An Unsupervised Learning Approach to Text Line Detection in Complex Illuminated Medieval Manuscripts*

Lizeth Gonzalez-Carabarin[1] and Lisandra S. Costiner[2] **

[1] Department of Electrical Engineering
Eindhoven University of Technology
Eindhoven, The Netherlands
l.gonzalez.carabarin@tue.com
[2] Merton College, University of Oxford
lia.costiner@merton.ox.ac.uk

Abstract. This paper outlines a simple and effective clustering and filtering approach for text line detection in challenging illuminated medieval manuscripts from Western Europe. As illuminated medieval manuscripts were copied and illustrated by hand and do not have regular formats, they pose particular difficulties for traditional methods of text line extraction, which have been designed for printed books. This paper introduces an unsupervised learning approach to text line detection in challenging manuscripts based on clustering, using a k-means algorithm with a combination of three salient features that are well-suited for image processing. These are: the gradient in the y-direction, mean values of rows, and grayscale values. The strength of the method lies in its reduced number of features, its computational lightness, low-memory use, transparency at every step of the process, and versatility. It can also be used on a single image, being per-page based, unlike supervised learning approaches which require large training datasets. This stands as an alternative to computationally-heavy algorithms such as neural networks, which have been increasingly used in recent years to solve such tasks.

Keywords: Text Line Extraction, Document Layout Analysis, Medieval Manuscripts, Unsupervised Learning, Artificial Intelligence, Digital Humanities.

1 Introduction

Digitization initiatives undertaken by libraries, museums and collections around the globe are rapidly increasing the number of manuscript images online. Given the large volume of such data, it is important to devise new ways to automatically process and extract relevant information from these images thus saving valuable human time. Digitized documents pose a number of challenges for the extraction of relevant information, the key ones being the location of areas of text and illustration and the detection of text lines.

* Supported by Merton College, Oxford.
** The two authors contributed equally to this article.

Medieval manuscripts are especially challenging for automatic text line detection. Each surviving book was hand produced so its page layout and illustrations vary widely. Furthermore, unlike in printed books, text and decoration in medieval manuscripts do not typically conform to uniform rectangular registers, which is the assumed layout for image segmentation [1], [2]. Instead, text lines in manuscripts can be irregular, and can be broken up by unframed images. Such documents, therefore, pose particular difficulties for traditional methods of text line detection designed for printed text, requiring instead the development of customized algorithms.

Text line detection is an essential component of automatic document analysis, being the basis of text extraction and ultimately optical character recognition (OCR). It is typically categorized under image segmentation algorithms [3]. A number of competitions have been particularly dedicated to the challenges of extracting text lines from challenging documents [4], [5]. Older approaches employed in document segmentation and text line extraction were adapted to specific types of records [8] [9], [10] , so there is a need for a global or generic approach that will be able to adapt to different types of documents. Recent developments have tended to focus on the use of neural networks [2], [5], [6], [7], [12], [13], [14], which have proved successful, as have hybrid approaches [1]. Although effective, neural networks (NNs) require manually-annotated data for training, expending large amounts of human time. They are also computationally heavy, and are black boxes, meaning that their inner workings are not understood. New approaches with increased versatility, stability, generality, ability to perform multi-scale analysis, and to handle color remain a desiderata [3]. There is a need, therefore, for a generic tool that is flexible in dealing with a range of documents, is low on processing power, and white-box, allowing every step to be queried and understood.

This paper proposes such a technique for the automatic identification and extraction of lines of text from Western medieval manuscripts with complex layouts that include text and image. It introduces an unsupervised learning approach to text line detection based on clustering, using a k-means algorithm [17] with a combination of three salient features that are well suited for image processing. Although k-means has been applied for document segmentation previously, the number of features used in these approaches was large, increasing the computational cost [1]. The current methodology differentiates itself by relying on only three features, namely the gradient in the y-direction, mean values of rows and gray-scale values. It is also able to detect useful clusters based on statistical information about cluster centroid positions. Moreover, as compared to previous approaches to line extraction and segmentation which work uniquely on pictures of single pages, this approach can be applied to images of open books which show two facing pages, a more difficult arrangement for image processing.

