

Automating the Extraction of Structured Data from Large Newspaper Corpora using Layout Analysis, OCR and Generative AI

NIKOS KONTONASIOS, Information Systems Laboratory, FORTH-ICS and Computer Science Department, University of Crete, Greece

YANNIS TZITZIKAS, Information Systems Laboratory, FORTH-ICS and Computer Science Department, University of Crete, Greece

PAVLOS FAFALIOS, Information Systems Laboratory, FORTH-ICS and School of Production Engineering and Management, Technical University of Crete, Greece

Historical newspapers are invaluable resources that capture the cultural, social, economic, and political dimensions of their time, providing unique insights into past events and daily life. However, the potential of these archives remains underutilised due to the limitations of traditional research methods, which rely on laborious manual examination of individual pages or articles. Digital archives have begun to address this issue by enhancing accessibility, but the lack of effective tools for parsing and analysing the *contents* still restricts their utility for large-scale historical research. Optical Character Recognition (OCR) technology plays a foundational role in converting scanned images into searchable text. However, OCR alone cannot adequately handle the complexities of historical newspapers, which often feature degraded image quality, inconsistent typography, and intricate layouts. In parallel, the effective segmentation and ordering of newspaper contents is essential for generating well-structured and logically ordered data from specific newspaper sections. This paper focuses on the development of a pipeline that addresses these challenges, using the daily newspaper *Le Sémaphore de Marseille* as a case study (35,703 issues covering the period 1827 to 1944). The proposed system integrates layout analysis, OCR, and Generative AI-driven information extraction techniques to extract specific data elements from scanned images, namely ship arrivals data, and to store them in a machine-readable format for further analysis. This well-structured output enables historians to analyse historical trade patterns, economic trends, and regional interactions more efficiently and at a much larger scale. An evaluation of the key pipeline components demonstrates the effectiveness of the system, achieving an F1-score of 96% in paragraph segmentation and information extraction, while also highlighting the need for further improvement in the critical OCR component.

CCS Concepts: • **Information systems** → **Information extraction**; • **Computing methodologies** → **Information extraction**; • **Applied computing** → **Document analysis**.

Additional Key Words and Phrases: Data Extraction, Maritime History, Historical Newspapers, Layout Analysis, Text Block Ordering, OCR, Paragraph Segmentation, Generative AI, ChatGPT

ACM Reference Format:

Nikos Kontonasios, Yannis Tzitzikas, and Pavlos Fafalios. 2026. Automating the Extraction of Structured Data from Large Newspaper Corpora using Layout Analysis, OCR and Generative AI. *ACM J. Comput. Cult. Herit.* 1, 1 (January 2026), 29 pages. <https://doi.org/10.1145/3800678>

Authors' Contact Information: Nikos Kontonasios, nkonto@ics.forth.gr, Information Systems Laboratory, FORTH-ICS and Computer Science Department, University of Crete, Heraklion, Crete, Greece; Yannis Tzitzikas, tzitzik@ics.forth.gr, Information Systems Laboratory, FORTH-ICS and Computer Science Department, University of Crete, Heraklion, Crete, Greece; Pavlos Fafalios, fafalios@ics.forth.gr, Information Systems Laboratory, FORTH-ICS and School of Production Engineering and Management, Technical University of Crete, Chania, Crete, Greece.



This work is licensed under a Creative Commons Attribution 4.0 International License.

© 2026 Copyright held by the owner/author(s).

ACM 1556-4711/2026/1-ART

<https://doi.org/10.1145/3800678>

1 Introduction

Newspapers, as historical documents, serve as invaluable sources of information, capturing a wide range of events, trends, and societal shifts [7, 25, 30]. They document not only political and economic changes, but also cultural, social and technological developments. For historians, newspapers constitute rich and multifaceted resources, offering unique insights into the daily lives of people, their concerns, and the ways in which they perceived and responded to contemporary events. Through reports, personal stories and advertisements, newspapers reflect the historical context in which they were published and provide a valuable window into the past.

Despite their significance, the full potential of newspaper archives remains largely untapped due to limitations inherent in traditional research methods. Large collections of historical newspapers are often distributed across multiple institutions, and accessing relevant information has traditionally required the manual examination of individual articles or entire pages. As these archives continue to grow, the challenge of extracting structured, searchable and reusable data becomes increasingly pressing. Although digital archives have significantly improved access to historical newspapers [6], the absence of effective tools for parsing and analysing their content means that much of the potential of these resources remains unrealised.

A central objective in historical research is the conversion of physical, often degraded, newspapers into machine-readable formats. This process, commonly referred to as *newspaper digitisation*, involves more than document scanning. It requires the transformation of page images into structured data that can be searched, analysed and interpreted [16]. Optical Character Recognition (OCR) [17] plays a key role in this process by converting textual content within images into machine-readable characters. However, although modern OCR systems perform well on clean and homogeneous documents, they often face challenges when applied to historical newspapers due to issues such as degraded image quality, inconsistent typography, and complex or unconventional layouts [18].

For these reasons, OCR alone is insufficient to address the full complexity of newspaper digitisation. Historical newspapers typically exhibit intricate layouts with multi-column structures, overlapping elements, and irregular text arrangements. This creates a significant *information segmentation* problem, namely the task of identifying and separating page components, such as articles, advertisements, and tables, into coherent and logically ordered units [5, 8]. Since OCR systems rely on properly ordered input, effective layout analysis and segmentation are critical steps in the digitisation pipeline. These challenges are particularly pronounced in historical newspapers whose layouts may vary not only between newspapers but also across individual issues of the same newspaper. Furthermore, the physical condition of archival material often introduces additional difficulties, including skewed pages, fading ink and background noise. Therefore, accurately segmenting newspaper pages into their individual sections while preserving their logical reading order remains an important yet non-trivial task.

This paper focuses on the extraction of well-structured data from historical newspaper archives in order to support large-scale historical research. By addressing the segmentation problem inherent in newspaper pages, we aim to develop a robust processing pipeline capable of transforming raw page images into structured, machine-readable data suitable for scholarly analysis. As a case study, we examine the extraction of *ship arrivals* data, such as ship names, names of captains, ports of origin, and cargo, from the regional newspaper *Le Sémaphore de Marseille* (1827–1944). The daily issues of this newspaper are preserved as scanned images by the National Library of France (BnF)¹. This information appears in a dedicated “ARRIVALS” section in almost all newspaper issues, offering valuable insights into historical trade routes, economic activity, and regional interactions. For maritime historians, such data can shed light on patterns of global commerce and local economies that would otherwise remain difficult to study systematically. The structured output of the proposed pipeline, provided in the form of machine-readable CSV files, enables large-scale analysis and facilitates the investigation of trends over time and across regions.

¹<https://www.retronews.fr/titre-de-presse/semaphore-de-marseille>

Figure 1 illustrates the proposed pipeline. The pipeline combines layout analysis with AI-driven information extraction techniques and is specifically tailored to the characteristics of *Le Sémaphore de Marseille*. While this newspaper provides an ideal test case, the methods presented do not necessarily generalise directly to other newspapers with substantially different layouts or characteristics. Focusing on a specific newspaper allows for a detailed examination of segmentation and information extraction challenges and highlights the importance of customisation when applying such techniques to different types of archival collections. At the same time, the pipeline is designed in a modular manner, enabling adaptation to other document types and data extraction requirements. In comparison to related work, such as [32] and [21], the key novel feature of the proposed approach lies in its ability to extract well-structured data from targeted newspaper sections based on specific historical research needs, thereby surpassing the conventional approach of plain text extraction.

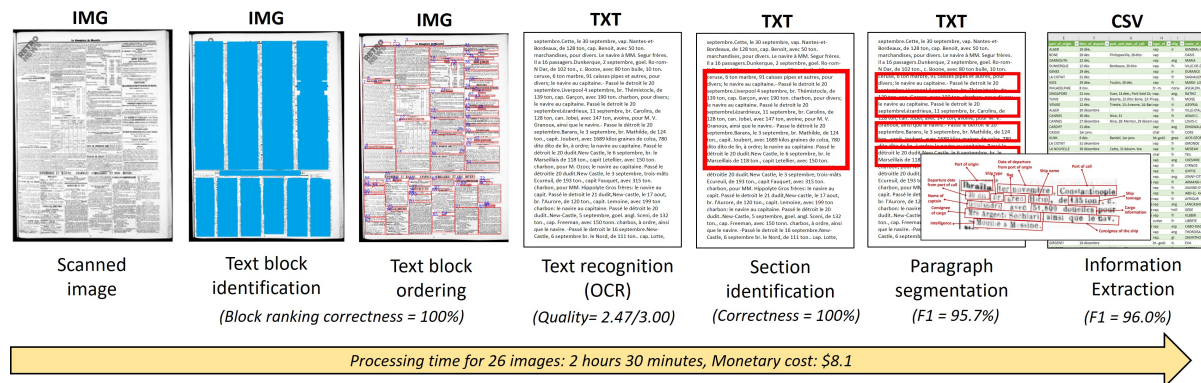


Fig. 1. Overview of the information extraction process.

To assess the reliability and effectiveness of the proposed pipeline, an evaluation of its components was conducted. This evaluation provides insight into the overall performance of the pipeline, highlighting both its strengths and areas that require improvement. In summary, the pipeline achieves robust accuracy in text block ordering, acceptable OCR performance with an average score of 82%, and high performance in paragraph segmentation and information extraction, with F1-scores of 0.957 and 0.960 respectively. These results demonstrate the potential of advanced computational methods to support historical research while also illustrating the challenges posed by the imperfections inherent in digitised historical material.

In summary, this paper makes the following key contributions:

- it introduces a modular pipeline for the extraction of well-structured data from targeted sections of scanned newspaper pages, detailing the tools used in each processing step;
- it proposes a prompt engineering strategy to guide a generative AI model in extracting specific data elements from noisy OCR text;
- it evaluates the performance of each pipeline component using issues of the newspaper *Le Sémaphore de Marseille* as a test case, guided by concrete historical research requirements.

The remainder of this paper is organised as follows: Section 2 describes the context of this work, defines the problem, and reviews related literature. Section 3 presents and formalises the data extraction pipeline, details its individual components, and discusses aspects of automation and applicability. Section 4 reports the evaluation results, including processing times and the associated monetary costs of the tools used. Finally, Section 5 concludes the paper and discusses directions for future research.

2 Context, Problem Description and Related Work

2.1 The PortADa project and the *Le Séaphore de Marseille* newspaper corpus

The work presented in this paper is situated within the framework of the PortADa project (MSCA Staff Exchanges, No. 101129889)², which is positioned in the field of maritime history. The project focuses on the study of ship entries at the ports of Barcelona, Marseille, Havana, and Buenos Aires from 1850 to 1914, with the aim of generating an open-access database containing thousands of records related to 19th century maritime trade. This comprehensive database is expected to provide valuable insights into the economic, social, geopolitical, and cultural dynamics of the period, thereby laying the foundation for new research in these areas.

