and vertical CCs using different alternatives are presented in (Appendix A). Alternatives in the results of both categories are ordered by their F1 score from the largest to smallest.

In horizontal CC classification result, it can be easily seen that most alternatives have relatively high accuracy even though they have low F1 score. The F1 score reflects the unbalance in the training/testing data problem (see Appendix A). It is possible that even when all true instances are misclassified, the accuracy can still be higher than 90% since the true instances are only about 6.17% out of the total samples set. That is the reason we use F1 score as the main driving metric to measure the performance throughout our experiments. Regarding the imputation strategy, the custom approach gives similar overall performance to the statistical-mean strategy. The statistical-median strategy gives much inferior performance; thus, we deprecate the use of it. In respect to the usefulness of PCA, among the top eight approaches, half of them have incorporated PCA. Regarding the scaling methods, the standardization approach is apparently better than the normalization one. In the last comparison between RF and SVM classifiers, among the top eight with the best scaling technique, SVM is shown to outperform RF classifier.

In the vertical CC classification result, most alternatives give similar performance in line to their performance on the horizontal CC classification. The statistical-mean and custom imputation strategies have similar performance. SVM shows better classification performance with PCA. Normalization is still the worse scaling method.

3.1.2. Experiment II - custom features and transfer learning features

In this experiment, we include the use of transfer learning features and explore different alternatives' performance as before. However, we made some changes based on the discovery in experiment I. A more balanced dataset is used. Since the mean and custom imputation strategy for custom features give similar result, only custom imputation strategy is retained here. Normalization scaling is not applied with custom features due to its poor performance. Additionally, only the best alternative using SVM is retained since the RF alternative did not work so well. As for the parameter's setup for SVM and RF classifiers, the ones performing the best overall in the previous experiment were retained. This gives us C=1, kernel=rbf for SVM and n_estimators=40, max_features=number_of_features for RF. Furthermore, two previous binary classification (horizontal/vertical CC classification) process are in-

Table 5
Basic setup of experiment II.

Dataset	A random sample set of features data extracted from all training image data. • Non-line CC instances: 3648 • True horizontal-line CC instances: 3648 • True vertical CC-line instances: 3648
Validation strategy	Split dataset into 70% for training and 30% for evaluation

Table 6
Basic setup of experiment III.

Dataset	A random sample set of features data extracted from all training image data. • Non-line CC instances: 60000 • True horizontal-line CC instances: 10000 • True vertical CC-line instances: 10000
Validation strategy	Cross validation with 2 folds stratified split of training data

Table 7
Classification result of selected alternatives.

	TL_null_null_null_SVM	ct_ct_PCA_stdr_SVM	ct_ct_null_stdr_RF
F1_micro avg[5]	0.936	0.918	0.914

tegrated into multi-label classification process in this experiment. The experimental basic setup is listed below in Table 5.

The results of this experiment are presented in (Appendix B). It is obvious that all alternatives' performance (i.e., F1 score) have benefited from the balanced dataset. The horizontal CC components are still the most difficult class to classify. When SVM is fed with custom features, the alternative "ct_ct_PCA_stdr_SVM"[3] produces the best result in this category. Whereas, when SVM is fed with transfer learning features, the alternative "TL_null_null_null_SVM"[4] performs the best in the transfer learning features category (see Appendix B). These two best alternatives in each category along with the best alternative using RF were further tested in experiment III with a larger data set.

3.1.3. Experiment III - performance comparison of selected alternatives

This experiment is a successive experiment of the previous one which aims at testing the performance consistency of the three selected alternatives when a larger data set is used. To randomize the selection and to avoid overfitting the models, cross validation is used. The experiment's basic setup is listed in Table 6.

The results of this experiment are presented in Table 7. Since we use cross validation with multi-label classification, *F1_micro avg* is used to measure the overall performance of classification of three classes. This time, alternative "TL_null_null_null_SVM" has the best performance as compared to "ct_ct_PCA_stdr_SVM". Since these two alternatives perform well consistently in all these experiments, we adopted them to produce the final evaluation results. The take-home message from these experiments is two-fold: "*transfer learning utilizes deep learning extracted features; their robustness and saliency improve by increasing the number of samples. This remark reinforces the conjecture that traditional machine learning algorithms* (*i.e.*, *SVM with custom features*) *may, in certain cases, outperform deep learning models if the data set is limited*.", and "*With a few targeted and descriptive handcrafted features, it is feasible in the studied task to arrive at an accuracy comparable to that of deep complex features, while allowing better scalability to big data processing*."

