HAL
open science

GREYC@FinTOC-2022: Handling Document Layout and Structure in Native PDF Bundle of Documents

Emmanuel Giguet, Nadine Lucas

To cite this version:

Emmanuel Giguet, Nadine Lucas. GREYC@FinTOC-2022: Handling Document Layout and Structure in Native PDF Bundle of Documents. 4th Financial Narrative Processing Workshop (FNP 2022), Jun 2022, Marseille, France. pp.100-104. hal-03741656

HAL Id: hal-03741656

https://hal.science/hal-03741656v1

Submitted on 1 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Proceedings of the 4th Financial Narrative Processing Workshop @ LREC 2022, pages 100–104
Marseille, 24 June 2022
© European Language Resources Association (ELRA), licensed under CC-BY-NC-4.0

GREYC@FinTOC-2022: Handling Document Layout and Structure in Native PDF Bundle of Documents

Emmanuel Giguet, Nadine Lucas
Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC
14000 Caen, France
emmanuel.giguet@cnrs.fr, nadine.lucas@unicaen.fr

Abstract

In this paper, we present our contribution to the FinTOC-2022 Shared Task "Financial Document Structure Extraction". We participated in the three tracks dedicated to English, French and Spanish document processing. Our main contribution consists in considering financial prospectus as a bundle of documents, i.e., a set of merged documents, each with their own layout and structure. Therefore, Document Layout and Structure Analysis (DLSA) first starts with the boundary detection of each document using general layout features. Then, the process applies inside each single document, taking advantage of the local properties. DLSA is achieved considering simultaneously text content, vectorial shapes and images embedded in the native PDF document. For the Title Detection task in English and French, we observed a significant improvement of the F-measures for Title Detection compared with those obtained during our previous participation.

Keywords: Document Structure Extraction, Document Layout Analysis

1. Introduction

FinTOC-2022 comes as part of a series of shared tasks dedicated to financial document processing. It follows previous editions of FinTOC relating to Financial Document Structure Extraction, in particular FinTOC-2021 organized by (El Maarouf et al., 2021).

In the FinTOC competitions, two tasks are proposed by the organizers: *Title Detection* and *Table of Content Structure Extraction*, aiming at identifying and organizing the headers of the document according to its hierarchical structure. In this edition, three languages are considered: English, French and Spanish.

Table of Content (ToC) extraction can be considered in two complementary ways: (i) extracting a logical structure that has been explicitly marked in a ToC and map it to the titles present in the document, and (ii) creating a ToC from scratch when no such section is present in the document. In the latter case, the process consists in detecting the titles and computing their level in the hierarchy of titles.

Keeping to the tradition of our earlier work, we choose to enrich an end-to-end pipeline aiming at fully structuring documents from the native PDF files. Our intention is to build a language independent solution and a domain independent system. Our motivation is to better understand abstract structuring processes where *contrast* and *positioning* are key features. Therefore, titles and tables of content should be derivative outputs of our system.

In this work, we choose to pay attention to the global structure of the documents, and to highlight how the global structure might help to improve the analysis of local inner structures. Therefore, for the first time, we consider financial prospectus as a *bundle of documents*, i.e., a set of merged documents, each with their own layout and structure. And that's what they are.

In our approach, Document Layout and Structure Analysis (DLSA) first starts with the boundary determination of each document of the bundle using general layout features. Then, the process applies inside each individual document, taking advantage of its local properties. Our preliminary work shows that computing background properties on the whole document, or inside a sliding window, is a non-sense and may lead to analytical errors when processing bundle documents. *Background style* and more broadly any deduction related to the document should be computed within the document space.

The paper is organized as follows. In section 2. we briefly present the datasets. In section 3. we present financial prospectuses as document bundles and we analyse the document structure and layout of each part. In Section 4., we describe our structuring approach. In Section 5. we present a discussion about our results, and we draw some perspectives for future work.

