Historical Document Digitization through Layout Analysis and Deep Content Classification

Andrea Corbelli, Lorenzo Baraldi, Costantino Grana and Rita Cucchiara
Dipartimento di Ingegneria "Enzo Ferrari"
Università degli Studi di Modena e Reggio Emilia
41125 Modena, Italy
Email: {name.surname}@unimore.it

***Abstract*—Document layout segmentation and recognition is an important task in the creation of digitized documents collections, especially when dealing with historical documents. This paper presents an hybrid approach to layout segmentation as well as a strategy to classify document regions, which is applied to the process of digitization of an historical encyclopedia. Our layout analysis method merges a classic top-down approach and a bottom-up classification process based on local geometrical features, while regions are classified by means of features extracted from a Convolutional Neural Network merged in a Random Forest classifier. Experiments are conducted on the first volume of the "Enciclopedia Treccani", a large dataset containing 999 manually annotated pages from the historical Italian encyclopedia.**

I. INTRODUCTION

Document digitization is a very important aspect of document preservation. In particular, regarding historical documents, the digitization process is useful to improve not only the durability of the documents, but also their accessibility. Often times historical documents have been transcribed, but a plain text version lacks of all the metadata available in the original version, regarding images, layout arrangements and linking to the original page. An extensive digitization process involves, along with the creation of a digital copy, also the extraction of the information contained in a document. Simple optical character recognition (OCR) is only one of the necessary steps that lead to the creation of a fully digital version of a document, which include the recognition of all the different elements that might appear in a document, such as images, tables and formulas. All these should be to be segmented and classified to obtain a truly digital version of the document.

Despite its long history, layout analysis is still a difficult task due the great number of possible layout configurations and content arrangements. In this paper we present a document analysis pipeline which includes a layout segmentation algorithm and a content classification method to classify the actual content of the segmented regions. Moreover, in case a manual transcription is available, we propose a technique to map OCR-ed text to transcription, to enrich the digitized version with additional metadata.

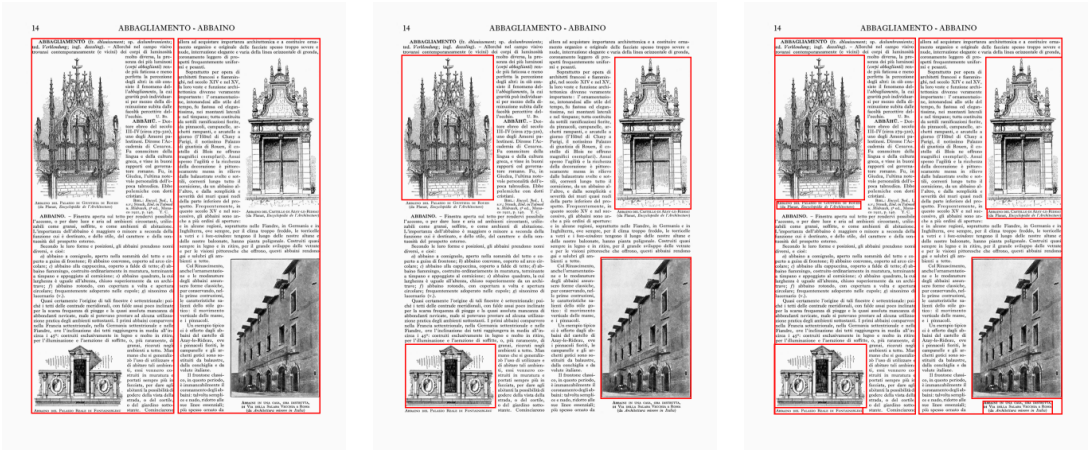
We test our pipeline on the Italian historical encyclopedia, the "Enciclopedia Treccani", published between 1929 and 1936. The encyclopedia consists of 35 volumes, for a total of around 60000 articles. In our experiments we focus on the first volume which contains all the different layout arrangements found in the encyclopedia and which has been manually annotated by us. A possible outcome of this work is indeed the digitization of the entire encyclopedia.

