

Chapter 2

DOCUMENT LAYOUT ANALYSIS SYSTEMS

In this chapter, we first review some of the systems in the area of document layout analysis area. In Section 2.2, we will describe a real document layout analysis system: a PDF-XML conversion system, in which most of the dissertation research was conducted.

2.1 Introduction

The goal of document image layout analysis is to specify the geometry of the maximal homogeneous regions and the spatial relations of these regions in a document image. Specifically, the problems in this field include: text word segmentation, text line segmentation, text block segmentation, table structure understanding, zone content classification, etc. Lots of research work have been done on these problems. Some systems are reported in the literature [40, 54, 61]. A survey of document image analysis research can be found in [71].

Most of the current document layout analysis systems assume some prior knowledge of the typographical and layout conventions of document. The required information about the document properties ranges from very specific and precise to fairly general ideas. By considering the knowledge representation and structure recognition approaches, the structure analysis algorithms can be categorized into two main types: rule or grammar based, and statistical model based. The grammar based approaches use multiple grammars to represent the recognition algorithm. The recognition task is formulated as a parsing problem using one dimensional string pattern matches.

Hand-crafted rules and models tend to be brittle and only work for a specific kind of document. On the other hand, in statistical-based approaches, the probabilities are estimated from an extensive training set. The off-line probabilities estimated in the training then drive all decisions in the on-line analysis modules. Thus it has the advantages such as domain independent and easy to be extended.

Jain and Yu [40] developed a bottom-up method to partition a page into columns of *text, drawings, images, table regions, and rulers*. Following the application of a hierarchical Hough transform to connected components, the estimated skew is accommodated by introducing generalized text lines. Foreground pixels are grouped into rectangular blocks with adjacent same-length horizontal runs preserved as nodes in a Block Adjacency Graph (BAG). The BAG nodes are successively grouped into connected components, text lines, and region blocks, and simultaneously inserted into a novel page model. The segmented regions are classified using empirical rules and thresholds. The model is a tree where the branches of the root correspond to the five types of entities, and the leaves are the BAG nodes. Selected results from performance tests on 150 varied page images are illustrated in color.

Lee et.al [54] present a knowledge-based method for sophisticated geometric structure analysis of technical journal pages. The proposed knowledge base encodes geometric characteristics that are not only common in technical journals but also publication-specific in the form of rules. The method takes the hybrid of top-down and bottom-up techniques and consists of two phases: region segmentation and identification. The knowledge rules are also divided into region segmentation and identification rules according to the stage where they are applied. Generally, the result of the segmentation process does not have a one-to-one matching with composite layout components. Therefore, the proposed method identifies nontext objects, such as images, drawings, and tables, as well as text objects, such as text lines and equations, by splitting or grouping segmented regions into composite layout components. Experimental results with 372 images scanned from the *IEEE Transactions on Pattern*

Analysis and Machine Intelligence show that the method has performed geometric structure analysis successfully on more than 99 percent of the test images.

Liang et. al [61] give a formal definition of the document image structure representation and formulate document image structure extraction as a partitioning problem: finding an optimal solution partitioning the set of glyphs of an input document image into a hierarchical tree structure where entities within the hierarchy at each level have similar physical properties and compatible semantic labels. A unified methodology that is applicable to construction of document structures at different hierarchical levels is proposed. An iterative, relaxation-like method is used to find a partitioning solution that maximizes the joint probability. All the probabilities used in the partitioning process are estimated from an extensive training set of various kinds of measurements among the entities within the hierarchy. The probabilities estimated in the training then drive all decisions in the on-line document structure extraction. A text-line extraction algorithm using this framework is implemented. The algorithm was evaluated on the UW-III database of some 1600 scanned document image pages. An area-overlap measure is used to find the correspondence between the detected entities and the ground-truth. For a total of 105,020 text lines, the text-line extraction algorithm identifies and segments 104,773 correctly, an accuracy of 99.76%.

Among the current systems, some specific problems, such as table structure understanding, zone content classification, are either not solved [61] or solved by sophisticated empirical rules [40, 54]. Thus the current systems are strongly domain dependent. To build a document layout analysis which performs nearly perfectly, we have to develop algorithms to be proved out on large data sets with suitable performance metrics for each of these problems. My dissertation work focus on table structure understanding, zone content classification and text word segmentation problems. We use statistical instead of rule based algorithms to solve these problems. We develop and use large image data sets with carefully made ground truth data. We discuss different performance metrics for each of the problems and use them to

evaluate our algorithms. Most of my research was done in a real document layout analysis system scenario: *PDF-XML Conversion System*. Together with other group members, we built this system. The system is running on a commercial server and is supplying automatic document conversion service to the public. In Section 2.2, we give a brief introduction to this system.

2.2 A Document Layout Analysis System: PDF-XML Conversion System

This system is designed to convert a *PDF* document into a well partitioned *SGML/XML* representation, say *GPML*, which reliably identifies the basic structural page components present in typical printed documents [96]. The design of the system architecture is given in Figure 2.1. There are four modules in the system. The *PDF2CSV* module converts a *PDF* file into a *CSV* (Comma Separated Value) file. The *CSV2DAFS* module translates a *CSV* file into a *DAFS* (Document Analysis File Structures) file, where the *DAFS* file consists of only one basic-level structure of glyphs. The *UW DLA* (Document Layout Analysis) module partitions the basic-level *DAFS* structure to produce a hierarchical *DAFS* structure. The hierarchical *DAFS* structure then can be converted to a *GPML* file, by the *DAFS2GPML* module. Our sponsor provided to the UW team the *PDF2CVS* converter and UW team worked on the remaining three modules.

In Figure 2.1, *PDF* refers to the Portable Document Format [65] which is designed by Adobe®. *CSV* refers to Comma Separated Values, which specifies different entities with their geometric positions and various properties. In the following subsections, we give a brief introduction to XML, *DAFS* structure, document structure analysis module and markup module.

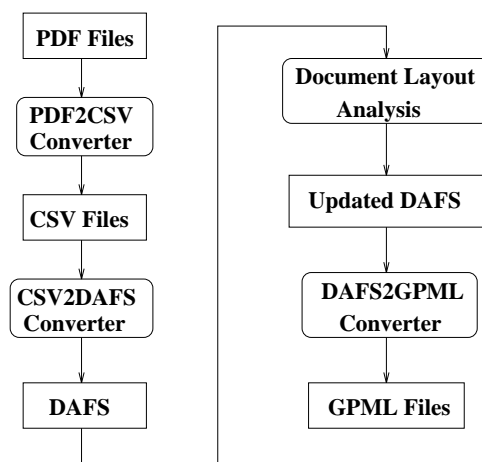


Figure 2.1: The Process Diagram of the *PDF-XML* conversion system

2.2.1 About XML Language

With the advance of web technology, