Analysis of the Logical Layout of Documents

6

Andreas Dengel and Faisal Shafait

Contents

A. Dengel (✉)
German Research Center for Artificial Intelligence (DFKI GmbH), Kaiserslautern, Germany
e-mail: dengel@dfki.de; Andreas.Dengel@dfki.de

F. Shafait
Multimedia Analysis and Data Mining Competence Center, German Research Center for Artificial Intelligence (DFKI GmbH), Kaiserslautern, Germany
School of Computer Science and Software Engineering, The University of Western Australia, Perth, Australia
e-mail: shafait@dfki.de; faisal.shafait@dfki.de

D. Doermann, K. Tombre (eds.), *Handbook of Document Image Processing and Recognition*, DOI 10.1007/978-0-85729-859-1_6,

Abstract

Automatic document understanding is one of the most important tasks when dealing with printed documents since all post-ordered systems require the captured but process-relevant data. Analysis of the logical layout of documents not only enables an automatic conversion into a semantically marked-up electronic representation but also reveals options for developing higher-level functionality like advanced search (e.g., limiting search to titles only), automatic routing of business letters, automatic processing of invoices, and developing link structures to facilitate navigation through books. Over the last three decades, a number of techniques have been proposed to address the challenges arising in logical layout analysis of documents originating from many different domains. This chapter provides a comprehensive review of the state of the art in the field of automated document understanding, highlights key methods developed for different target applications, and provides practical recommendations for designing a document understanding system for the problem at hand.

Keywords

Bibliographic meta-data extraction • Document structure extraction • Document understanding • Information extraction • Invoice processing • Logical labeling • Logical layout analysis

Introduction

Logical layout analysis or document understanding refers to the field that is concerned with logical and semantic analysis of documents to extract human understandable information and codify it into machine-readable form. This information comprises not only textual aspects such as event dates, names, or identification numbers but also logical objects which may be derived from document structure as well as from information, which is added from background databases because of content-driven relationships. In order to do so, document understanding systems provide technology to automatically transform meaningful information from a raster image into a formal representation. Hence, it is a form of reasoning in which the meaning of communication is deduced from the combination of the written text and

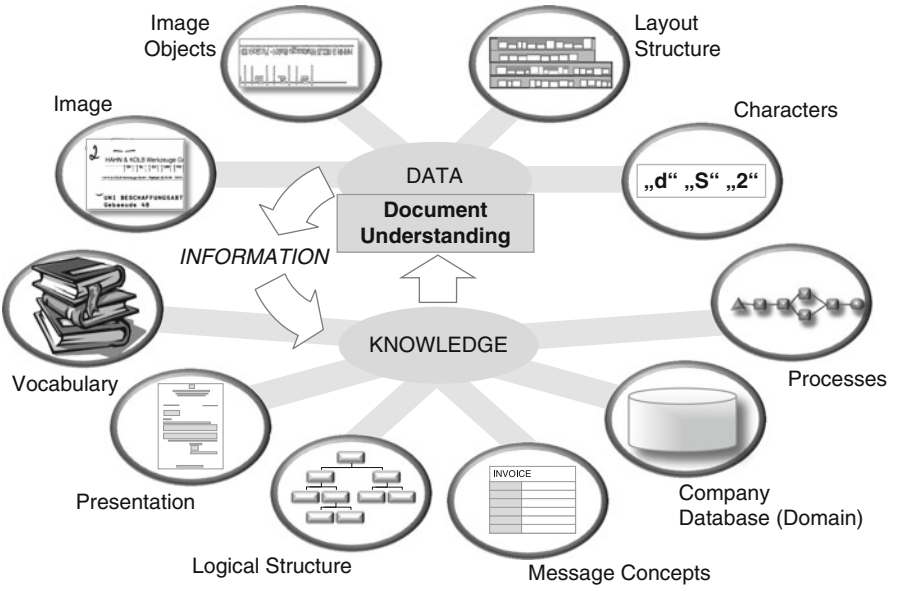

Fig. 6.1 Document understanding as the process to transform data into information by applying knowledge

its presentation, the logical structure of a message, the context, knowledge of the language, as well as knowledge about the world (see Fig. 6.1).

The output of document understanding may be of different quality ranging from an editable description or a representation useful for document sorting, routing, or filing up to a structured message with attribute-value pairs describing all aspects of communication necessary to drive particular workflows. To achieve this quality of information, the level of data present in a document has to be enriched to the desired level. Before doing so, a paper document has to be converted into an electronic form which is suitable for the application of document understanding techniques.

At a first glance a document is an object describing data of different types:

- Pictographic data where the whole image is represented as a sequence of orthogonal pixel runs
- Segment data describing homogeneous color or texture image regions
- Layout data describing the presentation of objects, i.e., their geometry, order, and nesting
- Character data indicating that multiple glyph images have the same ASCII code

Initially, none of these different data types have information value. An analogous situation occurs when an ordinary European holds a letter in her/his hands written in Kanji or Hangul characters. For her/him, this letter is nothing more than just data: an image with segments arranged in a certain order and some of them look the same.

The conversion from paper to electronic data of the four qualities described above is the initial step of document understanding providing an electronic representation that is already sufficient for many applications, such as full text indexing, electronic editing, or reuse of document content.

History and Importance

Documents are means of communication among individuals or institutions. They are created to announce conferences or new products, to summarize agreements, to comprise paragraphs of contracts, or even law, to publish innovative research results, or just to arouse, to promise, to inform, to question, to command, and to convince others. Documents are always present where information meets with human beings whose work gets done through documents. With the intention to assist these work processes, the question is how to make documents data useful. Document understanding plays a major role in extracting structured data from these documents such that it can readily be used effectively.

One of the stimulating factors that drove the initial progress in the field of document understanding was the advent of digital libraries. The objective of an electronic library is to create an electronic analog of a traditional library environment in each user's home or office. One of the first initiatives in this direction was the RightPages image-based electronic library for alerting and browsing technical articles, developed at AT&T Bell Laboratories [33]. The system provided users with online library services comprising stacks of journals. This was made possible by a complete document understanding pipeline including preprocessing, layout analysis, logical labeling, and text processing. A user interface was also developed that gave its user the look and feel of a conventional library. Another system, called Gobbledoc [23, 27], was developed by Nagy et al. to facilitate storage and browsing in digital libraries containing a large collection of technical journals. They developed an integrated segmentation and logical labeling approach for journal articles that became widely known as the recursive X-Y cut algorithm.

Desktop published also emerged as a new trend in the publishing industry in the mid-1980s. When reusing already printed material, a high demand for automating keyboard input data arose where large amounts of documentation had to be converted into a computer-readable form for data entry. Hence, text reader systems that could automatically convert a page into an electronic representation were required. Tsujimoto and Asada [35] developed a complete text reading system to fulfill these demands, contributing new approaches for document analysis, document understanding, and character segmentation/recognition. They defined document understanding as a transformation from a geometric structure into a logical one. A small number of rules were developed to carry out this transformation, based on the assumption that document layout is designed to seamlessly convey its logical components to humans.

Contrary to the previously mentioned complete systems for document analysis and processing on heterogeneous collections of documents, approaches for improved performance on a limited set of target documents also emerged at the same time. Bayer et al. [3, 29] developed a specific language called FRESCO for representing concepts of structured documents. FRESCO modeled the conceptual entities of structured documents from the very low level of connected components to high-level logical objects for a particular document class, like an IEEE journal.

Using syntactic analysis, desired logical objects (like title, authors, and page number of a technical article) were extracted.

Increasing use of personal computers in office environments also created the so-called media gap. Many companies desired to convert existing as well as incoming paper documents into an electronic representation for better information management including content-based retrieval and distribution. Hence a pressing need for document analysis systems that could be used as intelligent interfaces between paper and electronic media arose. Dengel [10, 12] presented a system called ANASTASIL to identify important conceptual parts (logical objects) within business letters, like sender, receiver, or company-specific printings. The system worked independently of text recognition and utilized only geometric information sources to assign logical labels to the segmented image regions, which provides the basis for an expectation-oriented text recognition, i.e., using controlled vocabularies.

The abovementioned early initiatives in the field of document understanding triggered a lot of research in this area in the later years. The next section outlines how the field evolved over the years while catering the demands of a rapidly changing market.

Evolution of the Problem

Document understanding systems after over two decades of research have become not only more robust and accurate but also more versatile. Naturally, expectations from document understanding systems have also significantly increased. The capabilities of a document understanding system can be judged from various perspectives. Complexity and diversity of documents that can be handled by a particular system, besides performance (speed and accuracy) on target document collection, are often used to characterize it. Document complexity refers in this context to the layout as well as to the number of document objects on the document page. Diversity of a document collection refers to the number of domains (e.g., business letters, invoices, magazine articles) of documents present in that collection. From these perspectives, document understanding systems can be roughly classified into four classes of increasing capabilities [1]:

- Systems processing simple documents of one specific domain
- Systems able to process simple documents from multiple domains
- Systems processing complex documents of one specific domain
- Systems able to process complex documents from multiple domains

It should be noted that systems able to process complex documents are a superset of systems processing only simple documents. However, the systems able to process documents from multiple domains are not necessarily a superset of system processing one specific domain, as they might not achieve the same performance as that of specifically designed systems.

If one looks at the chronological development of the field, one can see that system presented in the early 1990s focused mainly on processing simple documents of one

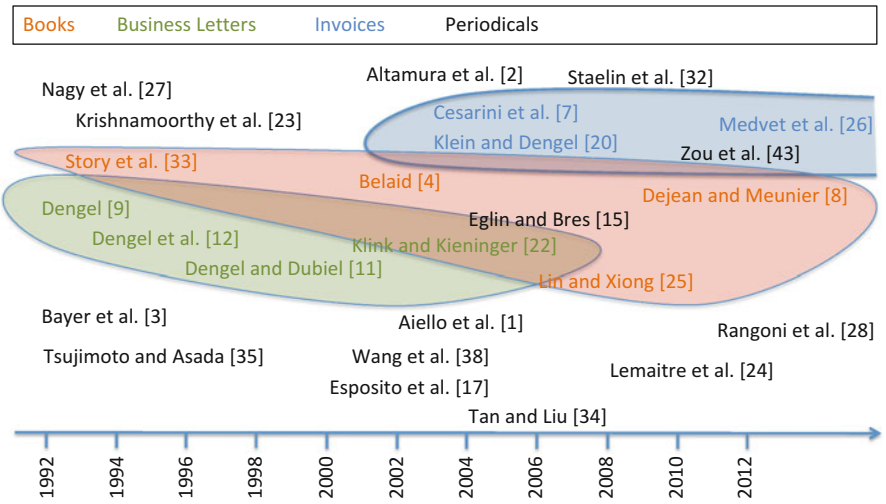

Fig. 6.2 Evolution of the application domains of document understanding systems

specific domain. Some prominent examples of such systems are the RightPages system from AT&T Bell Laboratories [33], the FRESCO system from Bayer et al. [3], and the Gobbledoc system by Nagy et al. [27] for analyzing journal articles and the ANASTASIL system by Dengel [12] for logical labeling of business letters. One exception is the system by Tsujimoto and Asada [35], which can process simple documents from multiple domains including journal articles, letters, manual, newspaper, and magazines. An overview of the evolution of different application fields of document understanding systems is shown in Fig. 6.2.

Further progress in the document analysis field led also to development of more robust logical labeling systems, particularly with the capabilities of handling more complex documents. Towards the end of the last century, many approaches were already proposed to handle complex documents of specific domains. Examples of representative work in that direction are the DAVOS business letter analysis system by Dengel and Dubiel [11] and the INFORMys invoice reading system by Cesarini et al. [6]. Both of these systems are capable of learning and extracting logical structure of documents with diverse layouts.