The rest of this paper is structured as follows: Section 2 gives a brief discussion of the state of the art in layout analysis and content classification, Section 3 explains the main components of our pipeline, and Section 4 reports the performance evaluation and a comparison against the state of the art.

II. RELATED WORK

Layout analysis has been an active area of research since the seventies. There are two main approaches to this task, namely *bottom up* and *top down*. Top-down methods, such as XY cuts [1], [2] or methods that exploit white streams [3] or projection profiles are usually fast but tend to fail when dealing with complex layouts. Bottom-up methods are instead more flexible and process the image page from the pixel level and subsequently aggregate into higher level regions but with an higher computational complexity.

These approaches are usually based on mathematical morphology, Connected Components (CCs), Voronoi diagrams [4] or run-length smearing [5]. Many other methods exist which do not fit exactly into either of these categories: the so called mixed or hybrid approaches try to combine the high speed of the top-down approaches with the robustness of the bottom-up ones. Chen *et al.* [6] propose a method based on whitespace rectangles extraction and grouping: initially the foreground CCs are extracted and linked into chains according to their horizontal adjacency relationship; whitespace rectangles are then extracted from the gap between horizontally adjacent CCs; CCs and whitespaces are progressively grouped and filtered to form text lines and afterward text blocks. Lazzara *et al.* [7] provide a chain of steps to first recognize text regions and successively non-text elements. Foreground CCs are extracted, then delimiters (such as lines, whitespaces and tab-stop) are detected with object alignment and morphological algorithms. Since text components are usually well aligned, have a uniform size and are close to each other, the authors propose to regroup CCs by looking for their neighbors. Filters can also be applied on a group of CCs to validate the link between two CCs. Another algorithm based on whitespace analysis has been



(a) XY-Cut (b) Detected illustrations (c) Final result

Fig. 1. The page layout segmentation pipeline. First the Recursive XY-Cut algorithm is applied to detect candidate regions inside the page; then, illustrations are detected using local autocorrelation features. A second application of the XY-Cut algorithm gives the final segmentation.

proposed by Baird *et al.* [8]: the algorithm uses the white space in the page as a layout delimiter and tries to find the biggest background empty rectangles to extract connected regions.

On a different note, document content classification is also a problem that has been faced by many researchers. Regarding the applied strategies, existing algorithms can be subdivided in two main categories: rule-based and statistical-based. Some papers present algorithms to classify whole pages into different categories, such as "title page", "table of contents", etc. [9], while a different approach is to classify homogeneous regions in a document into different classes, such as "text", "image", "table", etc. Is interesting to note that many papers face this problem trying to distinguish only one class, for example Zanibbi *et al.* [10] focuses on mathematical expressions, Hu *et al.* [11] on tables, Chen *et al.* [12] and Pham [13] on logo detection and Futrelle *et al.* [14] on diagrams. These approaches solve only part of the classification problem.

Regarding multi-class algorithms, many of them exploit rules built specifically for certain document classes, for example Krishnamoorthy *et al.* [15] and Lee *et al.* [16] developed algorithms to identify entities in scientific journals and papers, Mitchell *et al.* [17] identifies entities in newspaper pages and Jain and Yu [18] identifies various types of entities in different types of documents. These approaches often rely on hand-built heuristics, strictly tied to specific document types. Other approaches use statistical features and image features to classify document regions in different categories. The algorithm proposed by Sivaramakrishnan *et al.* [19] distinguishes between nine different classes extracting features for each region using run length mean and variance, number of black pixels and aspect ratio of the region. Fan and Wang [20] use density features and connectivity histograms to classify regions into text, images and graphics. Li *et al.* [21] proposed an algorithm that models images using two-dimensional HMMs and Wang *et al.* [22] proposed an algorithm based on the representation of a region by means of a 25-dimensional vector and on a classification tree.

III. PROPOSED APPROACH

A. Layout Segmentation

Layout segmentation aims at segmenting an input page into coherent regions, like text, tables and images, and is therefore a necessary step for classifying the contents of a document. In the following, we propose a top-down layout analysis approach, which builds upon the classic XY-Cut algorithm [1] and is able to deal with more complex layouts.