Our work specifically addresses the extraction of ship arrival records for the port of Marseille during the period 1850 to 1914. To achieve this, the *Le Séaphore de Marseille* newspaper serves as the primary source. Founded in 1827 by printers Joseph-François Feissat and Pierre Alexandre-Henri Demonchy, it is Marseille's oldest newspaper. Originally encompassing diverse themes such as news, political information, art reviews, exhibition reports and others, it gradually (from the 20th century) specialised in maritime news. From its inception in 1827, to its final issue in 1944, the newspaper diligently reported all ship arrivals in Marseille. Thus, its digitisation offers a rich source for historical analysis of maritime trade during the 19th and early 20th centuries. The newspaper's archives span from 19 December 1827 to 19 August 1944, with 35,703 issues published during this period. The full archives are available to the public through the Bibliothèque Nationale de France (BNF) dedicated newspapers website³. A sample issue is depicted in Figure 2⁴.

2.2 Problem Statement

Given the scanned images of a newspaper issue, the task is to extract well-structured data located in a specific section of the newspaper contents. In the case of *Le Séaphore de Marseille*, the focus is on ship arrivals data, which appears in a section entitled ARRIVÉE (Figure 3). In this section, information about each ship arrival is presented in short paragraphs arranged vertically on the page. This layout is preserved across different columns on the same page, and when a page is filled, the section continues onto the next page. Each ship arrival paragraph contains the following data elements:

- *port of origin*, which usually appears at the beginning of the paragraph
- *date of departure from the port of origin*, which always follows the port name and consists of a number and a month or the word "cour"
- *port and date of call*, referring to any intermediate stops made by the ship during its journey and the corresponding dates of departure
- *ship type*, typically written in lowercase and in abbreviated form, for example "vap", "br", "br-goel", or "bat"
- *ship's flag*, written in lowercase, e.g., "grec" for Greek
- *ship name*, which usually appears after the ship's flag
- *ship tonnage*, which (when present) appears after the ship name and usually begins with "de"
- *name of the captain*, which usually begins with "cap" or "c."
- *cargo information*, which usually begins with "avec" and includes the quantity, unit of measurement, and type of cargo
- *consignee of the cargo*, which (when present) always follows the cargo information and usually begins with "pour" or "p"
- *consignee of the ship*, which typically appears after the cargo information but may also be absent, and often begins with expressions such as "le navire", "nav", or "navire", or may simply consist of a name or entity

²<https://cordis.europa.eu/project/id/101129889>

³<https://www.retronews.fr/titre-de-presse/semaphore-de-marseille>

⁴<https://www.retronews.fr/journal/le-semaphore-de-marseille/6-janvier-1886/1191/3066417/1>

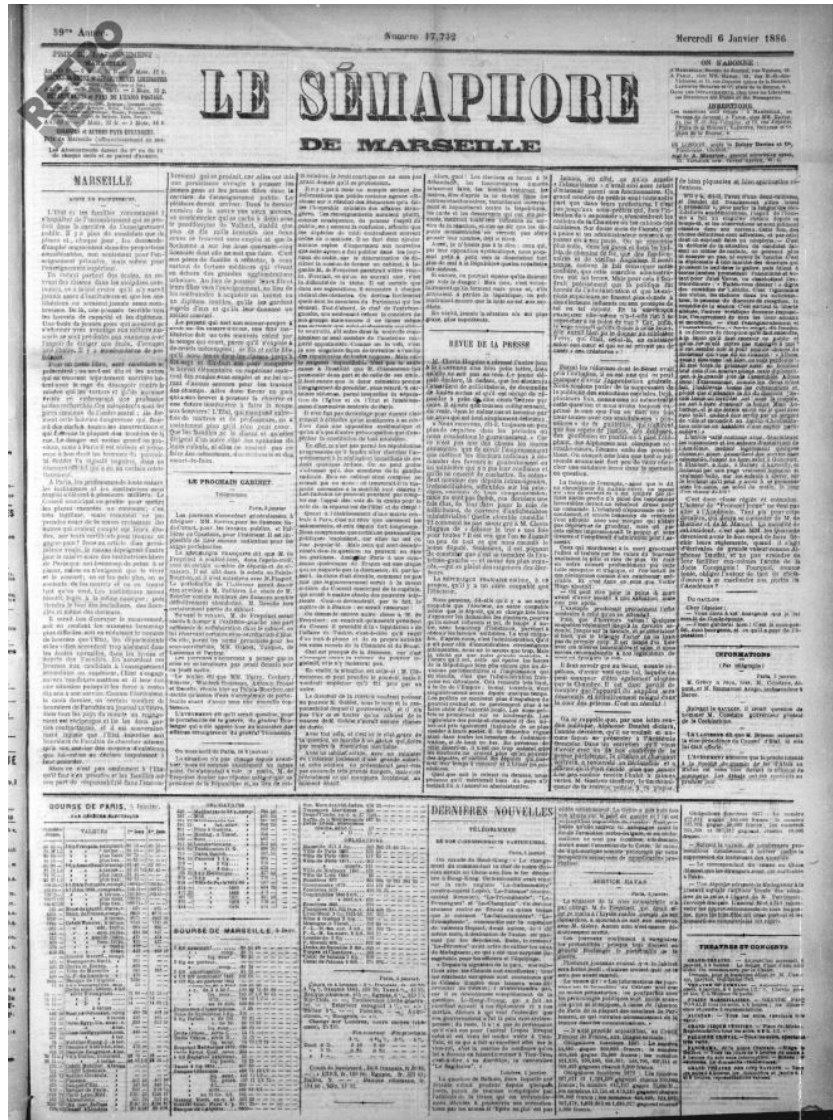


Fig. 2. Le Sémaphore de Marseille, Issue of January 6, 1886 (source: retronews.fr)

- *intelligence*, which includes events or interactions during the voyage, such as passing the Strait of Gibraltar or passenger information, and always appears last when present.

The objective is to extract this information for all ship arrivals and store it in a machine-readable CSV format. Achieving this goal, however, requires addressing several interconnected challenges inherent in historical newspaper digitisation and information extraction.



Fig. 3. Example of section providing information about ship arrivals (source of newspaper image: retronews.fr).

2.3 Key Challenges

The first key challenge is the segmentation of complex newspaper page layouts into distinct text blocks while preserving the correct reading order. Newspapers often feature irregular layouts, overlapping text, and inconsistent column structures, which makes it difficult to discern the logical flow of information. Without addressing this issue, subsequent steps in the data extraction process become unreliable.

The next challenge lies in accurately locating and isolating the specific section of the newspaper that contains ship arrivals information. Newspapers typically include a wide variety of content, such as advertisements, editorials, and general news, which are interspersed in ways that vary from issue to issue. Identifying the relevant section within this diverse content is therefore critical for focused data extraction.

Once the ship arrivals section has been isolated, parsing its text into meaningful units, such as individual paragraphs corresponding to ship arrivals, presents an additional set of difficulties. Historical newspapers often lack clear visual delimiters for paragraphs, which can lead to ambiguities when distinguishing one ship’s information from another. Resolving this issue is essential to ensure that the extracted information is accurate and logically grouped.

Finally, transforming the unstructured text of each paragraph into a structured format requires the identification of specific data elements related to each ship arrival, such as the ship name, captain, and cargo. This task is further complicated by variations in terminology, inconsistent formatting, and domain-specific nuances that are typical of historical maritime records. Failure to address these issues may result in incomplete or incorrect data extraction.

In addition, the poor quality of historical newspaper images, caused by physical deterioration, noise such as fading and smudging, skew, and page curvature, can significantly affect the performance of text recognition. This often results in low-quality text output, thereby exacerbating the challenges described above.

This work aims to address these challenges through a series of integrated solutions, resulting in an automated pipeline that transforms historical newspaper archives into structured datasets suitable for research and analysis.

2.4 Related Work

Effective approaches for addressing the challenges discussed above generally follow a two-step process: document layout analysis and text extraction. Document layout analysis focuses on detecting and segmenting page components such as headlines, columns, and text blocks, while ensuring structural coherence and preserving the correct reading order. Text extraction is then applied to convert the identified text regions into searchable and machine-readable formats.

2.4.1 Document Layout analysis. In the field of document layout analysis, multiple techniques have been developed to address two primary challenges: page segmentation, which involves identifying the structural components of a page, and the extraction of the correct reading order of text blocks. Various approaches address one or both of these aspects, each with specific strengths and limitations depending on the document type and the complexity of its layout.

One such approach is presented by Bandyopadhyay et al. [4], who developed a method focused on segmenting document images, particularly suited to the identification of columns and paragraphs. Their technique employs image smoothing to detect runs of white and black pixels, facilitating the removal of large-font text and image areas that might interfere with segmentation. This approach is especially effective in cases where column separation lacks explicit black lines, a situation often encountered in structured documents without strict bounding borders.

Meunier [23] proposed a distinct solution to address the problem of reading order, particularly in documents where bounding boxes are generated by PDF converters. By adapting the established XY-cut method [24], which was originally developed for block segmentation, Meunier's method focuses on arranging document components into a logical sequence, which is essential for maintaining coherent text flow in complex documents.

A more extensive pipeline for scientific PDF documents was proposed by Klampfl et al. [22]. Their approach utilises PDFBox output to cluster characters into words, words into lines, and lines into blocks, resulting in a segmented document layout. To determine reading order, they incorporate a binary relational model for spatial intervals in both the X and Y dimensions, originally suggested by Aiello et al. [2], together with a custom block neighbourhood algorithm to further refine block ordering.

These traditional methods perform well for PDF-based documents or clean document images with minimal noise, straight column alignment, and limited skew. However, the analysis of historical newspaper images introduces additional challenges, including high noise levels, irregular column layouts, and frequent skew. In such cases, deep learning-based tools, such as LayoutParser by Shen et al. [28] and Arcanum's newspaper segmentation API⁵, have emerged as robust alternatives. These tools are designed to better handle complex layouts by leveraging advanced machine learning techniques to accurately identify and order blocks in diverse and degraded document images, thereby enhancing the reliability of layout analysis in historical and unstructured document contexts.

In summary, while traditional methods offer effective solutions for structured PDF documents or high-quality images, deep learning approaches are increasingly valuable for historical and complex document types, where layout inconsistencies are more prevalent.

2.4.2 Text extraction. After document layout analysis identifies the structure of a document, text extraction techniques such as Optical Character Recognition (OCR) and Handwritten Character Recognition (HCR) are applied to convert detected text regions into searchable digital text. These technologies are essential for extracting both printed and handwritten text from images and support a wide range of applications, including document

⁵<https://www.arcanum.com/en/newspaper-segmentation/>

digitisation and assistive technologies. Over time, OCR and HCR methods have advanced to better handle diverse text formats and degraded image quality.

Several OCR tools are widely used for text extraction, including Tesseract⁶, EasyOCR⁷, and Google Vision⁸. Tesseract, an open-source tool supported by Google [29], has been applied in a variety of contexts, such as serial number extraction from wine labels by Čakić et al. [9] and licence plate information retrieval by Agbemenu et al. [1]. EasyOCR, another open-source tool, has gained attention due to its high accuracy under specific conditions. Salehudin et al. [27] analysed its performance under image degradation, while Chaitra et al. [10] demonstrated its robustness in text recognition for complex scenes. These studies highlight the versatility and reliability of EasyOCR in challenging environments. Google Vision, a cloud-based OCR service, has been employed in various applications, including the conversion of text in medical images to speech for visually impaired users, as demonstrated by Edupuganti et al. [11], and the extraction of text from student theses archived as images in university libraries, as shown by Arief et al. [3].

While general-purpose OCR tools such as Tesseract, EasyOCR, and Google Vision address a wide range of text extraction needs, specialised tools like Transkribus [19] have been optimised for HCR tasks. Transkribus is particularly effective for historical and handwritten documents, as it can better accommodate the variability of handwriting styles.

Beyond OCR and HCR, alternative approaches to text extraction have been explored in specific contexts. For example, Terasawa et al. [31] proposed a method for recognising characters in old Japanese newspapers, in which each character is represented as a feature vector and recognised through sequence matching. This approach assumes accurate character segmentation and can be effective in scenarios where traditional OCR methods perform poorly, such as in the processing of historical documents. Finally, the FAST CAT system [14, 26] supports researchers in manual but collaborative data transcription and curation. This system is well suited to handwritten historical documents with varying writing styles and repetitive content, such as logbooks, civil registers, and similar records, where OCR often fails to produce high-quality text.

Overall, the field of text extraction has diversified to include a wide range of tools and techniques adapted to specific tasks and environments, ranging from the extraction of printed and handwritten text in digital libraries to specialised applications in scene text recognition and assistive technologies.

2.4.3 Integrated Approaches. The accurate transformation of newspapers into structured information requires an integrated approach that combines effective layout analysis with robust text extraction methods. Recognising the close interplay between these two processes, this work proposes a comprehensive solution that integrates them in order to address the specific challenges posed by historical newspaper digitisation. This need has also been acknowledged by other researchers, who have developed a range of techniques to handle the complexities of newspaper layouts, languages, and scripts. Table 1 provides an overview of these works, with additional details discussed in the following paragraphs.

Kanungo et al. [20] developed a system for overlaying extracted text onto newspaper images. Their approach uses the XY-cut algorithm to segment pages into text blocks and a custom OCR tool built using a commercial development kit. Unlike other methods, this system does not prioritise reading-order reconstruction, as its primary objective is to preserve the visual structure of the newspaper by overlaying text onto the original image. While this approach maintains high fidelity to the original layout, it is less suitable for tasks that require coherent reading flow, such as content analysis or large-scale archival search.

Zeni et al. [32] designed a system for extracting and clustering articles from African newspaper archives, using Tesseract for OCR. Their system successfully reconstructs the reading order of articles, but it does not specify

⁶<https://github.com/tesseract-ocr/tesseract>

⁷<https://github.com/JaidedAI/EasyOCR>

⁸<https://cloud.google.com/vision/>

Table 1. Summary of related work on integrated approaches.

Paper	Use Case	Layout Analysis	OCR
Kanungo et al. [20]	USA newspaper archives	XY-cut algorithm	Custom, based on a commercial tool
Zeni et al. [32]	African newspaper archives	Custom	Tesseract
Kaur et al. [21]	Gurumukhi newspaper archives	Articles manually cropped, article column segmentation using white space and pitch, line segmentation using median calculation and projection profile, word and char segmentation using horizontal and vertical histograms	Random Forest classifier

the exact method used for reading-order extraction. Instead, the authors refer to several existing techniques that could be applied for layout analysis, suggesting flexibility in adapting the system to different newspaper structures. While this approach enables accurate organisation of articles within digital archives, the lack of explicit implementation details for reading-order extraction leaves aspects of the method underspecified.

Kaur et al. [21] developed a system for digitising newspapers written in the Gurumukhi script, relying on a manual cropping process to segment each article from the newspaper image. This manual segmentation eliminates the need for automated layout analysis or reading-order determination across newspaper articles, allowing the system to focus primarily on script recognition on a specific article. Each article is further segmented into columns, lines, words, and characters using a variety of techniques, and six feature extraction techniques are applied at the character level. A random forest classifier is then used for character classification. Although this method achieves high accuracy for the Gurumukhi script, its dependence on manual article segmentation limits its scalability and suitability for large-scale or heterogeneous newspaper collections.

2.4.4 Summary and Positioning. Building on these foundational efforts, the proposed work aims to address the limitations identified in previous approaches by developing an *automated* and *modular* solution tailored to the extraction of well-structured information from specific parts of complex newspaper layouts. By integrating advanced layout analysis with AI-driven information extraction techniques, the proposed solution seeks to bridge the gap between preserving document structure and enabling reliable, large-scale data extraction, thereby supporting further analysis and research on digitised historical archives.

3 Approach

We have developed a specialised information extraction pipeline tailored to meet the specific needs posed by historical newspapers. This pipeline is designed to efficiently process the intricate layouts characteristic of historical newspapers and to systematically extract relevant information from digitised newspaper pages. By automating the identification, organisation, and extraction of key data points from newspaper images, the pipeline ensures that information is captured consistently and accurately, while also supporting scalability.

In the following, we first provide an overview and formalisation of the data extraction pipeline (Section 3.1) and then describe each of its stages in detail (Sections 3.2-3.7). Finally, we discuss aspects related to the automation of the pipeline and its applicability to different document formats (Section 3.8).

3.1 Overview and Formalisation of the Data Extraction Pipeline

The data extraction pipeline, depicted in Figure 4, consists of the following stages, each of which contributes crucially to transforming raw images into structured and machine-readable data in tabular form:

- **Text Block Identification:** This initial stage establishes the layout and structure of the newspaper page by separating the text into individual, clearly defined blocks. Identifying these blocks provides insight

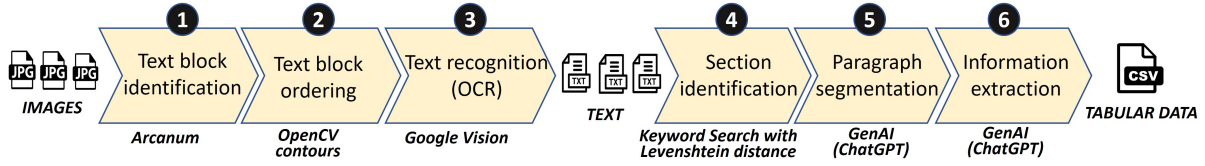


Fig. 4. Data Extraction Pipeline

into the arrangement of columns, headings, and paragraphs, which is essential for accurate reading and interpretation. In addition, isolating smaller text blocks can significantly improve OCR performance, as processing smaller and more homogeneous regions typically yields more accurate recognition results than applying OCR to the entire image. By ensuring precise segmentation, this stage lays the foundation for correctly ordered and readable text in subsequent steps of the pipeline.

- **Text Block Ordering:** The text blocks identified in the previous stage are often not arranged in sequential reading order due to the complex layouts characteristic of historical newspapers. To address this issue, a dedicated software application was developed to arrange the identified blocks into their correct sequence, thereby ensuring logical text flow and coherent reading order.
- **Text Recognition (OCR):** Once the text blocks have been ordered, they are processed using Google Vision AI to convert the visual content into machine-readable text. Although this tool achieves high recognition accuracy, the resulting text may still contain errors, often due to poor image quality.
- **Section Identification:** The OCR output is analysed to identify the target section of interest, in this case the *ship arrivals* section. This stage involves scanning the recognised text for keywords and phrases that typically indicate the beginning and end of the relevant section.
- **Paragraph Segmentation:** The identified *ship arrivals* section is further divided into individual paragraphs, each corresponding to a distinct ship arrival. This segmentation is performed using OpenAI, leveraging its text processing capabilities to detect paragraph boundaries and formatting cues within the text.
- **Information Extraction:** Finally, detailed information about each ship arrival, including the ship name, arrival date, port of origin, cargo, and other relevant information, is extracted from the segmented paragraphs by exploiting OpenAI's natural language processing capabilities.

Pipeline formalisation. Given a set of scanned newspaper page images $I = \{I_1, I_2, \dots, I_n\}$, the objective is to extract a list of ship arrival records R in tabular form. The pipeline operates sequentially on each image I_i . Text block identification takes as input a scanned page image I_i and produces a set of text blocks B_i corresponding to distinct layout components of the page. Text block ordering then takes B_i as input and outputs an ordered set of blocks that follows the natural reading order of the newspaper layout. OCR processes the ordered text blocks and produces textual content T_i . Section identification analyses T_i to locate the target newspaper section containing ship arrival information and outputs the corresponding section text S_i . Paragraph segmentation takes S_i as input and produces a set of paragraphs P_i , each corresponding to an individual ship arrival. Finally, information extraction processes each paragraph $p \in P_i$ and outputs a structured record that is appended to the final result set R . Algorithm 1 formally describes this process in algorithmic form.

3.2 Text Block Identification

The tool used for this task is Arcanum's newspaper segmentation API⁹. It takes as input an image of a newspaper page and produces a JSON file containing a list of "articles" and a representation of the layout components,

⁹<https://www.arcanum.com/en/newspaper-segmentation/>

Algorithm 1 Historical Newspaper Data Extraction**Require:** Scanned newspaper page images $I = \{I_1, I_2, \dots, I_n\}$ **Ensure:** Structured ship arrival records R in tabular form

```

1:  $R \leftarrow []$ 
2: for all  $I_i \in I$  do
3:    $B_i \leftarrow \text{IDENTIFYBLOCKS}(I_i)$  {Text block identification}
4:    $B_i \leftarrow \text{ORDERBLOCKS}(B_i)$  {Text block ordering}
5:    $T_i \leftarrow \text{OCR}(B_i)$  {Text recognition}
6:    $S_i \leftarrow \text{IDENTIFYSECTION}(T_i)$  {Section identification}
7:    $P_i \leftarrow \text{SEGMENTPARAGRAPHS}(S_i)$  {Paragraph segmentation}
8:   for all  $p \in P_i$  do
9:      $r \leftarrow \text{EXTRACTFIELDS}(p)$  {Information extraction}
10:     $R \leftarrow \text{APPEND}(R, r)$  {Add extracted record}
11:   end for
12: end for
13: return  $R$ 

```

namely a mask of the detected blocks. This mask is provided as a two-dimensional NumPy array with the same aspect ratio as the original image, where each pixel is assigned a block identifier. For each article, the JSON output provides: i) a confidence score associated with the article (floating-point value), and ii) a list of the article's blocks. For each block, the following information is included: its bounding box (expressed as a list of values in the order xmin, ymin, xmax, ymax), its label (with possible values including Artifact, Advertising, Picture, Table, Title, Abstract, and Body), a confidence score associated with the assigned label (floating-point value), and a unique block identifier (integer value).

Figure 5 shows an example of the blocks identified by Arcanum on a newspaper issue page. Three primary issues can be observed in the identified blocks:

- (1) Some blocks contain no relevant information, e.g., blocks 1 and 2.
- (2) Some blocks are nested within other blocks, e.g., blocks 19 and 29.
- (3) Some blocks are not arranged in the correct reading order, e.g., blocks 13, 14, 66, 67, and 68.

To address the first issue, the *label* attribute of each block was used. Blocks such as 1 and 2, which are empty, are assigned the label *Artifact*. In the example shown in Figure 5, block 3 is also classified as an artifact. Its removal does not pose any problems, as it contains only the newspaper title, which is not essential information for the extraction task. Empirical testing confirmed that blocks labelled as artifacts consistently lack relevant content, which allows for the safe removal of all such blocks.

To address the second issue, the *bounds* attribute of each block was exploited, as it specifies the coordinates of the block. By analysing these coordinates, any block that was fully contained within another block was identified and removed, thereby preventing duplicate information in the OCR output. The remaining blocks, after addressing both issues, are shown in Figure 6.

