Information Fusion 108 (2024) 102398

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full length article

Continuous document layout analysis: Human-in-the-loop AI-based data curation, database, and evaluation in the domain of public affairs

Alejandro Peña [a,*], Aythami Morales [a], Julian Fierrez [a], Javier Ortega-Garcia [a], Iñigo Puente [b], Jorge Cordova [b], Gonzalo Cordova [b]

[a] BiDA-Lab, Universidad Autonoma de Madrid, Madrid, 28049, Spain
[b] VINCES Consulting, Madrid, 28049, Spain

ARTICLE INFO

Keywords:
Document layout analysis
Document understanding
Legal domain
QCD-based detection
Natural language processing
Human-in-the-loop

ABSTRACT

In the digital era, the amount of digital documents generated each day have being increasing exponentially with the years, to a point where it is unfeasible to process them manually. Thus, there has been growing interest from different sectors to develop automatic tools to process digital documents in an automatic manner. Yet useful, this task is challenging, due to both the large variability and the multimodal nature inherent to the problem. In most cases, a text-only approach often falls short in comprehending the information conveyed by diverse components of varying significance. In this regard, Document Layout Analysis (DLA) has been an interesting research field for many years, whose objective it to detect and classify the basic components of a document. Thus, is an interesting task to obtain a first understanding on how the different components of the document interact with each other. In this work, we used a semi-automatic procedure to annotate digital documents with different layout labels, including 4 basic layout blocks and 4 text categories. We apply this procedure to collect a novel database for DLA in the public affairs domain, the PALdb database, using a set of 24 data sources from the Spanish Administration. The database comprises 37.9K documents with more than 441K document pages, and more than 8M labels associated to 8 layout block units. The results of our experiments validate the proposed text labeling procedure with accuracy up to 99%. We also present a novel application of Quickest Change Detection (QCD) techniques on the DLA domain, which we use to continuously detect changes in the layout of the documents from multiple sources.

1. Introduction

Nowadays, the Portable Document Format (PDF), originally developed by Adobe and standardized in 2008 [1], has become one of the most important file formats for digital document storing and sharing. The reason behind its success is the possibility to present documents including a variety of components (e.g. text, multimedia content, hyperlinks, etc.) in a format independent from the software, hardware, and operating system. Furthermore, this file format allows encryption, compression, digital signature, and even interactive editing (e.g. form filling).

The advantages of the PDF format have converted it in a basic document tool for governments, administrations, and enterprises. However, despite its usefulness, automatic processing of digital PDF documents remains as a difficult task. To correctly process and extract information from a document, it is required first to understand how the different components of the document are structured and how they interact with each other. For instance, processing information contained in a table usually requires to previously detect its basic structure. Even when it comes to text processing, text blocks in documents can be grouped into a variety of semantic levels (e.g. body text, titles, captions, etc.), which have different relevance and presentation formats. The way in which basic elements are presented in a document to effectively transmit its message is known as document layout. Once the document layout is clear, then modern Natural Language Processing (NLP) technologies (e.g., Transformers [2,3] with attention mechanisms [4]) can be applied for generating useful outputs from segmented text blocks.

Document Layout Analysis (DLA) is a task that aims to detect and classify the basic components of a document. As we previously introduced, this task is a key component within the automatic document processing pipeline. Nevertheless, its usefulness is proportional to its difficulty. The main reason behind this fact is the large variability

* Corresponding author.
E-mail address: alejandro.penna@uam.es (A. Peña).
[1] https://www.op.europa.eu/en/web/forum/european-union

https://doi.org/10.1016/j.inffus.2024.102398
Received 30 November 2023; Received in revised form 25 March 2024; Accepted 30 March 2024
Available online 6 April 2024

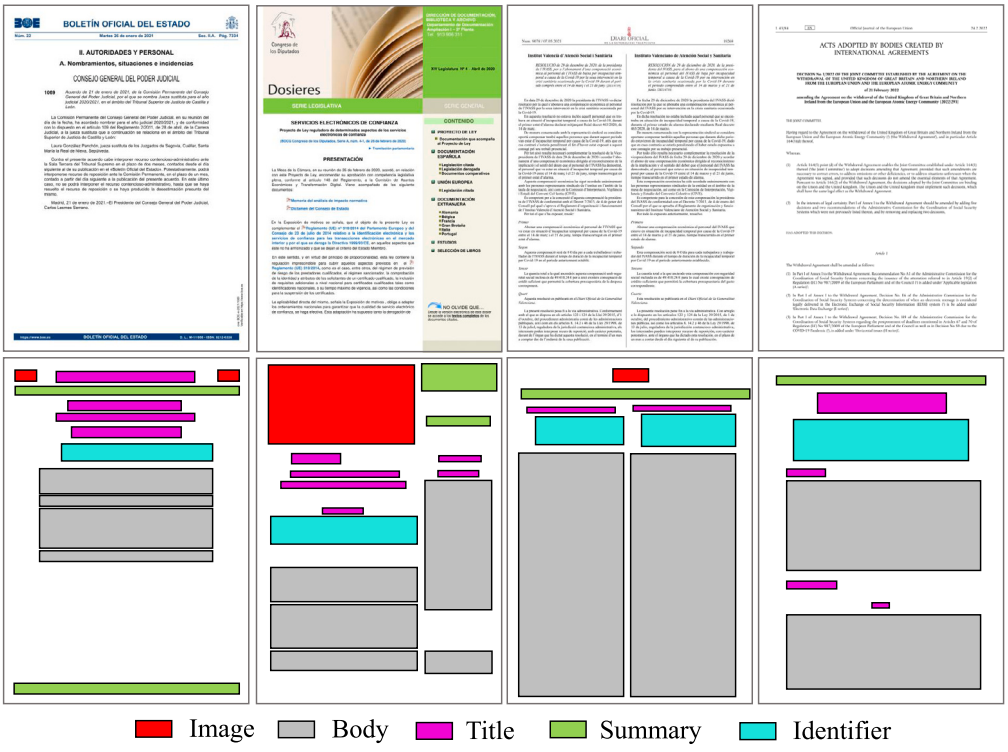

Fig. 1. Visual examples of page document images from different official gazettes (upper row). From left to right we present: (i) Spanish official gazette (i.e., BOE); (ii) public affairs document with the results of a vote in the Spanish Parliament; (iii) Spanish regional official gazette (i.e. DOGV); and (iv) Official Journal of the European Union. The bottom row presents illustrations of their layout components simplified by colored block (i.e., each color represents a document layout component).

inherent in the problem. In this work, we propose a method to semi-automatically annotate a large number of digital PDF documents with their basic layout components. Our method combines a document collection procedure, the use of PDF miners to extract layout information, as well as a human-assisted process for data curation. We use this pipeline to generate a corpus of official documents for DLA in the legislative domain, which we call Public Affairs Layout (PAL) database. The source of the documents in this work are official gazettes from different institutions of the Spanish Administration. Official gazettes are periodical publications,[1] in which administrations include legislative/judicial information and announcements. Fig. 1 presents 4 visual examples of public affairs document pages (top row), along with manual annotations of their layouts (bottom row). We included two document images from Spanish official gazettes (first and third samples), a document image from the EU official gazette (fourth sample), and a public affair document image with a summary of legislative activities in the Spanish Parliament (second sample). Although all of them span from the same domain (i.e., public affairs), they exhibit different ways of presenting their components. Notice, for instance, how the Spanish gazettes present a one- and a two-column format respectively for the text blocks. The EU gazette is relatively similar to the first Spanish gazette sample. In the second sample (i.e. the only one that is not part of an official gazette) the use of images and colors in its template favors visual design hardening layout analysis. Note that in this last example the text layout is not structured as a two-column format, but rather it corresponds to a layout where the text in the right column serves as an index. These examples illustrate the large variability of the problem at hand.

Looking again to the top rows in Fig. 1, we can appreciate that, despite the fact that the documents originate from different administrations and countries, all of them present common features related with the spatial location and visual characteristics of the text blocks. Independently of the language of the document, a reader can easily identify the different text blocks (e.g., titles, body, summary). The long-term vision of our work is to develop AI-powered tools around these commonalities able to adapt (with human intervention if needed) to the specificities of particular documents deviating from known patterns.

With that vision in mind, the present work presents methods and resources to develop AI-based tools for the automatic processing of massive databases of public affairs documents. The main aim is leveraging recent advances in the Document Layout Analysis and Natural Language Processing (NLP) domains to allow both citizens and corporations to have quick access to changes in regulations. The system's block diagram is illustrated in Fig. 2, in which we can appreciate three different modules: (i) a Harvester or Document Capturing module; (ii) a Document Layout Analysis module; and (iii) a Natural Language Processing module [5]. The Harvester constantly monitors a set of predefined public affairs data sources, and automatically collects any new document in them by using web scraping tools. Then, the DLA module extracts the main components of the document, and classifies the text blocks into different semantic categories with the objective of enriching the following NLP tasks. Finally, a NLP module processes the text blocks using Large Language Models (LLMs) to extract relevant information from the documents, such as the topics or the entities cited within them. The present paper focuses on the DLA module (see the highlighted components in Fig. 2), with the following contributions:

- We present a new publicly available database for DLA in the domain of public affairs, the Public Affairs Layout database (PALdb).[2] The database was collected from a set of 24 data sources from the Spanish administration and comprises nearly 37.9 K documents, 441K document pages, and 8M layout labels.
- We provide layout information extracted from the documents, including the pre-processed cleaned text from the text blocks detected. Thus, in addition to the layout information database, a large corpus of public affairs text data in Spanish, as well as 4

[2] https://www.github.com/BiDAlab/PALdb

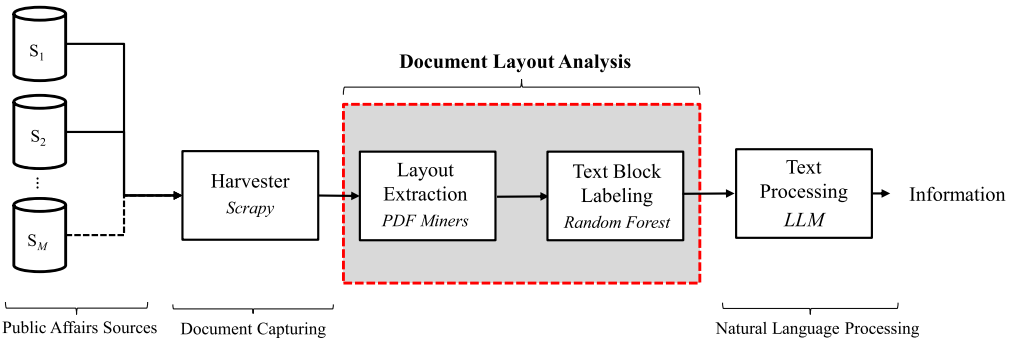

Fig. 2. Block diagram of an automatic public affairs document analysis system. We highlighted the components of the DLA module, which are detailed in this work.

other co-official languages in Spain, was collected and is available for NLP pre-training and domain adaptation.
- We assess our text labeling strategy with different experiments, in which we prove the usefulness of the information extracted to classify text blocks into different semantic classes defined after an empirical analysis of the data sources.
- We also experiment with the use of QCD techniques [6–8] to continuously detect changes in the layout style of documents from a specific source, and discuss how this technique can be a key component in an automatic document processing framework. To the best of our knowledge, this is the first time that this kind of time-adaptive scheme is scientifically documented to continuously detect document layouts.