2 Dataset

Although for text line detection in medieval documents with challenging layouts, a number of annotated datasets have been created [11], [15], no such datasets exist for illuminated medieval manuscripts. For the current study, a new challenging dataset was specifically created to contain images that might cause difficulties in the automatic detection of text lines. This include books with a variety of layouts and decorations,

different types of texts (devotional and medical), written in various scripts, and produced in a number of geographical regions in different time periods. The layout of these manuscripts include single or double columns of text. The text is often interrupted by images, such as historiated initials, borders, marginal decoration, and unframed drawings. The selection of images was chosen from digitised manuscripts downloaded from the freely available digital collection of the Bodleian Library in Oxford [16]. The 120 images used in this study, derive from the following manuscripts in Oxford's Bodleian Library: MS Canon. Misc 476 (a fourteenth-century Italian Life of the Virgin and of Christ), MS Add. A. 185 (a fifteenth-century French Book of Hours), MS Ashmole 1462 (a twelfth-century English manuscript containing medical and herbal texts), MS Auct. 2.2 (a fourteenth-century English choir psalter), MS Buchanan e 7 (a fifteenth-century Italian Book of Hours).

3 Clustering Based on k-means

The algorithm used in this approach is a k-means clustering algorithm [17]. The k-means algorithm partitions a set P of N observations, $P = \{x_1, x_2, \cdots, x_N\}$, into k clusters ($C_1, C_2, \cdots, C_k$), (where $C_1 \cup C_2 \cup \cdots \cup C_k = P$), such as to minimise:

$$\min_{C_1, C_2, \cdots, C_k} \sum_{i=1}^{k} \sum_{x \in C_i} \left\| x - \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j \right\|^2 \tag{1}$$

In our case, $x_1, x_2, \cdots, x_N$ are vectors in R^p, where p is the number of features, and the distance between points is the Euclidean distance, $\|x_i - x_j\|^2$, and the centroid m_i of the cluster C_i is the mean:

$$m_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j \tag{2}$$

In order to find the clusters and centroids in equations (1) and (2), the k-means algorithm starts by selecting k centroids m_i, which are randomly placed among the data points. Then, the method iterates the following two steps: (a) clusters each data point to their nearest mean/centroid and (b) calculates new means/centroids as the average of each cluster (equation 2).

Other studies have proposed similar approaches for layout segmentation based on k-means, however they use a large number of features (tens of them) [1]. This paper proposes the use of three features after a pre-processing stage, which has the advantage of increasing efficiency in terms of processing time and resources.

4 Pre-Processing Stage

The input image is in RGB format, and it is represented as a multidimensional matrix of m x n x d, where m is the number of rows, n is the number of columns and d is the

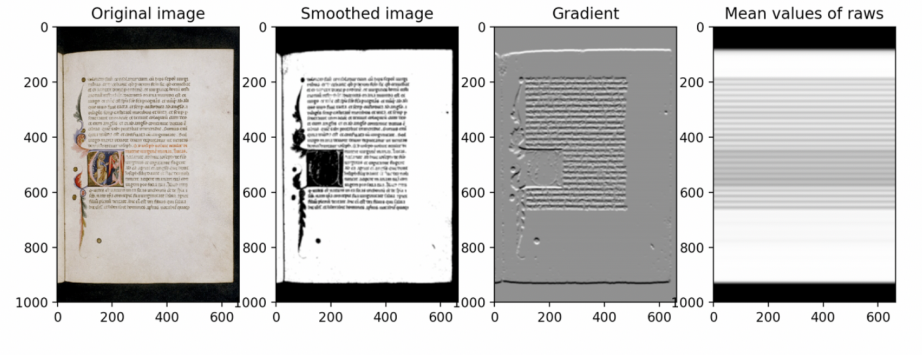

Fig. 1: (a) Original image, (b) smoothed image, (c) gradient, and (d) mean values of rows.

depth (number of channels). The values range from 0 to 255. The first step is to obtain the gray format based on:

$$I_G = 0.2989 * d_1 + 0.5870 * d_2 + 0.1140 * d_3 \tag{3}$$

Where $d = [d_1, d_2, d_3]$, representing R, G and B values respectively. Here, I_G is a matrix of shape m x n. After converting the image into grayscale, a uniform filter is applied using a kernel size of 13 (selected through trials) in order to obtain a smoother format $I_{Gsmooth}$.