The third issue was the most challenging to address and also the most critical, as failure to resolve it would significantly complicate the information extraction process by producing text blocks in an incorrect reading order. To tackle this issue, a three-step approach was developed, which is described in detail in Section 3.3.

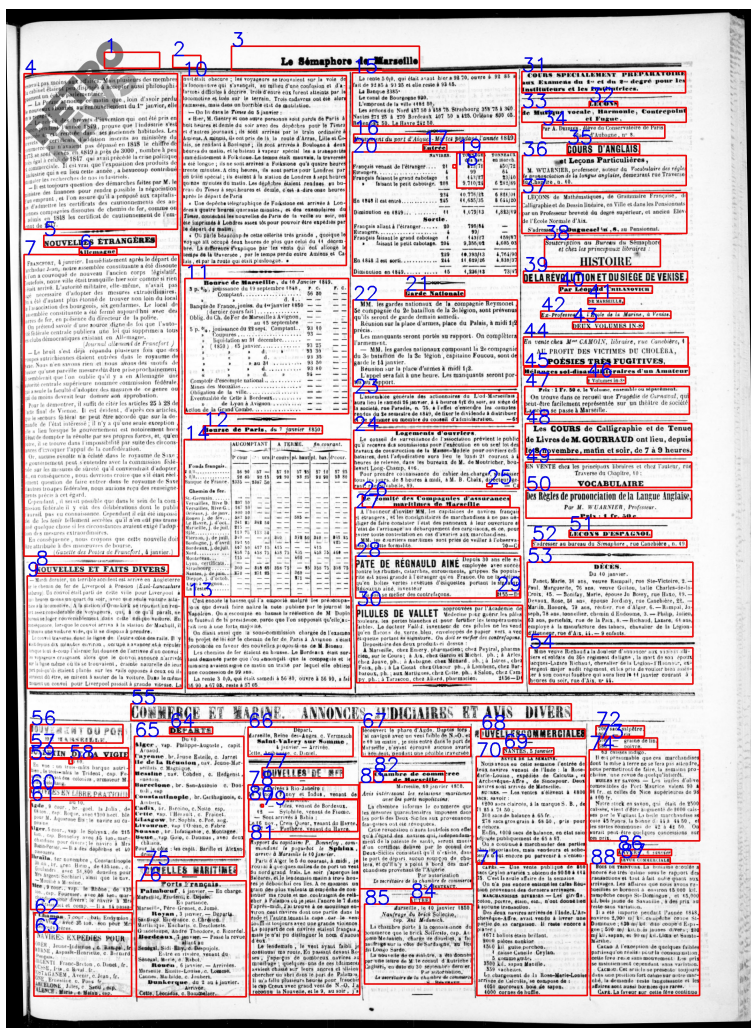


Fig. 5. Example of the blocks identified by Arcanum. Issue of Jan 11, 1850 (source of original newspaper image: retronews.fr).

3.3 Text Block Ordering

While tools such as Arcanum and other layout analysis software can effectively identify and extract text blocks from scanned images, they frequently output these blocks in an incorrect reading order when confronted with intricate page structures. To ensure a logical text flow that accurately reflects natural reading patterns, it is essential to reorder the extracted blocks, thereby addressing the challenges posed by the non-linear arrangements typical of historical newspaper layouts. The three steps implemented to address this problem are described below.

3.3.1 Segment Identification. The initial step in this process involves determining the number of segments present on a newspaper page. For example, as shown in Figure 6, the page is divided into two main segments: a top segment consisting of four columns and a bottom segment consisting of six columns. To identify these

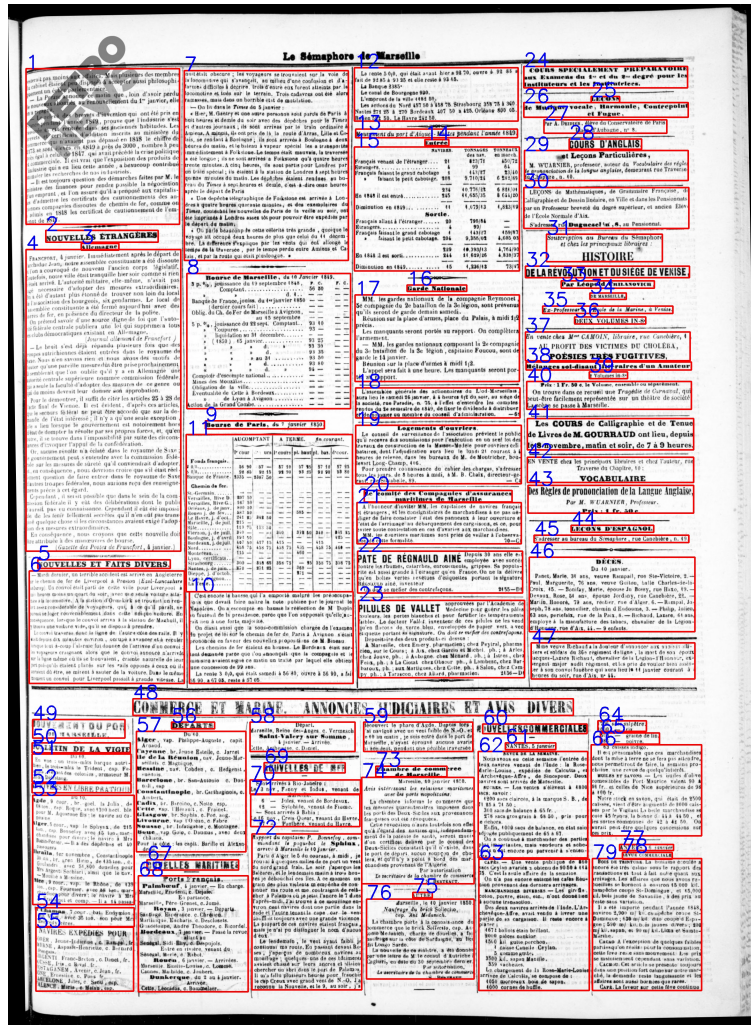


Fig. 6. Example of the blocks after removing the unnecessary blocks. Issue of January 11, 1850 (source of original newspaper image: retronews.fr).

segments, a cyan colour fill is first applied to all text blocks using OpenCV¹⁰, which facilitates segment detection through OpenCV contours¹¹. The blocks are then horizontally stretched so that all blocks belonging to the same segment merge together, thereby highlighting the distinct segments of the page. In addition, the height of the blocks is slightly reduced during this process to prevent unintended merging of segments caused by image skew or page curvature, ensuring accurate segment identification.

By merging the blocks in this manner, the overall structure of the page is simplified, which makes it easier for the contour detection algorithm to distinguish between different sections. Figure 7 illustrates the result after

¹⁰<https://opencv.org/>

¹¹https://docs.opencv.org/4.10.0/d3/d05/tutorial_py_table_of_contents_contours.html

block stretching has been applied to the image shown in Figure 6. In this case, the page is divided into three distinct segments rather than the two initially identified. This occurs because the title "COMMERCE ET MARINE. ANNONCES JUDICIAIRES ET AVIS DIVERS" between the two main segments forms an additional, separate segment.

Although this additional segment may appear unexpected, it is advantageous, as it enables more precise identification of the page layout. This proves particularly useful in the subsequent step of the process.

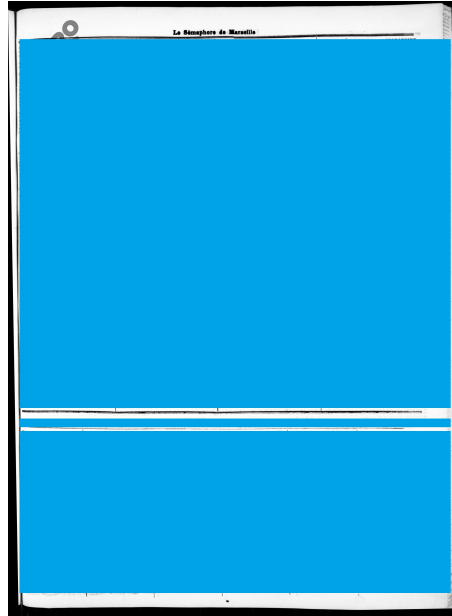


Fig. 7. Example of the horizontal splitting of a newspaper page. Each cyan rectangle is a different segment.

3.3.2 Column Identification. The second step in the process is the identification of columns within each segment of the newspaper page. Referring again to the example shown in Figure 6, the top segment contains four columns, while the bottom segment contains six. To identify these columns, a method similar to that used in the previous step is applied. Initially, a cyan colour fill is applied to each block, after which the blocks are stretched vertically so that all blocks belonging to the same column merge together, thereby making the column structure visible.

An important detail in this process is that each block is slightly reduced along the x-axis before being stretched vertically. This minor reduction in width is essential to prevent unintended merging of adjacent columns. Due to image skew or curvature in the original layout, some blocks may partially extend into neighbouring columns, even if they primarily belong to a single column. Without this adjustment, such distortions could lead to incorrect merging of adjacent columns.

Once the columns in the top segment have been identified using OpenCV contours, the same procedure is applied to the bottom segment. It is also worth noting that the title "COMMERCE ET MARINE. ANNONCES JUDICIAIRES ET AVIS DIVERS", which forms its own distinct segment, does not interfere with the column merging in the other segments when the blocks are stretched vertically. This ensures that the columns within each segment are clearly and accurately identified.

After the shrinking and stretching operations, the columns for each segment become clearly visible. Figures 8a, 8b, and 8c illustrate the columns of the first (top), second (title), and third (bottom) segments, respectively.

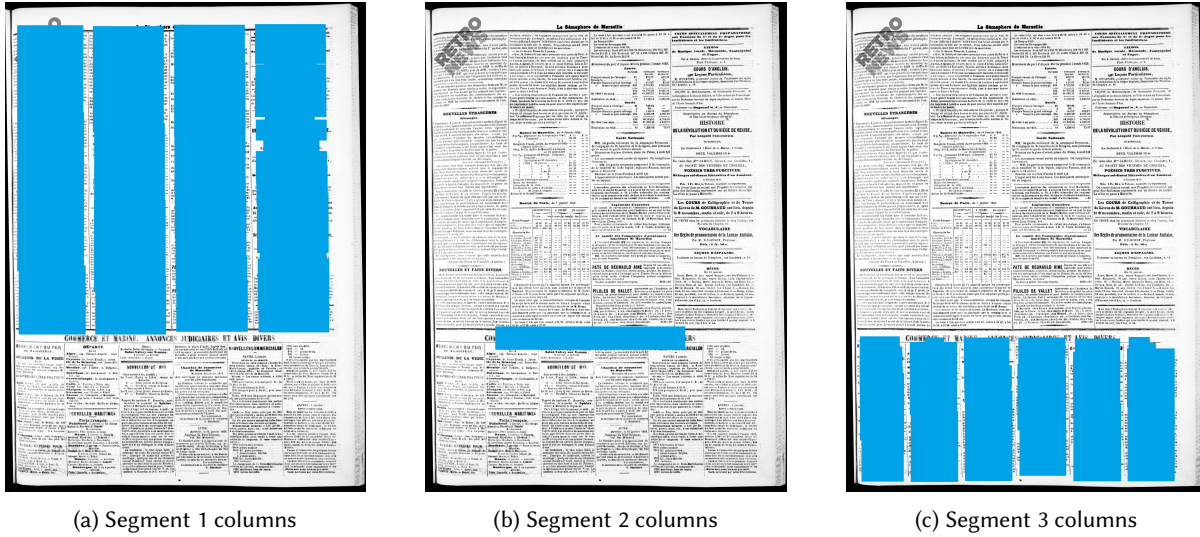


Fig. 8. Example of the columns in each segment (source of original newspaper images: retronews.fr).