A preliminary version of this article was published in [9]. This work significantly improves [9] in the following aspects:

- We have extended the related works section, with a review of the state-of-the-art in document understanding technologies, a task closely related with both DLA and the research project in which this work is framed.
- We provide further context on the nature of the data sources used to collect the proposed PAL database.
- We have extended the text labeling experiments of [9] with two new scenarios, in which we assess both multi-source classifiers, and the behavior of the proposed classifiers when dealing with documents from unseen data sources.
- The new experiments demonstrate the generalization capacity of the proposed models with consistent performance over 85% classification accuracies.
- We have also expanded our experiments with an application of QCD-based techniques to the DLA domain, and develop how these techniques can be useful in the development of time-adaptive document processing systems with an experiment on the proposed PAL database.

The rest of the paper is structured as follows. Section 2 provides a review of different works concerning document layout analysis and automatic digital document processing. In Section 3 we present our semi-automatic procedure to collect and annotate our legislative document database, as well as the database details. Section 4 presents the different experiments of this work. Finally, Section 5 summarizes the main conclusions.

2. Related work

The literature on Document Layout Analysis (DLA) differences between two types of PDF documents: (i) native or digital-born documents; and (ii) document images. The former are those originally created from a digital version of the documents, while the latter are scanned images captured from physical documents, or digital-born documents which were converted to images. This distinction leads to different approaches on how to extract their main layout components.

With regard to PDF native documents, the availability of all document information within the internal PDF structure makes the use of PDF miner tools the most common approach to extract layout information. A wide range of tools are available for layout extraction in digital documents. However, for a long time these tools have mainly focused on text extraction [10], and have been limited by the way the PDF format processes their components (e.g., it specifies where and how to place individual components, without using high level semantic information about them). This behavior also makes it difficult to detect layout elements such as tables, as there is no label to identify them. Modern PDF miners have learned to work with this structure, but elements such as tables remain difficult to detect. We can cite here systems such as OCR++ [11], which converts PDFs to XML documents using the *pdf2xml*[3] library, and then extracts text information from these XMLs. The authors of [10] conducted an evaluation of 14 text extraction tools, including their own, and proposed a benchmark for text extraction methods from digital PDF documents. They collected a database consisting on 12K scientific articles from *arXiv*[4], which they annotated by parsing the corresponding TeX files. More recently, in [12] the authors proposed an automatic method to annotate a large corpus of digital PDF articles, by matching the output of the *PDFMiner*[5] library with the XML representations of the articles. They released a page image database, known as PubLayNet[6], and later made available the original native PDF documents they used to create it.

On the other hand, DLA on document images has been addressed as an image processing task with the use of Computer Vision techniques. By not having access to internal information of the documents, especially when working with scanned documents, databases in this domain have been mostly annotated by hand, which ultimately limited their sizes. For instance, we can cite here the datasets collected for the ICDAR document processing challenges [13–15], which included complex documents with realistic layouts, the ICDAR 2013 Table Recognition Challenge dataset [16], or both the UW III and UNLV datasets. Other works present their own manually annotated databases based on scientific articles [17,18], with the previously mentioned PubLayNet [12] being the largest database (i.e. nearly 350K page images) thanks to an automatic annotation method.

Early approaches for conducting DLA on document images included text segmentation techniques [19,20], or the use of HOG features [21, 22] to perform this task. More recently, deep learning-based methods have been applied, specially the use of R-CNN object detection models [23,24]. In [18] a combination of F-RCNN [24] with contextual

[3] https://www.sourceforge.net/projects/pdf2xml
[4] https://www.arxiv.org
[5] https://www.github.com/euske/pdfminer
[6] https://www.github.com/ibm-aur-nlp/PubLayNet

information was proposed to conduct this task. The authors of PubLayNet [12] used both F-RCNN and M-RCNN [23] in their experiments with the proposed database. A texture-based CNN was used in [25] to classify document blocks detected by an OCR. Oliveria and Viana proposed [17] the use of 1-D CNNs as both an efficient and fast solution for DLA. They use the running length algorithm [26] to detect regions of information in grayscale images, then define blocks as regions connected after a 3×3 dilatation operation. Finally, the network classifies the blocks using both vertical and horizontal projections of these blocks.

Closely related with DLA, there are some methods that perform what is known as Document Image Understanding (DIU), which aims to process documents automatically to perform certain tasks (e.g. document classification, extracting and structuring textual information, visual question answering, etc.). In this sense, the work of [27] proposed a pre-training method that combines layout, visual and textual information in a BERT-based Transformer architecture [2], known as LayoutLM. They hypothesized that including both visual and layout cues in an information fusion setup would improve the performance of NLP models when processing visually rich documents. An improved version of LayoutLM was developed later [28], which included spatial self-attention, and new pre-training tasks to help the network model the cross-modality interaction between text, image and layout. A multilingual version of these models was recently published for multilingual document understanding [29].

3. Semi-automatic document layout annotation

In this Section, we will present our new database for Document Layout Analysis in the domain of public affairs, the Public Affairs Layout database (PALdb), along with the procedure to collect and annotate it. More specifically, Section 3.1 presents a brief description of the data sources of this work. Then, Section 3.2 introduces our data collection method, and the details of the final database. Section 3.3 describes the tools used to extract layout information from the documents, the features extracted, and the different layout components of the database. Finally, in Section 3.4 we explain our semi-automatic method to classify the text blocks extracted into different semantic categories, and the following data curation procedure.

3.1. Database sources

Spain has a wide variety of sources of public affairs documents. The Spanish Constitution[7] approved in 1978 established the public nature (in its article 9.3) of all the norms and provisions. Hence, all the judicial, royal and governmental decrees and norms, as well as the laws approved by the congress, must be published in the *Boletín Oficial del Estado (BOE)*, the official gazette of Spain, in order to be considered legally valid. This nature, in addition to its daily publication, makes the *BOE* the most important source of legal information in Spain. In addition to the *BOE*, there are two relevant publications that include the main decisions and events of the two chambers composing the Spanish Parliament (i.e., the Congress of Deputies and the Senate). These two chambers represent the legislative power of Spain, which lead them to have their own gazettes, the *Boletín Oficial del Congreso de los Diputados (BOCD)*, and the *Boletín Oficial del Senado (BOS)*.

Spain is a decentralized state that delegates part of its power and governance to various administrations responsible for managing the territories that comprise the country. This system, known as the "State of Autonomies", divides the territory into 17 autonomous communities and 2 autonomous cities, each with its own regional government, legislation, and culture (e.g.,some of these territories even have its own co-official language, such as Basque or Catalan). Thus, beyond the official gazette, Spain have 19 different regional gazettes, which address the main legislative changes in their territories.

Finally, there are two important local gazettes that we considered worth to mention in addition to the previous ones. The two main cities of Spain, Madrid and Barcelona, publish the main resolutions and news of their city councils in two official gazettes, namely the *Boletín del Ayuntamiento de Madrid (BOAM)* and the *Boletín Oficial del Ayuntamiento de Barcelona (BOAB)*. Due to the significant relevance of both cities, not only in Spain but worldwide, these publications are normally considered in the same level as the regional gazettes.

3.2. Document collection

Our data collection involves the 24 public affairs data sources from the Spanish Administration that we introduced in the previous section. The main characteristics of these data sources are summarized as follows:

1. The availability of historical repositories of PDF files, with more than ten years of almost daily publications (there are usually no publications on the weekend).
2. The diversity of document layouts, which are different for each source, and have changed over the years (e.g. compare first and third samples in Fig. 1).
3. The existence of 5 co-official languages in the Spanish territory,[8] and the freedom to publish official document in their own format, have led to a rich diversity of languages in these sources. We consider this an important point, due to the lack of data available in those languages to develop NLP systems.

Table A.1 in the Appendix presents the full list of data sources used in this work, including the links to the original repositories, and the languages employed in each one of them. Note that from now on we will refer to each data source with a numerical identifier according to Table A.1. Since we had access to the historical repositories of all the publications in various Spanish institutional websites, we used an automatic web scrapping method to download all the documents (i.e., the Harvester module depicted in Fig. 2). We then filtered documents published before 2014, and used the most current ones in our work. This allowed us to discard scanned-image PDF files corresponding to old publications, which are left out of the scope of this work. We use as web scraping backbone the Python library Scrapy[9], concretely the "Spider" class, and define how each website would be parsed. We would like to deepen here on the public nature of all these documents. According to the Spanish legislation, these documents have to be publicly published in order for their content to be legally valid. Once published, their download and distribution is completely legal. The only constraint in this sense, naturally, would be to try to modify their content maliciously to spread miss-information or fake news. However, this is not the intention of the work, since we are only interested in process them to provide information on the layout of the documents, and to extract the text without modification (i.e., no modification on the content, nor to the original documents, is conducted). Thus, no agreement to process them is required, as long as we respect the previous considerations.