5 Feature Extraction

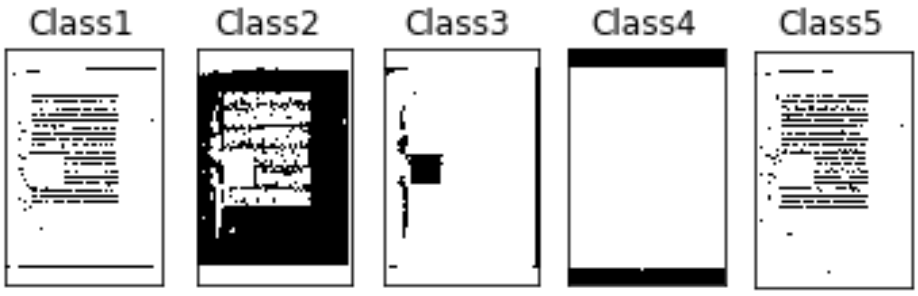

Fig. 2: Five classes obtained using k-means for one manuscript image. Each class represents one cluster. The points in the clusters are represented in black.

After pre-processing, three features are proposed for clustering. These are the gradient in the vertical y-direction, the mean values of rows, and the gray-scale values of $I_{Gsmooth}$. The reason behind the use of gradient values of the smooth version is that it provides the upper and lower edges of the text lines (Fig. 1c). The second feature, the mean value of rows, also provides information regarding the text lines (Fig. 1d). Zero values in $I_{Gsmooth}$ correspond to black pixels, while 1 values correspond to white pixels. Therefore, minima of mean row values (close to 0) are likely to correspond to black text lines, whereas maxima (close to 1) are likely to correspond to the white space between text lines or the spaces at the top and bottom of the page. Finally, the third feature, the grey values of $I_{Gsmooth}$ may provide information regarding empty spaces and image location (Fig. 1b). For an N-dimensional array, the gradient is based on central differences in the interior and first differences in the boundaries for the matrix. The central differences are given by:

$$\delta_h[f](x) = f(x + h/2) - f(x - h/2) \tag{4}$$

And for the boundaries:

$$\delta_h[f](x) = f(x + h) - f(x) \tag{5}$$

Based on previous equations, the gradient was calculated, and a matrix I_Δ of size m x n was obtained. Additionally, the mean value for one row is calculated according to:

$$\mu_{smooth,i} = \frac{\sum_{j=1}^{n} x_{i,j}}{n} \tag{6}$$

The I_μ m x n matrix of row means associates to each pixel within a row i, the mean value $\mu_{smooth,i}$. After the computation of the three matrices, $I_{Gsmooth}$, I_Δ, and I_μ, which contain the values of features, standardisation was performed on the concatenated matrices:

$$I_z = \frac{I_X - \mu_I}{\sigma_I} \tag{7}$$

where μ_I and σ_I are the mean and the standard deviation of I_X, which contains the concatenation of $\{I_{Gsmooth}$,I_Δ,$I_\mu\}$. To better understand the full pre-processing stage, Algorithm 1 shows the pseudocode.

6 Classification Based on k-means

Once the three features are computed and standardised, a k-means algorithm is used to compute five clusters. An example of produced clusters from one manuscript image is shown in Fig. 3. Each class represents one cluster. The points in the clusters are represented in black. As can be observed, Class 1 and Class 5 provide useful information regarding the location of text lines.

These five clusters were computed across the entire corpus of 120 images. Fig. 3 plots these 120 x 5 clusters. The observations made for Fig. 3 are visible in all 120 images. The centroid values that are in the extremes of 2 and -2 correspond to gradients at

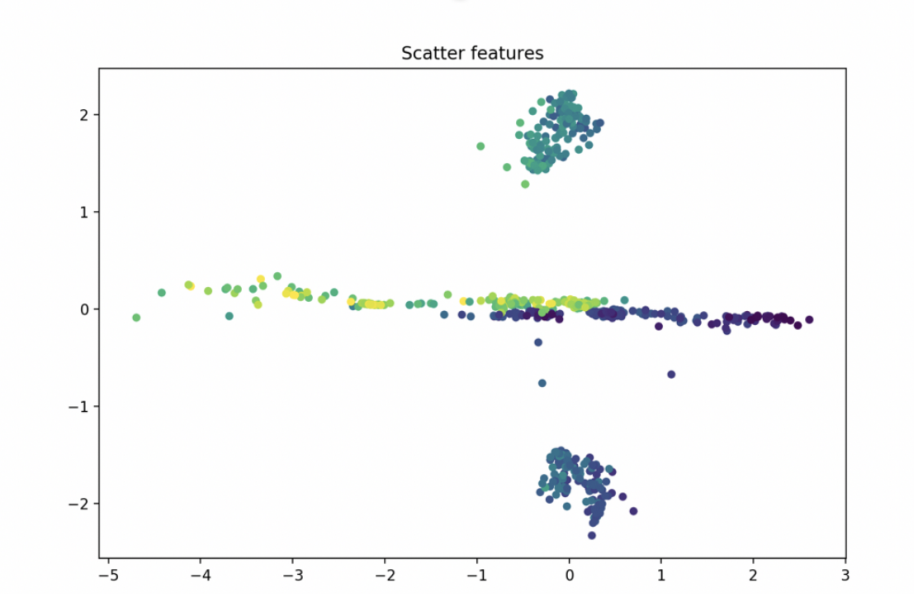

Fig. 3: Centroids of each of the 120 images in our dataset. Each image has 5 centroids. The top and bottom clusters show that the algorithm has clearly detected the upper and lower boundaries of text lines.

the upper and lower boundaries of text lines. In Fig. 3, the well-separated top and bottom clouds of centroids, with values of 2 and -2 respectively, show that the algorithm has clearly detected the upper and lower boundaries of text lines. Further post-processing can be performed to extract unique horizontal lines of text. Using the mean value of rows, text lines can be differentiated from the white horizontal space separating them (black color having a value of 0, while white having a value of 1).

Other results of k-means can be used to improve the extraction of text lines. In this process, it is important to note that digitised manuscripts come in a range of formats. Some capture a single page, others capture facing pages, while certain others feature both the page and some space that falls outside of it. The regions outside of the page, which typically appear black on a photograph or scan, need to be removed (see Fig. 1a). These areas are detected by the algorithm as seen in Fig. 2, class 4. The algorithm also detects the white pixels of the page, corresponding to unwritten and unillustrated areas (represented in black in Fig. 2, class 2). These observations can be used to further eliminate upper and lower margins, and improve the accuracy of the text line extraction.

The pseudocode with the procedure of clustering, post-processing and line detection is shown in Algorithms 1 and 2. After obtaining the class which contains the lines (***class_lines***), a post-processing stage is applied in order to eliminate borders (***class_borders***). This is followed by an enhancing stage, in which a uniform filter and

gradient is obtained. This enhances horizontal lines of ***class_lines***. Finally, a Hough Lines algorithm is applied [20].

Algorithm 1 Pseudo-code for pre-processing

```
procedure PRE-PROCESSING
    img_gray = RGB_TO_GRAY(original_image)
    img_bw = BW_THRESHOLD(img_gray > BW_THRESHOLD)
    img_bw_smooth_1 = UNIFORM_FILTER(img_bw)
    mean_rows_gray_v = MEAN(img_bw_smooth_1, rows)
    img_bw_smooth_2 = img_bw_smooth_1
    loop:
    if i < smoothness_level then return true
        img_bw_smooth_2 = UNIFORM_FILTER(img_bw_smooth_2)
        img_bw_smooth_2 =GRAD_Y(img_bw_smooth_2)
        i ← i + 1
    img_grad = img_bw_smooth_2
    X_data={ img_bw_smooth_1 img_grad mean_rows_gray_v }
```

Algorithm 2 Algorithm for text line extraction

```
procedure TEXT EXTRACTION
    clusters = K_MEANS(X_data)
    class_lines [cluster == cluster_lines] = 0
    loop:
    if cluster_borders then return true
        class_borders[cluster == cluster_borders] = 0
        class_borders = UNIFORM_FILTER(class_borders)
        class_lines[class_borders == 1] = 0
    class_lines_smooth = UNIFORM_FILTER(class_lines)
    loop:
    if i < smoothness_level_2 then return true
        class_lines_smooth = UNIFORM_FILTER(class_lines_smooth)
        class_lines_smooth = GRAD_Y(class_lines_smooth)
        i ← i + 1.
    lines = HOUGH_LINES(class_lines_smooth)
```