2. Datasets

The training set and test set of the shared tasks are composed of financial documents written in French, English and Spanish. The documents are distributed as native PDF documents. The French and English sets contain financial prospectuses. The Spanish set contain financial information reports.

lang.	train set	test set
English	79	10
French	81	10
Spanish	80	10

Table 1: Document count in datasets

	train set			test set		
lang.	min	max	avg.	min	max	avg.
en	3	405	77	66	136	102
fr	2	155	26	12	28	20
sp	15	318	118	92	444	198

Table 2: Page count in datasets

3. Document Structure and Layout of Prospectus Document Bundles

The document layout and the document structure of the financial prospectuses are interesting to observe. Most of them are indeed bundles of documents: each individual document of the bundle has its own structure, its own layout, including headers, footers, and even sometimes page numbering counters. The bundles may result of a simple merge of PDF documents.

Concerning the bundle structure, three main parts should be considered: (1) the Key Investor Information Document (KIID), (2) the prospectus, (3) the regulatory terms.

3.1. The Key Investor Information Document

The first part of the bundle is a two-page factual document which provides key information to the investor. It is also called Key Investor Information Document (KIID). Its structure is guided and supervised by authorities. In some bundles, the first part is made of a series of concatenated KIIDs.

The KIID tends to look like a commercial document – although it is not one – with an attractive presentation, a coherent color palette for text attributes, text backgrounds, logos, short and understandable texts and figures. The mandatory structure and the synthetic nature of the document lead to high text density with small font sizes, small text interline, small vertical spaces between document objects.

3.2. The prospectus

The second part of the bundle, also called prospectus, is a detailed written presentation combining descriptions, written texts, and tables. The text structure is more free but there are similarities from one document to another. Inconsistencies in the numbering of list items or titles can be observed, as stated by (Bourez, 2021).

The prospectus has a more straightforward presentation that can be observed in technical reports. The text structure is rich and quite complex with multi-level headings, combined to embedded lists of several types: numbered list, bulleted list, checkbox list, description list, tabular list. The tables may also be complex with multiple page-spanning layout.

3.3. The regulatory terms

The third part is a regulatory section which is organized in titles and articles. The text has a formal legal style. The regulatory terms has a traditional sober and rigorous layout, with often centered titles, and independent page numbering counters for titles and articles.

Most of the prospectuses are published without a table of content (ToC), which means you can not rely on a ToC detection and parsing module to achieve the tasks. Some prospectuses may include a cover page or may be complemented by appendices. All these characteristics make the challenge all the more interesting.

4. Method

The experiment is conducted on native PDF documents. In line with the work presented in FinSBD-2 task by (Giguet and Lejeune, 2021a) and FinTOC-2021 (Giguet and Lejeune, 2021b), we choose to implement an end-to-end pipeline from the PDF file itself to a fully structured document. This approach allows to control the entire process. Titles and Table of Contents that we generate for the shared tasks are derivative outputs of the system.

4.1. Document Preprocessing

The document content is extracted using the `pdf2xml` command (Déjean, 2007). Three useful types of content are extracted from the document: text, vectorial shapes, and images.

Text Preprocessing

`Pdf2xml` introduces the concepts of token, line and block, as three computational text units. We choose to only rely on the "token" unit. In practice, most output tokens correspond to words or numbers but they can also correspond to a concatenation of several interpretable units or to a breakdown of an interpretable unit, depending on character spacing. We choose to redefine our own "line" unit in order to better control the coherence of our hierarchy of graphical units. We abandon the concept of "block" whose empirical foundations are too weak.

Vectorial Shapes Preprocessing

Using `pdf2xml` allows to rely on vectorial information during document analysis. Text background, framed content, underline text, table grid are crucial information that contributes to sense making. They simplify the reader's task, and contribute in a positive way to automatic document analysis.

Most vectorial shapes are basic closed path, mostly rectangles. Graphical lines or graphical points do not exist: lines as well as points are rectangles interpreted by the cognitive skills of the reader as lines or points. In order to use vectorial information in document analysis, a preprocessing stage builds composite vectorial shapes and interprets them as background colors or borders. This preprocessing component returns shapes that are used by our system to detect framed content, table grids, and text background. It improves the detection of titles which are presented as framed text and it avoids considering table headers as titles.