In some cases, text blocks span multiple columns, as illustrated in Figure 9a with block 50. Without appropriate adjustment, stretching such blocks could cause adjacent columns to merge, thereby distorting the column structure. To address this issue, the Canny edge detection algorithm¹² is applied in combination with OpenCV contours to detect the vertical lines that separate columns within each newspaper segment, as shown in Figure 10a. Blocks that intersect these vertical lines are then split at the detected boundaries, ensuring that each column remains properly separated. This approach preserves the intended layout and column structure, as demonstrated in Figure 9b, where blocks 86 and 92 are correctly divided.

Another layout challenge in *Le Séaphore de Marseille* arises from the presence of a table in the bottom right corner of certain issues, which disrupts the standard layout. To address this issue, the same approach used for column identification was applied, but with the detection of horizontal lines instead of vertical ones, as illustrated in Figure 10b. To ensure that the detected horizontal line corresponded to the separation between the table and the main content, rather than to a boundary between two horizontal page segments, a width threshold was introduced. This threshold enabled more accurate identification of the correct line. Once the appropriate horizontal line was detected, all blocks located below it were removed, as they belonged to the table and were not relevant to the extraction of ship arrivals information.

3.3.3 Segment and Column Sorting. The third and final step of the process involves organising each segment and its corresponding columns to ensure a coherent reading order. First, the segments are sorted from top to bottom on the page. Next, within each segment, the columns are sorted from left to right. Finally, within each column, the blocks are ordered from top to bottom.

This systematic ordering ensures that the text blocks follow the natural reading sequence of the newspaper layout. The final organised structure of the issue page shown in Figure 6 is illustrated in Figure 11, where the logical flow of segments, columns, and blocks is clearly established. This arrangement is essential for accurate information extraction in the subsequent stages of the pipeline.

¹²https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html



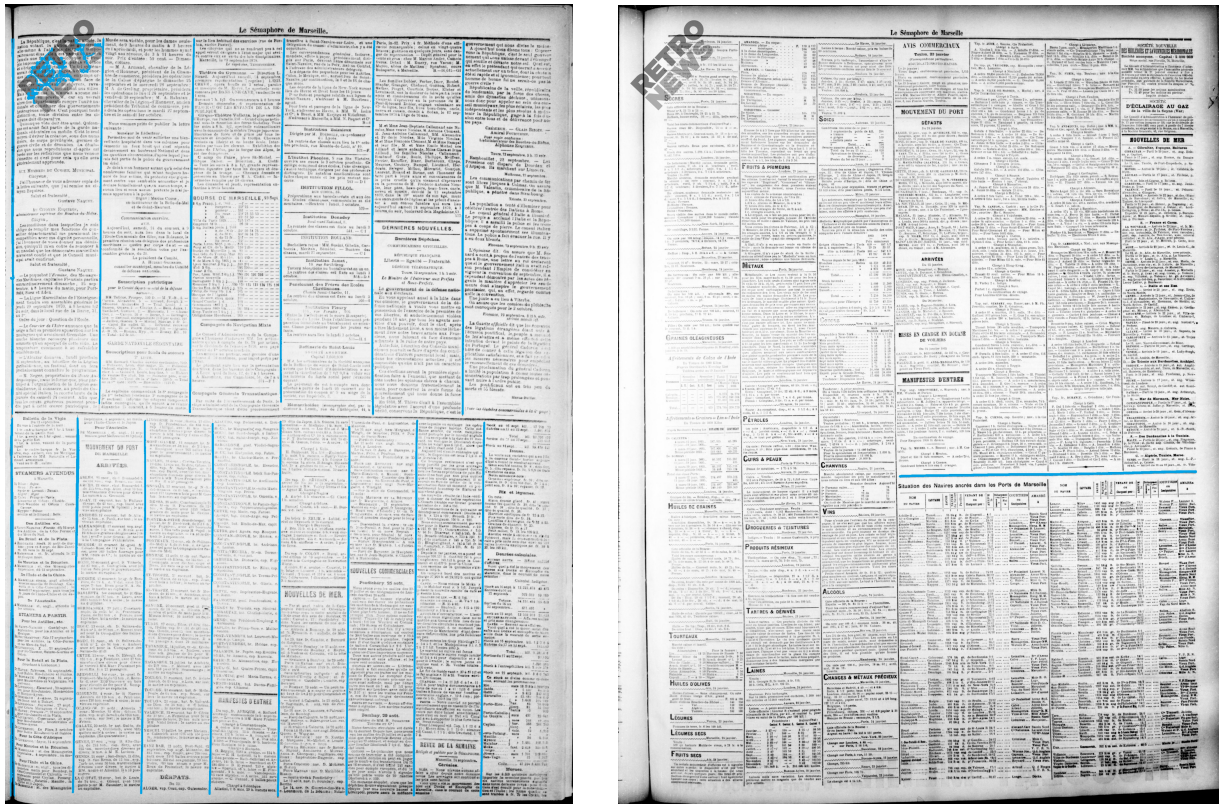
Fig. 9. Example of blocks spanning multiple columns and their adjustment (source of original newspaper images: retronews.fr).

3.4 Optical Character Recognition (OCR)

The OCR process encountered several challenges that affected the accuracy of text recognition on historical newspaper pages. One of the primary issues was the slight inclination or tilt of some pages, which caused the OCR tool to misinterpret text blocks by merging content from multiple columns. This misalignment disrupted the logical flow of the text, as the OCR system struggled to distinguish between adjacent columns. In addition, the curvature of column-separating lines, as illustrated in Figure 12a, further complicated OCR interpretation by distorting the boundaries between columns and, in some cases, incorrectly merging or splitting text blocks.

To address these challenges, several OCR tools were tested to assess their performance on historical newspaper pages. Tools such as Tesseract, EasyOCR, and Google Vision (described in Section 2.4.2) were evaluated with respect to their ability to handle the issues presented by degraded documents, including misalignment, column separation, and noise artefacts. While each tool exhibits particular strengths, the evaluation indicated that Google Vision produced the most accurate results in our case. It demonstrated a stronger ability to cope with tilted pages and noisy, degraded text, and provided more reliable text extraction than the other OCR tools tested.

The importance of the preceding steps of text block identification and ordering became evident during this evaluation, as they allow each text block to be processed independently by the OCR tool. By isolating and correctly ordering individual blocks, structural problems such as column merging and skew-related errors were



(a) Example of vertical lines detected. Issue of Sep 24, 1870. (b) Example of horizontal line detected. Issue of Jan 25, 1895.

Fig. 10. Example of vertical and horizontal lines detected (source of original newspaper images: retronews.fr).

significantly reduced. With each block clearly defined and sequentially arranged, the OCR tool was able to process them more effectively, resulting in more accurate and reliable text extraction.

Another significant challenge was the presence of ink marks, stains, and other noise artefacts on the pages, as shown in Figure 12b. These imperfections obscured characters and words, making accurate recognition difficult. This issue was particularly problematic for numerical data, which are critical but lack the contextual cues that can sometimes assist in interpreting misrecognised letters. For example, distinguishing between "400 barrels of wine" and "100 barrels of wine" based solely on OCR output was often not possible, requiring manual verification against the original document to ensure data accuracy.

To further mitigate these challenges, additional image preprocessing techniques could be applied to improve OCR accuracy. While image deskewing was performed in this work to correct page inclination and align text blocks, other techniques, such as noise reduction and contrast enhancement, could be explored to address issues related to ink marks and stains. The application of such methods would likely improve the OCR tool's ability to cope with the imperfections inherent in historical documents and enhance the overall accuracy of text extraction.

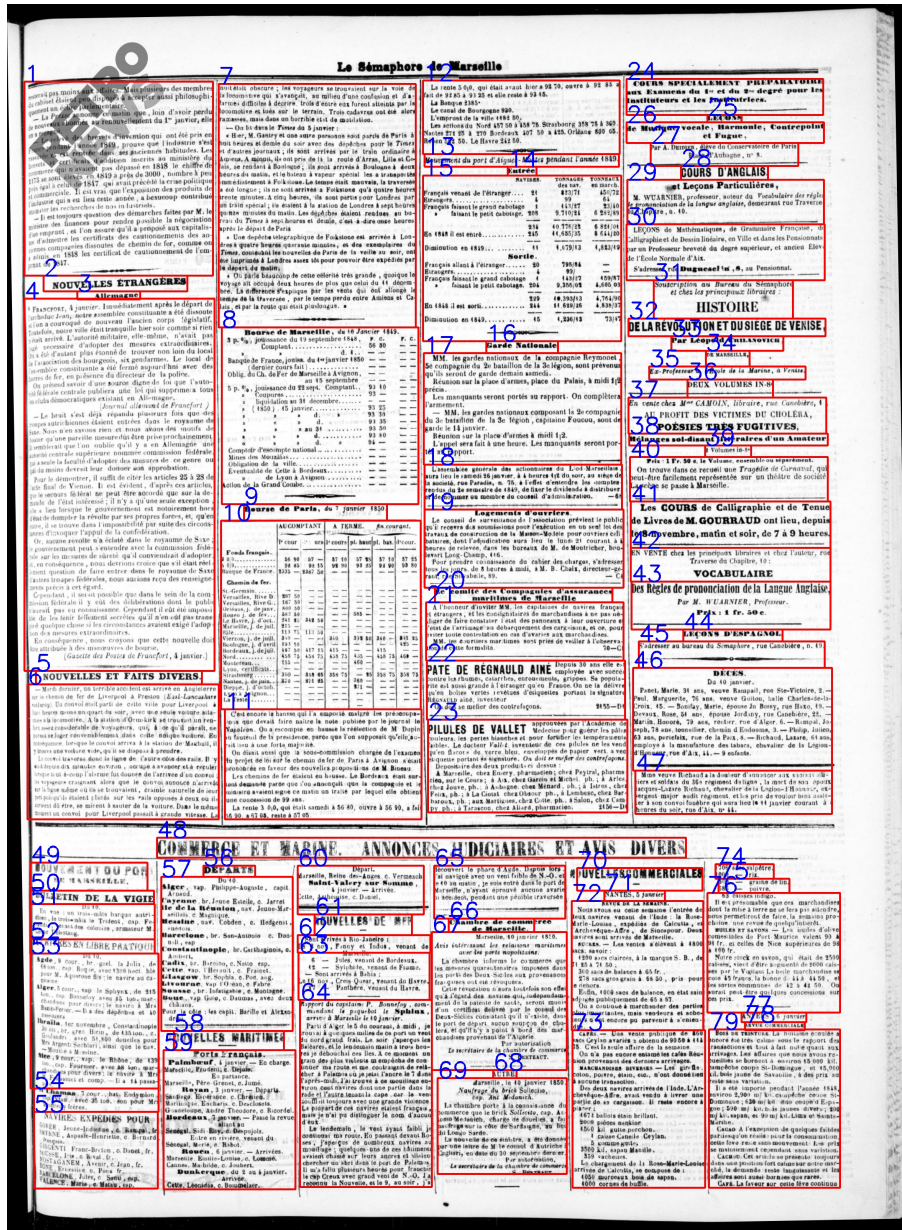
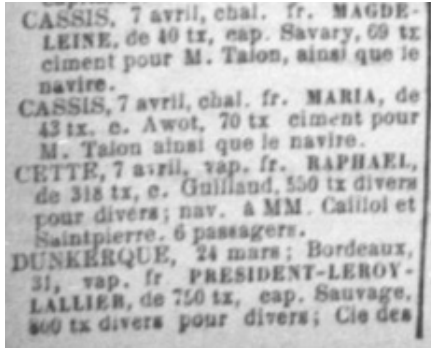


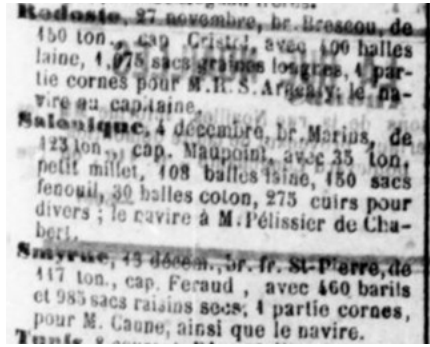
Fig. 11. Example of the blocks ordered correctly. Issue of January 11, 1850 (source of original newspaper image: retroneues.fr).