The XY-Cut algorithm is applied as the first step of our layout segmentation pipeline. This classic top-down algorithm recursively splits a region of interest into two, based on the spatial arrangement of white pixels. The algorithm exploits the pixels' vertical and horizontal projections in order to find low density regions in the projections' histograms. These low density points correspond to white spaces separating two different layout elements and therefore the region is split accordingly.

The XY-Cut algorithm is suitable to find rectangular regions surrounded by white space, but in real layouts it is not uncommon to find images within text areas (see the image in Fig. 1 for an example). The involved text area, instead of being a plain rectangle, is identifiable as a rectilinear polygon which can't be recognized by the XY-Cut algorithm. To address this problem an additional analysis step is performed in order to detect images and illustrations in the page.

The image detection problem is approached exploiting local autocorrelation features and the method we propose is inspired by the algorithm proposed in [23]. The assumption is that text areas are clearly distinguishable thanks to the text lines pattern which is absent in images, thus the autocorrelation matrix of a region is used as an effective descriptor for this task. The image is subdivided into square blocks of size $n \times n$, and for each block the autocorrelation matrix $C(k,l)$, with $k,l \in [-n/2, n/2]$, is computed. Then, the autocorrelation

TABLE I

STRUCTURE OF THE CONTENT CLASSIFIER CNN. THE NETWORK TAKES A INPUT A GRAY-SCALE $n \times n$ PATCH, AND CONSISTS OF A SEQUENCE OF CONVOLUTIONAL (CONV), MAXPOOLING (MP) AND FULLY-CONNECTED (FC) LAYERS. NEURONS FROM ALL LAYERS USE RELU ACTIVATIONS, EXCEPT FROM THE FC2 LAYER WHICH USES A SOFTMAX ACTIVATION.

Layer	Kernel size	Stride size	N. filters	Input shape	Output shape
conv1	11×11	1×1	32	$(1, n, n)$	$(32, n, n)$
mp1	2×2	2×2	–	$(32, n, n)$	$(32, n/2, n/2)$
conv2	5×5	1×1	32	$(32, n/2, n/2)$	$(32, n/2, n/2)$
mp2	2×2	2×2	–	$(32, n/2, n/2)$	$(32, n/4, n/4)$
conv3	3×3	1×1	32	$(32, n/4, n/4)$	$(32, n/4, n/4)$
mp3	2×2	2×2	–	$(32, n/4, n/4)$	$(32, n/8, n/8)$

Layer	Input size	Output size
fc1	$(32 \cdot n^2/64)$	1024
dropout	1024	512
fc2	512	7

matrix is encoded into a directional histogram $w(\cdot)$, in which each bin contains the sum of the pixels along that direction. Formally,

$$w(\theta) = \sum_{r \in (0, n/2]} C(r \cos\theta, r \sin\theta) \tag{1}$$

We compute the directional histogram in the range $\theta \in [0, 180]$, and quantize θ with a step of 1, and r with a step of 1 pixel. The histogram is then concatenated with the vertical and horizontal projections of the autocorrelation matrix, to enhance the repeating pattern of the text lines. The resulting descriptor is fed to a two-class SVM classifier with RBF kernel, trained to distinguish between blocks of text and blocks of illustrations.

Once the original image is split by the first XY-Cut application, each region is subdivided into $n \times n$ blocks which are then classified into text and images. Illustration boundaries are then detected by finding the connected components created by the illustration blocks.

The last step of this process involves the removal of the detected illustrations in order to use the XY-Cut algorithm again on the same regions. The final result of the segmentation process is the union of the regions found by the XY-Cut applications and the illustrations regions. Fig. 1 shows an example of a page through the various phases of the segmentation process.

B. Content Classification

Once the input document has been divided into a set of coherent regions, these can be classified according to the class they belong to. Notice that in a structured document blocks of different classes are separated by whitespaces, and it is reasonable to assume that each region belongs to a unique class. In the case of the "Enciclopedia Treccani", there are seven different classes: text, tables with border, borderless tables, images, graphics, scores and mathematical formulas.