Images Preprocessing

`Pdf2xml` extracts images from the pdf. They may be used in different context such as logos in the title page, figures in the document body. An other interesting feature lies in the fact that certain character symbols are serialized as images, in particular specific item bullets such as arrows or checkboxes. They are indistinguishable from a standard symbol character by the human eye.

We choose to handle images as traditional symbol characters, so that they can be exploited by the structuration process, in particular by the list identification module. Identical images are grouped, and a virtual token containing a fake character glyph is created. The bounding box attributes are associated to the token and a fake font name is set. These virtual tokens are inserted at the right location by the line builder module thanks to the character x-y coordinates. This technique significantly improves the detection of list items and, as a consequence, the recognition of the global document structure.

4.2. Document Layout and Structure Analysis

Document Delimitation in the Bundle

As stated above, the delimitation of individual documents inside a bundle is the main contribution of our work for this edition. This problem has been examined in specific studies (Taghva and Cartright, 2009). In previous work we have also faced quite similar problems: the delimitation of parts and chapters in OCRed books (Giguet and Lucas, 2010).

The experimentations we carried out reveal that (1) frequency of hashes of font family concatenated with font size, (2) colors palettes, (3) text content and position in headers and footers, (4) page number sequence and position in headers and footers are interesting features to compute document boundaries.

In this prototype, we rely on (1) text content and position in headers and footers, and (2) page numbers sequence and position in headers and footers to detect a new individual document. Due to time constraint, we did not include information related to font attributes and colors.

The process detects inconsistencies in the sequence of headers and footers in order to split the bundle: appearance of new content, disappearance of content, change of position of the content, break or reset in page number series.

Detecting Header and Footer Areas

Header and footer area boundaries are computed from the repetition of similar tokens located at similar positions at the top and at the bottom of contiguous pages (Déjean and Meunier, 2006). We take into account possible odd and even page layouts.

Header and footer pattern is inferred from a set of a maximum of twenty contiguous pages. While this number is arbitrary, we consider it is enough to consider the pattern reliable in case of odd and even layouts. Once the pattern is inferred, we check if it is still applicable on the following pages. If not, a document limit is detected, a new document is created, and the header and footer pattern induction process is launched.

A special process detects page numbering and computes the shift between the PDF page numbering and the document page numbering. Page numbering is computed from the repetition of tokens containing decimals and located at similar positions at the top or at the bottom of contiguous pages. These tokens are taken into account when computing header and footer boundaries.

Page Layout Analysis

Page Layout Analysis (PLA) aims at recognizing and labeling content areas in a page, e.g., text regions, tables, figures, lists, headers, footers. It is the subject of abundant research and articles (Antonacopoulos et al., 2009).

While PLA is often achieved at page scope and aims at bounding content regions, we have taken a model-driven approach at document scope. We try to directly infer Page Layout Models from the whole document and we then try to instantiate them on pages.

Our Page Layout Model (PLM) is hierarchical and contains 2 positions at top-level: the *margin area* and the *main content area*. The *margin area* contains two particular position, the *header area* located at the top, and the *footer area* located at the bottom. *Aside areas* may contain particular data such as vertically-oriented text. The *main content area* contains *column areas* containing text, figures or tables. *Floating areas* are defined to receive content external to column area, such as large figures, tables or framed texts.

The positions that we try to fill at document scope are header, footer and main columns. First, pages are grouped depending on their size and orientation (i.e., portrait or landscape). Then header area and footer area are detected. Column areas are in the model but due to time constraints, the detection module is not fully implemented in this prototype yet.

Detecting the Table of Contents

The TOC is located in the first pages of the document. It can spread over a limited number of contiguous pages. One formal property is common to all TOCs: the page numbers are right-aligned and form an increasing sequence of integers.

These characteristics are fully exploited in the core of our TOC identification process: we consider the pages of the first third of the document as a search space. Then, we select the first right-aligned sequence of lines ending by an integer and that may spread over contiguous pages.

Linking TOC Entries and Headers

Linking Table of Content Entries to main content is one of the most important process when structuring a document (Déjean and Meunier, 2010). Computing successfully such relations demonstrates the reliability

of header detection and permits to set hyperlinks from toc entries to document headers.