The first decade of the twenty-first century has seen further evolution in document understanding systems based on ideas from a diversity of theoretical foundations (see Fig. 6.3). Not only more systems were presented to handle heterogeneous documents from one domain (e.g., [2, 7, 28]), but also systems capable of processing complex documents from different domains (e.g., [1, 15, 22, 26]) were proposed in the literature. The former category of methods uses knowledge about the domain of documents in a learning framework to handle a variety of complex documents of that domain. The latter category of methods, on the other hand, focuses on extracting knowledge about specific logical labels automatically

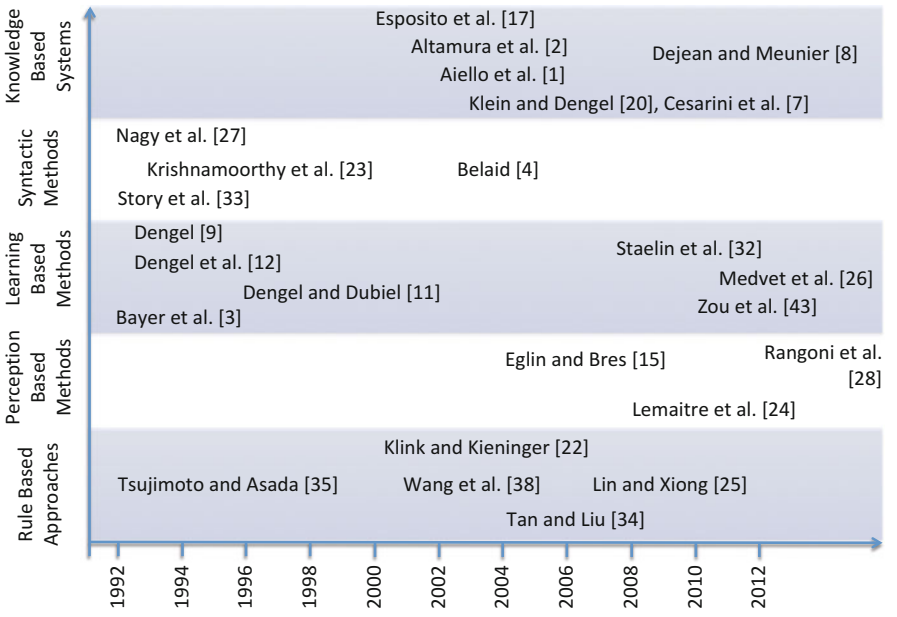

Fig. 6.3 An overview of evolution of underlying theoretical foundations of document understanding systems

from the training data comprising documents from multiple domains. Using this knowledge, they are able to characterize different logical labels based on their meaning and are thus capable of identifying those logical labels even in complex documents from different domains. However, the generalization capabilities of current document understanding systems are still far away from human capabilities. Further evolution of the field towards generalized systems that deliver high performance on heterogeneous document collections from multiple domains is expected to be seen in near future.

Applications

Understanding the contents of a document is crucial for many automated document analysis processes. Due to its vital role in making documents data useful, document understanding is a key component of document management systems employed in postal services, banks, incoming mail sorting of large organizations, and so on. Besides, digital library projects aimed at digitizing a particular collection of books, magazines, or newspapers also require to extract logical structure of the digitized material. Availability of logical structure facilitates navigation and advanced search inside the document as well as enables better presentation of the document in a possibly restructured format. Although document understanding is being used today

in a very wide spectrum of applications, the most prominent areas where it brings the highest value are outlined below. These application areas are described in more detail in section "Application Areas."

- **Creation of Digital Libraries:** In digital library projects, the aim is to digitize the books in the library so that they can be read online. The main role of logical labeling in this context is to identify table of contents pages in the books and to link individual entries in the table of content pages to the corresponding chapters/sections. Besides, if the document contents need to be reflowed, it is also desirable to identify page headers/footers, footnotes, and section headings. Similarly, when digitizing scholarly material like technical journals, it is often required to extract titles and authors of each article to facilitate advanced search within the digitized collection. Extraction of such logical entities is also a major application area of logical labeling.
- **Incoming Mail Sorting:** Many large organizations receive a large number of paper documents in incoming mail. To allow better information management and distribution, these incoming paper documents are converted into a structured electronic representation. A document understanding system is required at this stage to route the documents to the appropriate department/person in the company. This is achieved, for instance, by identifying key components representing certain concepts in a letter, like sender, receiver and subject, and then using business rules to appropriately forward the letter to the corresponding department.
- **Analysis of Invoices:** Another common application of document understanding is in automatic analysis of invoices. Invoices are also involved in daily workflows of many companies. Automatic analysis of invoices to extract header and position data and to make plausibility checks of the invoice items is the main contribution of document understanding in this domain.

Main Difficulties

In order to extract all relevant information from a document image, the employment of knowledge is of vital significance. Knowledge is a term used very often with little meaning. However, for the task of document understanding, one may refer to knowledge as various intuitively comprehensible sources that one uses in daily business to capture the important bits of information driving different decisions and processes (cf. Fig. 6.1).

First of all, there is the terminology of communication consisting of the vocabulary of a language enriched by special designators and identifiers of a domain. A layout structure may be knowledge as well, especially if it is typical of a certain class of documents. Consider, for example, the rectangles in the document image shown in Fig. 6.4. Although they represent simple geometric data, it can be easily reasoned that they describe the typical layout of an invoice. Furthermore, it is easy to give hypotheses about where certain logical objects such as recipient or position data may be located. Neglecting the aspect of presentation, it is possible to describe,

Fig. 6.4 Without having any knowledge about the textual contents of a document, in many cases it is possible to identify the category of that document just by looking at its layout. It is easy to identify that the document image in this figure is an invoice

for example, a business letter by just its defining logical constituents, such as sender, recipient, and subject. The structure can be further refined in some aspects. For example, a recipient is composed of a name and an address.

In addition to these complementary structural views, the various message concepts can be more restrictively defined in relevance to the context in which a document is appearing. For instance, in a corporate environment, the documents one receives correspond to one's role and tasks in that company, for example, "notification of claim" and "appraisal report" in an insurance transaction, or "invoice" and "delivery note" in a purchasing department. Furthermore, all of these sources are only valuable if the insured person, property, and value are known, or if knowledge about customers, suppliers, and products is accessible either from the company's databases or other devices.

If a human being is going to interpret the semantics behind a document message, he or she makes use of these kinds of knowledge sources either as tacit experience or by employing explicit expertise on the computer. It is one of the big challenges to combine these knowledge sources in a way that the relevant messages captured in the bulk of documents can be extracted automatically.

If this was possible, the extracted information might be added to the knowledge repository and used for later processing. For example, if a company receives an invoice, the existing information from the quote and from the order may be utilized as knowledge. Thus the process described in Fig. 6.1 also shows aspects of learning.

Summary of the State of the Art

In transforming the model shown in Fig. 6.1 to a computer implementation, it is important to first consider the diversity of documents encountered in different application of document understanding systems. Based on variations in structural aspects with respect to layout and logical composition, documents can be categorized into three major classes:

1. Highly structured documents, which have a static form built on precisely defined page layout. They can be described by geometric templates including the regions of interest (ROIs) where the relevant information to be extracted is located. Approaches for analyzing highly structured documents, like form processing systems [14,42], are described in ▸Chap. 19 (Recognition of Tables and Forms) and hence are not covered in this chapter.
2. Semi-structured documents allowing a partial employment of templates. These are documents having only some regular geometric features to identify either the class of document or some of the ROIs, i.e., dynamic forms or business letters which have preprinted headers or footers and an empty page body for the message. Analysis of semi-structured documents is the major focus of this chapter.
3. Loosely structured documents which cannot be characterized by a geometric template. They contain typical logical objects hidden in an irregular page layout.

For all these classes of documents, there have been various approaches published in the past, some of which are applied to a single domain of documents, while others are applied to a number of different domains. Most techniques combine geometric template matching and character recognition in order to extract keyword lists generated by post-OCR fuzzy match approaches for indexing and retrieving the contained text. However, there are also successful approaches extracting structured messages.

Figure 6.5 shows an attempt to categorize some of the important publications with respect to highly structured, semi-structured, and loosely structured documents. Although a large number of publications exist on logical layout analysis, the literature review here is limited largely to archival publications only (journal papers or book chapters).

Components of a Document Understanding System

Understanding a document image typically involves different processes (see Fig. 6.6). The exact order in which these processes are applied varies from one algorithm to another. Also, some algorithms might skip one or more of these processes, add some other processes, or apply them in a hybrid way. However, most of the document understanding systems use these processes in some form. Most of these processes have already been discussed in detail in previous chapters. However, for completeness, a brief outline of these processes and their role in a document understanding system is given here.

Document Structure Representation

Models for document structure representation usually consist of two parts: the physical and the logical part. The physical part encodes the layout information

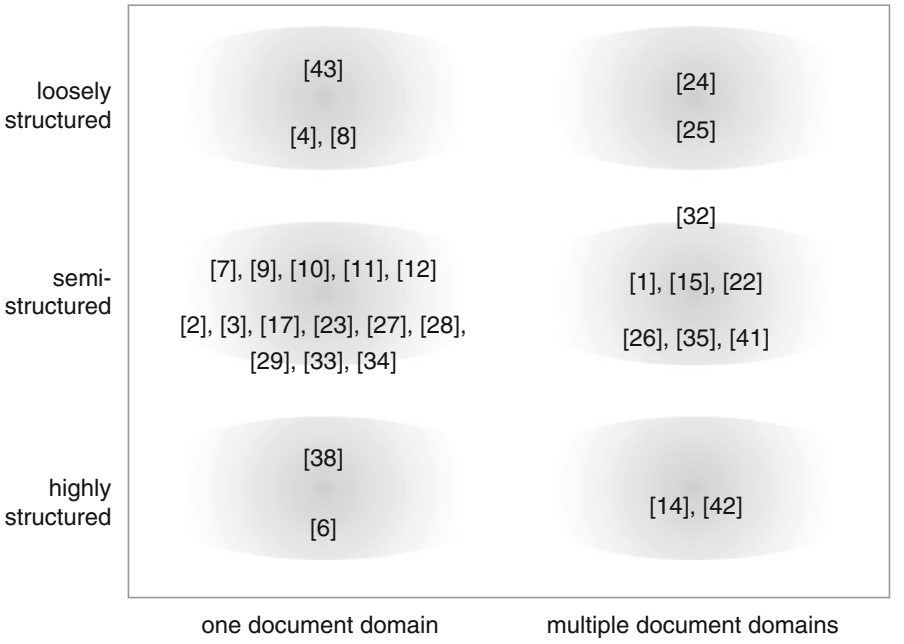

Fig. 6.5 An overview of the state-of-the-art logical layout analysis approaches for documents of varying complexities and domains

of the document (e.g., a page consists of some blocks, a block consists of some text lines, a text line consists of some words), whereas the logical part represents how the content of the document is organized into related logical units (e.g., title, heading, page number). Due to hierarchical nature of documents, one of the earliest and most successful document models was a geometric or X-Y tree [12, 23, 27, 35]. The tree-based document models were later generalized to graph-based models [1, 26] to handle more complex layouts (for a more detailed discussion of document models, please refer to ▸Chap. 7 (Page Similarity and Classification) – section "Page Representation").

The tree-based models consider a document page as a rectangle, having a characteristic width and height. To describe its spatial structure, the page is divided into smaller rectangles by horizontal and vertical cuts. Model cuts are placed in white-space areas such that they do not intersect with textual or graphical areas. The sub-rectangles can be recursively divided in the same way, until the layout of the page is described in sufficient detail. To annotate the logical structure, different rectangles are assigned a label which describes their logical meaning. Advantages of such models include simplicity, a natural representation of page hierarchy, direct correspondence between physical and logical structure, and direct integration with algorithms of page decomposition and logical labeling. However, they limit the

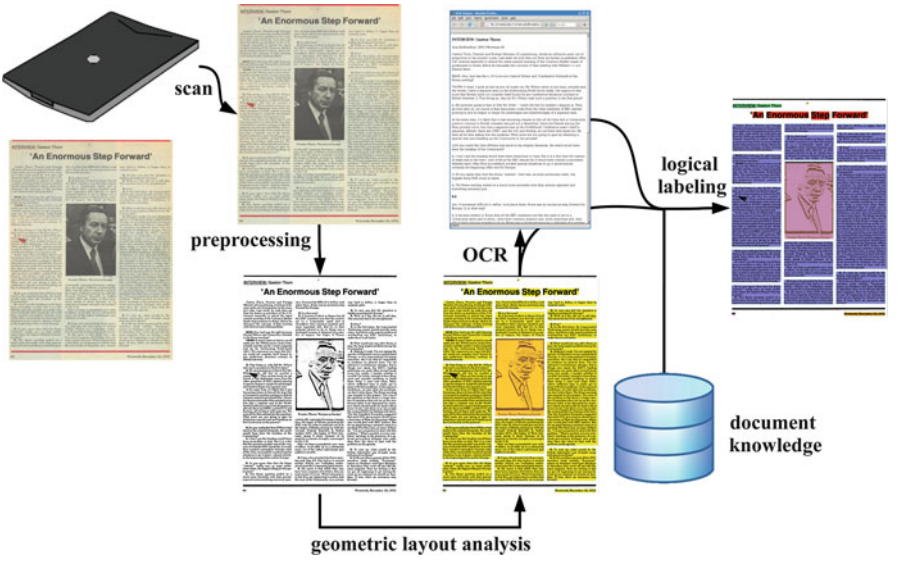

Fig. 6.6 An illustration of typical components of a document understanding system. The exact order in which these processes are applied (and which of the subcomponents are needed) varies from one approach to the other. For instance, many methods do not need OCR results for logical labeling

class of documents that can be handled to Manhattan layouts only. Besides, they only allow one layout structure per document page and one logical structure per page. However, complex document requires several views on both the layout and the logical information.

To adequately capture the structure of complex documents, generalized models represent a document as a set of layout or geometric structures [1, 26]. The set of layout structures is a collection of views such that each view represents a different layout interpretation of the document. Each layout structure itself is a set of geometric document objects and a set of geometric relations among them. Each type of geometric relation is represented as a graph. The vertices are document objects, and an edge represents a relation between the document objects. This graph can be a tree for a simple relation, but in general, it is a directed graph. The set of logical structure is represented in a similar way as a collection of views, where each view corresponds to a particular logical interpretation of the document. Due to this flexible representation, document objects are not required to have rectangular shapes, and one can define complex relations between document object. Besides, the possibility to have multiple views on the layout and logical structure benefits the labeling algorithms by allowing them to indicate multiple hypotheses as valid representations of the document.

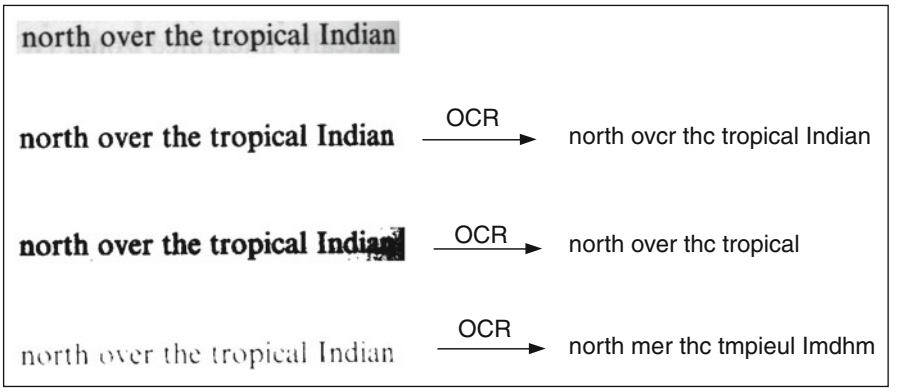

Fig. 6.7 The effect of choosing different binarization thresholds on the image and the resulting OCR output

Document Preprocessing

Binarization

Binarization is the process that converts a given input greyscale or color document image into a bi-level representation. The majority of document analysis systems have been developed to work on binary images [5]. The performance of subsequent steps in document analysis like page segmentation or optical character recognition (OCR) heavily depends on the result of binarization algorithm. Binarization with a high threshold results in merged components, which are difficult to recognize with an OCR system. On the other hand, binarization with a low threshold results in broken characters that are again a problem for OCR. An example of OCR results on differently binarized images of the same greyscale image is shown in Fig. 6.7.

Several approaches for binarizing a greyscale and colored documents have been proposed in the literature. Details of different binarization strategies are given in ▸Chap. 4 (Imaging Techniques in Document Analysis Processes) – section "Document Image Binarization."

Noise Removal

Different types of noise can be present in document images depending on the type of document degradation involved. Paper positioning variations usually result in marginal noise, making it hard to distinguish between the actual page contents and extraneous symbols from the neighboring page. Furthermore, non-textual noise (black borders, speckles etc.) may appear along the border of the page image as a result of binarization. Presence of border noise in document images might adversely affect the performance of page segmentation [30] or optical character recognition [31] modules in a document understanding system. Reliable removal of marginal noise under a wide diversity of conditions is still a challenging

problem. Details of different state-of-the-art noise removal algorithms can be found in ▸Chap. 4 (Imaging Techniques in Document Analysis Processes) – section "Document Image Enhancement."

Skew and Orientation Detection

In large-scale document digitization, skew and orientation detection plays an important role, especially in the scenario of digitizing incoming mail. The heavy use of automatic document feeding (ADF) scanners and moreover automatic processing of facsimiles results in many documents being scanned in the wrong orientation. These misoriented scans have to be corrected, as most subsequent processing steps assume the document to be scanned in the right orientation. In literature, no clear definition or distinction between page skew and orientation can be found. Skew detection methods are often presented with minimum and maximum rotation angles that can be detected. For orientation detection, some authors consider only upside up and upside down as possible orientations, whereas others also consider upside left and upside right as possible orientations. Although methods for skew and orientation detection are still being proposed [18, 37], these are generally considered as more or less solved problems for typical application scenarios (for a more details, please refer to ▸Chap. 4 (Imaging Techniques in Document Analysis Processes) – section "Document Image Normalization").

Geometric Layout Analysis

Text/Non-text Classification

Text/non-text classification is a key component of geometric layout analysis. Given a document image, the goal of page segmentation is to perform a decomposition of the document image into smaller zones or segments. The segments thus obtained are classified as containing text or non-text elements. The text segments or zones are then fed to a character recognition module to convert them into electronic format. If a page segmentation algorithm fails to correctly segment text from images, the character recognition module outputs a lot of garbage characters originating from the image parts. Figure 6.8 shows the case of an image merged with a text segment. When such a segment is fed to an OCR system, it outputs a large number of garbage characters in an attempt to classify the image portions as text. ▸Chapter 7 (Page Similarity and Classification) – section "Region Classification" outlines several methods to classify a given block or region of a page into text and non-text elements.

Page Segmentation

The task of page segmentation is to divide a document image into homogeneous zones, each consisting of only one physical layout structure (text, graphics, pictures, etc.). If the document contains more than one text column, the page segmentation algorithm should segment all text columns separately so that the text lines in different text columns are not merged together. Owing to the central role of page segmentation in OCR system, several page segmentation algorithms have

a

Figure 4 View of fruit shed naked (a) and (b) showing the complete androecial ring remaining within the tepals on the spike. (c) Fruit shed with all tepals attached.

b

tv _ _. - l 4
la) ` )âoơ , ` (C)
Â» ` fr /.1 Â·r , ` WM. âoơ
7 if Â¢âo Â¥, ` _ ` { Ã©>Ã©_j l; I X I I
K;~Â· - _ V . Â· ` _ ~ M" JF Q} { fÂ·',; Ai ' V _ I \' g,
âo _ it ~ _y, i * t
f jgj . __ it ` l âoơ f
1< .Â· âoơ Â·. V Â·- âoơÂ¢ âoơ âoơ âoơ F {U
Â·~ . 7 ~ Ã© . i/ ` I / _âoơ * âoơ K Â»
Figure 4 View of fruit shed naked la) and lb) showing the complete androecial ring remaining within the tepals on the spike. (c) Fruit shed with all tepals attached.

Fig. 6.8 The OCR result of an in-correctly segmented zone containing both images and text. The OCR system generates many garbage symbols from the non-text parts of the input page segment (**a**) Input page segment. (**b**) OCR result

been reported in literature for last three decades (please see ▸Chap. 5 (Page Segmentation Techniques in Document Analysis) Section "Analysis of Pages with Nonoverlapping Layout" for details). It should be noted that for logical layout analysis applications, it is also of vital significance that different text blocks corresponding to different semantic concepts (like author and title) are segmented into separate blocks in this step.

Table Detection and Labeling

In many practical applications, the documents to be analyzed (e.g., bank statements or invoices) do contain important core information in tables. To extract tabular information, one has to take into account that tables have no fixed position but rather can be found more or less anywhere on a page. Furthermore, tables appear in a large variety of styles. Therefore, to locate and analyze tables in document images, two broad categories of table structure extraction approaches have been proposed in the literature, namely, model-based approaches and model-free approaches.

Model-based table recognition allow the definition of specific table models which describe textual or layout features [16]. Model-free approaches, on the other hand, attempt to locate tables based on their geometric structure [39]. Model-based approaches are not universally applicable and fail for a large number of tables. However, since they are domain specific, they generally work better than model-free

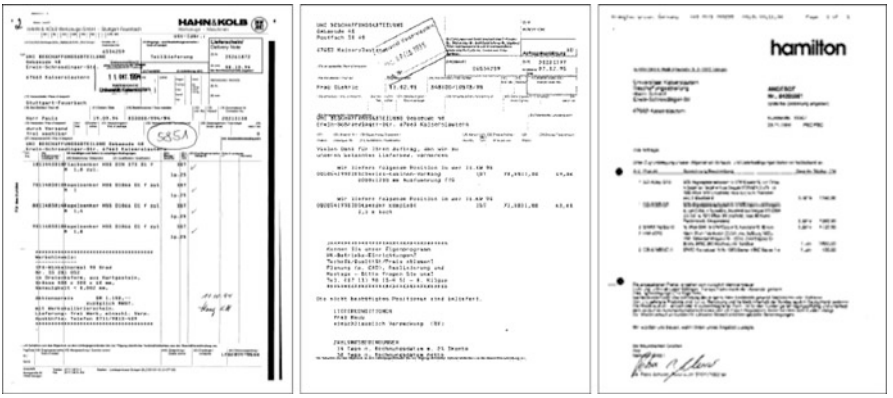

Fig. 6.9 Typical examples of documents capturing the same message concepts but showing different degrees of structural complexity

approaches on the targeted domains. The main advantage of the model-based approaches is that they can rely on logical table models, which are defined by the column headers and corresponding content types specified via regular expressions. These models are then used to guide the analysis process. As a result, tables cannot only be located but also categorized or even semantically interpreted at the same time. Details of different table structure understanding approaches can be found in ▸Chap. 19 (Recognition of Tables and Forms) – section "Consolidated Systems and Software."

Document Categorization

Document categorization is usually an important prerequisite for performing logical layout analysis, since the logical labels that one needs to extract from a document heavily depend on its category. For instance, in a scientific publication, one might be interested in the author, title, or abstract, whereas in a business letter, the semantic entities one would like to extract might be completely different (like sender, receiver, and date). In large-scale applications dealing with heterogeneous document collections, there is a need to categorize the documents before one can meaningfully extract logical components from a document.

One typical example is the bulk of documents at the mail entry point of a medium to large-sized organization. Such organizations receive thousands of documents daily, with widely varying structural complexity. The examples in Fig. 6.9, for instance, carry the same message concept "invoice" with the same relevant bits of information but show different levels of complexity. Furthermore, there may be single-page or multipage documents of different formats and paper quality. In many practical examples, documents having a different structural complexity as well as different formats are found in a single mail envelope. For example, in the domain of

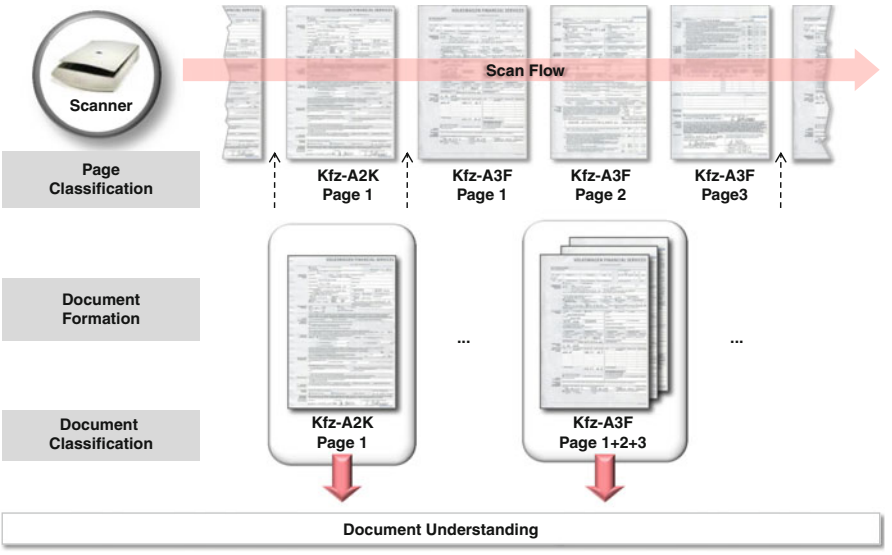

Fig. 6.10 A typical flow of a scanning process illustrating how page and document classification are employed before document understanding

health insurance, medical invoices are often sent with prescriptions or medical estimates. It is important to note here that a single document usually consists of multiple pages. Hence during the scanning process, one either needs to explicitly specify separator pages (e.g., placing a blank page after all pages of a document), or algorithms need to be developed to automatically cluster consecutively scanned pages into individual documents. A typical flow of the scanning process is shown in Fig. 6.10.

Consequently, there is a need to categorize many thousands of documents a day and further to determine how to proceed depending on the topic described. Document categorization addresses the task to automatically determine that a document belongs to a certain predefined class allowing routing or archiving of documents, the search for notes in online service systems, to respond to customer requests, to extract messages relevant to workflow, and many more practical applications. Categorization relies on methods that organize documents by structure or by content. Details of different methods for page or document classification can be found in ▸Chap. 7 (Page Similarity and Classification) – section "Page Classification."

Logical Labeling

The design of a logical labeling system requires carefully choosing a document structure representation that is able to capture the structure of most complex documents one expects the system to encounter and to develop a methodology using

that document model to extract relevant logical labels from a target document. The next section outlines key approaches for logical labeling grouped according to their theoretical foundations.

Techniques for Logical Labeling

A large number of techniques have been presented in the literature for logical labeling as outlined in section "Summary of the State of the Art." These techniques originate from many different concepts in computer science and artificial intelligence. In the following, an overview of different categories of document understanding techniques is given based on their underlying principles.

Rule-Based Approaches

When a document is created, some of the logical information is encoded in the document using layout typesetting conventions, e.g., by using a specific font size and style to represent a heading. Therefore, the layout structure of a printed document carries a significant amount of information about its logical structure. Particularly for simple documents, a human reader can usually determine the logical structure from layout and typesetting information only. These formatting rules used to produce the page can also be employed in the reverse process of logical labeling. Hence, a large number of document understanding approaches in the literature (e.g., [22, 25, 34, 35, 38]) perform logical labeling based on document-specific rules and use many features derived from the document. These features may include the relative and absolute position of a block on a page, the relative and absolute position of a field within a block, general typesetting rules of spacing, and the size and font style of the text. However, the wide variation of formatting style across different documents usually does not allow a comprehensive coverage of rules for a large number of document classes. Hence, most of the rule-based approaches are restricted to particular domains.

A representative rule-based method for document structure understanding is presented by Klink and Kieninger [22]. Their system consists of several stages. First, the scanned document is segmented, and the characters and their fonts are recognized by a commercial OCR system. Then, common document structures such as header, footer, lists, and tables are recognized to refine the output of the OCR system. Finally, domain-dependent logical labeling is performed on the identified text blocks. A rule-based approach is used in the logical labeling step. For each label, exactly one rule is defined. A rule in their system is not a simple if-then rule but rather consists of a logical expression which contains the so-called rule units combined with the logical operations AND ($\wedge$), OR ($\vee$), NOT ($\neg$), and parentheses. Hence, for assigning a particular label, logical expressions like $(A \wedge B) \vee (C \wedge \neg D)$ can be evaluated. While evaluating the logical expression, each rule unit (e.g., A or B) is matched against the block actually investigated. A rule unit is divided into two parts defining self-related attributes and cross-related attributes.

The self-related attributes define only the features of the block actually inspected and consist of both layout (geometric) and textual features. These might include dimensions of the block, its position in the page, number and alignment of lines, font attributes such as font family, size and bold format as well as a list of strings contained in the block, e.g., "with best regards" or a regular expression. The cross-related attributes consist of geometric relations (e.g., relative position of two blocks), textual relations (e.g., common words among two blocks), and label relations (e.g., presence of a specific labeled block directly above the current block). Based on these rule units, logical expressions can be defined to represent a particular label within a domain. In the recognition phase, a fuzzy rule matching approach is used to assign logical labels to the input page segments.

Syntactic Methods

Syntactic approaches for logical labeling attempt to draw an analogy between the structure of a document or page and the syntax of a language. The analogy is attractive not only due to the availability of mathematical linguistic tools but also due to the hierarchical nature of documents themselves. Syntactic logical labeling methods can be grouped into two broad categories. The first category of methods [23, 27, 33] is those that achieve logical labeling by building the parse tree for a page according to a specified grammar. The second category of methods [1, 4, 25] uses natural language processing on the output of an OCR system to guide the labeling process.

A representative method of the first category was presented by Nagy et al. [27]. They presented the X-Y tree data structure that transforms a 2D image analysis problem into a hierarchy of 1D string matching problems. Using conventional syntactic formulation, parsing a string effectively segments it into sub-strings that specify both the partitioning of the corresponding block on the page and the logical label of each partition. Hence, both segmentation and labeling of the page take place at the same time.

Abdel Belaïd [4] presented a logical labeling method based on part-of-speech (POS) tagging. The main idea behind this approach is that title and authors of an article could be identified using their specific linguistic characteristics (like author list contains several nouns corresponding to person names). The method parses the ASCII text output of an OCR system through a linguistic analysis tool to assign labels (like proper noun, common noun, adjective, article and number) to each word. Then, several rules are applied on the labeled strings to extract author and title sub-strings.

Perception-Based Methods

Despite the large variety of layouts and formatting conventions, most humans can easily get the logical information embedded in a page. This has inspired many researchers to use concepts from human perception and cognitive psychology for

logical labeling [15, 24, 28]. The most recent work in this direction is by Rangoni et al. [28]. They employ a perceptive neural network – which has been developed to be an analogy to human perception – for logical labeling. The main idea of perceptive neural networks is to integrate knowledge as interpretable concept associated to each neuron resulting in an architecture with local representation. Both physical and logical structures are incorporated as concepts in the neurons. A training step allows learning the relationships between the two structures from samples. The recognition is not only a classic feed-forward propagation but also performs many perceptive cycles. A perceptive cycle consists of forwarding the physical features, getting the logical output, and, if an ambiguity occurs, correcting the input vector. Due to this perceptive cycle, the system can refine the recognition progressively. Additionally, they incorporated a time-delay neural network to take into account the results of the previous perceptive cycles.

Learning-Based Methods

Learning-based methods make use of raw physical data to analyze the document. For that purpose, no knowledge or static rules are given. The underlying idea is to let the system learn the labeling function by itself and stop relying on rules and heuristics of an expert. A large number of learning-based approaches for logical labeling have been proposed in the literature [7, 9, 11, 12, 26, 32, 43]. While these approaches employ a broad spectrum of learning techniques and use different ways of using these learning methods for logical labeling, the underlying concept of data-driven methodology remains the same. The Biblio system by Staelin et al. [32] is a good example of learning-based methods. It uses example-based machine learning to adapt custom-defined document and metadata types. Like traditional machine learning systems, Biblio has two modes of operation: training and recognition. Both modes of operation use a cascading series of classifiers. During training, many example documents of a single type are fed to the system. The system uses both textual and layout features. Textual features are based on metadata dictionaries that are used to represent the words associated with a given metadata type. The function of these dictionaries is to give the probability that the given text stream contains text representing a particular type of metadata. This is achieved by training a support vector machine (SVM). The SVM engine uses a dictionary that contains a unique ID for every word it encounters. Each time a new word is encountered, a unique ID is generated, and the word is permanently stored in the dictionary. During training, the SVM engine builds input vectors using IDs from the dictionary to identify words in the document. This allows the SVM to create support vectors that identify the most probable words associated with a particular type of metadata. Other layout and textual features (like bounding boxes of page elements, font size, and ascender/descender ratios) are directly extracted from the document. These features along with the probabilities generated by the SVM for each meta-data type are used to train multiple neural networks. Each neural network outputs the probability for each metadata type directly. A majority voting is then used to obtain the final logical labels.

Knowledge-Based Systems

Generic logical structure information of a document is encoded as a set of layout and typesetting conventions for that document class. These conventions can be regarded as document knowledge for that particular class of documents. Many researchers have taken knowledge-driven approaches [1, 2, 7–9, 17] to reversely engineer the document authoring process and obtain the desired logical structure. Usually a basic distinction of document knowledge into two classes is adopted: Class-Independent Domain Knowledge (CIDK) and Class-Dependent Domain Knowledge (CDDK). The CIDK describes similar characteristics of logical objects in different documents in one domain, regardless of the classes. A domain of documents is defined in this context as a group of documents that can be clustered in terms of the subject. For instance, technical journals, tax forms, business letters, and invoices represent different domains of documents. Hence, documents in one domain can be characterized by their logical similarities, and these similarities are encoded in the CIDK. The CDDK, on the other hand, describes the characteristics of the logical objects of a particular class of documents (like articles from a specific journal). A system will be more effective in document understanding, when it uses CDDK for the target document. However, the system will not be able to handle documents coming from different classes. Hence, robust systems able to process a broad class of documents first use CIDK and then refine the results further by applying CDDK.

Knowledge-based approaches for document understanding can be further divided into two groups: those that employ machine learning to use document knowledge in the recognition phase (like [1]) and those that extract rules from the document knowledge to guide the recognition process (e.g., [7]). Aiello et al. [1] extract a set of geometric and textual features of each document object and its relation to other document objects. These features come from commonsense reasoning and statistical methods and constitute the knowledge base. Based on these features, a decision tree learner is trained to assign one of the logical labels to a document object. Cesarini et al. [7], on the other hand, use a learning scheme to construct the knowledge base. However, application of the knowledge base for document understanding is done through a rule-based system.

Case-Based Reasoning

Case-based reasoning (CBR) systems work by solving new problems based on the solutions of similar past problems. This concept was used by van Beusekom et al. [36] for logical labeling of title pages of journal articles. Their method takes a set of labeled document layouts and a single unlabeled document layout as input and finds the best matching layout in the set. Structural layout similarity and textural similarity on the block level are used to establish a similarity measure between layouts. Once a best matching layout is found, correspondence is established between the text block in the new document and those in the labeled document retrieved from the training set. Based on this correspondence, labels are simply transferred to the blocks in the new document.

Application Areas

Logical layout analysis is a key component in document understanding solutions for a large number of domains. A summary of different target domains where logical layout analysis serves a major role in the digitization workflow and the corresponding state-of-the-art approaches in those domains is given in Table 6.1.

Demands from a logical layout analysis system vary a lot from one application domain to the other. This is not only because of different set of desired logical labels to be extracted from the different domains but also because the kind of features that can be used to extract logical labels varies across these domains. In this section, the perspective of a target domain will be taken. In doing so, typical expectations from a logical layout analysis system for that domain will be described, challenges faced in reaching those expectations will be highlighted, and some prominent approaches tackle those challenges will be summarized.

Books

Digital libraries have become an important player in today's information age for bringing the library to the user. Besides the efforts of several individual libraries to bring their content online and worldwide accessible, last decade has seen a series of mass digitization projects, such as Google Book Search, Million Book Project, and the efforts of Open Content Alliance. The trigger for these activities was the initiative by Google Inc. in its ambitious launch of the Book Search project. They aimed at digitizing all the books in the world and making them accessible online.

The initial steps in most digital library projects involve efficient capture of book pages and converting the captured book pages into searchable text by using commercial OCR software. After that, the next step is to organize the logical units (e.g., chapters in a book or articles in a journal) of a book into a structured representation to facilitate further information retrieval. The goal is to improve the user experience in search and browsing through the book. To enable these tasks, the logical layout analysis module aims at two major tasks:

1. Identifying the start of each chapter and sections inside a chapter
2. Detecting table of contents (ToC) pages, recognizing individual entities in ToC pages, and linking them to the actual chapter/section headings

Several approaches have been presented in literature for solving these tasks [4, 8, 13, 25, 33]. In practice, the two problems are usually not solved independently but are tackled together since ToC provides rich information about possible chapter/section headings, which can be used to improve the accuracy of chapter identification step.

Although identifying a ToC page seems trivial at the first glance due to more or less regular structure, yet everyday documents show a variety of ToC pages (see Fig. 6.11). There are different ways to display the organization to the reader (font size, indentation, capitalization, dot leader, item marks, etc.), different numbering systems, different levels of information present, different order of the

Table 6.1 A summary of state-of-the-art logical layout analysis approaches for different target domains

Application / Technique	Rules	Perception	Learning	Syntactic	Knowledge
Books	[25]			[4, 33]	[8]
Business letters	[22, 35]		[9, 11, 12]		
Invoices			[26]		[7, 20]
Periodicals	[22, 34, 35]	[15, 28]	[3, 32, 43]	[23, 27]	[1, 2, 17]
Other applications	[38]	[24]	[26]		

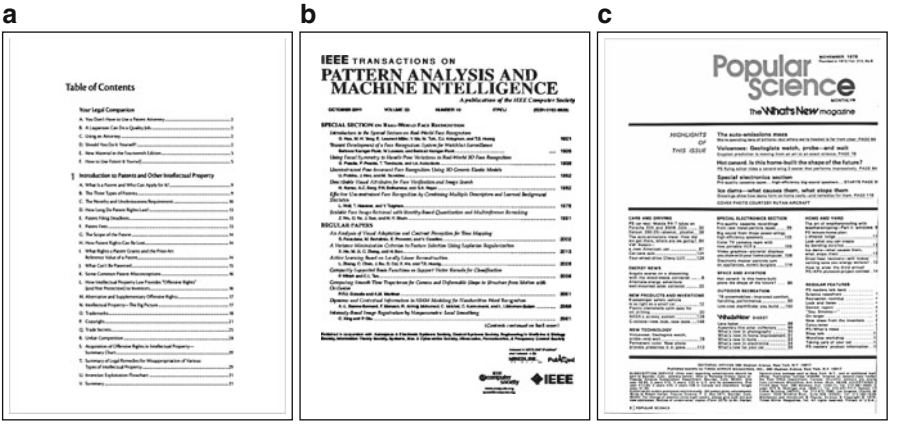

Fig. 6.11 Different samples of table of contents (ToC) pages ranging from simple (book) to quite complex (magazine) layouts (**a**) Book. (**b**) Journal. (**c**) Magazine

main fields like author, title, or page number (every field can appear on the left, on the right, or in the middle), and so on. Furthermore, a ToC page can contain other miscellaneous information besides article references, like copyright notice or subscription information. However, since ToC pages serve a particular function (referring to other pages in the same document), they do exhibit some functional regularity. Déjean and Meunier [8] define a ToC as an element having the following properties:

1. Contiguity: A ToC consists of a series of contiguous references to some other parts of the same document.
2. Textual similarity: Each reference has a high textual similarity with the referred part.
3. Ordering: The references appear in the same order in the document as the referred parts.
4. Optional elements: A ToC may include elements whose role is not to refer to any other part of the document, e.g., decorative text or logical headlines grouping ToC elements themselves.

5. No self-reference: All references in a ToC page refer outside the contiguous list of references forming the ToC.

It is important to note here that when resolving references from ToC entries to actual articles in a document, distinction should be made between logical page numbers and physical page numbers. Logical page numbers are the page numbers printed on the ToC pages, whereas physical page numbers are the result of the scanning process. For instance, if all the pages in a book are scanned, the front cover page will be the first physical page. Since the ToC entries refer to logical page numbers, one has to map these to physical page numbers to enable navigation to the correct target page.

A comparative overview of different ToC analysis methods is given in Table 6.2. Among pioneering works for ToC analysis was the RightPages image-based electronic library system [33] developed at AT&T Bell Labs. The system provided services to alert its users to the arrival of new journal articles matching their interest and to let them browse through the alerted article electronically. To enable navigation through different articles, ToC of the corresponding journal was analyzed, and its entries were linked to page numbers of the respective articles. This was done by specifying a grammar for each of the different ToC formats encountered in the indexed journals. The grammar used information about the relative positions of the blocks to determine their semantic structure. The specification was converted into a parse tree that was used to determine the structure of each new ToC page using a grammar specific to the particular journal title. For instance, association was made between the title of an article and its page number. This association was used to determine the page number automatically when a user clicked on the title of an article and then to retrieve and display the appropriate page.

Abdel Belaïd [4] presented a labeling approach for automatic recognition of ToC entries for a digital library system called Calliope. The main objective of Calliope is to index ToC pages from many periodicals and to allow scanning of the corresponding articles on demand. Hence, for automatic recognition of ToC pages, one needs to identify individual entries in a ToC page and further decompose these entries into the corresponding constituents (title, authors, and page number). The method developed by Belaïd is based on part-of-speech (POS) tagging of the OCR result of ToC pages. The tagging process consists of three stages: tokenization, morphological analysis, and syntactic grouping and disambiguation. The tokenizer isolates each textual term and separates numerical chains from alphabetical terms. The morphological analyzer contains a transducer lexicon that produces all legitimate tags (like adjective, article, preposition, common noun, proper noun) for words appearing in the lexicon. The syntactic grouping and disambiguation module employs another transducer which assigns a token to "author" or "title" by examining its context. As a result, each article reference is decomposed into its title, authors, and page number fields.

Lin and Xiong [25] presented TOCDAS (ToC Detection and Analysis System) to detect and analyze ToC pages in a wide range of documents without explicit modeling or training. The goal was to convert MIT Press' 3,000 out-of-print books, journals, and magazines from paper to high-quality electronic version enabling new services such as online reading and print on demand. A commercial OCR system SDK was used to extract bounding boxes for words, text lines, and non-text regions

Table 6.2 An overview of key approaches for book structure extraction

Reference	Input features	Target domain(s)	Logical labels	Key idea	Evaluation metric	Experimental dataset
Story et al. [33]	Page segments, OCR results	Books (journals)	Article segmentation and linking on ToC pages	A grammar for each type of ToC page is specified; logical structure is given by the parse tree of a page	Not specified	Private
Belaid [4]	OCR result	Book	Article segmentation and functional tagging in ToC pages	parts-of-speech tagging	Labeling accuracy	Private
Lin and Xiong [25]	Page segments, OCR results	Books	Table of content pages, article segmentation, linking, and functional tagging	Textual similarity and parts-of-speech tagging	Labeling accuracy	Private
Dejean and Meunier [8]	Page segments, OCR results	Books	Table of content pages, article segmentation, and linking	Knowledge-based approach exploiting link structure of ToC pages; logistic regression classifier for decision making	Precision, recall	Private

in addition to plain ASCII text output. The unique feature of their approach is its content association algorithm, through which ToC detection and a significant part of ToC analysis are conducted simultaneously, helping to verify each other. The contents on ToC pages are associated with those on the body pages using observations that text information about an article (like its title) and its start page number will be listed both on the article title page and the ToC page. Due to the distinct nature of these two kinds of information, two algorithms, general text mining and page number matching, are designed to handle them separately. A confidence score is then calculated for the candidate ToC page by adding together the general text mining score and page number matching score. A candidate page is marked as a real ToC page if its confidence score is above a given threshold. Since the identification of a ToC page is based on its association with the contents of other parts of the document, one automatically gets the references from different entries of ToC page to their corresponding articles.

Déjean and Meunier [8] present a method for automatically identifying ToC pages in a scanned book, recognizing different entries in a ToC page, and finally linking the recognized entries to the target chapters/sections. The method aims at developing a generic method to be applied on any document, without any specific knowledge about the document or the collection it may belong to. Besides, the same method can be used to detect other organizational tables (e.g., table of figures) in the document. The key idea of their approach is to first use functional knowledge to identify ToC pages and then to apply formal or layout knowledge to improve the accuracy of the method in a second step. The method computes pairwise textual similarity between all text blocks. Then, a functional definition of a ToC page based on contiguity, textual similarity, ordering, presence of optional elements, but ignore self-references, is used to generate and score ToC candidates. A ToC candidate is a set of contiguous text blocks, from which it is possible to select one link per block to provide an ascending order for the target text blocks. Finally, a Viterbi best-path algorithm is applied to select the best candidate ToC page. Moreover, its references are identified. The results of this step are further refined by training a binary classifier to determine for each link whether it belongs to the ToC or not. The classifier uses formal knowledge about the document itself by learning a feature vector consisting of layout and textual features. This results in a refined ToC that can be used to structure a document.

To evaluate and compare automatic techniques for deriving structure information from digitized books, a book structure extraction competition was organized at ICDAR 2009 [13]. The task that participants faced was to construct hyperlinked tables of contents for a collection of 1,000 digitized books selected from the INEX book corpus [19]. Two hundred out of those one thousand books did not contain a printed ToC. Four organizations participated in the contest, and their submissions were evaluated using commonly used precision, recall, and F-measure metrics. The results showed that the method from Microsoft Development Center Serbia achieved the best results with an F-measure of 41.5 %. Evaluation on a separate subset of books without an explicit ToC page showed that existing algorithms for this task are still premature, with the winning algorithm achieving an F-measure of 7 % only.

Invoices

Processing of invoices, either automatically or manually, is a daily task in most of the companies. The volume of invoices to be processed largely depends on the type of business of a company. For instance, insurance companies receive a large number of reimbursement claims every day. These invoices contain similar information, but the information items are distributed according to a large number of different layout styles. The information items are usually categorized into two broad classes:

1. Header data: data about the invoicing party, invoice date, invoice number code, total invoice amount, due date for payment, etc.
2. Table or position data: details about the invoice elements line by line

Automatic extraction of these information items from a given invoice is the goal of invoice reading systems. Due to the presence of table data, table spotting and understanding play a major role in automatic invoice processing. This section focuses on techniques for extraction of head data. The readers are referred to section "Table Detection and Labeling" for a discussion about table structure extraction and analysis.

When processing invoices, several logical labels need to be extracted from the invoice head data. Since financial transactions are based on invoices directly, accurate extraction of these labels is very critical. Therefore, systems for invoice processing use as much knowledge as possible to verify the extracted information from the invoices. Medical invoices can be considered as a use case here, without loss of generality. A medical insurance company may receive thousands of invoices each day. When processing a particular invoice, the first logical label that is required is the identity of the patient (first name, last name, insurance number). Then, the date of the treatment needs to be extracted to verify whether the person was insured at the time of treatment. After that, the treatment details (table data) have to be extracted to do plausibility checks (are calculations correct, does the reference number look genuine, is treatment cost in line with similar treatments of other patients, does treatment suit the diagnosis, does the patient's insurance policy cover this treatment, etc.). Finally, the total amount to be paid and payment target should be identified. If payment is to be made to a doctor/agency, the bank account details of the doctor/agency should also be extracted (and verified).

It is important to note here that all of the abovementioned information fields to be extracted from an invoice have a vital importance to ensure correct transaction to the recipient and to prevent insurance fraud. To reliably extract information, these fields are usually distinguished into different classes based on how they can be verified [21]:

1. Enumerable fields: These are the fields for which the extracted value can be judged as correctly recognized by matching it to a database having all possible values for that field. For instance, a person's identification can be regarded as correctly extracted if the identified first name, last name, and date of birth match with the company database.
2. Labeled fields: These are the fields that are most often simply identified by a keyword present in the field, like "From:…," "Total," and "Account number."

One of the earliest invoice processing systems was presented by Cesarini et al. [6] (for a comparative overview, please see Table 6.3). The system was designed to handle only form-like invoices and was tested on a dataset consisting of utility bills from Italian TELECOM and Italian electricity supplier company ENEL. Since the system was limited to form-like invoices, they used a three-step approach. In the first step, form registration was performed to align the target form with a reference form (prototype). Then, information fields were located by establishing correspondences between the marked information fields in the prototype with the fields in the target form. Finally, the identified information fields were recognized by using a connectionist-based model, using a multilayer perception as an autoassociator.

To analyze multi-class invoices, Cesarini et al. [7] developed a knowledge-based system. Two levels of knowledge for the invoice domain were elaborated. Logical similarities of invoices were captured by general knowledge about the domain (CIDK) as well as class-specific knowledge (CDDK). CIDK describes similar characteristics of logical objects in different invoices, regardless of the classes, whereas CDDK represents a collection of the characteristics of logical objects of particular classes of invoices. Both these knowledge sources are constructed by a learning procedure on a learning set consisting of a small number of invoices of each class. When an invoice has to be analyzed, both knowledge sources are used (for details, please see section "Knowledge Based Systems").

A commercial system called smartFIX was presented in [20] for analyzing invoices from a variety of sources. The system implements a comprehensive document understanding pipeline, from scanning and document classification to information extraction and verification. Since the system is deployed in several health insurance companies in Germany, it has a thorough coverage of logical labels to support different workflows in the companies. A total of 107 labels are supported, although on average, about 20 appear on a single document.

Medvet et al. [26] present a probabilistic approach for logical labeling of invoices. Their approach is model based, where a model represents documents of the same class, i.e., invoices from the same firm. When an invoice from a new firm is received, the operator can make the corresponding model using a GUI. Given a new document from the same firm, the logical labels are identified as the sequence of blocks that are most probable given the model.