3.5 Section Identification

Figure 3 has illustrated a typical "Arrivals" data section from the newspaper, which is usually located on the third page. In some cases, however, this section may span the second and third pages or the third and fourth pages, depending on the publication layout. If the complete "Arrivals" section is not detected on the third page, the



(a) Curvature



(b) Noise

Fig. 12. Problems with the newspaper source.

second and fourth pages are also processed to ensure that the full section is captured. This strategy provides flexibility in handling variations in section placement while avoiding unnecessary processing.

Due to potential inaccuracies in OCR output, identifying the “Arrivals” section is not as straightforward as searching for exact matches of the keywords “ARRIVÉES” and “DEPARTS”. OCR artefacts, spelling variations, and character recognition errors frequently alter the appearance of these terms. To address this issue, the Levenshtein distance method is applied to locate the word most similar to “ARRIVÉES” and to identify the first occurrence of a word resembling “DEPARTS” that appears after it. As “DEPARTS” may occur multiple times within the text, whereas “ARRIVÉES” typically appears only once, this approach helps isolate the relevant section with improved accuracy.

In the example shown in Figure 11, a smaller subsection entitled “NAVIRES EXPÉDIÉS POUR” (block 55) appears within the “Arrivals” section. This subsection contains information that is not relevant to the extraction target. It is therefore identified using keyword matching based on the Levenshtein distance, and once detected, both the subsection header and all subsequent content are removed. This filtering step ensures that only the relevant arrival data remain, thereby improving the accuracy and focus of the information extraction process.

3.6 Paragraph Segmentation

Accurate paragraph segmentation is essential for extracting structured information about individual ship arrivals from the “Arrivals” section of the newspaper. Each paragraph typically corresponds to a single ship arrival and often begins with the name of the port of origin, which provides a clear anchor point for parsing the relevant details. By separating the text into distinct paragraphs, the extraction process is streamlined, ensuring that each paragraph is centred on the information of a single ship. This approach simplifies subsequent processing steps and reduces ambiguity during data extraction.

For this segmentation task, OpenAI’s language model (ChatGPT 4o) is employed using a carefully designed prompt that preserves the original structure of the text. The full prompt is provided in Appendix A. This prompt instructs the AI to segment the text into paragraphs without modifying or enriching the content, thereby preserving any errors or inconsistencies introduced during the OCR process. In addition, the paragraphs are grouped according to their arrival date. By focusing exclusively on paragraph segmentation, this step ensures that each paragraph accurately reflects the original layout of the source text and provides a clear and reliable structure for the subsequent information extraction phase.

3.7 Information Extraction

The final stage of the processing pipeline focuses on extracting specific information from each segmented paragraph corresponding to individual ship arrivals. The extracted information includes details such as the port of origin, date of departure, ship type, cargo information, the captain's name, and other relevant attributes associated with each arrival. A visual representation of the information to be extracted is presented in Figure 13.

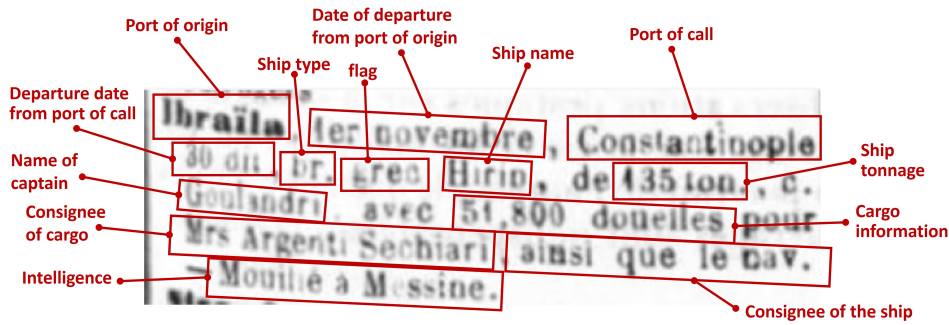


Fig. 13. Example of the information to be extracted from a ship arrival paragraph.

We considered two main approaches to information extraction: regular expressions and Large Language Model (LLM)-based methods. Regular expressions are effective for detecting patterns in structured text. However, they are highly sensitive to irregularities and noise. In the case of historical newspapers, the source data often lacks a consistent format, particularly with respect to punctuation, and the OCR output is frequently affected by inaccuracies caused by issues such as image quality and layout complexity. As a result, developing robust regular expressions for this task would be both time-consuming and challenging, as they would require continuous testing and adjustment to accommodate OCR errors.

To overcome these challenges, an LLM-based approach was chosen, specifically utilising OpenAI's language model. This approach provides greater flexibility, allowing the system to process a wide range of text variations and interpret information even in the presence of OCR imperfections.

To ensure that the model focuses on the relevant information, a detailed prompt was designed to guide the extraction process, in which the fields to be extracted are explicitly specified. In addition, the model was instructed to return its output in JSON format, thereby ensuring structured results. The full prompt is provided in Appendix B. This prompt constrains the model to operate on the raw input text without attempting to interpret or correct OCR errors, thereby preserving the authenticity of the data. By defining clear fields and requiring each text fragment to be used only once, the prompt reduces the risk of overlapping or duplicate information across fields and ensures consistency and clarity in the extracted data.

Finally, the structured JSON output is transformed into a CSV (spreadsheet-like) format, as an alternative that further facilitates data handling by maritime historians.

3.8 Pipeline Automation and Applicability

The pipeline has been fully automated for processing scanned documents of the newspaper *Le Sémaphore de Marseille*. To apply the pipeline to a different document format, a set of preprocessing adaptations must be considered at each stage of the pipeline, as outlined below.

- **Text Block Identification:** It is first necessary to ensure that Arcanum's newspaper segmentation API is compatible with the target document images. Arcanum recommends an image resolution of 150 DPI for

optimal performance. However, if the target images are particularly complex, the segmentation process may result in timeout errors. In such cases, reducing the image DPI can help ensure successful processing. In addition, if the target document contains images or visual elements that span multiple text columns, these elements should be removed. This can be achieved by using the *label* attribute in the JSON output to identify and exclude blocks labelled as *Picture*. Such elements can interfere with the text block ordering stage later in the pipeline.

- **Text Block Ordering:** If the target document does not contain a table in the bottom-right corner similar to that found in *Le Sémaphore de Marseille*, the logic for detecting and removing this table should be disabled. Conversely, if a table or another element that disrupts the column structure appears elsewhere in the document, the logic should be adapted to identify and exclude it accordingly. For documents with significantly different layouts, particular attention should be paid to the detection of vertical (column-separating) lines. When using the Canny edge detection algorithm, the edge detection thresholds must be adjusted to suit the characteristics of the target document. If the detected edges are too weak or too strong, column-separating lines may not be identified correctly, leading to errors in column detection. In cases where clear vertical separators are absent but blocks still span multiple columns, alternative strategies for splitting blocks should be considered, such as detecting whitespace between words or text blocks.
- **Optical Character Recognition (OCR):** This stage of the pipeline does not require structural changes to support different document types. However, OCR performance can often be improved through document-specific preprocessing techniques, such as image dewarping, noise reduction, or contrast enhancement. A different OCR tool may also be substituted without affecting the rest of the pipeline, provided that it can process isolated text blocks and produce textual output.
- **Section Identification:** The keywords used for section identification must be adapted to the target document. For example, if the objective is to identify a "Departures" section rather than an "Arrivals" section in the *Le Sémaphore de Marseille*, a broader keyword such as "MOUVEMENT DU PORT" could be used to locate the relevant section. Within this section, the keyword "DEPARTS" can then be searched to isolate the specific subsection of interest, particularly when the term appears multiple times throughout the document.
- **Paragraph Segmentation:** This is required only in cases where information is distributed across multiple repeating paragraphs. To adapt this stage, the LLM prompt should be modified to reflect the paragraph structure of the new document. For instance, the phrases or markers that indicate the beginning of a paragraph may differ from "Du xx mmm" and should be updated accordingly. In addition, any document-specific characteristics, such as recurring keywords, patterns, or delimiters, should be incorporated into the prompt to ensure accurate segmentation.
- **Information Extraction:** To adapt this step to a different document type, the LLM prompt should be updated to specify the fields to be extracted and their expected structure. If the target document contains different types of information or follows a different format, the JSON schema should be modified accordingly. It is also beneficial to include common OCR errors specific to the new document in the prompt. By explicitly accounting for typical errors, such as character substitutions or formatting inconsistencies, the model can better manage inaccuracies in the extracted text.

4 Evaluation

The evaluation was conducted using a test sample comprising all 26 issues of the newspaper published in January 1895¹³. In the following, we first report the processing time of each pipeline component (Section 4.1). We then present the evaluation results for the major components of the pipeline, namely Text Block Ordering (Section 4.2),

¹³The scanned documents for this month are available through the BNF's dedicated newspapers website: <https://www.retronews.fr/journaux/semaphore-de-marseille>.

Text Recognition / OCR (Section 4.3), Paragraph Segmentation (Section 4.4), and Information Extraction (Section 4.5). Finally, we report the costs associated with using the commercial tools Arcanum, Google Vision, and OpenAI GPT-4o on the test data (Section 4.6). The CSV file containing the final extracted data is publicly available¹⁴.

4.1 Processing Time

The size of the input data and the processing times for each pipeline component are summarised in Table 2. The most time-consuming component is Text Recognition (OCR) using Google Vision, which required approximately two hours. All other components operated with processing times on the order of minutes.

Table 2. Input data and time taken for each step of the processing pipeline.