As a result, the PALdb comprises 37, 910 documents, in which we have 441.3 K document pages and 138.1M tokens. Attending to the layout labels, we can find 1M images, 118.7 K tables, 14.4 K links, and 7.1M text blocks. The database is divided into a train set and a validation set, where text labels were validated by a human supervisor, as we will comment later in Section 3.4. The train set is composed of 36, 466 documents, 422K document pages and 130.5M tokens, with 1.1M images, 145.2 K tables, 16.3 K links, and 8.8M text blocks.

[7] https://www.boe.es/boe/dias/1978/12/29/pdfs/A29313-29424.pdf

[8] https://www.bilingualkidspot.com/2020/09/10/languages-spoken-in-spain-official-language-more/

[9] https://www.scrapy.org/

Table 1
Description of the validation set of the PALdb. We provide statistics on the number of documents, pages and tokens for each source, along with the number of examples of each layout category.

Source	#Doc.	#Pages	#Tokens	Layout components						
				#Images	#Tables	#Links	#ID	#Title	#Summary	#Body
1	193	602	182.5K	1206	179	1	2296	2463	192	5540
2	16	231	143.3K	0	0	0	715	1040	11	2138
3	14	224	48.9K	19	150	0	743	1369	62	2562
4	28	857	329.5K	80	141	0	2735	6287	199	9518
5	50	403	176.6K	845	106	2	810	1647	114	5293
6	30	884	189K	65	449	105	3034	3108	144	6718
7	122	393	205.9K	2	93	1	786	1695	128	7166
8	44	649	299.1K	5593	176	3	793	1702	283	7052
9	96	570	139.1K	6	346	103	1709	1189	102	5114
10	13	1046	736.8K	1023	279	268	275	5394	214	13.3K
11	75	476	170K	102	141	3	928	1496	76	4277
12	41	742	367.3K	725	128	145	2232	5146	251	15.7K
13	43	310	118K	600	17	27	930	727	55	3742
14	43	281	131.2K	865	98	275	774	1164	42	3670
15	50	225	69.5K	8	38	0	659	852	50	2737
16	13	383	204.4K	1064	140	22	382	1209	73	4100
17	142	315	143.9K	7	72	372	941	1283	157	4899
18	35	1064	297.2K	12 304	273	5	2127	1410	224	9766
19	10	1302	532.5K	1511	4384	0	2592	1962	470	15.4K
20	40	1049	348.3K	1049	63	0	2098	1951	44	32.1K
21	32	887	323.8K	862	114	268	1779	2320	293	15.8K
22	57	549	453.3K	626	309	0	547	1941	210	4534
23	61	4771	1.45M	15 468	1919	413	9608	12 563	882	48.9K
24	197	1064	454.8K	1071	17	23	1049	2006	282	14.4K
Total	**1444**	**19276**	**7.52M**	**45.1K**	**9632**	**2036**	**40.5K**	**61.9K**	**4558**	**244.4K**

Table 1 presents the number of documents, pages, tokens, and samples from each layout category included in the validation set of the PALdb. The statistics in Table 1 refer to data validated by a human supervisor, as we will comment later in Section 3.4. We set a minimum of 10 documents and 200 pages for each data source. Note that some sources present significant differences in the pages per document rate (e.g. for Source 1 the relation is roughly 3 pages per document, while for Source 19 is 130). This is a direct consequence of the nature of the documents we had access to. While all the documents we downloaded are official gazettes, these gazettes were available in two different formats: (i) full gazette contained in one document; and (ii) the gazette divided in different individual documents, each one containing a section or announcement. Most gazettes are available in both formats, but the access we had to these using the web scrapping spiders varied between publications. Another interesting relation is the number of tokens vs the number of pages, which is significantly higher for two gazettes, namely Source 22 (i.e. 825.7 tokens per page) and Source 10 (i.e. 704.4 tokens per page). These publications are the only ones among the 24 to present a two-columns format (see the third sample in Fig. 1 for a visual example of Source 22), so the number of tokens per page is naturally higher.

All the documents we collected are PDF native format, that is, they are not scanned images of a document, but rather they were originally created from a digital version of the document. This fact allowed us to use PDF miner tools to identify the main document layout components, and extract information related to these elements. In the following section, we will introduce the different layout components extracted in this work, as well as our semi-automatic annotation method.

3.3. Layout components extraction

We considered 4 main layout categories or *blocks* in this work: (i) image; (ii) table; (iii) link (i.e. a region in the document associated to an external URL); and (iv) text block. We used 2 different PDF Miner libraries to extract these components from the documents:

- **PyMuPDF**.[10] This is a Python binding for MuPDF, a powerful PDF viewer and toolkit. We use this tool to extract image, link, and text blocks from each page of the documents. PyMuPDF not only allows us to detect these blocks (i.e. it returns their position as a 4-tuple bounding box (x_0, y_0, x_1, y_1)), but also returns information about them (e.g. the raw text of the blocks, font type, size, etc.)
- **Camelot**.[11] A Python library to extract tables from PDFs. This library detects the position of the tables in each page by getting both the vertical and horizontal lines composing the table, and computing their boundaries (again as a 4-tuple bounding box). Then, it extracts their information in a *pandas dataframe* that preserves their structure, which can be exported to different formats.

It is important to note that these libraries work only for PDF native documents, therefore any document image scanned included in them was treated as a simple image. For each input PDF file, we extracted the layout information, and stored it in a CSV file. The information from all the tables detected was also exported to independent CSV files. We also generated as output a version of the input PDF file annotated with layout information, using the functions provided by PyMuPDF. This allowed us to visually assess the results of the extraction.

When extracting tables with Camelot, we adapted the bounding boxes to PyMuPDF's coordinate system, which considers the origin (0, 0) in the top left corner (see Fig. 3 for a visual example). We also consider the case of rotated tables (i.e., wide tables that appear rotated to fit in a whole page), which present another coordinate system different from normal tables.

We extracted the features presented in Table 2 for the 4 layout blocks studied in this work. As we commented before, thanks to PyMuPDF's functionalities, we had access to different font characteristics from the text blocks, including size, font type, bold, or italic information. This allowed us to define several handcrafted-features describing the text blocks. Note here that our approach is limited to the data contained in the PDF structure of the document. Hence, the

[10] https://www.pymupdf.readthedocs.io/en/latest/
[11] https://www.pypi.org/project/camelot-py/

Table 2
Layout features extracted for each layout block (i.e. Image, Table, Link, or Text Block) detected by our algorithm.

Feature	Description	Layout blocks			
		Image	Table	Link	Text
f_1	Page number in which the block was detected, starting from 0	✓	✓	✓	✓
f_{2-5}	A 4-tuple (x_0, y_0, x_1, y_1) bounding box that defines the block's region in the page (see the coordinate system in Fig. 3)	✓	✓	✓	✓
f_{6-7}	A 2-tuple (x_c, y_c), where $x_c = (x_0 + x_1)/2$, $y_c = (y_0 + y_1)/2$	✓	✓	✓	✓
f_{8-11}	A 4-tuple with the block distance to each limit of the page (see the coordinate system in Fig. 3)	✓	✓	✓	✓
f_{12}	Important data about the block (output CSV file path for tables, URL for links, and the pre-processed text for text blocks)	✗	✓	✓	✓
f_{13}	Proportion of bold tokens in the text block	✗	✗	✗	✓
f_{14}	Proportion of italic tokens in the text block	✗	✗	✗	✓
f_{15}	Average font size of the text block	✗	✗	✗	✓
f_{16}	A tuple with the different font types in the text block	✗	✗	✗	✓
f_{17}	Proportion of capital letters in the text block	✗	✗	✗	✓
f_{18}	Number of elements separated by simple space in the text block	✗	✗	✗	✓
B	Type of block detected (0 for Image, 1 for Table, 2 for Link, and 3 for Text)	✓	✓	✓	✓
L	Layout attribute label (0 for *Image*, 1 for *Table*, 2 for *Link*, 3 for *Identifier*, 4 for *Title*, 5 for *Summary*, and 6 for *Body*)	✓	✓	✓	✓

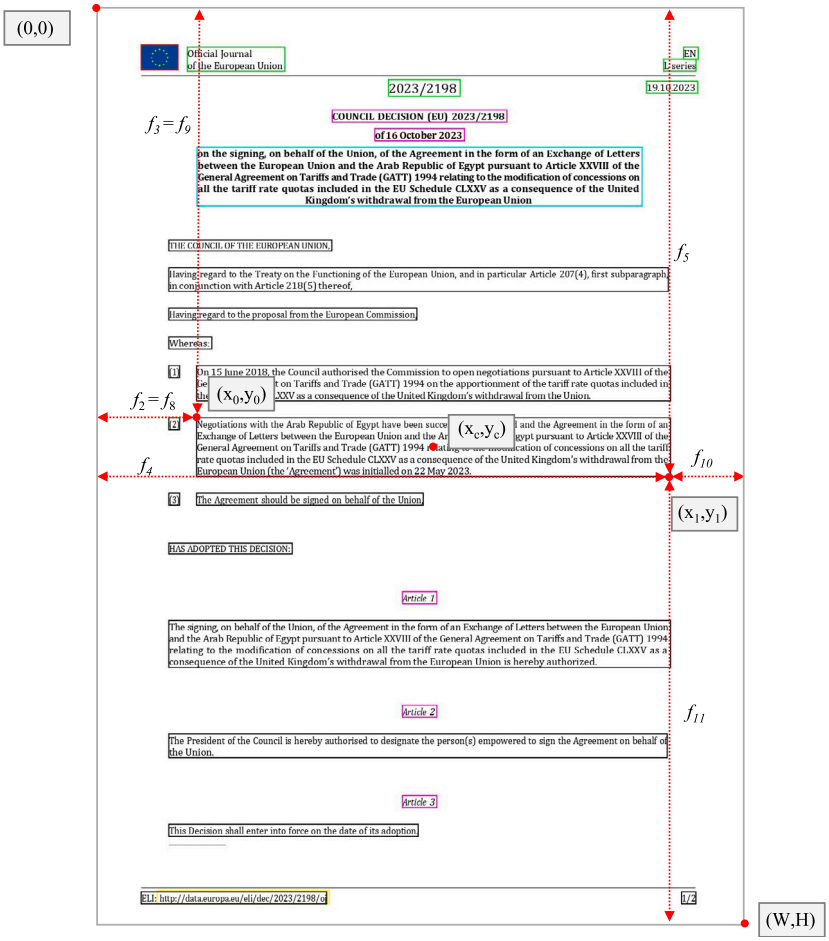

Fig. 3. Document page image from the Official Journal of the European Union. The color codes for the annotations are as follows: red for *Image*; yellow for *Link*; green for *Identifiers*; pink for *Titles*; cyan for *Summary*; and black for *Body*. We also illustrate the coordinate system, and different positional features for an example text block.

feature extraction depends on this information, and ultimately, on the editor used to create the files. We extracted text blocks in reading order (i.e. following a top-left to bottom-right schema) at line level, merging close lines with similar font features (except for size, which we have averaged using the number of tokens of each size in the resulting blocks). In this step, we pre-processed the raw text to remove line breaks, excessive white spaces, and \uFFFF Unicode characters, which were found instead of white spaces in the raw text extracted from Source 5 documents. It is worth to mention here the case of Source 4 documents, where the raw text appeared without white spaces when trying to extract it. This was probably due to the original PDF editor, which, instead of using white spaces, just put each word in its corresponding place. We could extract each word individually with PyMuPDF, and reconstruct the original text using their block and line

references. Finally, we removed any text blocks with an overlap over 70% with a table detected, as its information should be stored in the corresponding CSV table file.