Business Letters

To cater the needs of modern offices, many companies convert incoming paper documents into a structured electronic representation that allows better information management and distribution. Business letters constitute a major part of these documents. Therefore, logical labeling of business letters has been a topic of research since early days of document analysis (see Table 6.4). Automatic analysis and understanding of business letters involve identification of key components representing certain concepts in the letter, like sender, receiver and date. Fortunately, business letters typically follow some specific layout structures, and hence the

Table 6.3 An overview of key approaches for invoice analysis

Reference	Input features	Target domain(s)	Logical labels	Key idea	Evaluation metric	Experimental dataset
Cesarini et al. [6]	Raw image	Invoice	Data fields in form-like invoices	Registering new invoices to manual prototypes using attributed relational graphs	Labeling accuracy	INFORMys
Cesarini et al. [7]	Page segments, geometric layout tree, OCR result	Invoices	Invoice number, total amount	Knowledge-based approach using rules based on layout features and contents	Success rate, labeling, accuracy, label segmentation accuracy	Private
Klein and Dengel [20]	Page segments, OCR result	Invoices	107 labels appearing in medical invoices	Document management system specifically aimed at commercial-grade invoice processing	Classification accuracy	Private
Medvet et al. [26]	Text blocks and their OCR result	Invoices, patents, datasheets	Eight labels in invoices, 11 labels in patents, 8 labels in datasheets	Probabilistic modeling of known layouts, user interaction to model a new layout	Success rate	Ghega

Table 6.4 An overview of key approaches for business letter understanding

Reference	Input features	Target domain(s)	Logical labels	Key idea	Evaluation metric	Experimental dataset
Dengel [9]	Page segments, geometric layout tree	Business letters	Sender, recipient, date, subject	Geometric trees representing different layouts are learned, labeling done using a decision tree	Not specified	Private
Dengel et al. [12]	Page segments, geometric layout tree	Business letters	Sender, recipient, date, subject	Logical object arrangements are learned using layout trees, labeling done using a probabilistic parsing of the tree	Not specified	Private
Tsujimoto and Asada [35]	Geometric layout tree	Business letters, technical journals, magazines, newspapers	Title, abstract, subtitle, paragraph, header, footer, page number, caption	Transform geometric layout tree to logical tree using a set of rules	Not specified	Private
Dengel and Dubiel [11]	Page segments, geometric layout tree	Business letters,	Sender, recipient, date, logo, subject, footer, body text	Unsupervised learning to build a geometric tree based on attribute of logical objects	Precision, recall	Private
Klink and Kieninger [22]	Page segments, OCR result	Business letters, technical journals	Business letters (header, addressee, subject, date, sender, closure) Journals (header, title, author, abstract, bibliography)	Human expert defines rules, which are then matched with a fuzzy combination to label new documents	Precision, recall	UW-I Private

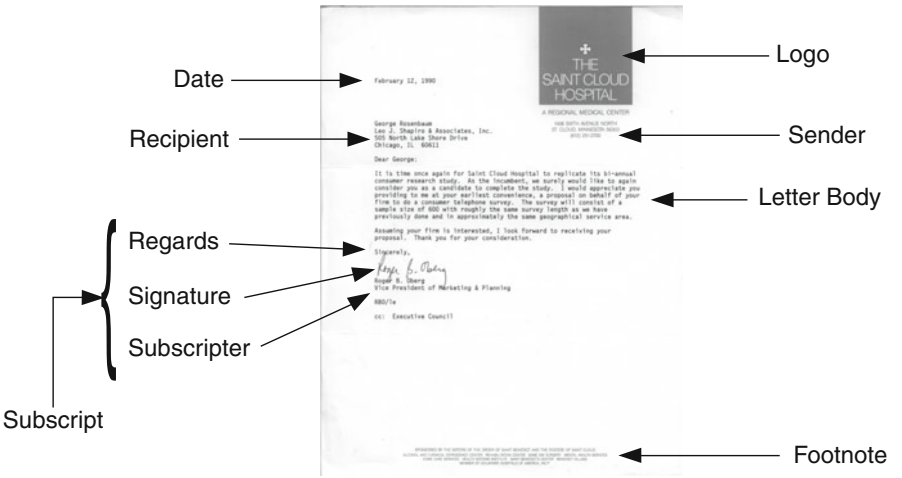

Fig. 6.12 Typical logical components of a business letter

layout of a letter already provides rich information about the logical components of the letter. For instance, when one looks at the block arrangement in Fig. 6.4, it is easy to deduce where the recipient and date are located.

When specifying the logical components to be extracted from a business letter, one also should take care of the granularity with which these labels should be extracted. A typical business letter and its different logical components are shown in Fig. 6.12. It is important to note that the logical structure divides the contents of the letter into a hierarchy of logical objects. In the example in Fig. 6.12, only top-level logical objects are shown, except the *subscript* that is further divided into *subscriber*, *signature*, and *regards*. Other logical components could also be further subdivided, for instance, the *recipient* could be decomposed into *recipient name* and *recipient address*, whereas *recipient name* could further be split into *first name* and *last name*.

Although business letters have a lot of structural similarity, in practice one still encounters a wide variety of relative locations of different logical components. Furthermore, not all components might be present in a particular letter. Some sample letters highlighting the diversity in layouts of letters from the UNLV database are shown in Fig. 6.13.

One of the earliest works in the domain of business letter understanding was the ANASTASIL (Analysis System to Interpret Areas in Single-sided Letters) system [9, 10] by Dengel. The system identified important logical objects within business letters, like recipient, sender, or company-specific printings. The system utilized only geometric knowledge sources and worked completely independent of text recognition. The sources included global geometric knowledge about logical object arrangements, and local geometric knowledge about formal features of logical objects (e.g., extensions, typical font sizes). To model knowledge about more

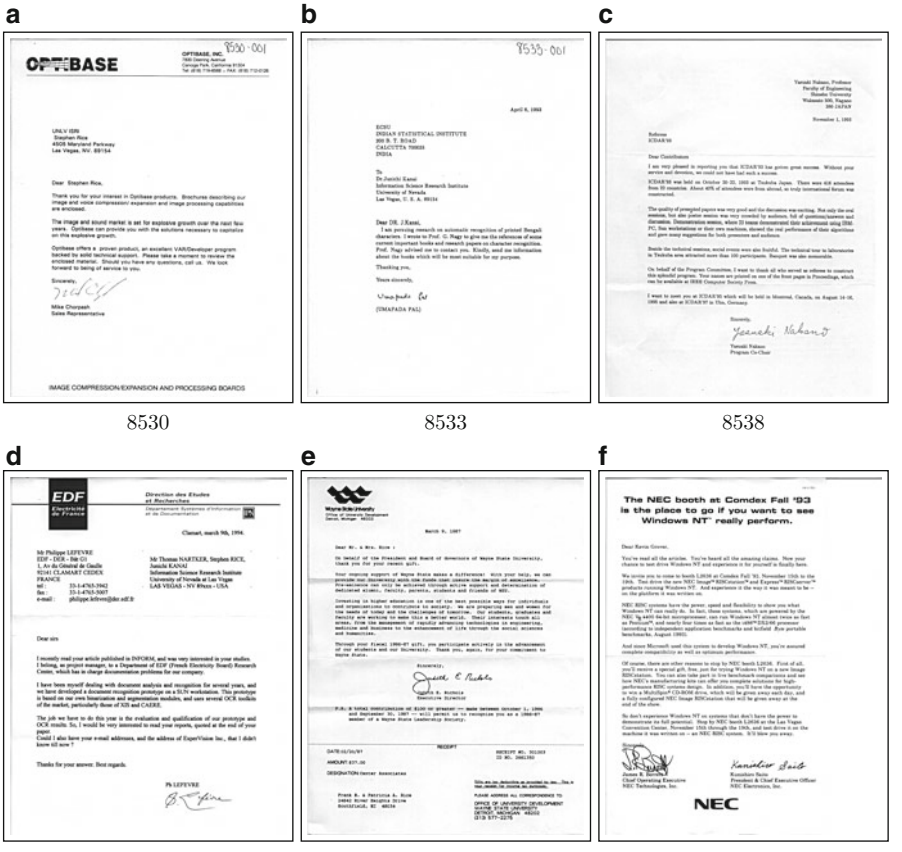

Fig. 6.13 Different samples from the UNLV database illustrating diversity in letter formats: (**a**) Date is missing. (**b**) No logo, header, or footer present. (**c**) No recipient name or address specified. (**d**) Sender details are on the *left side*, whereas recipient details are on the *right side*. (**e**) Sender and recipient details are provided at the *bottom* of the letter. (**f**) Multiple subscript blocks present

than one document layout and to obtain a compact knowledge representation, they used a geometric tree. The geometric tree described a global hierarchy of possible logical object arrangements. The tree was constructed using training data. When identifying logical labels of a given page image, the geometric tree is used as a decision tree.

While the ANASTASIL system focused on locating logical objects in a document image, its successor system ΠODA [12] also extracted text from the identified logical objects. Since logical labeling provided the identification of document constituents, the subsequent text recognition and verification were performed in an expectation-driven manner. For controlled selection of vocabularies, syntactic knowledge was attached to some of the generic logical classes. This knowledge

determined the order in which subordinate logical objects (e.g., ZIP code, city, and street in an address block) can occur and, therefore, controlled the access to corresponding logical vocabularies.

Dengel and Dubiel [11] described a system (DAVOS) capable of both learning and extracting document logical structure. DAVOS learned document structure concepts by detecting distinct attribute values in document objects. The structural concepts were represented by relation patterns defined by a cut-and-label language. A geometric tree was used to represent the concept language. Unsupervised decision tree-based learning techniques were used to build the tree. Forty letters were used to train the system, and then the learned geometric trees were used to classify another set of 40 unknown letters.

Klink and Kieninger [22] presented a hybrid approach for document structure understanding. They make use of layout (geometric) as well as textual features of a given document. Based on these features, rules are formulated to indicate how one might observe a specific layout object in a document. Besides, the rules also express dependencies between different layout objects. For each block in a document image, it is checked which rules in the rule base are satisfied. This matching of rules is done on a fuzzy basis. Finally, different matched rules are combined into a single probability score for the logical label of that block.

Technical Journals, Magazines, and Newspapers

Periodicals constitute a major portion of printed material, whether they be technical journals, newspapers, or magazines on a wide variety of subjects. Digitization of such periodicals has been a major driving force in document analysis research (see Table 6.5), since it is desirable to have centralized databases that index articles within a field. For instance, in the case of technical journals, several indexing projects exist to give a comprehensive overview of research in particular fields. A prominent example is MEDLINE, the flagship bibliographic citation database of the US National Library of Medicine that indexes more than 11 million articles from over 4,800 indexed titles in the fields of medicine and health sciences.

To facilitate automatic data entry of bibliographic information, one needs to identify titles and authors of different articles. Furthermore, automatic detection of headlines is desirable to allow navigation within an article. A major challenge in automatic bibliographic metadata extraction projects is to deal with a large variety of layouts that journals or magazines follow. Besides, articles within a particular journal may also show significant variations in position of different logical blocks depending on their length or presence of other optional logical elements. An example illustrating these interclass variations in layouts is shown in Fig. 6.14.

Pioneering work in the field of understanding technical journal articles were by Nagy et al. [23, 27] and Tsujimoto et al. [35]. The approach by Nagy et al. relied on page grammars to decompose a page image into its different logical components with a recursive X-Y cut method. Their method transforms a two-dimensional segmentation and labeling problem into a one-dimensional segmentation and labeling

Table 6.5 An overview of key approaches for analysis of articles from technical journals, magazines, and newspapers

Reference	Input features	Target domain(s)	Logical labels	Key idea	Evaluation metric	Experimental dataset
Tsujimoto and Asada [35]	Geometric layout tree	Business letters, technical journals, magazines, newspapers	Title, abstract, subtitle, paragraph, header, footer, page number, caption	Transform geometric layout tree to logical tree using a set of rules	Not specified	Private
Nagy et al. [27]	Raw image	Technical journals	Title, abstract, page number, keywords, section titles, captions, etc.	Syntactic segmentation and labeling of pages using block grammars defined for each layout type	Success rate	Private
Altamura et al. [2]	Page segments, geometric layout tree	Technical journals	Title, author, abstract, body text, etc.	Machine learning to extract rules for logical labeling	Labeling accuracy	Private
Klink and Kieninger [22]	Page segments, OCR result	Business letters, technical journals	Business letters (addressee, subject, date, sender, closure) Journals (header, title, author, abstract, bibliography)	Human expert defines rules, which are then matched with a fuzzy combination to label new documents	Precision, recall	UW-I Private
Aiello et al. [1]	Geometric features, content features, neighborhood relationships	Technical journals, magazines newspapers	Title, body, caption, page number	A knowledge base is created based on input features. A decision tree learner is used to assign logical labels	Precision, recall	UW-II MTDB

Tan and Liu [34]	Raw image	Newspapers	Headlines	Rules applied to smeared image	Precision, recall	Private
Eglin and Bres [15]	Page segments	Technical journals, magazines, newspapers	Title, sub-title, paragraph, footnote, caption	Distinguish labels of page segments based on their visual saliency	Labeling accuracy	MTDB
van Beusekom et al. [37]	Page segments	Technical journals	Title, author, abstract, affiliation	Case-based reasoning method: labeling based on correspondence of blocks with most similar layout	Labeling accuracy	UW-II MARG
Staelin et al. [32]	Page segments, OCR results	Journal articles, grant deeds	Journal articles (title, author, pages, date, etc.), Grant deeds (deed number, county, grantor, grantee, etc.)	A hierarchy of self-tuning classifiers (neural networks and SVMs)	Precision, recall	Private
Zou et al. [43]	OCR result, HTML tags	Technical journal	Bibliographic references locating and parsing	Machine learning to first identify and then parse references	Labeling accuracy	Private
Rangoni et al. [28]	Page segments, OCR results	Technical journals	21 logical labels (title, author, abstract, page number, etc.)	Time-delay neural network analogous to human perception	Labeling accuracy	MARG

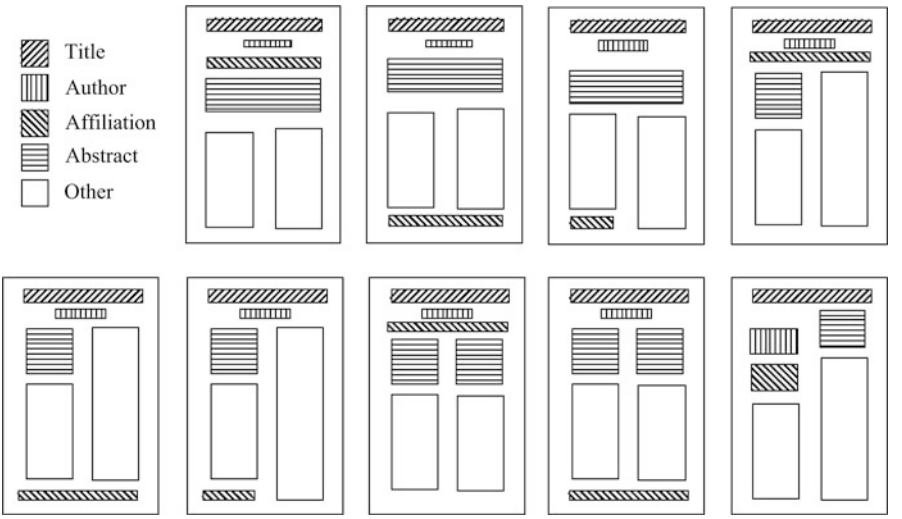

Fig. 6.14 Variations in layout of different journals with respect to position of various logical entities in the MARG dataset

problem without distinguishing between physical layout and logical structure. Tsujimoto et al. [35] represented document physical layout and logical structure as trees and tackled document understanding as the transformation of a physical tree into a logical one using a set of generic transformation rules.