Content classification is carried out by exploiting a combination of deeply learned features and classical local features encoding techniques. Given an input region, a Convolutional Neural Network (CNN) is used to produce local features from squared $n \times n$ blocks, which are then encoded according to their mean and variance. The final descriptor, which is global with respect to a region, is then classified using a Random Forest classifier.

A CNN is a neural model composed by a sequence of Convolutional, Spatial Pooling and Fully connected layers. Each Convolutional layer takes as input a $c \times w \times h$ tensor, and applies a set of convolutional filters to produce an output $c' \times w \times h$ tensor, where each of the c' output channels is given by the application of c' learned convolutional features to each of the c input channels. The input tensor of the first Convolutional layer is the input image itself, which in our case is a grey-level image, thus having $c = 1$. Spatial pooling layers, instead, downsample their input tensor on spatial dimensions, while preserving the same number of channels. This is usually done through a max-pooling operation, that computes the maximum on a square $k \times k$ kernel which slides over the input image with stride k, thus reducing a tensor with size $c \times w \times h$ to size $c \times \lfloor w/k \rfloor \times \lfloor h/k \rfloor$. Eventually, the output tensor of the last Spatial Pooling layer is flattened, and given to a sequence of Fully Connected (FC) layers. Given an input vector with size l, a FC layer with l' neurons learns a $l \times l'$ matrix of weights and l' biases, and each output neuron is given by the dot product of the input vector and a column of the weight matrix, plus the bias. Convolutional and FC layers are usually followed by an activation function, which creates non-linearities inside the network.

To perform content classification, we design a CNN which takes as input a square $n \times n$ patch extracted from a region, in a sliding window manner. The network contains three Convolutional-MaxPooling stages, plus two FC layers. The last FC layer has 7 neurons, and outputs the probability of each class, while the overall network is trained with Stochastic Gradient Descent by minimizing a categorical cross-entropy function over the activations of the last FC layer. Details of the architecture are given in Table I.

Once the network has been trained, it is able to predict the class of a $n \times n$ squared block. In order to classify an entire region, we take all the blocks of a region, describe them with the activations of the last but one FC layer (fc1 in Table I), and encode the resulting set of feature vectors according to their mean and covariance matrix, plus geometrical statistics of the region. In particular, for a region $\mathcal{R}$, having coordinates $(\mathcal{R}_x, \mathcal{R}_y, \mathcal{R}_w, \mathcal{R}_h)$, and containing a set $\mathcal{B}$ of squared blocks, the following feature vector is computed:

$$d(\mathcal{R}) = \left[\frac{\mathcal{R}_x}{W}, \frac{\mathcal{R}_y}{H}, \frac{\mathcal{R}_w}{W}, \frac{\mathcal{R}_h}{H}, \frac{\mathcal{R}_w}{\mathcal{R}_h}, \mu(\mathcal{B}), \sigma(\mathcal{B}) \right] \tag{2}$$

where W and H are the page height and width. $\mu(\mathcal{B})$ and $\sigma(\mathcal{B})$ are respectively the mean vector and the flattened upper-

left triangular of the covariance matrix computed over the descriptors of all blocks in $\mathcal{B}$. The resulting set of descriptors is finally given to a Random Forest classifier, which is used to classify input regions.

C. OCR Mapping

After the layout segmentation and classification steps we are able to distinguish between text and non-text regions. Historical documents are often written using fonts that are not used anymore and thus present greater challenges for OCR systems, which have to be tuned on specific sets of characters to produce a reliable transcription of the content.

On the other hand is not uncommon for historical documents to be manually transcribed. The "Enciclopedia Treccani" is no exception to this practice. We present here a technique to map OCR-ed text, read using an open-source OCR system trained on generic documents, and thus containing errors, to the manual transcription. This also allows for an enrichment of the extracted text by means of the metadata encoded in the manual transcription, such as paragraphs subdivision, titles, bulleted and numbered lists and bibliography notes.