Note that not all the text blocks have the same semantic role within a document. Some document components, such as tables and images, usually have a clearly defined purpose. However, a text block may be a paragraph inside the body text, a title, or a caption, among other options. Furthermore, these semantic roles of the text are usually denoted in a document by using different layout features (e.g. the use of bold or italics, variety of fonts, different sizes, specific positions, etc.). For this reason, we inspected the documents from each source, and defined 4 different text categories within them:

1. **Identifier**. A text block that identifies the document. Here we can find the date, the number of the publication, or even explicit identifiers (e.g. in Source 1 announcements documents, you can usually find an identifier in the bottom right of the page, with structure "BOE-A-year-announcement number").
2. **Summary**. A text block that can be found at the beginning of some announcements, and summarizes their content.
3. **Title**. A text block that identifies different sections within the document, or that has a significant higher importance than the body text. Titles usually present different font characteristics than regular text blocks.
4. **Body**. The text blocks composing the main content of the document.

Some visual examples of these text categories can be found in Fig. 3. Considering the text categories, layout components in our documents are labeled into 1 among 7 possible categories. As these text categories have a semantic meaning in our documents, but may not have in some other applications, we have made an explicit distinction between *block* (B) and *label* (L) in our database (see Table 2). Note that for tables, links, and images, these attributes are the same. In the next section we will describe our semi-automatic method to classify each text block into one among the previously introduced text categories, based on the text features extracted.

3.4. Text block labeling and data curation

As we introduced in the previous section, we defined 4 different text categories for the text blocks of our documents. After extracting the layout components into layout files, we had a set of 18 different features describing each text block. We proceeded to annotate each text block based on the features in a two-step process, which is illustrated in Fig. 4.

In the first step, or initialization step (Step 0 in Fig. 4), we defined a set of heuristic rules after an initial inspection of the feature values for each source and text category. An example of such rules could be "*Text blocks with a proportion of bold tokens over 0.5 and size higher than* 14 *are Titles*". The goal of the heuristic rules was not to perform a perfect labeling of the text blocks, but to have an initial set of labeled documents with possible noisy labels. We defined these rules based only in the font text features (f_{13}-f_{18} in Table 2), except the font types (f_{16}). In the case of the *Identifier* class, the presence of some key words in the text (f_{12}) was used as well (e.g., the Spanish words for the days of the week). We developed an application to help a human supervisor validate the resulting noisy-labeled documents, and correct wrong labels. For an input file, the application displayed each page annotated with the current labels. The application then detected in which part of the page the supervisor was clicking. In case of clicking inside a block's bounding box, the application changed the label to the next value. This allowed us to obtain an initial set of correctly labeled documents for each document source in a "quick" manner. All the documents were validated by an unique human supervisor (expert in analyzing this kind of documents) to prevent subjective biases of different supervisors. Note, however, that the labeling criteria were predefined by a group of experts, and based on these criteria, the human supervisor conducted the aforementioned validation. The resulting validated documents were finally assessed by the previously cited group of experts, who agreed with the labeling carried out after a visual inspection of a significant percentage of the documents. Thus, the groundtruth labels for this set are considered to be reliable for this application (i.e. gold standard).The resulting validated documents were finally assessed by a few other experts, who agreed with the labeling criteria.

Once we had the initial set of validated documents for a specific source, we moved to the next step (Step 1 in Fig. 4). We considered a threshold of 50 pages as the minimum set to proceed. In this step, we started by training a classifier for each source with its validated documents. We decided to use a Random Forests classifier for this task, as the nature of our rule-based labeling was close to the hierarchical logic employed by this classifier. In this case, we use all the features available except for the text (f_{12}). We normalized the positional features (f_2 - f_{11}) with respect to the page dimensions. We also codified font types by creating a dictionary for each unique set of fonts found within the blocks.

We qualitatively assessed that, by using the trained models to validate new documents, the number of errors significantly dropped. Our experiments in Section 4.1 will demonstrate quantitative results supporting this fact. Hence, from this point we continued with a Human-in-the-loop AI-based data curation process. The use of the AI-based labeling speeded up the validation process significantly, and many errors from the rule-based labeling were properly corrected. This allowed us to increase the validated sets, and retrain the models with more data, which ultimately ended up reducing further the errors. We repeated periodically this iterative process, and end up with models whose outputs were satisfactory enough. At this point, we used the AI-labeling process to create a large set of unvalidated data whose labels were clean enough for a training set. A visual inspection of an arbitrary selection of documents confirmed this hypothesis. Furthermore, our experiments in the next section will provide quantitative measures based on which an estimation of the reliance of these labels can be made.

4. Experiments

In this Section, we present the experimental setup and results of this work. Our experiments can be divided into two groups. The first group of experiments (Section 4.1) aims to validate the usefulness of the layout features extracted from the text blocks to train text labeling classifiers in different scenarios. The second part of our experiments (Section 4.2) presents an application of QCD-based techniques [7,8] on the layout detection domain, where the objective is to automatically detect changes in the document layout of an specific source.

4.1. Automatic text blocks labeling

As we previously introduced in Section 3.4, we applied a data curation method based on the use of a text labeling classifier to annotate the text blocks in our database. In this Section, we provide different experiments to quantitatively assess our strategy, and the usefulness of the text features extracted (see Table 2) to discriminate between different semantic text categories. Recalling Section 3.3, we defined 4 different text categories in our work after an initial inspection of the data sources: (i) *Identifier*; (ii) *Title*; (iii) *Summary*; and (iv) *Body*. Among these, the most frequent one is the *Body* category, with *Summary* on the other hand being the least frequent (see Table 1).

We will evaluate the performance on three different text labeling scenarios: (i) training one model for each data source; (ii) training a model for 5 different sources; and (iii) training a model in a *leave-3-out* setup, where we use 21 sources for training, and leave 3 for

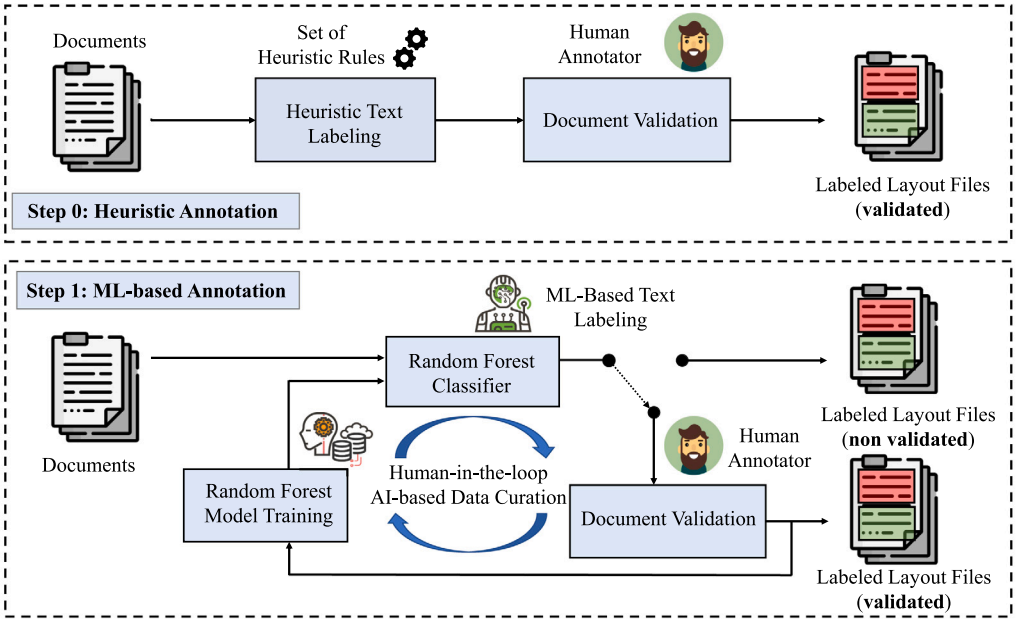

Fig. 4. Text block labels' data curation process. During Step 0 we assign labels to text blocks using a set of heuristic rules, which are validated and corrected by a human supervisor. Then, in Step 1 we use the validated documents to train a text labeling classifier per source, and use these classifiers to annotate new documents, hence reducing the number of errors and accelerating the labeling process.

evaluation. In all 3 cases, we choose a Random Forests (RF) model as classifier, inspired by the low number of features, and the hierarchical decision logic of the problem, for which we consider the RF model to be the perfect suit. We used as input for the classifiers all the features depicted in Table 2, except for feature f_{12} (i.e. the raw text data itself). We normalized features f_2 - f_{11} with respect to the dimensions of the document. For feature f_{16}, we created a dictionary to assign a value to each unique configuration of text fonts (remember that we can find different font types within the same text block, which we stored in a tuple format). We also added an additional feature, an identifier of the source, so that the classifier could learn to distinguish the source-dependent text layout patterns.