Once TOC is detected, each TOC Entry is linked to its corresponding page number in the document. This page number is converted to the PDF page number thanks to the page shift (see section 4.2.). Then header is searched in the related PDF page. When found, the corresponding line is categorized as header.

Table Detection

Table detection to exclude table content from the main text stream. It allows to exclude tables when searching for list items, sentences or titles.

The table detection module analyzes the PDF vectorial shapes. Our algorithm builds table grids from adjacent framed table cells. The framed table cells are built from vectorial shapes that may represent cell borders. The table grid is defined by the graph of adjacent framed table cells.

Unordered List Structure Induction

Unordered lists are also called *bulleted lists* since the list items are supposed to be marked with bullets. Unordered lists may spread over multiple pages.

Unordered list items are searched at page scope. The typographical symbols (glyphs) used to introduce items are not predefined. We infer the symbol by identifying multiple left-aligned lines introduced by the same single-character token. In this way, the algorithm captures various bullet symbols such as squares, white bullets... Alphabetical or decimal characters are rejected as possible bullet style type. Images of character symbols are transparently handled thanks to virtual tokens created during the preprocessing stage.

The aim of the algorithm is to identify PDF lines which corresponds to new bulleted list item (i.e., list item leading lines). The objective is not to bound list items which cover multiple lines. Indeed, the end of list items are computed while computing paragraph structures: a list item ends when the next list item starts (i.e., same bullet symbol, same indentation) or when less indented text objects starts.

Ordered List Structure Induction in PDF Documents

Ordered list items are searched at document scope. We first select numbered lines thanks to a set of regular expressions, and we analyse each numbering prefix as a tuple $\langle P, S, I, C \rangle$ where *P* refers to the numbering pattern (string), *S* refers to the numbering style type (single character), *I* refers to the numbering count written in numbering style type (single character), and *C* refers to the decimal value of the numbering count (integer).

The numbering style types are defined as follows: Decimal (D), Lower-Latin (L), Upper-Latin (M), Lower-Greek (G) Upper-Greek (H), Lower-Roman (R), Upper-Roman (S), Lower-Latin OR Lower-Roman (?), Upper-Latin OR Upper-Roman (!).

To illustrate, the line “A.2.c) My Header" is analysed as ⟨ A.2.L), L, c, 3 ⟩.

Lines are grouped in clusters sharing the same numbering pattern. A disambiguation process assigns an unambiguous style type to ambiguous lines. The underlying strategy is to complement unambiguous yet incomplete series in order to build coherent, ordered series.

Paragraph Structure Induction

The aim of paragraph structure induction is to infer paragraph models that are later used to detect paragraph instances. The underlying idea to automatically infer the settings of paragraph styles.

Paragraphs are complex objects: a canonical paragraph is made of a leading line, multiple body lines and a trailing line. The leading line can have positive or negative indentation. In context, paragraphs may be visually separated from other objects thanks to above spacing and below spacing.

In order to build paragraph models, we first identify reliable paragraph bodies: sequences of three or more lines with same line spacing and compatible left and right coordinates. Then, leading lines and trailing lines are identified considering same line spacing, compatible left and/or right coordinates (to detect left and right alignments), same style. Paragraph lines are categorized as follows: L for leading line, B for body lines, T for trailing line. Header lines are categorized H. Other lines are categorized as ? for undefined.

In order to fill paragraph models, paragraph settings are derived from the reliable paragraphs that are detected. When derived, leading lines of unordered and ordered list items are considered to create list item models.

Once paragraph models and list item models are built, the models are used to detect less reliable paragraphs and list items (i.e., containing less than three body lines). Compatible models are applied and lines are categorized L, B (if exists) or T (if exists). Remaining undefined lines are categorized considering line-spacing.

5. Results and discussion

The document-wise approach we presented was evaluated on both tasks at FinTOC 2022 : *Title Detection* and *Table of Content extraction*. However, due to lack of time, we only produced results for the Title Detection task. Results for the second task are not relevant.