The OCR system output must include the bounding boxes of the text lines, since their position is used to determine a preliminary paragraph splitting. We consider that each paragraph starts with a tabbed line, as Figure 2 shows.

We assume that the manual transcription is hierarchically organized in articles and paragraphs. Following this structure and since a brute force search approach is not possible we decided to build a two-level hierarchy of word-based inverted indexes. The first level is built at the article level, where each word votes for the articles in which it appears. The lower level consists of an inverted index for each article, these are built using the same principle but considering only the words in the article itself, in this case then, each word votes for the paragraphs in which it appears. All the stop words are removed from the text during this process since they do not carry substantial information and only add computational load. The inverted indexes are all prebuilt before the matching process.

The matching process analyzes each paragraph independently and does not use any information about the previous and following paragraphs in order to avoid error propagation. During the matching the inverted index hierarchy is followed from top to bottom, starting from the article level. Each word is looked up in the article-level inverted index and an histogram is built. Each bin corresponds to the number of votes that each article has received. The articles that have received more votes are the best candidates to contain the searched paragraph. For each top-5 article the same process is repeated at the lower level of the inverted index hierarchy. Each word of the paragraph votes for the paragraphs in which it appears within a specific article and the top-5 candidate paragraphs are then compared with the searched one using the Levenshtein distance. The Levenshtein distance measures the the difference between two character sequences as the number of single-character edits (insertions, deletions and substitutions) and it's suitable to determine which paragraph is the most similar

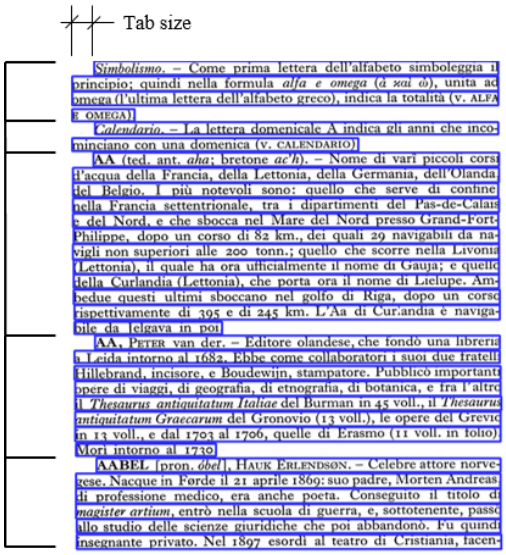


Fig. 2. Paragraph subdivision example. In this figure the text lines' bounding boxes are the output of the OCR system. Tabbed lines are used to mark the beginning of a new paragraph.

despite errors that might have occurred during the OCR phase. The best match from each article is then compared with the others and the one with the lowest Levenshtein distance is considered to be the best match for the searched paragraph.

A pseudo-code description of the algorithm is given in Algorithm 1.

IV. EXPERIMENTAL EVALUATION

The evaluation of layout segmentation performances can be approached at two different levels, the region level and the pixel level. The region level approach takes into account the semantically coherent regions extracted from a page and tries to find the best matching between them and ground truth annotated regions in order to determine an accuracy value for the entire page. The shape of a region may vary depending on the segmentation methodology. The pixel level approach evaluates the class assigned to each pixel in the segmented page and compares it with the class assigned to the same pixel in the ground truth annotation. The segmentation accuracy is calculated as the percentage of correctly classified pixels in the page. In both cases, the overall accuracy of the segmentation method can be calculated as the mean accuracy over multiple pages.

Since our hybrid method produces polygonal regions as output results we have chosen a region level evaluation methodology. In particular we used a method proposed by Phillips and Chabra [24] used for the ICDAR 2003 page segmentation competition [25]. We used the suggested *acceptance threshold* and *rejection threshold*, respectively 0.85 and 0.05 and we used intersection over union as a similarity metric to evaluate matching scores between different regions. Block size n was set to 64 in all experiments.