In both Scenario 1 and Scenario 2 we use 80% of the documents from the validation set for training, and 20% for testing. Note that we decided to make the train/test splitting at document level, instead of page level, so we could take into account potential intra-document biases (i.e., the existence of significant layout differences within a document with respect to the classic source's template, where annexes, for instance, play an important role). In these two scenarios, we repeated the experiment 10 times with arbitrary train/test splits, and report as results the mean and the standard deviation of both overall and per-class accuracy. Concerning Scenario 3, we use all the documents from the training sources as train set, and evaluate the resulting classifiers on the test sources. Consequently, we did not repeat this experiment several times, as both train and test split are always the same for each configuration.

The results of Scenario 1 are presented in Table 3. We can observe that the best results per-class are those obtained for *Identifier*, with a mean accuracy over 99% in all cases except for two sources, Source 6 and Source 22. These two cases also show a standard deviation higher than 3, which further highlights the increased difficulty in detecting identifiers in these sources. The good results obtained for the *Identifier* class could be expected, as these text blocks present low variability in the documents (i.e., they usually have the same position in the documents, number of tokens and font characteristics). After them, the *Body* class presents the best results, with all the mean accuracies over 95%, and low standard deviations in general. On the other hand, the "worst" performance is obtained for the *Title* category. We considered this class as the less objective during labeling, as different considerations of what is a title can be correct, especially when encountered within the body text of the announces, or in the annexes. This subjective nature make titles harder to classify. Another reason behind this performance is the potential mistakes between *Title* and *Body* classes, as the class imbalances penalizes errors more for the former. Nevertheless, the mean performance for the *Title* class is over 84% in all cases, with 19 sources reaching a performance over 90%. Attending to the overall accuracies, all the sources obtain good results, which surpass 96% for 20 different sources. The lowest performance is obtained in the Source

Table 3
Overall and per-class accuracies of the text block classifiers for each data source. We report the accuracy in terms of $mean_{std}$ after repeating 10 times the experiment with arbitrarily selected train/test splits.

Source ID	Classification accuracy (%)				
	Overall	ID	Title	Summary	Body
1	$98.08_{1.17}$	$99.54_{0.41}$	$96.20_{4.35}$	$99.47_{1.05}$	$98.48_{1.18}$
2	$96.59_{3.61}$	$100.0_{0.00}$	$91.57_{6.10}$	100_{0}	$97.99_{2.36}$
3	$98.26_{1.66}$	100_{0}	$96.72_{3.53}$	$97.75_{4.53}$	$98.91_{0.08}$
4	$99.03_{1.00}$	$99.99_{0.02}$	$98.33_{1.81}$	$96.55_{4.68}$	$99.24_{0.78}$
5	$96.26_{2.67}$	100_{0}	$94.03_{4.95}$	100_{0}	$96.22_{3.73}$
6	$95.93_{2.61}$	$98.67_{3.51}$	$88.48_{7.07}$	$90.98_{6.44}$	$98.92_{0.80}$
7	$97.90_{1.02}$	100_{0}	$96.56_{1.76}$	$99.20_{1.60}$	$97.87_{1.61}$
8	$94.81_{2.03}$	100_{0}	$86.39_{9.85}$	$89.08_{4.67}$	$97.62_{1.55}$
9	$97.19_{1.61}$	100_{0}	$89.10_{6.56}$	$98.52_{2.26}$	$98.66_{1.48}$
10	$98.70_{0.95}$	$99.85_{0.30}$	$97.60_{2.47}$	$96.84_{4.48}$	$99.26_{0.46}$
11	$96.54_{2.89}$	$99.96_{0.13}$	$96.02_{4.06}$	$98.75_{2.50}$	$95.45_{4.21}$
12	$99.35_{0.55}$	100_{0}	$98.03_{1.95}$	$98.16_{2.66}$	$99.74_{0.42}$
13	$98.25_{1.91}$	100_{0}	$93.90_{6.84}$	$97.98_{4.07}$	$98.74_{1.52}$
14	$96.52_{1.95}$	$99.69_{0.63}$	$94.65_{5.36}$	100_{0}	$96.27_{3.17}$
15	$94.64_{3.27}$	$99.33_{1.18}$	$89.13_{7.40}$	$93.09_{6.37}$	$95.54_{3.56}$
16	$93.37_{3.20}$	$99.89_{0.34}$	$84.35_{10.72}$	$61.08_{12.35}$	$96.67_{2.30}$
17	$97.16_{0.83}$	100_{0}	$93.25_{2.99}$	$97.37_{2.00}$	$97.68_{0.84}$
18	$98.94_{0.91}$	$99.93_{0.09}$	$94.95_{4.23}$	$97.96_{3.49}$	$99.35_{0.79}$
19	$98.00_{1.21}$	$99.91_{0.19}$	$90.50_{3.53}$	$99.58_{0.65}$	$98.89_{1.09}$
20	$99.93_{0.07}$	100_{0}	$99.44_{1.47}$	$99.09_{2.73}$	$99.95_{0.05}$
21	$99.38_{0.52}$	100_{0}	$96.61_{3.14}$	$93.96_{5.88}$	$99.76_{0.45}$
22	$98.22_{1.28}$	$98.87_{3.38}$	$96.87_{2.39}$	$95.03_{5.55}$	$98.90_{0.84}$
23	$97.36_{1.01}$	$99.96_{0.04}$	$91.08_{3.65}$	$99.55_{0.47}$	$98.25_{1.62}$
24	$99.27_{0.44}$	100_{0}	$95.25_{3.97}$	$99.63_{0.75}$	$99.79_{0.06}$

Table 4
Overall and per-class accuracies of the 5-source text block classifiers. We report the accuracy in terms of $mean_{std}$ after repeating 10 times the experiment with arbitrarily selected train/test splits.

Source IDs	Classification accuracy (%)				
	Overall	ID	Title	Summary	Body
1, 11, 21, 7, 17	$97.69_{0.66}$	$99.49_{0.70}$	$94.21_{1.38}$	$95.17_{3.09}$	$98.55_{0.77}$
4, 6, 21, 9, 18	$95.93_{2.25}$	$99.98_{0.03}$	$92.98_{5.6}$	$96.04_{3.39}$	$96.22_{3.44}$
12, 20, 18, 15, 2	$98.69_{0.48}$	$99.92_{0.13}$	$95.48_{2.39}$	$97.25_{1.57}$	$99.10_{0.42}$
12, 3, 19, 10, 2	$97.75_{1.27}$	$99.89_{0.11}$	$95.38_{2.49}$	$97.39_{1.64}$	$98.17_{1.45}$
23, 16, 19, 6, 13	$96.29_{1.40}$	$96.96_{4.54}$	$92.09_{2.80}$	$96.32_{1.34}$	$97.36_{2.49}$
24, 7, 11, 22, 14	$98.78_{0.32}$	$99.90_{0.12}$	$96.32_{1.03}$	$98.16_{1.12}$	$99.10_{0.49}$
24, 3, 9, 5, 10	$97.81_{1.48}$	$99.96_{0.09}$	$94.49_{3.55}$	$98.24_{1.53}$	$98.45_{2.09}$
1, 8, 13, 17, 20	$98.60_{0.61}$	$99.83_{0.29}$	$95.74_{1.37}$	$93.86_{3.35}$	$98.90_{0.67}$
4, 15, 7, 14, 22	$98.10_{0.30}$	$99.51_{0.61}$	$95.49_{1.35}$	$97.25_{1.57}$	$98.56_{0.44}$
19, 10, 11, 22, 21	$98.45_{0.38}$	$99.89_{0.10}$	$96.62_{1.00}$	$96.22_{3.23}$	$98.86_{0.31}$
16, 14, 13, 11, 22	$98.04_{0.91}$	$99.93_{0.10}$	$93.21_{3.99}$	$93.19_{3.17}$	$99.01_{0.46}$

16, which also shows an outlier-like performance in the summaries (i.e. 61.08%), and the lowest performance in titles (i.e. 84.35%). Note that this scenario represents the final stage of our data curation process (see Section 3.4), in which one model per source was trained on the validation set, and employed to annotate the training documents of the corresponding source. Considering the groundtruth text labels of the validation set as the gold standard for our application, we can assume that the performance achieved here approximates the expected noise in the text labels of the training set of PALdb.

Moving on now to Scenario 2, we present the results for different configurations in Table 4. These 5-source configurations were selected randomly. In general, all the classifiers obtained satisfying results, with a minimum overall accuracy of 95.93% (i.e. the classifier trained on Sources 4, 6, 21, 9, and 18, see second row of Table 4). The results per-class follow the same trend, with a minimum value of 92.09% for the *Title* class. While the accuracy of both *Identifier* and *Body* classes are close to the ones obtained in the previous scenario (see Table 3), we can observe an improvement in performance for the *Title* category, as we find the accuracy in the range 92% – 97% in all cases. The standard deviation of this metric is also lower than the ones exhibited by the individual classifiers. Note that these results are also obtained in configurations combining one- and two-columns text formats (i.e., the ones considering Source 22 or Source 10). Despite having 5 different possible layout sources, the use of the source identifier as input clearly helped the classifier to learn specific patterns of each source, therefore leading to accurate predictions during inference.

Now that we have assessed the performance of both individual and multi-source text block classifiers, in Scenario 3 we assess the generalization capabilities of the classifiers when making predictions on documents from sources never seen during training. We chose a Leave-3-out configuration, where only 3 sources were used as testing sources, and employed the remaining for training. Table 5 presents the results of this scenario. We grouped the 24 sources so that all of them are evaluated once. We can observe a significant drop in performance in all cases. In both *Title* and *Body* categories the drop in performance is more subtle compared to the previous scenarios. In just one case

Table 5
Overall and per-class accuracies in a leave-3-out setup.

Test source IDs	Classification accuracy (%)				
	Overall	ID	Title	Summary	Body
1, 14, 19	91.21	77.45	85.38	89.64	95.75
13, 11, 10	87.51	69.08	70.83	34.78	97.74
12, 7, 22	86.43	48.56	67.39	10.98	96.95
24, 2, 6	80.79	34.35	86.53	0.92	90.35
3, 20, 21	90.82	33.47	68.07	39.24	99.25
4, 9, 17	78.06	28.99	78.52	32.01	92.28
23, 16, 8	88.07	90.18	74.76	1.62	92.90
15, 5, 18	84.45	39.00	73.60	19.57	97.38

(i.e., the case where Sources 4, 9, 17 conform the test set) the accuracy for the *Body* class is lower than 90%. This behavior is not surprising, as these classes are easier to generalize. On the other hand, both *Identifier* and *Summary* classes present features more dependent of the source layout, which would explain their worst results. For instance, the identifiers obtained the best performance in previous experiments, as their positional features are usually well defined within the documents of a certain source. However, most of the sources place identifiers in page borders, so depending of the train sources they can be easier to recognize than summaries. In the case of summaries, we found that they usually present a set of similar features within each source (i.e., bold, italic, size, font type, or in certain cases, even positional features), which ultimately makes them the most source-dependent text blocks. This led to a significant performance decay, with some configurations seeing their performance drop dramatically.