Altamura et al. [2, 17] presented a structured document understanding system called WISDOM++. The presented system implements the complete processing pipeline including preprocessing, page segmentation, block classification, document classification, and logical labeling. The logical labeling module associates page layout components with basic logical components. The mapping is achieved by matching the document description against models of classes of documents and models of logical components interesting for that class. Both layout structures and models are described in a first-order language, where unary and binary function symbols, called attributes and relations, are used to describe properties of single layout components and their relative position. A first-order learning system is applied for the induction of rules used for the layout-based classification and understanding of documents. A free version of WISDOM++ software package is available at http://www.di.uniba.it/∼malerba/wisdom++/.

Aiello et al. [1] presented a knowledge-based system for logical structure extraction from a broad class of documents. To keep the system generic enough, they considered extraction of only title, body, caption, and page number as logical entities. The contents of a document were described using textual and geometric features. The content features of a document object (block) consist of its aspect ratio, area ratio, font size ratio, font style, number of characters, and number of lines. The geometric features describe global arrangement of objects on the page as well as

spatial relation between objects. For global arrangement, neighborhood relations are used (two objects are considered neighbors if they share an edge in the Voronoi diagram of the centers of the document objects). Spatial relations between objects are described as thick boundary rectangle relations on both axes. These relations include precedes, meets, overlaps, starts, during, finishes, equals, and their inverse. A feature vector consisting of all of these features is used to train a C4.5 decision tree classifier to distinguish the desired logical objects (title, body, caption, page number).

Identification of title and author areas in a technical document image using a Delaunay triangulation-based method was presented in [41]. First, the positions and alignment of small text line regions are measured by different triangle groups. Then, character stroke widths are calculated from the constrained Delaunay triangulation. Finally, the rules defining spatial features and font attributes of the title and author region are applied to single line text regions to extract title and author regions.

A method for assigning functional labels to pre-segmented homogeneous page segments using a psycho-visual approach was presented by Eglin and Bres [15]. Their method classifies text blocks into headings, body paragraphs, and highlighted text according to their visual saliency and perceptual attraction power on the reader's eye. The main idea is to characterize text blocks using their complexity, visibility, and geometric characteristics. The complexity captures the frequency of transitions between text components, whereas the visibility estimates the density of these transitions. The geometric features correspond to the dimensions and location of each text block on the page. Finally, a decision tree is trained on these features to perform functional labeling.

A system for extracting headlines from newspaper microfilms was presented by Tan and Liu [34]. The goal was to provide automatic indexing of news articles by extracting headlines from digitized microfilm images to serve as news indices. Since the aim was to extract only prominent headlines, a method based on the run-length smearing algorithm (RLSA) [40] was proposed to extract headlines without the need for detailed layout analysis. The method was based on computing statistics of blocks returned by RLSA algorithm to first distinguish text blocks from non-text blocks and then to differentiate a headline from other text blocks.

An example-based machine learning system to adapt to customer-defined document and metadata type, called Biblio, was presented in [32]. One key element of Biblio's design was that it did not require a skilled operator to tune the underlying machine learning algorithm. For training the system, many example documents of a single type are fed to the system. Each document type has its own set of neural networks and support vector machines for detailed processing. SVMs are used to create metadata dictionaries that represent the words associated with a given metadata type. These dictionaries are used by neural networks while looking for evidence of data specific to a document type. Multiple neural networks are trained and combined in the form of committees to improve the overall accuracy of the system.

Rangoni et al. [28] have presented a dynamic perceptive neural network model for labeling logical structures in technical documents. Their system combines

capabilities of model-driven and data-driven approaches. The main idea is to integrate knowledge as interpretable concepts in a perceptive (also called transparent) neural network. The system performs several recognition cycles and corrections while keeping track of and reusing the previous outputs. This dynamic behavior is achieved by using a time-delay neural network. Using several cycles, their system is able to make corrections to the input when ambiguous inputs are encountered.

Other Application Areas

A large diversity of documents exists around us that is not limited to one of the abovementioned categories (see Table 6.6). Different researchers have developed specialized techniques for understanding non-stereotypical documents. In the following, a few prominent contributions for different categories of documents are outlined.

Lemaitre et al. [24] presented a method inspired by perceptive vision phenomena for naturalization decree register treatment. The analysis of naturalization decree registers aimed at extracting register numbers and surnames for each act contained in the document. The method was applied on more than 15,000 registers from the French national archives, dated between 1883 and 1930, representing about 85,000 pages. To be robust against noise and poor quality of the documents, the authors used a multi-resolution approach. Different resolutions were built successively by low-pass filtering the initial image. Then, for each chosen resolution, suitable features were extracted. Since the low-resolution version contains only a global perspective of the document, initial detection of surname and register number regions is done at that level. This detection is then further refined at higher resolution to detect the actual locations of the desired logical objects.

Wang et al. [38] proposed a specialized method to analyze tender documents. In a tender, there is no indexing information such as a table of contents (ToC) to indicate different logical units of the document. Its hierarchical logical structure is embodied in citing relationships between its different pages. A tender document can be divided into several logical parts, called bids. Every bid can be divided into either sub-bids or basic pages, and every sub-bid is also composed of basic pages. The main part of every page is a form containing several items. An item is the basic unit that can be priced in a tender document. At the end of every part, such as a sub-bid, bid, and the whole document, there exist several pages containing the summary information of this part. The logical structure of the tender includes not only the structure within a single page but also the relationship between different pages (e.g., a bill page may refer to several sub-bill pages). Hence, instead of only a sequential order, a citing order also exists between pages. This means that one page is not only before or after another page, but it also may be cited by or cite other pages. Since the citing information is essential for generating the hierarchy of the tender, the extraction of the logical structure of the tender is therefore more difficult than that of a common document. The system developed by Wang et al. called VHTender consists of three steps. First, preprocessing is done to correct skew angle of the

Table 6.6 An overview of document understanding methods for certain niche domains

Reference	Input features	Target domain(s)	Logical labels	Key idea	Evaluation metric	Experimental dataset
Wang et al. [38]	Page segments, geometric layout tree, OCR result	Tender documents	Extraction of citing relationships between pages to reconstruct hierarchy of a tender	A knowledge base encodes domain-specific rules. A rule learning method is applied to learn and then extract logical labels	Labeling accuracy (percentage of correctly assigned labels)	Private
Stælin et al. [32]	Page segments, OCR results	Journal articles, grant deeds	Journal articles (title, author, pages, date, etc.) Grant deeds (document number, county, grantor, grantee, etc.)	A hierarchy of self-tuning classifiers (neural networks and SVMs)	Precision, recall	Private
Lemaitre et al. [24]	Raw image	Naturalization decree	Serial number, candidate name	Multi-resolution image analysis mimicking human perception	Precision, recall	Private
Medvet et al. [26]	Text blocks and their OCR result	Invoices, patents, datasheets	Eight labels in invoices, 11 labels in patents, 8 labels in datasheets	Probabilistic modeling of known layouts, user interaction to model a new layout	Success rate (percentage of documents with no labeling error)	Ghega

scanned pages. Then, the page analyzer extracts the geometric layout of the page. In the last step, the page grouper module synthesizes the layout information of every page into a physical structure tree of the tender and analyzes the citing information to deduce and create the logical structure tree. A knowledge base storing domain-related rules is used to aid both page analyzer and page grouper modules.

Logical labeling of patent applications was investigated by Medvet et al. [26]. Their probabilistic modeling approach was designed for application scenarios in which the set of possible document classes may evolve over time. Patent applications make a suitable test case for this approach since different patent sources have different layouts. The labels that they automatically extracted from patent applications were title, applicant, inventor, representative, filing date, publication date, application number, publication number, priority, classification, and first line of abstract. When a patent application from a new source is to be indexed, the operator can make the corresponding model using a GUI. Later, when a new application from the same source is encountered, the logical labels are identified as the sequence of blocks that are most probable given the model.

Experimental Validation

Performance Measures

To evaluate a logical labeling algorithm, one needs to take into account the perspective and scope of the target application. When logical labeling is done individually as a module of a rather complex document understanding system, the target of evaluation is to find out the percentage of correctly identified labels. From the perspective of an algorithm/system developer, one is further interested in identifying different classes of errors quantitatively, whereas end users are typically only interested in overall performance of the algorithm (irrespective of the types and number of different kinds of errors). Similarly, in many cases, the results of other modules like page classification or OCR are combined to give an overall error rate of the system.

Consider an automatic system for processing business letters as a use case. When evaluating such a system, one is interested in finding out what percentage of *actual* labels, e.g., insurance number, has been correctly recognized. However, the incoming mail sorting system might not identify a letter correctly and label it as a form. That would increase the error rate of the system despite the fact that logical labeling itself had no chance of fixing this error. Similarly, the presence of OCR errors would also reduce the accuracy of the system. Hence, what is actually being measured, even when using the same metrics, might vary from one publication to the other.

The most general and widely used measures in different domains of logical labeling are precision and recall. These measures are well understood in information retrieval community and applying them to a logical labeling problem is straightforward. The precision and recall are defined for every logical label l as

$$\text{Precision}_l = \frac{|D_l \cap G_l|}{|D_l|} \qquad \text{Recall}_l = \frac{|D_l \cap G_l|}{|G_l|}$$

where D_l represents the set of document objects classified by the system as logical objects with label l and G_l represents the set of document objects with the label l in the ground truth. The overall precision and recall of the system can be computed as the average precision and recall for all the labels. For a more detailed discussion of these measures, please refer to ▸Chap. 30 (Tools and Metrics for Document Analysis Systems Evaluation).

Datasets

Public ground-truth datasets are crucial for progressing the state of research in many fields of computer science by providing common grounds for comparing different algorithms. The need for such datasets was identified early in the document analysis community, and several ground-truth datasets emerged for different target applications. For logical labeling, there are also several ground-truth datasets available. A brief overview of these datasets is given here. The reader is referred to ▸Chap. 29 (Datasets and Annotations for Document Analysis and Recognition) for details.

University of Washington Datasets (UW-I/II/III)

The UW datasets are probably the most widely used datasets in the document analysis community. The UW-III dataset, which is the latest one in the series, consists of 1,600 scanned pages from technical journal articles with an extensive ground truth for each page. These datasets have been used for validation of logical labeling approaches in [1, 22].

Oulu MediaTeam Documents Datasets (MTDB)

The MTDB is a collection of scanned documents and the corresponding ground-truth data about the physical and logical structure of the documents. It consists of 500 document images from a wide diversity of sources (journal articles, newspaper articles, advertisement, correspondences, etc.). Use of this dataset for evaluating logical labeling methods can be found in [1, 15].

Medical Articles Records Ground-Truth Datasets (MARG)

The MARG dataset contains title pages of medical journal articles from various publishers. The title pages have varying layouts. For each title page, ground-truth boxes containing the title, authors, affiliation, and abstract are provided. This dataset was used in [28] for evaluating their methods.