3.1.4. Experiment IV- constructing final classifiers

In this experiment, two classifiers of adopted two alternatives are trained to use in the future classification process. A larger dataset is used to train the classifiers. Theoretically, this will give a better result if the classifier is fed with more information about the different classes. Experiment basic setup is list Table 8. The evaluation results are presented in Table 9 where the adopted alternatives give plausible results.

3.2. Time complexity

As for the setup environment, the processes of CLAHE, Hessian filtering with mask operation, Gabor filtering, morphological opening and transfer learning feature extraction are performed on a MATLAB environment, other operations are performed on python environment. Graphic processing unit (GPU) was used in transfer learning feature extraction. Multi-thread technique is used to refine images. Since the underlying code of the different solutions is implemented in a different environment, it is not feasible to establish a direct and fair time complexity assessment. Nevertheless, it is safe, and expected as well, to say that the solutions with machine learning are 18 times slower than the traditional ones (*HessMask* & *GaborMask*).

These processes are carried out on a computer running on Windows 10 OS, with Intel i5-6200U CPU, GeForce 940M GPU and 12G RAM. Approximate time cost of each solution is presented in Fig. 12.

3.3. Human visual system scoring

In order to evaluate the effect of the different approaches on the visual output and to measure their performances, we made an evaluation group consisting of 6 members (excluding the authors) to grade the table layout extracted masks produced by the different algorithms from 70 images. Each image produced four images with the four adopted solutions, therefore, the graders where presented with 280 images in total in addition to the original RGB images. Grading specifications are listed in Table 10.

[3] Alternative using custom features, custom imputation, PCA, standardization scaling and SVM classifier.

[4] Alternative using transfer learning features and SVM classifier without PCA and scaling.

[5] Scikit-learn provides five approaches to determine the type of averaging, the'micro' method is used here which calculates metrics globally by counting the total true positives, false negatives and false positives.

Table 8
Basic setup of experiment IV.

Dataset	A random sample set of features data extracted from all training image data. • Non-line CC instances: 48639 • True horizontal-line CC instances: 32426 • True vertical CC-line instances: 16213
Validation strategy	Split dataset into 90% as training set and 10% as evaluation set

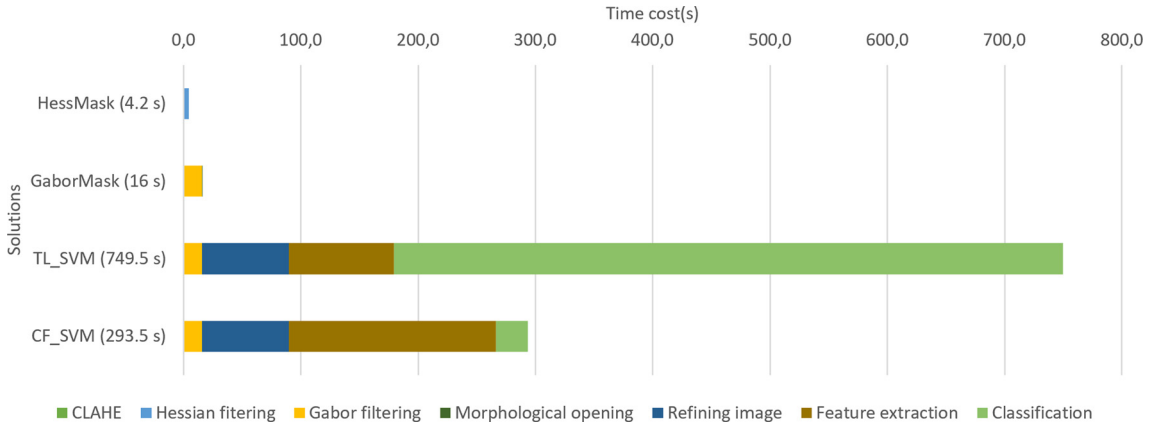

Fig. 12. Time complexity of the studied solutions.