Algorithm 1: OCR Mapping Algorithm

```
Input  : OCR of a paragraph: OCRParagraph
Output: Best matching manually annotated paragraph:
        bestMatch
Words ← extractWords(OCRParagraph);
globalHisto = {};
forall the word ∈ Words do
    updateHisto(globalHisto, word, globalInvIndex)
end
Sort(globalHisto);
candidates = {};
for i ← 0 to 4 do
    articleInvIndex ← LoadInvIndex(Histo[i]);
    articleHisto = {};
    forall the word ∈ Words do
        updateHisto(articleHisto, word, articleInvIndex);
    end
    Sort(articleHisto);
    articleCandidates = {};
    for k ← 0 to 4 do
        articleCandidates ←
        LevenshteinDistance(articleHisto[k],
        OCRParagraph);
    end
    candidates ← min(articleCandidates);
end
bestMatch ← min(candidates);
```

Our dataset consists of all the 999 pages of the first volume of the "Enciclopedia Treccani"[1]. The whole dataset has been manually annotated by three different people and presents various layout configurations. Each page is accurately scanned at a maximum resolution of 2764×3602 and no color information is retained since the encyclopedia was not meant to be printed in color. The overall dataset contains 9489 text regions, 965 images, 126 graphics, 121 tables with border, 80 mathematical formulas, 21 music scores and 77 borderless tables. To train the SVM classifier for layout analysis, we split the dataset into a training and test set, the first one containing 2/3 of the samples of each class. In this case text samples are considered positive, and samples from all other classes are considered negatives.

We compare our method with the standard XY-Cut algorithm and with the Whitespace Analysis algorithm proposed by Baird in [8]. Results are shown in Table II: as it can be seen, our hybrid approach outperforms other classic algorithms by a large margin. Some segmentation samples are also reported in Figure 3, which shows some of the possible layout arrangements found in the encyclopedia, with entities ranging from tables, formulas, musical notation and images. Different colors are related to different classes of images and also to the paragraph subdivision.

[1]http://www.treccani.it

TABLE II
PAGE SEGMENTATION EXPERIMENTAL RESULTS.

	XY-Cut	Whitespace Analysis	Our method
Accuracy	61.8%	71.4%	93.8%

TABLE III
OCR MAPPING ERRORS.

	Wrongly mapped	Too short	Total
Error %	1.7%	0.8%	2.5%

Regarding content classification, the proposed Deep Network was implemented using Theano, and we employed the Random Forest classifier included in the OpenCV library. Since the dataset splits previously described are heavily unbalanced, the network was trained with mini-batches containing the same amount of samples for each class, randomly chosen from the training set. The same train and test splits were used to train the Random Forest classifier. Content classification performances are shown in Table IV.

On a different note, the OCR mapping performance has proven to be very good, with an accuracy of 97.5%. Nevertheless a drawback of this algorithm involves very short paragraphs and titles. Paragraphs composed by a single word or two very short words often do not carry much information about their belonging to a specific article unless they are composed of very uncommon words. For this reason we heuristically set up a threshold and only paragraphs with more than 15 characters have been considered by the algorithm. We have evaluated the algorithm sampling 1160 paragraphs read from the encyclopedia using the open-source Tesseract OCR software [26]. An error breakdown, split into too short and wrongly mapped paragraphs, is presented in table III.

V. CONCLUSIONS

In this paper we presented a layout analysis pipeline and a document content classification method. The layout analysis process is based on the classic XY-Cut algorithm and on a SVM classifier used to detect illustrations. Content classification is approached using the combination of a Convolutional Neural Network and a Random Forest classifier used to distinguish between seven different classes of layout entities. We also provided an algorithm to map OCR-ed text to a manual transcription which exploits inverted indexes built on the manual transcription. Experimental results prove the effectiveness of our approach when tested on an historical document, the "Enciclopedia Treccani".

REFERENCES

[1] J. Ha, R. M. Haralick, and I. T. Phillips, "Recursive xy cut using bounding boxes of connected components," in *ICDAR*. IEEE, 1995, vol. 2, pp. 952–955.