In table 3 and 4 we present the results we obtained respectively for *Title Detection* at FinTOC 2021 and FinTOC 2022.

	Prec	Rec	F1
French	0.842	0.485	0.606
English	0.913	0.338	0.465

Table 3: Results for Title Detection at FinTOC 2021

	Prec	Rec	F1
French	0.766	0.610	0.671
English	0.812	0.794	0.793
Spanish	0.293	0.507	0.206

Table 4: Results for Title Detection at FinTOC 2022

These results are encouraging and show a significant improvment of the F-measure on French and English test sets. These improvements are partly due to the handling of document bundles: it permits a better computation of background style and constrast among styles. The very low precision for the Spanish test set is due to a specific table layout not covered by our table detection system. The improvement of this module should solve the problem.

The rationale of our method is to have an end-to-end pipeline from the PDF file itself to a fully structured document. The approach is systemic: any improvment in a particular module benefits to all other modules, and more broadly to the global system. The advantage of this approach is to solve every problem where it has to be solved. The drawback, for such a challenge, is that we have to model and implement all the document objects recognition modules and their interaction to obtain competitive results.

Today, our system includes an individual document delimitation module to handle bundles of document, a basic page layout analysis module, a header/footer detection system, a basic table detection module, a list detection module and a paragraph induction module. They all seem to contribute in a good way to the document structuration process. They all have to be improved. New components should be developed, in particular, a graph detection module. Still, we believe there is a great interest in representing a fairly unusual but ambitious way to deal with the document structure as a whole.

6. Bibliographical References

Antonacopoulos, A., Bridson, D., Papadopoulos, C., and Pletschacher, S. (2009). A realistic dataset for performance evaluation of document layout analysis. In *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pages 296–300, 01.

Bourez, C. (2021). FINTOC 2021 - document structure understanding. In *Proceedings of the 3rd Financial Narrative Processing Workshop*, pages 89–93, Lancaster, United Kingdom, 15-16 September. Association for Computational Linguistics.

Déjean, H. and Meunier, J.-L. (2006). A system for converting pdf documents into structured xml format. In Horst Bunke et al., editors, *Document Analysis Systems VII*, pages 129–140, Berlin, Heidelberg. Springer Berlin Heidelberg.

Déjean, H. and Meunier, J.-L. (2010). Reflections on the inex structure extraction competition. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, page 301–308, New York, NY, USA. Association for Computing Machinery.

Déjean, H., (2007). *pdf2xml open source software*. Last access on July 31, 2019.

El Maarouf, I., Kang, J., Aitazzi, A., Bellato, S., Gan, M., and El-Haj, M. (2021). The Financial Document Structure Extraction Shared Task (FinToc 2021). In *The Third Financial Narrative Processing Workshop (FNP 2021)*, Lancaster, UK.

Giguet, E. and Lejeune, G. (2021a). Daniel at the FinSBD-2 task : Extracting Lists and Sentences from PDF Documents: a model-driven end-to-end approach to PDF document analysis. In *Second Workshop on Financial Technology and Natural Language Processing in conjunction with IJCAI-PRICAI 2020*, Proceedings of the Second Workshop on Financial Technology and Natural Language Processing, pages 67–74, Kyoto, Japan, January.

Giguet, E. and Lejeune, G. (2021b). Daniel@FinTOC-2021: Taking advantage of images and vectorial shapes in native PDF document analysis. In *Proceedings of the 3rd Financial Narrative Processing Workshop*, pages 70–74, Lancaster, United Kingdom, 15-16 September. Association for Computational Linguistics.

Giguet, E. and Lucas, N. (2010). The book structure extraction competition with the resurgence software for part and chapter detection at caen university. In *Comparative Evaluation of Focused Retrieval - 9th Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2010), Revised Selected Papers*, pages 128–139.

Taghva, K. and Cartright, M.-A. (2009). Document boundary determination using structural and lexical analysis. In Kathrin Berkner et al., editors, *Document Recognition and Retrieval XVI*, volume 7247, pages 22 – 26. International Society for Optics and Photonics, SPIE.