4.1.1. Results discussion

In the context of the text block labeling experiments, we have analyzed 3 different scenarios, namely: (i) training a specific text labeling classifier for each source; (ii) training a model for 5 different sources; and (iii) training a model in a leave-3-out setup, where sources composing the test set are not used for training. From the results of the first two scenarios (see Tables 3 and 4 respectively) we can extract that the proposed text features are useful to distinguish between different semantic classes. Furthermore, the RF classifier represents an effective model to conduct this task. However, a significant gap in performance can be seen between those scenarios and the results of Scenario 3 (see Table 5), where some classes (i.e. *Identifier* and *Summary*) exhibit significantly low accuracies.

We decided to inspect further the results of this last scenario, and the nature of the errors. We present in Fig. 5.a the confusion matrix of the classifier evaluated with Sources 24, 2, and 6 (i.e. the fourth row of Table 5), while Fig. 5.b presents the one obtained with Sources 4, 9, and 17 (i.e. the sixth row of Table 5). Note that we selected on purpose for this analysis the two models that obtained the worst overall accuracies. In both cases, the classes *Identifier* and *Summary* obtained significantly worse results than the ones in previous scenarios. For the *Identifier* class, in both classifiers more than 40% of the test instances were labeled as *Body*. As we commented in the previous paragraph, identifiers are usually text blocks with specific positions in the page, but their text features are close to the body text. In the absence of source specific knowledge, a trained classifier would mistake *Identifier* samples with *Body* instances. We have another situation for the *Summary* category, where most of the errors were labeled as *Title*. As we discussed before, summaries usually have font features different from the body text, but source-specific enough to prevent the classifier to assign the correct label, hence end up being classified as *Title*. Note that in this scenario we also included the source identifier as input, but it did not provide any significant information about the test sources.

In the light of these results, we can conclude that the data curation strategy applied (see Fig. 4) is adequate. The application of annotation models to sources that were not seen during training poses the risk of obtaining a significant number of labeling errors. Although the overall

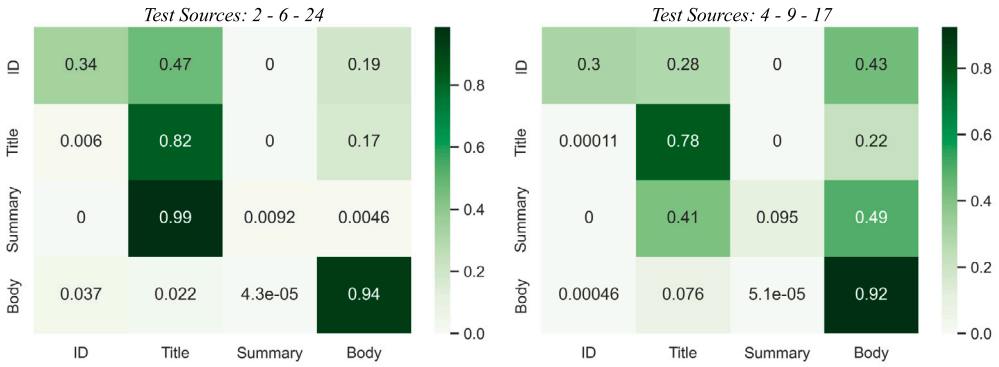

Fig. 5. Confusion matrices of the text block classifiers. Left figure presents the results of the classifier evaluated with Sources 2, 6 and 24. Right figure presents the results of the classifier tested with Sources 4, 9, and 17.

accuracies in Scenario 3 are not that far from the other scenarios, the per-class accuracies of some specific classes are clearly degraded. Note that these classes are not that frequent, compared to more generic classes such as *Title* or *Body*, so the overall accuracy is acceptable anyway. By including at least a few examples from each of the sources in the training data, the model is able to learn source-specific features for all the classes. Therefore, we can obtain accurate classification of all the text blocks using either of the first two scenarios.

4.2. QCD-based automatic layout change detection

In this section, we propose a novel application of Document Layout Analysis tools. Suppose that we want to develop an automatic document understanding system, in which one module performs DLA as a pre-processing step previous to the main task, for which it is crucial to have access to layout information. Assume that this system receives as input documents from a set of pre-defined sources with well-known layouts, so the layout detection module is trained on them. This system is similar to the one presented in Fig. 2. But, what if the layout of some of these sources significantly changes? The performance of the whole system would be at risk, since the layout module would perform worst on documents from those sources. The same would be true if documents from sources other than those for which the system was designed began to enter the system. In such scenario, it would be desirable to have a system that detects changes in the layout in order to react as quick as possible to them, and here is where Quickest Change Detection (QCD) algorithms [6–8] can play a role.

QCD algorithms are a family of statistical signal processing techniques, whose objective is to detect changes in certain statistical properties, while maintaining a trade-off between detection delay and false alarm rate. These algorithms are based on computing the cumulative sum of the scores of previous events instances, until a predefined threshold is reached. Returning to the original problem of detecting layout changes, we would start by training a source layout classifier, which learns to identify if the source of the input document is among the sources of interest (i.e., class $\omega = Y$, for Yes, it is within known sources) or not (i.e., class $\omega = N$, for No, it is not within known sources) based on layout features $\mathbf{f}$. After training the classifier, we compute the probability scores for both classes using a test set, and estimate the probability density functions (pdfs) corresponding to the trained classifier $p(\mathbf{f}|\omega = Y)$ and $p(\mathbf{f}|\omega = N)$. For a new, unseen input document U, represented by its feature vector $\mathbf{f}^U$, we then compute those pdfs to generate the log-likelihood ratio:

$$L = \log \frac{p(\mathbf{f}|\omega = N)}{p(\mathbf{f}|\omega = Y)} \tag{1}$$

Note that the likelihood function of a distribution can be viewed as a measure on how well the input data fits with the trained model. If the input document $\mathbf{f}^U$ comes from one of the sources of interest, is likely to present a higher denominator than nominator in Eq. (1). Otherwise, it is expected a greater value of numerator if document $\mathbf{f}^U$ does not belong to any source of interest, as established in the current layout classifier from which the pdfs were generated. By computing the log-likelihood ratio, we are conducting a statistical inference of the true class of the document. We can then use the ratio as a score, and consider previous documents using the following to compute the Change Detection score at time step t:

$$S_t = \max(S_{t-1} + L, 0) \tag{2}$$

where $S_0 = 0$. The use of the maximum in Eq. (2) prevents the score to tend to extremely negative values after observing a long sequence of documents belonging to class Y. As the system receives documents from class N, or presenting an unusual layout for class Y sources, the score S_t will increase across time steps t, until a certain threshold τ is reached (i.e., $S_t > \tau$). Depending on the threshold, we can control the trade-off between false-alarm rate and the delay to detect changes. Someone could argue that using the first source layout classifier is enough to prevent undesirable documents to be fed into the following stages of the system. By using QCD we measure to what extent the sequence of documents is consistent with what is expected in terms of document layout. Fig. 6 presents the block diagram of the problem at hand.

We simulated the previously described situation by applying QCD-based continuous detection to 14 sources in PALdb. We defined a set of sources for our system, which includes samples from Sources 1, 4, 5, 6, 10, 12, 13, 14, 18, 20, 21, 22, 23, and 24. From these, we selected 4 sources as positive class or class Y (4, 10, 21, and 22). The other 10 sources from the previous list are labeled as negative class or class N. The aim of the experiment is to evaluate the performance to detect changes in the layout style of positive sources. We trained a Random Forest classifier to distinguish between documents from both classes, using as input a feature vector $\mathbf{f}$ describing each document. We included in the feature vector the median of each positional feature (i.e. f_2 - f_{11} in Table 2) for images, titles, identifiers, summaries, and body text components in the first 2 pages. To train the classifier, we use 50% of the documents, and then use 40% to validate the performance of the classifiers. Furthermore, this validation set is used to obtain class probabilities. Note that we left the remaining 10% of the documents to evaluate the QCD detection algorithm, as this set should be different from the classifiers' train and test sets. Once we had the probabilities values from the validation set, we estimated the probability density function for each class using Kernel Density Estimation (KDE) with Gaussian kernels, and start computing the detection score as indicated in Eq. (2).