Process	Input Data	Time Taken
Text Block Identification (Arcanum)	26 Images	5 minutes
Text Block Ordering (OpenCV)	26 JSONs & Images	2 minutes 30 seconds
Text Recognition / OCR (Google Vision)	4821 Image Blocks	2 hours
Paragraph Segmentation (OpenAI)	25 Text Files	4 minutes 40 seconds
Information Extraction (OpenAI)	491 Paragraphs	17 minutes 40 seconds
Total	-	2 hours 30 minutes

4.2 Performance of Text Block Ordering

The Text Block Ordering component successfully identified the correct reading order of text blocks in all 26 images. The process was completed in 2 minutes and 30 seconds, as reported in Table 2. This result demonstrates robust accuracy, providing a reliable foundation for the subsequent stages of the pipeline.

4.3 Performance of Optical Character Recognition (OCR)

The OCR stage processed a total of 4,821 text blocks extracted from the 26 newspaper images. Although all images were processed, one image exhibited significant physical damage (a fold on the arrivals section), which rendered the extracted information invalid. Nevertheless, this image was included in the overall OCR processing time, which amounted to 2 hours, as reported in Table 2.

To evaluate OCR quality, 491 paragraphs from the arrivals sections were assessed using a three-level scoring scheme based on the severity of recognition errors:

- **Score 3:** No errors or only minor errors (e.g., letter "c" misread as "e"), which do not affect critical information.
- **Score 2:** Errors affecting key information, such as the captain's or ship's name, that can be resolved through cross-referencing.
- **Score 1:** Errors in numerical data, such as cargo quantities, which are critical and difficult to validate.

As shown in Table 3, 65.4% of the paragraphs (321) were either error-free or contained only negligible errors, while 17.9% (88) contained critical numerical errors. The average OCR score was 2.47 out of 3.00, corresponding to an overall accuracy of 82.3%. These results indicate acceptable overall OCR performance. However, the relatively high proportion of critical errors in numerical fields underscores the importance of careful post-processing and validation during the data curation phase.

¹⁴https://users.ics.forth.gr/~fafalios/deltos-transcriber/ShipArrivals_SM_1895_01.csv

Table 3. Distribution of paragraphs by OCR score.

Score	Number of Paragraphs	Percentage
1	88	17.9%
2	82	16.7%
3	321	65.4%
Total	491	100%
Average Score	2.47/3.00 (82.3%)	

4.4 Performance of Paragraph Segmentation

Paragraph segmentation was performed using OpenAI's GPT-4o model on the text produced by the OCR stage. This step required 4 minutes and 40 seconds, as reported in Table 2, and resulted in 489 paragraphs, compared to the expected 491.

The performance was evaluated using the standard metrics of precision, recall, and F1-score. Precision is defined as the ratio of correctly extracted instances (in our case paragraphs) to the total number of extracted instances, while recall is defined as the ratio of correctly extracted instances to the total number of correct instances. The F1-score is the harmonic mean of precision and recall and provides a single measure that balances both metrics. Formally, precision, recall, and F1-score are defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \quad \text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

where TP denotes the number of true positives, FP the number of false positives, and FN the number of false negatives.

The performance results are reported in Table 4. The paragraph segmentation stage achieved a precision of 0.959, a recall of 0.955, and an F1-score of 0.957, demonstrating strong and reliable performance. The observed errors primarily stemmed from two sources:

- (1) **Merged paragraphs:** In three cases, two paragraphs were incorrectly merged due to missing terminal punctuation in the first paragraph.
- (2) **Misallocated fields:** On one page, paragraphs lacking arrival times were segmented incorrectly, with the model using unrelated fields, such as the port of origin, as placeholders. In addition, one erroneous paragraph, labelled "er.", was produced as a result of OCR errors.

Although limited in number, these issues highlight areas where further refinement of the segmentation logic could improve overall accuracy.

Table 4. Performance scores for paragraph segmentation.

Metric	Value
Correct Results	469
Total Predictions	489
Expected Results	491
Precision	0.959
Recall	0.955
F1-Score	0.957

4.5 Performance of Information Extraction

The Information Extraction process used the segmented paragraphs (489 in total, including 469 correctly segmented ones). The extraction took 17 minutes and 40 seconds (Table 2). Only the 469 correctly segmented paragraphs were evaluated for accuracy since paragraph segmentation is a different problem unrelated to information extraction.

4.5.1 Evaluation Metrics and Results. Each of the 12 information fields was evaluated independently. Table 5 reports, for each field, the number of correct predictions, the total number of predictions, and the number of expected results. A prediction was considered correct when the appropriate value was extracted into the correct field, based exclusively on the OCR output, irrespective of OCR-induced inaccuracies. The key performance metrics (precision, recall, and F1-score) are given for each field in Table 6. The key findings are outlined below:

- **High precision and recall:** Most fields achieved precision and recall values above 0.95, with average values of 0.971 for precision, 0.950 for recall, and 0.960 for the F1-score.
- **Strong performance for individual fields:** Fields such as "Name of Captain" (F1-score: 0.993) and "Tonnage of Ship" (F1-score: 0.985) exhibited near-excellent performance.
- **Challenges for specific fields:** The "Port and Date of Call" field recorded the lowest F1-score (0.895), primarily due to information grouping errors, where values from multiple fields were incorrectly combined into a single field.

Table 5. Evaluation Results for Information Extraction.

Field	Correct Results	Total Predictions	Expected Results
Port of Origin	458	468	469
Date of Departure	458	468	469
Port & Date of call	209	225	242
Type of Ship	445	469	469
Ship Flag	409	409	469
Name of Ship	457	469	469
Tonnage of Ship	456	463	463
Name of Captain	465	469	468
Cargo Information	445	469	469
Consignee of Cargo	379	398	414
Consignee of Ship	446	451	465
Intelligence	213	218	217
Total	4841	4976	5083

4.5.2 Common Errors. The most frequent extraction errors arose from the following issues:

- **Data grouping errors:** In some cases, information belonging to separate fields was incorrectly combined into a single field. For example, the "Type of Ship" field occasionally included data that should have been assigned to the "Ship Flag" field.
- **Partially correct entries:** Fields containing only partially correct information were classified as incorrect, which affected precision and recall values. For instance, if one port was missing from the "Port and Date of Call" field, the entire entry was marked as incorrect, even when all other ports and dates of call were accurately extracted.

Despite these challenges, the overall performance remained robust, demonstrating the effectiveness of using GPT-4o for this stage of the pipeline.

Table 6. Performance Metrics for Information Extraction.

Field	Precision	Recall	F1-score
Port of Origin	0.979	0.977	0.978
Date of Departure	0.981	0.979	0.980
Port & Date of Call	0.929	0.864	0.895
Type of Ship	0.949	0.949	0.949
Ship Flag	1.000	0.872	0.932
Name of Ship	0.974	0.974	0.974
Tonnage of Ship	0.985	0.985	0.985
Name of Captain	0.991	0.994	0.993
Cargo Information	0.949	0.949	0.949
Consignee of Cargo	0.952	0.915	0.933
Consignee of Ship	0.989	0.959	0.974
Intelligence	0.977	0.982	0.979
Average	0.971	0.950	0.960

4.6 Monetary Cost

The monetary costs associated with the tools used in the pipeline are summarised below:

- **Arcanum:** At a cost of \$0.005 per image, processing the 26 newspaper images resulted in a total cost of \$0.13.
- **Google Vision:** A total of 4,821 text blocks were processed. The pricing depends on the monthly usage tier: the first 1,000 units per month are free, units from 1,001 to 5,000,000 are charged at \$0.0015 per unit, and units beyond 5,000,000 are charged at \$0.0006 per unit. In our case, the first 1,000 units were free, while the remaining 3,821 units incurred a cost of \$5.73.
- **OpenAI GPT-4o:** The Paragraph Segmentation and Information Extraction stages processed approximately 43,500 and 524,500 input tokens, and produced approximately 3,400 and 59,300 output tokens, respectively. Input tokens were charged at \$0.0025 per 1,000 tokens, resulting in a cost of \$1.42. Output tokens were charged at \$0.01 per 1,000 tokens, resulting in a cost of \$0.90. The combined cost for these two stages was therefore \$2.32.

The total cost for processing all 26 newspaper issues amounted to \$8.18, demonstrating the cost-effectiveness of the proposed pipeline.

5 Conclusion

This paper highlights the transformative potential of advanced technologies for addressing the challenges of structured information extraction from historical newspapers. By designing and implementing a systematic pipeline tailored specifically to the historical newspaper *Le Séraphore de Marseille*, this work demonstrates how a dedicated methodology can convert raw archival material into structured, machine-readable data that can support large-scale historical analysis.

The pipeline demonstrated effective performance in processing historical newspaper data, achieving robust accuracy in Text Block Ordering, acceptable OCR quality with an average score of 2.47 out of 3.00, and strong performance in Paragraph Segmentation, with an F1-score of 0.957. The Information Extraction stage also performed reliably, with an average F1-score of 0.960 across the twelve information fields. The complete pipeline processed 26 newspaper issues in approximately 2 hours and 30 minutes. These results highlight both the strengths of the proposed system and areas for future refinement, underscoring its potential to advance historical research.

A key strength of the pipeline lies in its modularity, which allows targeted adjustments and improvements to individual components. This flexibility enables the system to be adapted to other document types or extraction tasks, offering broad applicability across a range of historical research scenarios.

In the short term, the primary objective is the complete extraction of ship arrival records from the PortADa project corpus, which comprises approximately 20,000 newspaper issues covering the period from 1850 to 1914. In parallel, the development of a dedicated database to store and organise the extracted information would streamline data access, querying, and analysis. Such a database would significantly enhance historians' ability to investigate trends in maritime trade and regional interactions over time. To support this effort, we plan to adopt a *Linked Data* approach, following [13, 15], with a focus on semantic interoperability and data integration.

Looking ahead, several promising directions for future work can be identified. Improving preprocessing techniques, such as incorporating image dewarping and advanced noise removal, could further enhance OCR performance, particularly when dealing with the degradation and poor quality typical of historical documents. Another important direction is the integration of regular expressions into the Information Extraction stage. This would enable a systematic comparison with AI-driven methods and provide a benchmark for improving extraction accuracy and reliability. In addition, analysing recurring errors made by the LLM during Paragraph Segmentation and Information Extraction offers an opportunity to refine the prompts used in these stages. By developing more targeted and task-specific prompts, the accuracy of the extracted results could be further improved, leading to overall gains in pipeline performance.

From a historical research perspective, linking the extracted data with related maritime history resources, such as the SeaLiT Knowledge Graphs [12], which include information about ships, ports, sailors, and related entities from the same historical period, can enrich historical analyses and provide deeper insights into maritime trade networks.

Acknowledgments

This work received funding from the European Union's Horizon Europe research and innovation programme under the Marie Skłodowska-Curie grant agreement No 101129889 (Project PortADa).

The authors sincerely thank Apostolos Delis, Kalliopi Vasilaki, Jordi Ibarz Gelabert, and Josep Cañellas, as well as all researchers involved in the PortADa project, for their valuable support throughout the course of this work.

The authors also gratefully acknowledge the Bibliothèque nationale de France (BnF) for providing access to the digitised newspaper archives that made this research possible.