7 Results

Results are shown in Fig. 4 for different manuscripts with various layouts and types of illumination. The green marks represent the upper and lower boundaries of text lines found by the algorithm. The algorithm appears to accurately identify lines of text in a number of challenging cases. These include images of single and double columns of

(a) Sample 1

(b) Sample 2

(c) Sample 3

Fig. 4: Text and image extraction for different samples of manuscripts. This shows that the algorithm can handle difficult cases such as: single and double columns of text, text lines which are broken by illustrations inserted within the text column, images including facing pages of the manuscript, and images including extraneous material, such as the space outside of a book.

text, text lines which are broken by unframed illustrations which are inserted within the text column, images of facing pages, and images that include extraneous material, such as the space outside of a book, which appears black in photographs.

The accuracy of the algorithm was evaluated by comparing its performance against the same dataset which had been manually labelled in red. The following measures were used to evaluate the performance of the current model: TP (True Positive) are correctly detected lines; FP (False Positive) are regions in the page erroneously marked as lines; FN (False Negative) corresponds to those lines which were not marked. This was done across all of our image dataset and the final accuracy was computed by averaging the results of each image. Evaluation results were obtained based on pixel accuracy. For the proposed database, a final accuracy of 87.2% was obtained. The accuracy was calculated according to the percentage of pixels categorized as 'text lines' by our algorithm that matched the manually labeled lines. Although this accuracy is below that of state-of-the-art algorithms [19], the method is useful because of its simplicity, and because it is less computationally heavy, being run without the assistance of GPU and external servers.

8 Discussion

We have presented an efficient algorithm which relies on unsupervised learning to successfully detect lines of text. The dataset was curated to include challenging cases, and indeed, some proved to be too difficult for the algorithm. The fifteenth-century French Book of Hours, Oxford, Bodleian Library, MS. Add. A.185, was one such case (Fig. 5).

For this particular manuscript, which includes a large border with foliate patterns and an illustration, the algorithm classified some areas of the decoration as text lines. These consisted mainly of vines which were sketched in pen. Although increasing the number of clusters did not improve the performance of our model, adding more features may. Another avenue for improvement that could be pursued is that of hybrid techniques (supervised and unsupervised learning). These would aim for model shrinkage (e.g. leveraging pruning and quantization for Deep Learning approaches) to further increase metrics while providing efficient inference models.

Fig. 5: Example of a challenging manuscript (Oxford, Bodleian Library, MS. Add. A. 185, fol. 106v). Red circles bring attention to areas of erroneous line detection.

9 Conclusion

This paper outlines a simple and effective approach for line detection in challenging illuminated medieval manuscripts. Traditional approaches of line detection assume that text regions are enclosed in rectangular registers, which is not true for many medieval books, especially those that contain illuminations. This approach is suited to such complexity. Although k-means and filtering have been previously used for similar page segmentation tasks, the uniqueness of this approach is its reliance on only three features. The strength of the method further lies in its transparency at every step of the process, low-memory use, potential to produce refined results, and versatility. This stands as an alternative to algorithms such as neural networks, which have been increasingly used

in recent years to solve such tasks – algorithms which are black-boxes, do not allow for querying of their decision-making process and are computationally intensive. Also, in contrast to supervised learning approaches, such as neural networks, that require a lot of human time for manual annotation of training data, this algorithm is unsupervised and demands no such investment. Furthermore, this algorithm works on single case, whereas others require large datasets on which to train. This approach, therefore, provides not only a solution for line detection in challenging images of documents with mixed textual and visual content, but more importantly leads towards algorithms with improved robustness, stability and versatility. Such results are important for scholars in the humanities, as a pre-processing step for textual extraction and ultimately optical character recognition (OCR).