Fig. 7 presents the evolution of the detection score (S_t) for an input document sequence extracted from the test set. Samples 0 to 17 are drawn from class Y (positive layout style), while the rest are examples

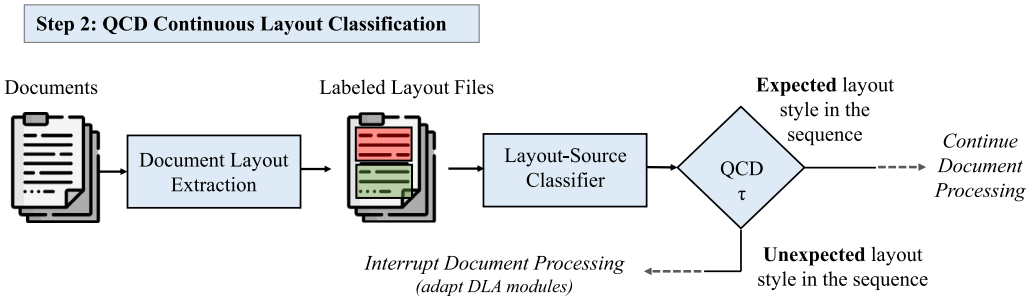

Fig. 6. Quickest Change Detection algorithm (QCD) for Document Layout Detection.

from class N (negative layout style). We also included two documents from Source 4 dated in 2009, whose layout varies from the current ones in our dataset (i.e. samples 15 and 16). We can observe that the score remains at 0 until sample 14. This behavior is expected for the class Y samples, as the likelihood ratio in Eq. (1) tends to negative values when the sample presents a greater likelihood with the main class distribution (i.e. remember that we set the cumulative score in Eq. (2) to have a minimum value of 0). However, when the 15-th sample is reached, the score slightly increases, despite being drawn from Source 4. Since this document presents an older layout from the ones in our dataset, its likelihood to come from unknown sources increases, so the score increases. The same applies for sample 16, where we also have an increase of 2. Note that in sample 17 the cumulative score decreases again, as this is a normal sample that pushes back the cumulative score to zero. From sample 18 onwards, the cumulative score only increases its value, since all the samples have a greater likelihood with the class N distribution. By analyzing the results in Fig. 7, we can see that the threshold selection affects the detection of events ($Y \rightarrow N$), and it controls the trade-off between missing those events and their detection delay. For instance, if we were to select a threshold around 12, as the one represented in Fig. 7, most of the samples from class N would be instantly detected. However, the two old Source 4 documents would be missed, as the following document with the expected layout (i.e. sample 17) pushes the score back to 0. In this scenario, it would require a longer sequence of samples with a different layout than the expected to detect the event ($Y \rightarrow N$), instead of a few examples that might be outliers. On the other hand, a more restrictive threshold, e.g. 2, would detect most samples different from the expected, thus reducing the delay and reducing the missed detections (most probably at the cost of increasing the false alarms).

Some other examples of the application of QCD are presented in Fig. 8. In all cases, we use the same corpus of data sources previously introduced, and change the sources composing the class Y subset. We also included in these scenarios an open set of documents, that is, documents from sources that were not seen during training (i.e., layout styles not included in classes Y or N). Note that all the scenarios in Fig. 8 present the same pattern of class Y/N documents, with samples 6 to 12 being open set samples, and the rest documents from class Y. Similar behaviors can be seen, where the cumulative detection score decreases until reaching its lowest value (i.e. 0) when facing samples from the main sources. Documents from the open set augment rapidly the cumulative score in general. It is interesting to appreciate that changes in the score are not fixed, but depend on the nature of the sources and the document layouts. Take for instance the results depicted in the top row of Fig. 8, where the score increments associated with the documents from the open set are comparable to the decrements caused by class Y documents. This denotes a highly confidence of the classifier when assigning the open set documents to class N. However, these increments associated with the open set are far from the ones seen in the systems illustrated in the bottom

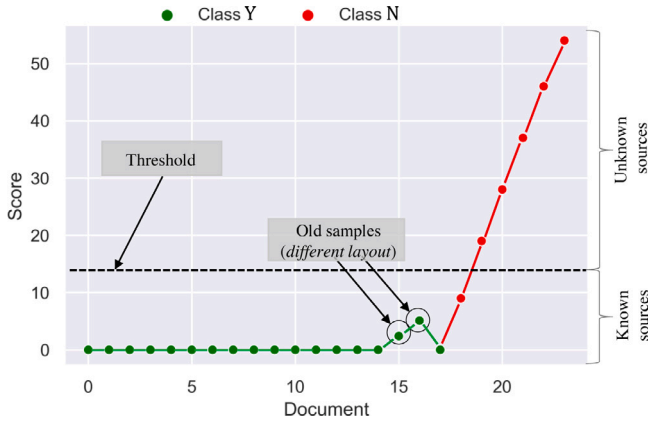

Fig. 7. Cumulative detection score for an input sequence of documents. Documents may come from sources of interest already considered in the system development (class Y), or outside those known sources (class N). With QCD we detect the transitions between Y and N. We included two old samples from the sources of interest to analyze the systems' reaction when facing a data sample of interest with an unexpected layout.

row of Fig. 8. In these cases, open set documents are not predicted as class N documents with such confidence, hence the cumulative score only augments slightly when facing them, compared to the previous scenarios. Recall that these sources were not seen during training, so it is not surprising the behavior found here. Nevertheless, in all cases the open set sources are detected as changes in the layout style. Since we can set the detection threshold to adapt it to these situations, it is not a problem that the score increases slowly, as long as the likelihood of these samples is distinct in comparison to known sources. This is a similar behavior to the one observed in Fig. 7 with the old samples. Being able to distinguish them, despite not seen during training, is a desirable property of the system that shows some protection against zero shot attacks, as our initial intention was to detect the presence of documents from sources different from the desired ones.

5. Conclusions

In this work, we developed a new procedure to semi-automatically extract layout information from a set of digital documents, and provide annotations about the main layout components. Our method is based on the use of web scraping tools to collect documents from different pre-defined sources, and extract layout information with classic PDF miners. The miners not only detect tables, links, images, and text blocks in the documents, but also provide us with different information about these blocks, including font characteristics of the text blocks. We defined a set of text features, which are useful to describe the text blocks and discriminate between semantic categories.

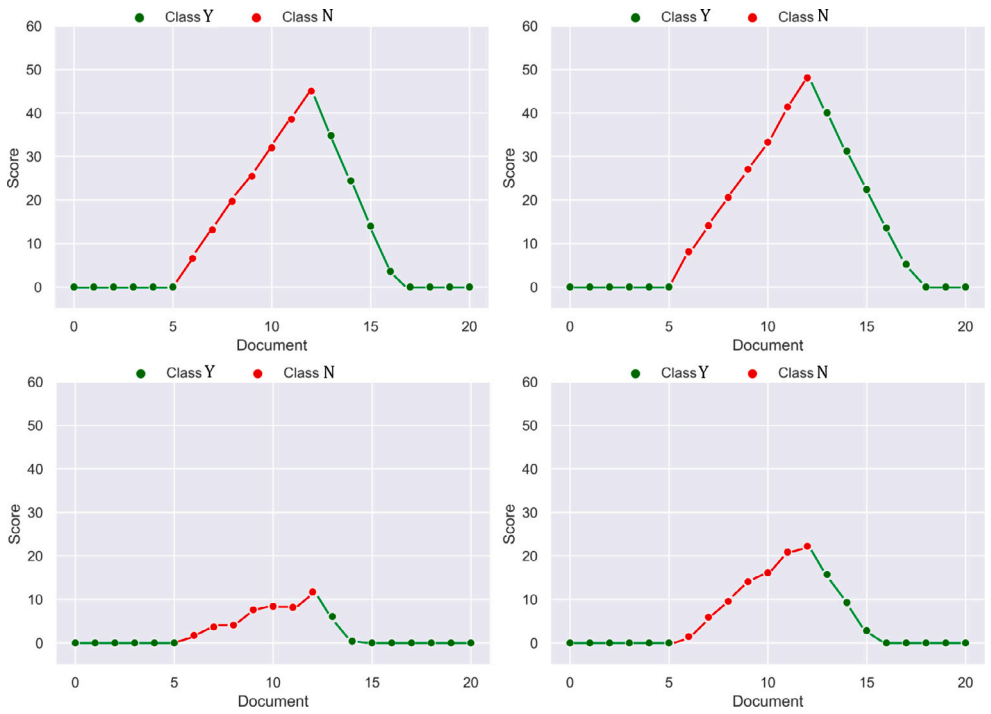

Fig. 8. Cumulative detection score (S_t) for different subsets of class Y and class N documents. These already considered in the system development (class Y), or outside those known sources (class N). With QCD we detect the transitions between Y and N. not seen during training. The class N samples depicted are all open set samples. In top-left, sources of interest are 20, 23, and 24, with Sources 1, 4, and 22 composing the open set. For bottom-left, Sources 13, 18 and 21 were employed as main set, leaving Sources 10, 12, and 13 as open set. Figures on the right employed the same splits that the left ones, switching the main and open set sources.

We applied our procedure to generate a new Document Layout Analysis (DLA) database in the domain of public affairs documents, the Public Affairs Layout database (PALdb), now available for research.[12] We employed 24 different sources of public affairs documents from the Spanish Administration. After an initial inspection of the documents from each source, we defined 4 different text categories, and classify the text block with these categories using a human-in-the-loop AI-based data curation process. Our data curation process trains text labeling models using human validated documents in an iterative way (i.e., the output of the models are validated and corrected by a human supervisor, leading to new validated documents to train more accurate models).

In the first part of our experiments, we assessed the usefulness of the text features to discriminate between the previously defined classes, thus validating our data curation procedure. We explored different scenarios using Random Forests classifiers, including one model per source, one model for 5 sources, and a leave-3-out configuration (i.e. train with all sources except for 3, which are left as test set). The results obtained in the first two configurations validate our strategy, while Scenario 3 show some limitations when dealing with source-dependent classes that have not been seen during training. As future work, we suggest exploring other text labeling models, such as recurrent models, which could preserve information about previous detected blocks, therefore allowing the flow of information between blocks in the page. Other text semantic classes could be also studied depending on the nature of the source documents. Also, note from the results of Scenario 3 that, despite the relative uniformity of the sources (legal/governmental communications), the inclusion of documents from each source is positive for data-driven models to capture the source-dependent features. In future work, we would like to extend the PALdb with new sources of public affairs documents, for instance, documents created by private organizations, or from sources others than official gazettes. We consider this point of special interest, not only for improving the diversity of the database, but to effectively broaden the range of scenarios that could be addressed by AI-based solutions for public affairs.

Finally, we explored the application of Quickest Change Detection (QCD) algorithms to the Document Layout Analysis domain with the objective of continuously detecting the presence of changes in time of the layout style of documents based on the layout information. Our experiments show how this algorithm can be helpful to detect if a sequence of documents presents the layout expected, while having control of the trade-off between missed detections and detection delay via threshold selection.

As we commented in Section 2, new lines of work around our developments are arising beyond DLA. One opportunity in this regard is focused in pre-training procedures for document processing that consider information from different domains in a multimodal setup [30], including layout information. The idea is to gain full insight of document information and be able to conduct several downstream tasks within the same system [31]. The advantages of such setup do not only comprise the use of supplementary information from different domains, but to exploit the correlations across domains. This is of special interest due to the success of Transformer-based architectures on capturing the dependencies across inputs modalities thanks to the attention mechanism, even when working with image data [32]. As future work, we would like to explore this new line of study, for which the PALdb may be suitable due to the inclusion of both textual and layout information. Compared to the system depicted in Fig. 2 for document processing, in which a serialized pipeline is presented, we will explore the development of new multimodal Transformer-based systems that leverage from different modalities to extract document information. This setup is of especial interest, as it has the potential to improve several downstream tasks in the domain of public affairs document processing, such as topic classification [5], content summarizing, or named entity recognition.