INFORMys Invoice Datasets

This dataset consists of 100 invoices from Italian TELECOM and 182 invoices from Italian electricity company ENEL. It was used in [6] to validate their invoice reading

system and is still available at http://www.dsi.unifi.it/∼simone/Dante/informys/demo/node24.html.

Ghega Datasheets and Patents Datasets

This dataset is composed of 110 datasheets of electronic components and 136 patent documents each divided into 10 classes. Datasheets of the same class share the producer and the component type. Multiple logical labels (model, type, case, power dissipation, etc.) are identified for each datasheet. For patent documents, each class represents one patent source. Several logical labels are identified (title, applicant, filing date, publication date, etc.) on each patent document. The dataset was used to demonstrate application of the document understanding approach in [26] on documents from different domains. The dataset is publicly available at https://sites.google.com/site/ericmedvet/research/ghega-dataset.

Recommendations

Choosing a logical labeling approach that best fits the needs of a project is not an easy task. One not only has to choose among a large variety of algorithms, but also the basic assumptions underlying different approaches need to be carefully investigated. Besides, choosing the right balance between specificity and generality is also crucial. While it is natural to desire having a general-purpose system, the cost of developing such a system might not be justified in the context of the project. Hence, a very important step is to do a thorough requirement analysis for the logical labeling module of the project. Questions one may ask at this stage are:

- What is the domain of documents that need to be processed? Is the project targeting documents from multiple domains? Is the domain of each document to be processed known a priori? Is the domain of documents to be processed static or dynamic?
- How variable is the layout of documents from a specific class? Do the documents always capture all logical components or are some of the elements optional?
- What is the intended role of the logical labeling module? Which set of logical labels are absolutely necessary for executing the project? Which labels can be considered optional?
- What is the scope of the project? Is it a one-time conversion project, or will the system remain operational to handle new documents every day?
- Who are the intended users of the system? Will the system run autonomously without a human operator? Will end users have the skills to adapt the system to new document classes?
- What are the requirements on the operational speed of the system? In other words, what is the volume of the documents to be processed each day by the system?
- How stringent are the accuracy requirements? What are the costs of different kinds of errors made by the system?

If the domain of each document to be processed is known, one can limit the search to specifically tailored methods for that domain (cf. Table 6.1). Otherwise, one of the generic methods (e.g., [1, 22, 26]) could be used. It is interesting to note that each of these methods has their particular strengths. The method by Klink and Kieninger [22] is a rule-based system in which an expert can define rule expressions (one rule for each desired logical label). Such rules could be defined by some domain expert who might not be from a computer science background. Hence, in extreme cases, one could develop such a system based only on the domain knowledge of the expert, without having any training data at hand. Application of these rules using fuzzy logic is also straightforward, and hence the system overall is easy to implement. The approach by Aiello et al. [1] is a knowledge-based approach that uses machine learning to draw inference based on the knowledge base. This approach is quite suitable if one has labeled training data available. The learning algorithm (a decision tree in this case) learns relations from data and automatically builds a model (set of rules) to assign logical labels based on the observed features of the document objects. The method presented by Medvet et al. [26] is designed for scenarios in which the set of target document domains is dynamic and may evolve over time. They provide a graphical user interface (GUI) with which an unskilled operator can train a new class/domain of documents by merely providing a few samples and defining regions of interest with their labels. A probabilistic approach is then used to automatically estimate the parameters of the newly defined class, which can then be used to label new documents of that class.

As a general recommendation, it should be noted that the amount of effort required for setting up a logical labeling system and the achievable error rates are both often underestimated. This is probably due to the ease with which humans can readily extract logical information from a given document. Hence, when designing a logical labeling system, one should not try to solve the problem in a more general way than required. This will not only save development time but also result in more accurate system for the target application.

Conclusion

Logical labeling is an integral part of most of the document analysis systems employed in digitization workflows. Owing to the central role of logical labeling in extracting semantic information from documents, a major effort in the document analysis community has been spent on it for over two decades. Hence, a large variety of methods exist for logical labeling using diverse concepts from artificial intelligence and computer science – not only present in the cited work, but manyfold proposed in the respective conference proceedings of this field. Initial efforts in logical labeling focused on processing documents from one particular domain with a limited variety of layouts. As the field matured, methods capable of handling documents from multiple domains emerged. In the meantime, several publicly available ground-truth datasets were developed. This allowed researcher not only to better perform comparative evaluations but also to benchmark on representative

samples of the target domain. Recent systems for logical labeling [28] report about 98 % accuracy on large public datasets containing documents with diverse layouts.

Despite the impressive advances in the functionality of the state-of-the-art logical labeling methods, their capabilities still remain far behind human skills. The ease and efficiency with which humans can just skim a document (even if the document is in a foreign language) to get its logical structure are yet to be matched by machines. Hence, much room for improvement remains in existing methods, and it is expected that more generic as well as accurate systems will be developed in the near future.

Cross-References

▸Datasets and Annotations for Document Analysis and Recognition
▸Imaging Techniques in Document Analysis Processes
▸Page Segmentation Techniques in Document Analysis
▸Page Similarity and Classification
▸Recognition of Tables and Forms
▸Tools and Metrics for Document Analysis Systems Evaluation

References

1. Aiello M, Monz C, Todoran L, Worring M (2002) Document understanding for a broad class of documents. Int J Doc Anal Recognit 5(1):1–16
2. Altamura O, Esposito F, Malerba D (2001) Transforming paper documents into XML format with WISDOM++. Int J Doc Anal Recognit 4(1):2–17
3. Bayer T, Franke J, Kressel U, Mandler E, Oberländer M, Schürmann J (1992) Towards the understanding of printed documents. In: Baird H, Bunke H, Yamamoto K (eds) Structured document image analysis. Springer, Berlin/New York, pp 3–35
4. Belaïd A (2001) Recognition of table of contents for electronic library consulting. Int J Doc Anal Recognit 4(1):35–45
5. Cattoni R, Coianiz T, Messelodi S, Modena CM (1998) Geometric layout analysis techniques for document image understanding: a review. Technical report 9703-09, IRST, Trento, Italy
6. Cesarini F, Gori M, Marinai S, Soda G (1998) INFORMys: a flexible invoice-like form-reader system. IEEE Trans Pattern Anal Mach Intell 20(7):730–746
7. Cesarini F, Francesconi E, Gori M, Soda G (2003) Analysis and understanding of multi-class invoices. Int J Doc Anal Recognit 6(2):102–114
8. Déjean H, Meunier J (2009) On tables of contents and how to recognize them. Int J Doc Anal Recognit 12(1):1–20
9. Dengel A (1992) ANASTASIL: a system for low-level and high-level geometric analysis of printed documents. In: Baird H, Bunke H, Yamamoto K (eds) Structured document image analysis. Springer, Berlin/New York, pp 70–98
10. Dengel A, Barth G (1988) High level document analysis guided by geometric aspects. Int J Pattern Recognit Artif Intell 2(4):641–655
11. Dengel A, Dubiel F (1996) Computer understanding of document structure. Int J Imaging Syst Technol 7:271–278
12. Dengel A, Bleisinger R, Hoch R, Fein F, Hönes F (1992) From paper to office document standard representation. IEEE Comput 25(7):63–67

13. Doucet A, Kazai G, Dresevic B, Uzelac A, Radakovic B, Todic N (2011) Setting up a competition framework for the evaluation of structure extraction from OCR-ed books. Int J Doc Anal Recognit 14(1):45–52
14. Duygulu P, Atalay V (2002) A hierarchical representation of form documents for identification and retrieval. Int J Doc Anal Recognit 5(1):17–27
15. Eglin V, Bres S (2004) Analysis and interpretation of visual saliency for document functional labeling. Int J Doc Anal Recognit 7(1):28–43
16. e Silva AC, Jorge AM, Torgo L (2006) Design of an end-to-end method to extract information from tables. Int J Doc Anal Recognit 8(2–3):144–171
17. Esposito F, Malerba D, Lisi F (2000) Machine learning for intelligent processing of printed documents. J Intell Inf Syst 14(2–3):175–198
18. Fan H, Zhu L, Tang Y (2010) Skew detection in document images based on rectangular active contour. Int J Doc Anal Recognit 13(4):261–269
19. Kazai G, Doucet A (2008) Overview of the INEX 2007 book search track (BookSearch'07). SIGIR Forum 42(1):2–15
20. Klein B, Dengel A (2003) Problem-adaptable document analysis and understanding for high-volume applications. Int J Doc Anal Recognit 6(3):167–180
21. Klein B, Agne S, Dengel A (2006) On benchmarking of invoice analysis systems. In: Proceedings of the international workshop on document analysis systems, Nelson, pp 312–323
22. Klink S, Kieninger T (2001) Rule-based document structure understanding with a fuzzy combination of layout and textual features. Int J Doc Anal Recognit 4(1):18–26
23. Krishnamoorthy M, Nagy G, Seth S, Viswanathan M (1993) Syntactic segmentation and labeling of digitized pages from technical journals. IEEE Trans Pattern Anal Mach Intell 15(7):737–747
24. Lemaitre A, Camillerapp J, Coüasnon B (2008) Multiresolution cooperation makes easier document structure recognition. Int J Doc Anal Recognit 11(2):97–109
25. Lin X, Xiong Y (2006) Detection and analysis of table of contents based on content association. Int J Doc Anal Recognit 8(2–3):132–143
26. Medvet E, Bartoli A, Davanzo G (2011) A probabilistic approach to printed document understanding. Int J Doc Anal Recognit 14(4):335–347
27. Nagy G, Seth S, Viswanathan M (1992) A prototype document image analysis system for technical journals. IEEE Comput 7(25):10–22
28. Rangoni Y, Belaïd A, Vajda S (2012) Labelling logical structures of document images using a dynamic perceptive neural network. Int J Doc Anal Recognit 15(2):45–55
29. Schürmann J, Bartneck N, Bayer T, Franke J, Mandler E, Oberländer M (1992) Document analysis – from pixels to contents. Proc IEEE 80(7):1101–1119
30. Shafait F, Breuel TM (2011) The effect of border noise on the performance of projection based page segmentation methods. IEEE Trans Pattern Anal Mach Intell 33(4):846–851
31. Shafait F, van Beusekom J, Keysers D, Breuel TM (2008) Document cleanup using page frame detection. Int J Doc Anal Recognit 11(2):81–96
32. Staelin C, Elad M, Greig D, Shmueli O, Vans M (2007) Biblio: automatic meta-data extraction. Int J Doc Anal Recognit 10(2):113–126
33. Story GA, O'Gorman L, Fox D, Schaper LL, Jagadish HV (1992) The rightpages image-based electronic library for alerting and browsing. IEEE Comput 25:17–26
34. Tan CL, Liu QH (2004) Extraction of newspaper headlines from microfilm for automatic indexing. Int J Doc Anal Recognit 6(3):201–210
35. Tsujimoto S, Asada H (1992) Major components of a complete text reading system. Proc IEEE 80(7):1133–1149
36. van Beusekom J, Keysers D, Shafait F, Breuel TM (2007) Example-based logical labeling of document title page images. In: Proceedings of the international conference on document analysis and recognition, Curitiba, pp 919–923
37. van Beusekom J, Shafait F, Breuel TM (2010) Combined orientation and skew detection using geometric text-line modeling. Int J Doc Anal Recognit 13(2):79–92

38. Wang S, Cao Y, Cai S (2001) Using citing information to understand the logical structure of document images. Int J Doc Anal Recognit 4(1):27–34
39. Wang Y, Phillips I, Haralick R (2004) Table structure understanding and its performance evaluation. Pattern Recognit 37(7):1479–1497
40. Wong KY, Casey RG, Wahl FM (1982) Document analysis system. IBM J Res Dev 26(6):647–656
41. Xiao Y, Yan H (2004) Location of title and author regions in document images based on the Delaunay triangulation. Image Vis Comput 22(4):319–329
42. Yu B, Jain AK (1996) A generic system for form dropout. IEEE Trans Pattern Anal Mach Intell 18(11):1127–1134
43. Zou J, Le D, Thoma G (2010) Locating and parsing bibliographic references in HTML medical articles. Int J Doc Anal Recognit 13(2):107–119

Further Reading

Aiello M, Monz C, Todoran L, Worring M (2002) Document understanding for a broad class of documents. Int J Doc Anal Recognit 5(1):1–16
Medvet E, Bartoli A, Davanzo G (2011) A probabilistic approach to printed document understanding. Int J Doc Anal Recognit 14(4):335–347
Rangoni Y, Belaid A, Vajda S (2012) Labelling logical structures of document images using a dynamic perceptive neural network. Int J Doc Anal Recognit 15(2):45–55