References

- [1] Andrew S Agbemenu, Jephthah Yankey, and Ernest O Addo. 2018. An automatic number plate recognition system using opencv and tesseract ocr engine. *International Journal of Computer Applications* 180, 43 (2018), 1–5.
- [2] Marco Aiello, Christof Monz, Leon Todoran, Marcel Worringer, et al. 2002. Document understanding for a broad class of documents. *International Journal on Document Analysis and Recognition* 5, 1 (2002), 1–16.
- [3] Rifiana Arief, Achmad Benny Mutiara, Tubagus Maulana Kusuma, et al. 2018. Automated extraction of large scale scanned document images using Google vision OCR in apache Hadoop environment. *International Journal of Advanced Computer Science and Applications* 9, 11 (2018).
- [4] A Bandyopadhyay, A Ganguly, and U Pal. 2011. Layout segmentation of scanned newspaper documents. *Journal of Computational Linguistics* 1, 1 (2011), 5.
- [5] Raphaël Barman, Maud Ehrmann, Simon Clematide, Sofia Ares Oliveira, and Frédéric Kaplan. 2021. Combining visual and textual features for semantic segmentation of historical newspapers. *Journal of Data Mining & Digital Humanities HistoInformatics* (2021).
- [6] Melodee Beals and Emily Bell. 2020. The atlas of digitised newspapers and metadata: Reports from Oceanic Exchanges. *Loughborough. DOI* 10 (2020), m9.
- [7] Adrian Bingham. 2010. The digitization of newspaper archives: Opportunities and challenges for historians. *Twentieth Century British History* 21, 2 (2010), 225–231.
- [8] Vincent Buntinx, Frédéric Kaplan, and Aris Xanthos. 2017. Layout analysis on newspaper archives. In *Digital Humanities 2017*. 409–412.

- [9] Stevan Čakić, Tomo Popović, Stevan Šandi, Srđan Krčo, and Anita Gazivoda. 2020. The use of tesseract ocr number recognition for food tracking and tracing. In *2020 24th International Conference on Information Technology (IT)*. IEEE, 1–4.
- [10] YL Chaitra, MJ Roopa, MT Gopalakrishna, MD Swetha, and CR Aditya. 2023. Text Detection and Recognition from the Scene Images Using RCNN and EasyOCR. In *International Conference on Information and Communication Technology for Intelligent Systems*. Springer, 75–85.
- [11] Sai Aishwarya Edupuganti, Vijaya Durga Koganti, Cheekati Sri Lakshmi, Ravuri Naveen Kumar, and Ramya Paruchuri. 2021. Text and Speech Recognition for Visually Impaired People using Google Vision. In *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*. IEEE, 1325–1330.
- [12] Pavlos Fafalios, Athina Kritsotaki, and Martin Doerr. 2023. The SeaLiT ontology—an extension of CIDOC-CRM for the modeling and integration of maritime history information. *ACM Journal on Computing and Cultural Heritage* 16, 3 (2023), 1–21.
- [13] Pavlos Fafalios, Yannis Marketakis, Anastasia Axaridou, Yannis Tzitzikas, and Martin Doerr. 2023. A workflow model for holistic data management and semantic interoperability in quantitative archival research. *Digital Scholarship in the Humanities* 38, 3 (2023), 1049–1066.
- [14] Pavlos Fafalios, Kostas Petrakis, Georgios Samaritakis, Korina Doerr, Athina Kritsotaki, Yannis Tzitzikas, and Martin Doerr. 2021. FAST CAT: collaborative data entry and curation for semantic interoperability in digital humanities. *Journal on Computing and Cultural Heritage (JOCCH)* 14, 4 (2021), 1–20.
- [15] Pavlos Fafalios, Georgios Samaritakis, Kostas Petrakis, Korina Doerr, Athina Kritsotaki, Anastasia Axaridou, and Martin Doerr. 2022. Building and Exploring a Semantic Network of Maritime History Data. In *Mediterranean Seafarers in Transition*. Brill, 509–535.
- [16] Paul Gooding. 2016. *Historic Newspapers in the Digital Age: Search All About It!* Routledge.
- [17] Karez Hamad and Mehmet Kaya. 2016. A detailed analysis of optical character recognition technology. *International Journal of Applied Mathematics Electronics and Computers* Special Issue-1 (2016), 244–249.
- [18] Johan Jarlbrink and Pelle Snickars. 2017. Cultural heritage as digital noise: nineteenth century newspapers in the digital archive. *Journal of Documentation* 73, 6 (2017), 1228–1243.
- [19] Philip Kahle, Sebastian Colutto, Günter Hackl, and Günter Mühlberger. 2017. Transkribus—a service platform for transcription, recognition and retrieval of historical documents. In *2017 14th iapr international conference on document analysis and recognition (icdar)*, Vol. 4. IEEE, 19–24.
- [20] Tapas Kanungo and Robert B Allen. 1999. *Full-text access to historical newspapers*. Citeseer.
- [21] Rupinder Pal Kaur, Munish Kumar, and Manish Kumar Jindal. 2020. Newspaper text recognition of Gurumukhi script using random forest classifier. *Multimedia Tools and Applications* 79, 11 (2020), 7435–7448.
- [22] Stefan Klampfl, Michael Granitzer, Kris Jack, and Roman Kern. 2014. Unsupervised document structure analysis of digital scientific articles. *International journal on digital libraries* 14 (2014), 83–99.
- [23] J-L Meunier. 2005. Optimized XY-cut for determining a page reading order. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*. IEEE, 347–351.
- [24] G NAGY. 1984. Hierarchical representation of optically scanned documents. In *Proc. 7th Int. Conf. Patt. Recogn.* 347–349.
- [25] Bob Nicholson. 2013. The Digital Turn: Exploring the methodological possibilities of digital newspaper archives. *Media History* 19, 1 (2013), 59–73.
- [26] Kostas Petrakis, Georgios Samaritakis, Thomas Kalesios, Enric Garcia i Domingo, Apostolos Delis, Yannis Tzitzikas, Martin Doerr, and Pavlos Fafalios. 2020. Digitizing, Curating and Visualizing Archival Sources of Maritime History: the case of ship logbooks of the nineteenth and twentieth centuries. *Drassana: revista del Museu Marítim* 28 (2020), 60–87.
- [27] MAM Salehudin, SN Basah, H Yazid, KS Basaruddin, MJA Safar, MH Mat Som, and KA Sidek. 2023. Analysis of Optical Character Recognition using EasyOCR under Image Degradation. In *Journal of Physics: Conference Series*, Vol. 2641. IOP Publishing, 012001.
- [28] Zejiang Shen, Ruochen Zhang, Melissa Dell, Benjamin Charles Germain Lee, Jacob Carlson, and Weining Li. 2021. Layoutparser: A unified toolkit for deep learning based document image analysis. In *Document Analysis and Recognition—ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I* 16. Springer, 131–146.
- [29] Ray Smith. 2007. An overview of the Tesseract OCR engine. In *Ninth international conference on document analysis and recognition (ICDAR 2007)*, Vol. 2. IEEE, 629–633.
- [30] Thomas Smits. 2014. Problems and possibilities of digital newspaper and periodical archives. *Tijdschrift voor tijdschriftstudies* 36 (2014), 139–146.
- [31] Kengo Terasawa, Takahiro Shima, and Toshio Kawashima. 2011. A fast appearance-based full-text search method for historical newspaper images. In *2011 International Conference on Document Analysis and Recognition*. IEEE, 1379–1383.
- [32] Mattia Zeni and Komminist Weldemariam. 2017. Extracting information from newspaper archives in Africa. *IBM Journal of Research and Development* 61, 6 (2017), 12–1.

A Prompt for Paragraph Segmentation

You are an AI tasked with organising and formatting text about ship arrivals by dividing it into sections and paragraphs.

Key Instructions:

- Each section in the input text begins with a phrase similar to "Du xx mmm," but note:
 - Due to OCR inconsistencies, the keyword "Du" may contain variations such as "Da", "De", "Bu", etc., or it may be absent altogether.
 - The "xx" number is optional.
 - The "mmm" part is an optional month.
- Your output should be a JSON object with the following structure:
 - Each section should have a key "arrival_date" with the entire detected phrase (e.g., "Du xx mmm" or any variant thereof) as its value.
 - Each "arrival_date" section contains an array, "paragraphs" which holds each paragraph under that section as a separate item in the array.
 - Each paragraph should preserve the exact text as it appears in the input, without corrections, changes, or additions.
- Additional details:
 - (1) **Section Handling:**
 - If the text contains no initial section marked by "Du xx mmm" or similar variant, set the "arrival_date" field to null.
 - (2) **Paragraph Structure:**
 - Each paragraph describes one ship's arrival with information such as name of ship, tonnage of ship, captain of ship, etc., and often starts with the port of origin.
 - Maintain each paragraph's content precisely as it appears in the input, with no changes to wording or formatting (except removing internal '\n' symbols).
 - Ignore any line beginning with "ARRIVÉES" or any variant of it.
- Generate the JSON output strictly according to these rules.

B Prompt for Information Extraction

You are an AI designed to extract specific information from French text, with some potential OCR errors. For each input provided, return the extracted information in a JSON object with the following fields. If a field is empty, set its value to null.

The response should be a JSON object with this structure:

```
{
  "port_of_origin": null,
  "date_of_departure_from_port_of_origin": null,
  "port_and_date_of_call": null,
  "type_of_ship": null,
  "ship_flag": null,
  "name_of_ship": null,
  "tonnage_of_ship": null,
  "name_of_captain": null,
  "cargo_information": null,
  "consignee_of_cargo": null,
  "consignee_of_ship": null,
  "intelligence": null
}
```

```
}

```

Information Fields and Their Order:

- (1) **Port of origin:** Usually appears at the start of the paragraph.
- (2) **Date of departure from port of origin:** Always appears after the port of origin. It consists of a number and a month or the word "cour".
- (3) **Port and date of call:** Intermediate stops made by the ship on its journey to the final destination (may include multiple ports or none).
- (4) **Type of ship:** Found in lowercase, e.g., "vap", "br", "br-goel", "bat", etc.
- (5) **Ship's flag:** Found in lowercase, e.g., "grec" for Greek.
- (6) **Name of the ship:** The ship's name as it appears in the text.
- (7) **Tonnage of the ship:** If it exists, it appears after the name of the ship and usually starts with "de".
- (8) **Name of the captain:** Usually starts with "cap" or "c".
- (9) **Information about the cargo:** Includes cargo quantity, unit of measurement, and type of cargo (usually starts with "avec").
- (10) **Consignee of the cargo:** If it exists it is always after the cargo information and usually starts with "pour" or "p".
- (11) **Consignee of the ship:** Usually appears after the cargo information but it can also be absent; typically begins with terms like "le navire", "nav", "navire", or may simply be a name or entity. In cases like "ainsi que le nav", include only the words from the text.
- (12) **Intelligence:** Includes events or interactions during the journey, such as passing the Strait of Gibraltar, passenger information, etc. Always appears last if it exists.

Notes:

- Parse the input text in reading order.
- When assigning values to fields, never change the values of the previous fields in the order.
- The text contains OCR errors (e.g., "ton" might appear as "fon", or "tx" as "ix", or "c" as "e"; digits may appear as similar-looking letters like "i" instead of "1").
- Don't try to correct those mistakes, just give the information as it appears in the text.
- If a field's information is not present in the text, set its value to null.
- Never use the same text more than once, if you assign part of the input text to a field then you cannot assign it to another field.
- Respond with a JSON object.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009