[12] https://www.github.com/BiDAlab/PALdb

Table A.1
List of data sources used to collect the PALdb.

Data Source	ID	Language	Access
Boletín Oficial del Estado	1	Spanish	boe.es/diario_boe
Boletín del Congreso de los Diputados	2	Spanish	congreso.es/indice-de-publicaciones
Boletín del Senado	3	Spanish	senado.es/web/actividadparlamentaria/publicacionesoficiales/senado/boletinesoficiales
Boletín de la Comunidad de Madrid	4	Spanish	bocm.es
Boletín de la Rioja	5	Spanish	web.larioja.org/bor-portada
Boletín de la Región de Murcia	6	Spanish	borm.es
Boletín del Principado de Asturias	7	Spanish	sede.asturias.es/ast/servicios-del-bopa
Boletín de Cantabria	8	Spanish	boc.cantabria.es/boces/
Boletín Oficial del País Vasco	9	Spanish, Basque	euskadi.eus/y22-bopv/es/bopv2/datos/Ultimo.shtml
Boletín de Navarra	10	Spanish, Basque	bon.navarra.es/es
Boletín de la Junta de Andalucía	11	Spanish	juntadeandalucia.es/eboja.html
Boletín de Aragón	12	Spanish	boa.aragon.es
Boletín de Islas Canarias	13	Spanish	gobiernodecanarias.org/boc
Boletín de Islas Baleares	14	Spanish, Catalan	caib.es/eboibfront/
Boletín de Castilla y León	15	Spanish	bocyl.jcyl.es
Boletín de la Ciudad de Ceuta	16	Spanish	ceuta.es/ceuta/bocce
Boletín de Melilla	17	Spanish	bomemelilla.es/bomes/2022
Diario de Extremadura	18	Spanish	doe.juntaex.es/
Diario de Castilla–La Mancha	19	Spanish	docm.jccm.es/docm/
Diario de Galicia	20	Galician	xunta.gal/diario-oficial-galicia/
Diari de la Generalitat Valenciana	21	Spanish, Valencian	dogv.gva.es/es
Diari de la Generalitat Catalana	22	Spanish, Catalan	dogc.gencat.cat/es/inici/
Boletín del Ayuntamiento de Madrid	23	Spanish	sede.madrid.es/portal/site/tramites/menuitem.944fd80592a1301b7ce0ccf4a8a409a0
Boletín del Ayuntamiento de Barcelona	24	Spanish, Catalan	w123.bcn.cat/APPS/egaseta/home.do?reqCode=init

CRediT authorship contribution statement

Alejandro Peña: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Aythami Morales:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Conceptualization. **Julian Fierrez:** Writing – review & editing, Writing – original draft, Project administration, Funding acquisition. **Javier Ortega-Garcia:** Writing – review & editing, Project administration, Funding acquisition. **Iñigo Puente:** Writing – review & editing, Project administration. **Jorge Cordova:** Writing – review & editing, Project administration. **Gonzalo Cordova:** Writing – review & editing, Software.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Alejandro Pena reports financial support was provided by Government of Spain Ministry of Universities. Aythami Morales reports financial support was provided by Spain Ministry of Science and Innovation. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

A link to the repository to obtain the database is included in the paper.

Acknowledgments

Support by VINCES Consulting under the project VINCESAI-ARGOS, and BBforTAI (PID2021-127641OB-I00 MICINN/FEDER). The work of A. Peña is supported by a FPU Fellowship (FPU21/00535) by the Spanish MIU. A. Morales is supported by the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with Universidad Autónoma de Madrid in the line of Excellence for the University Teaching Staff in the context of the V PRICIT (Regional Programme of Research and Technological Innovation). VINCES had an active role on the development of the work, through the guidance of the different authors belonging to the corporation. The rest of funding sources had no role/influence on the development of this work.

Appendix

See Table A.1.

References

[1] Document Management — Portable Document Format — Part 1: PDF 1.7, Standard, International Organization for Standardization (ISO), 2008.
[2] J. Kenton, M. Devlin, L. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of NAACL-HLT, 2019, pp. 4171–4186.
[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, et al., Language models are few-shot learners, Adv. Neural Inf. Process. Syst. 33 (2020) 1877–1901.
[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al., Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[5] A. Peña, A. Morales, J. Fierrez, I. Serna, et al., Leveraging large language models for topic classification in the domain of public affairs, in: Document Analysis and Recognition – ICDAR 2023 Workshops, Springer Nature Switzerland, San Jose, CA, USA, 2023, pp. 20–33.
[6] T. Banerjee, V. Veeravalli, Data-efficient quickest change detection in minimax settings, IEEE Trans. Inform. Theory 59 (10) (2013) 6917–6931.
[7] P. Perera, J. Fierrez, V.M. Patel, Quickest intruder detection for multiple user active authentication, in: 2020 IEEE International Conference on Image Processing, ICIP, IEEE, 2020, pp. 1341–1345.
[8] A. Acien, A. Morales, J. Fierrez, R. Vera-Rodriguez, et al., Active detection of age groups based on touch interaction, IET Biometrics 8 (1) (2019) 101–108.
[9] A. Peña, A. Morales, J. Fierrez, J. Ortega-Garcia, et al., Document layout annotation: Database and benchmark in the domain of public affairs, in: Document Analysis and Recognition – ICDAR 2023 Workshops, Springer Nature Switzerland, San Jose, CA, USA, 2023, pp. 123–138.
[10] H. Bast, C. Korzen, A benchmark and evaluation for text extraction from PDF, in: 2017 ACM/IEEE Joint Conference on Digital Libraries, JCDL, 2017, pp. 1–10.
[11] M. Singh, B. Barua, P. Palod, M. Garg, et al., OCR++: A robust framework for information extraction from scholarly articles, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 3390–3400.
[12] X. Zhong, J. Tang, A.J. Yepes, PubLayNet: Largest dataset ever for document layout analysis, in: 2019 International Conference on Document Analysis and Recognition, ICDAR, 2019, pp. 1015–1022.
[13] A. Antonacopoulos, D. Bridson, C. Papadopoulos, S. Pletschacher, A realistic dataset for performance evaluation of document layout analysis, in: 2009 10th International Conference on Document Analysis and Recognition, ICDAR, 2009, pp. 296–300.
[14] C. Clausner, C. Papadopoulos, S. Pletschacher, A. Antonacopoulos, The ENP image and ground truth dataset of historical newspapers, in: 2015 13th International Conference on Document Analysis and Recognition, ICDAR, 2015, pp. 931–935.
[15] C. Clausner, A. Antonacopoulos, S. Pletschacher, ICDAR2017 competition on recognition of documents with complex layouts-rdcl2017, in: 2017 14th IAPR International Conference on Document Analysis and Recognition, Vol. 1, ICDAR, 2017, pp. 1404–1410.
[16] M. Göbel, T. Hassan, E. Oro, G. Orsi, ICDAR 2013 table competition, in: 2013 12th International Conference on Document Analysis and Recognition, ICDAR, 2013, pp. 1449–1453.
[17] D. Oliveira, M. Viana, Fast CNN-based document layout analysis, in: 2017 IEEE International Conference on Computer Vision Workshops, ICCVW, 2017, pp. 1173–1180.
[18] C. Soto, S. Yoo, Visual detection with context for document layout analysis, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP, 2019, pp. 3464–3470.
[19] S. Bukhari, F. Shafait, T. Breuel, Improved document image segmentation algorithm using multiresolution morphology, in: Document Recognition and Retrieval XVIII, Vol. 7874, 2011, pp. 109–116.
[20] S. Eskenazi, P. Gomez-Krämer, J. Ogier, A comprehensive survey of mostly textual document segmentation algorithms since 2008, Pattern Recognit. 64 (2017) 1–14.
[21] T. Lang, M. Diem, F. Kleber, Physical layout analysis of partly annotated newspaper images, in: Proceedings of the 23rd Computer Vision Winter Workshop, 2018, pp. 63–70.
[22] A. Sah, S. Bhowmik, S. Malakar, R. Sarkar, et al., Text and non-text recognition using modified HOG descriptor, in: 2017 IEEE Calcutta Conference, CALCON, 2017, pp. 64–68.
[23] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision, CVPR, 2017, pp. 2961–2969.
[24] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, Adv. Neural Inf. Process. Syst. 28 (2015).
[25] S. Kosaraju, M. Masum, N. Tsaku, P. Patel, et al., DoT-Net: Document layout classification using texture-based CNN, in: 2019 International Conference on Document Analysis and Recognition, ICDAR, 2019, pp. 1029–1034.
[26] K. Wong, R. Casey, F. Wahl, Document analysis system, IBM J. Res. Dev. 26 (6) (1982) 647–656.
[27] Y. Xu, M. Li, L. Cui, S. Huang, et al., LayoutLM: Pre-training of text and layout for document image understanding, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1192–1200.
[28] Y. Xu, Y. Xu, T. Lv, L. Cui, et al., LayoutLMv2: Multi-modal pre-training for visually-rich document understanding, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Vol. 1: Long Papers), 2021, pp. 2579–2591.
[29] Y. Xu, T. Lv, L. Cui, G. Wang, et al., LayoutXLM: Multimodal pre-training for multilingual visually-rich document understanding, 2021, arXiv/2104.08836.
[30] A. Peña, I. Serna, A. Morales, J. Fierrez, et al., Human-centric multimodal machine learning: Recent advances and testbed on AI-based recruitment, SN Comput. Sci. 4 (43) (2023).
[31] Y. Huang, T. Lv, L. Cui, Y. Lu, et al., LayoutLMv3: Pre-training for document AI with unified text and image masking, in: Proceedings of the 30th ACM International Conference on Multimedia, Lisboa, Portugal, 2022, pp. 4083–4091.
[32] K. Han, H. Wang, Y. Chen, X. Chen, et al., A survey on vision transformer, IEEE Trans. Pattern Anal. Mach. Intell. 45 (1) (2023) 87–110.