References

1. Yang, Y., Pintus, R., Gobbetti, E., and Rushmeier H.: Automatic Single Page-Based Algorithms for Medieval Manuscript Analysis. Journal on Computing and Cultural Heritage 10(2). DOI:https://doi.org/10.1145/2996469 (2017).
2. Ares Oliveira, S., Seguin, B. and Kaplan, F.: dhSegment: A Generic Deep-Learning Approach for Document Segmentation. 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR) (2018).
3. Eskenazi, S., Gomez-Krämer, P. and Ogier, J.: A Comprehensive Survey of Mostly Textual Document Segmentation Algorithms since 2008. Pattern Recognition, 64, pp. 1-14 (2017).
4. Stamatopoulos, N., Gatos, B., Louloudis, G., Pal, U. and Alaei, A.: ICDAR 2013 Handwriting Segmentation Contest. Proceedings of the 2013 12th International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp. 1402–1406 (2013).
5. Grüning, T., Labahn, R., Diem, M., Kleber, F. and Fiel, S.: Read-Bad: a New Dataset and Evaluation Scheme for Baseline Detection in Archival Documents. 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pp. 351–356 (2018).
6. Fink, M., Layer, T., Mackenbrock, G. and Sprinzl M.: Baseline Detection in Historical Documents Using Convolutional U-Nets, 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), Vienna, Austria, 2018, pp. 37-42, doi: 10.1109/DAS.2018.34 (2018).
7. Alberti, M., Vögtlin, L., Pondenkandath, V., Seuret, M., Ingold, R. and Liwicki, M.: Labeling, Cutting, Grouping: an Efficient Text Line Segmentation Method for Medieval Manuscripts. arXiv:1906.11894 (2019).
8. Shafait, F., Keysers, D. and Breuel, T.: Performance Evaluation and Benchmarking of Six-Page Segmentation Algorithms. Pattern Analysis and Machine Intelligence, IEEE Transactions, 30(6), pp. 941–954 (2008).
9. Likforman-Sulem, L., Zahour, A. and Taconet, B: Text line Segmentation of Historical Documents: a Survey. IJDAR 9, pp. 123–138. https://doi.org/10.1007/s10032-006-0023-z (2007)
10. Pintus, R., Yang, Y., Gobbei, E., et al: A TaLISMAN: Automatic Text and Line Segmentation of Historical MANuscripts'. EUROGRAPHICS Workshop on Graphics and Cultural Heritage, (2014).
11. Diem, M., Kleber, F., Fiel, S., Grüning, T., and Gatos, B.: ScriptNet: ICDAR 2017 Competition on Baseline Detection in Archival Documents (cBAD) [Data set]. Zenodo. DOI: http://doi.org/10.5281/zenodo.257972 (2017).
12. Renton, G., Chatelain, C., Adam, S., Kermorvant, C. and Paquet, T.: Handwritten Text Line Segmentation Using Fully Convolutional Network. 2017 14th IAPR International Confer-

ence on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 2017, pp. 5-9, doi: 10.1109/ICDAR.2017.321 (2017).
13. Chen, K., Seuret, M., Liwicki M., Hennebert J. and Ingold R.: Page Segmentation of Historical Document Images with Convolutional Autoencoders. Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 1011–1015 (2015).
14. Barakat, B., Droby, A., Kassis, M. and El-Sana, J.: Text Line Segmentation for Challenging Handwritten Document Images using Fully Convolutional Network. 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 374-379 (2018).
15. Simistira, F., Seuret, M., Eichenberger, N., Garz, A., Liwicki, M., Ingold, R.: Diva-hisdb: a precisely annotated large dataset of challenging medieval manuscripts. Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference, pp. 471–476 (2016).
16. Digital Bodleian. https://digital.bodleian.ox.ac.uk/, last accessed 2021/3/20.
17. Jin, X., Han, J.: K-Means Clustering on Encyclopedia of Machine Learning. Springer US, pp. 563-564 (2010).
18. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. Advances in Neural Information Processing Systems, 27 (2014).
19. Barakat, B. K., Droby, A., Alasam, R., Madi, B., Rabaev I., Shammes, R. and El-Sana, J.: Unsupervised Deep Learning for Text Line Segmentation. arXiv:2003.08632 (2020).
20. Likforman-Sulem, L., Hanimyan, A. and Faure: A Hough Based Algorithm for Extracting Text Lines in Handwritten Documents. International Conference on Document Analysis and Recognition (ICDAR), pp. 774–777 (1995).