[2] F. Cesarini, M. Lastri, S. Marinai, and G. Soda, "Encoding of modified xy trees for document classification," in *ICDAR*. IEEE, 2001, pp. 1131–1136.

TABLE IV
CONFUSION MATRIX FOR CLASSIFICATION RESULTS OF THE CNN CLASSIFIER.

		Truth						
		Images	**Graphics**	**Tables**	**Formulas**	**Scores**	**Borderless T.**	**Text**
Prediction	**I**	72.4%	12.1%	2.2%	5.7%	1.1%	0.9%	5.7%
	G	8.3%	69.4%	0%	13.2%	0%	0%	9.0%
	T	1.6%	6.0%	30.9%	1.0%	0%	2.7%	57.8%
	F	3.6%	4.7%	0%	75.3%	0%	2.7%	13.7%
	S	9.6%	8.8%	0%	61.6%	15.2%	4.8%	0%
	BT	7.5%	3.5%	8.8%	34.0%	1.4%	13.6%	31.3%
	Txt	0.2%	0.3%	0.2%	1.7%	0%	0.3%	97.1%

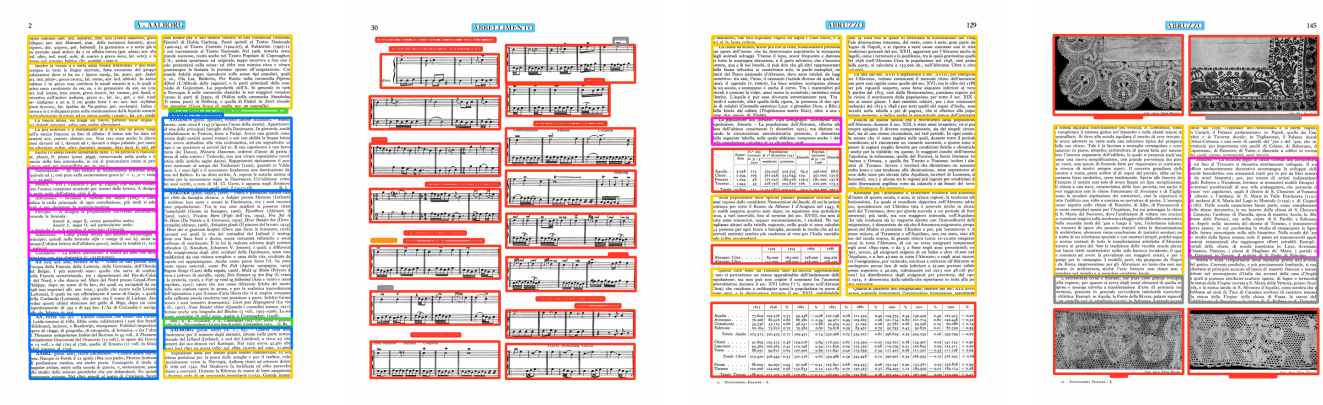


Fig. 3. Some examples of the possible layout arrangements found in the "Enciclopedia Treccani" and the extracted segmentations. Different colors are related to the different classes associated with each entity

[3] E. Appiani, F. Cesarini, A. M. Colla, M. Diligenti, M. Gori, S. Marinai, and G. Soda, "Automatic document classification and indexing in high-volume applications," *International Journal of Document Analysis and Recognition*, vol. 4, no. 2, pp. 69–83, 2001.

[4] K. Kise, A. Sato, and M. Iwata, "Segmentation of page images using the area voronoi diagram," *Comput. Vis. Image Und.*, vol. 70, no. 3, pp. 370–382, 1998.

[5] F. Sebastiani, "Machine learning in automated text categorization," *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[6] K. Chen, F. Yin, and C.-L. Liu, "Hybrid page segmentation with efficient whitespace rectangles extraction and grouping," in *ICDAR*. IEEE, 2013, pp. 958–962.

[7] G. Lazzara, R. Levillain, T. Géraud, Y. Jacquelet, J. Marquegnies, and A. Crépin-Leblond, "The scribo module of the olena platform: a free software framework for document image analysis," in *ICDAR*. IEEE, 2011, pp. 252–258.