Table 9
Classification results of the adopted alternatives. Suffix "F" indicates non-line class, Suffix "h" indicates horizontal line class and Suffix "v" indicates vertical line class.

	ct_ct_PCA_stdr_SVM	TL_null_null_null_SVM
F1_F	0.9815	0.9819
F1_h	0.9496	0.9561
F1_v	0.9917	0.9785
F1_micro avg	0.9755	0.9758
prec_F	0.9860	0.9795
prec_h	0.9378	0.9579
prec_v	0.9902	0.9893
recall_F	0.9771	0.9843
recall_h	0.9618	0.9544
recall_v	0.9932	0.9679

Table 10
Grading rule in the evaluation survey.

Grading scale	Integer number from 0 to 10. 10 denotes best performance (no misclassification). 0 denotes poor performance to recognize any true table line.
Grading criterion	rate each resulted image based on accuracy of line recognition, quality of recognized lines.

Table 11
Human visual system ranking of the different approaches.

	HessMask	GaborMask	TL_SVM	CF_SVM
Evaluator 1	6.26	7.21	**8.09**	7.70
Evaluator 2	4.47	6.17	**6.96**	6.83
Evaluator 3	4.86	**7.23**	7.13	6.70
Evaluator 4	5.39	6.30	**8.03**	7.74
Evaluator 5	5.04	6.00	**6.31**	6.30
Evaluator 6	3.87	4.69	**5.96**	5.84
Average	4.98	6.27	**7.08**	6.85

After the group's visual assessment of the resulted images, each assessor's grades for images of each solution are averaged. The final assessment results are listed below in Table 11.

These evaluation results show that most assessors gave above average ranking to all solutions. *TL_SVM*, using transfer learning features with SVM, is voted the best. *CF_SVM*, using custom features with SVM, comes the second. While, *HessMask*, using Hessian filter with mask operation, is selected the worst by the different raters. Examples of the images used for the subjective visual assessment are shown in (Appendix C).

3.4. Generalizability

The extension of the results of this study to other scenarios is feasible since the learned patterns are line segments that form the table geometric on a given scanned page. Therefore, the results reported in this study shall be insensitive to the language with which a given document is written. We believe extending the scope of machine learning to a more impactful environment in document retrieval and/or in optical character recognition, for example, is deemed important. In this paper, our work demonstrated non-intuitive improvements that would lend some contribution to current methods in the field.

4. Conclusion and future work

Based on the study we report in this work, we eventually concluded that the existing OCR systems, namely, Breuel's OCRopus method [11] is not able to effectively perform layout analysis tasks on scanned historical documents with table format. Traditional image processing techniques (e.g. 2D filtering, mask operations, morphological operations) are capable to a certain extent to extract the table layout of a document. However, the best achieved performance is by our CCs breakdown technique coupled with machine learning classification techniques whilst it is more time consuming than the former methods. Moreover, when it comes to transfer learning and deep convolutional network, the achieved high accuracy does prove their value and explain why they are so popular nowadays. To train the model for table layout analysis, we have used 2D filtering, extracted and manually annotated individual CCs, extracted their features and classified them, this obviously is a tedious task. Even so, blobs with fused text and line components are still not easy to tease apart in a few cases. In future work, we envision using the developed framework to automatically conduct

segmentation of the table layout to enhance our ongoing word-spotting research.

Declarations

Availability of data and material: The image data supplied by the company carry information pertaining to individuals which is confidential and therefore cannot be shared.

Funding: This work is part of the research project "*Scalable resource-efficient systems for big data analytics*" funded by the Knowledge Foundation (Grant: 20140032) in Sweden.

Authors' contributions: XL, carried out the analysis and wrote the manuscript. AC conceived the study and provided input and guidance for analysis, and JH supplied the data and provided the problem statement. All authors reviewed and revised the final paper.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The third co-author, Dr. Johan Hall, works at the Swedish company ArkivDigital AB which has supported this study.

Acknowledgements

The first author thanks ArkivDigital® AB for providing the data.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.bdr.2021.100195.

References

[1] A.M. Namboodiri, A.K. Jain, Document structure and layout analysis, in: Digital Document Processing, Springer, 2007, pp. 29–48.
[2] G. Nagy, S. Seth, M. Viswanathan, A prototype document image analysis system for technical journals, Computer 25 (7) (Jul. 1992) 10–22.
[3] H.S. Baird, S.E. Jones, S.J. Fortune, Image segmentation by shape-directed covers, in: 10th International Conference on Pattern Recognition, Proceedings, vol. 1, 1990, pp. 820–825.
[4] T. Pavlidis, J. Zhou, Page segmentation by white streams, Presented at the Proc. 1st Int. Conf. Document Analysis and Recognition, in: ICDAR, Int. Assoc. Pattern Recognition, 1991, pp. 945–953.
[5] T.M. Breuel, Two geometric algorithms for layout analysis, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 2423, 2002, pp. 188–199.
[6] L. O'Gorman, The document spectrum for page layout analysis, IEEE Trans. Pattern Anal. Mach. Intell. 15 (11) (1993) 1162–1173.
[7] K. Kise, A. Sato, M. Iwata, Segmentation of page images using the area Voronoi diagram, Comput. Vis. Image Underst. 70 (3) (Jun. 1998) 370–382.
[8] F.M. Wahl, K.Y. Wong, R.G. Casey, Block segmentation and text extraction in mixed text/image documents, Comput. Graph. Image Process. 20 (4) (1982) 375–390.
[9] G.E. Kopec, P.A. Chou, Document image decoding using Markov source models, IEEE Trans. Pattern Anal. Mach. Intell. 16 (6) (1994) 602–617.
[10] T. Pavlidis, J. Zhou, Page segmentation and classification, CVGIP, Graph. Models Image Process. 54 (6) (1992) 484–496.
[11] T.M. Breuel, The OCRopus open source OCR system, Presented at the Proceedings of SPIE - The International Society for Optical Engineering, vol. 6815, 2008.
[12] T.M. Breuel, Binary morphology and related operations on run-length representations, Presented at the VISAPP 2008 - 3rd International Conference on Computer Vision Theory and Applications, Proceedings, vol. 1, 2008, pp. 159–166.
[13] F. Shafait, D. Keysers, T.M. Breuel, Efficient implementation of local adaptive thresholding techniques using integral images, Presented at the Proceedings of SPIE - The International Society for Optical Engineering, vol. 6815, 2008.
[14] T.M. Breuel, Robust least square baseline finding using a branch and bound algorithm, Presented at the Proceedings of SPIE - The International Society for Optical Engineering, vol. 4670, 2002, pp. 20–27.
[15] F. Shafait, B. Van, D. Keysers, T.M. Breuel, Page frame detection for marginal noise removal from scanned documents, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), in: LNCS, vol. 4522, 2007, pp. 651–660.
[16] M. Thomas, High performance document layout analysis, Presented at the Proc. Symp. on Document Image Understanding Technology (SDIUT 03), 2003.
[17] J. Liang, J. Ha, R.M. Haralick, I.T. Phillips, Document layout structure extraction using bounding boxes of different entitles, in: Proceedings 3rd IEEE Workshop on Applications of Computer Vision, 1996. WACV'96, 1996, pp. 278–283.
[18] B. Gatos, D. Danatsas, I. Pratikakis, S.J. Perantonis, Automatic table detection in document images, in: Pattern Recognition and Data Mining, 2005, pp. 609–618.
[19] S.J. Perantonis, B. Gatos, N. Papamarkos, Block decomposition and segmentation for fast Hough transform evaluation, Pattern Recognit. 32 (5) (1999) 811–824.
[20] B.T. Ávila, R.D. Lins, A new algorithm for removing noisy borders from monochromatic documents, Presented at the Proceedings of the ACM Symposium on Applied Computing, vol. 2, 2004, pp. 1219–1225.
[21] Y. Tian, C. Gao, X. Huang, Table frame line detection in low quality document images based on Hough transform, in: The 2014 2nd International Conference on Systems and Informatics (ICSAI 2014), 2014, pp. 818–822.
[22] B.C.G. Lee, Line detection in binary document scans: a case study with the international tracing service archives, in: 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 2256–2261.
[23] Y. Zheng, C. Liu, X. Ding, S. Pan, Form frame line detection with directional single-connected chain, in: Proceedings of Sixth International Conference on Document Analysis and Recognition, 2001, pp. 699–703.
[24] A. Bansal, G. Harit, S.D. Roy, Table extraction from document images using fixed point model, in: Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing, New York, NY, USA, 2014, pp. 67:1–67:8.
[25] S.F. Rashid, A. Akmal, M. Adnan, A.A. Aslam, A. Dengel, Table recognition in heterogeneous documents using machine learning, in: Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, vol. 1, 2018, pp. 777–782.
[26] T. Kasar, P. Barlas, S. Adam, C. Chatelain, T. Paquet, Learning to detect tables in scanned document images using line information, in: 2013 12th International Conference on Document Analysis and Recognition, 2013, pp. 1185–1189.
[27] M.P. Viana, D.A.B. Oliveira, Fast CNN-based document layout analysis, in: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), 2017, pp. 1173–1180.
[28] P. Barlas, S. Adam, C. Chatelain, T. Paquet, A typed and handwritten text block segmentation system for heterogeneous and complex documents, in: 2014 11th IAPR International Workshop on Document Analysis Systems, 2014, pp. 46–50.
[29] S. Schreiber, S. Agne, I. Wolf, A. Dengel, S. Ahmed, DeepDeSRT: deep learning for detection and structure recognition of tables in document images, in: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 01, 2017, pp. 1162–1167.
[30] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, Presented at the Advances in Neural Information Processing Systems, vol. 2015-January, 2015, pp. 91–99.
[31] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (4) (2017) 640–651.
[32] MAURDOR campaign dataset [online]. Available: http://www.maurdor-campaign.org/. (Accessed 25 July 2018).
[33] A.F. Frangi, W.J. Niessen, K.L. Vincken, M.A. Viergever, Multiscale vessel enhancement filtering, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 1496, 1998, pp. 130–137.
[34] H.G. Feichtinger, T. Strohmer, Gabor Analysis and Algorithms: Theory and Applications, Springer Science & Business Media, 2012.
[35] N. Petkov, M. Wieling, Gabor filter for image processing and computer vision - examples, Gabor filter for image processing and computer vision [online]. Available: http://matlabserver.cs.rug.nl/edgedetectionweb/web/edgedetection_examples.html. (Accessed 19 June 2018).
[36] Contrast-limited adaptive histogram equalization (CLAHE) - MATLAB adapthisteq - MathWorks Nordic [online]. Available: https://se.mathworks.com/help/images/ref/adapthisteq.html. (Accessed 16 January 2019).
[37] K. Zuiderveld, Contrast limited adaptive histogram equalization, in: Graphics Gems IV, 1994, pp. 474–485.
[38] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst. Man Cybern. 9 (1) (Jan. 1979) 62–66.
[39] A. Van Opbroek, M.A. Ikram, M.W. Vernooij, M. De Bruijne, Transfer learning improves supervised image segmentation across imaging protocols, IEEE Trans. Med. Imaging 34 (5) (2015) 1018–1030.
[40] M. Mehdipour Ghazi, B. Yanikoglu, E. Aptoula, Plant identification using deep neural networks via optimization of transfer learning parameters, Neurocomputing 235 (2017) 228–235.
[41] Feature extraction using AlexNet - MATLAB & simulink - MathWorks Nordic [online]. Available: https://se.mathworks.com/help/deeplearning/examples/feature-extraction-using-alexnet.html?searchHighlight=alexnet&s_tid=doc_srchtitle. (Accessed 11 January 2019).