[8] H. Baird, S. Jones, and S. Fortune, "Image segmentation by shape-directed covers," in *ICPR*, Jun 1990, vol. i, pp. 820–825 vol.1.

[9] N. Chen and D. Blostein, "A survey of document image classification: problem statement, classifier architecture and performance evaluation," *International Journal of Document Analysis and Recognition*, vol. 10, no. 1, pp. 1–16, 2007.

[10] R. Zanibbi, D. Blostein, and J. R. Cordy, "Recognizing mathematical expressions using tree transformation," *IEEE TPAMI*, vol. 24, no. 11, pp. 1455–1467, 2002.

[11] J. Hu, R. S. Kashi, D. Lopresti, and G. T. Wilfong, "Evaluating the performance of table processing algorithms," *International Journal of Document Analysis and Recognition*, vol. 4, no. 3, pp. 140–153, 2002.

[12] J. Chen, M. K. Leung, and Y. Gao, "Noisy logo recognition using line segment hausdorff distance," *Pattern recognition*, vol. 36, no. 4, pp. 943–955, 2003.

[13] T. D. Pham, "Unconstrained logo detection in document images," *Pattern recognition*, vol. 36, no. 12, pp. 3023–3025, 2003.

[14] R. P. Futrelle, M. Shao, C. Cieslik, and A. E. Grimes, "Extraction, layout analysis and classification of diagrams in pdf documents," in *ICDAR*. IEEE, 2003, p. 1007.

[15] M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan, "Syntactic segmentation and labeling of digitized pages from technical journals," *IEEE TPAMI*, vol. 15, no. 7, pp. 737–747, 1993.

[16] K.-H. Lee, Y.-C. Choy, and S.-B. Cho, "Geometric structure analysis of document images: a knowledge-based approach," *IEEE TPAMI*, vol. 22, no. 11, pp. 1224–1240, 2000.

[17] P. E. Mitchell and H. Yan, "Newspaper document analysis featuring connected line segmentation," in *Proceedings of the Pan-Sydney area workshop on Visual information processing-Volume 11*. Australian Computer Society, Inc., 2001, pp. 77–81.

[18] A. K. Jain and B. Yu, "Document representation and its application to page decomposition," *IEEE TPAMI*, vol. 20, no. 3, pp. 294–308, 1998.

[19] R. Sivaramakrishnan, I. T. Phillips, J. Ha, S. Subramanium, and R. M. Haralick, "Zone classification in a document using the method of feature vector generation," in *ICDAR*. IEEE, 1995, vol. 2, pp. 541–544.

[20] K.-C. Fan and L.-S. Wang, "Classification of document blocks using density feature and connectivity histogram," *Pattern Recognition Letters*, vol. 16, no. 9, pp. 955–962, 1995.

[21] J. Li, A. Najmi, and R. M. Gray, "Image classification by a two-dimensional hidden markov model," *Signal Processing, IEEE Transactions on*, vol. 48, no. 2, pp. 517–533, 2000.

[22] Y. Wang, I. T. Phillips, and R. M. Haralick, "Document zone content classification and its performance evaluation," *Pattern Recognition*, vol. 39, no. 1, pp. 57–73, 2006.

[23] C. Grana, G. Serra, M. Manfredi, D. Coppi, and R. Cucchiara, "Layout analysis and content enrichment of digitized books," *Multimed. Tools Appl.*, Nov. 2014.

[24] I. T. Phillips and A. K. Chhabra, "Empirical performance evaluation of graphics recognition systems," *IEEE TPAMI*, vol. 21, no. 9, pp. 849–870, Sept. 1999.

[25] A. Antonacopoulos, B. Gatos, and D. Karatzas, "Icdar 2003 page segmentation competition.," in *ICDAR*. 2003, pp. 688–, IEEE.

[26] R. Smith, "An overview of the tesseract ocr engine," in *ICDAR*. IEEE, 2007, pp. 629–633.