[42] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (6) (May 2017) 84–90.
[43] ImageNet large scale visual recognition competition 2010 (ILSVRC2010) [online]. Available: http://image-net.org/challenges/LSVRC/2010/index. (Accessed 11 January 2019).
[44] R.M. Haralick, I. Dinstein, K. Shanmugam, Textural features for image classification, IEEE Trans. Syst. Man Cybern. SMC-3 (6) (1973) 610–621.
[45] GLCM texture features — skimage v0.15.dev0 docs [online]. Available: http://scikit-image.org/docs/dev/auto_examples/features_detection/plot_glcm.html. (Accessed 11 January 2019).
[46] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, 2005, pp. 886–893.
[47] 4.3. Preprocessing data — scikit-learn 0.20.2 documentation [online]. Available: https://scikit-learn.org/stable/modules/preprocessing.html. (Accessed 12 January 2019).
[48] 2.5. Decomposing signals in components (matrix factorization problems) — scikit-learn 0.20.2 documentation [online]. Available: https://scikit-learn.org/stable/modules/decomposition.html#pca. (Accessed 13 January 2019).
[49] 3.2. Tuning the hyper-parameters of an estimator — scikit-learn 0.20.2 documentation [online]. Available: https://scikit-learn.org/stable/modules/grid_search.html#grid-search. (Accessed 13 January 2019).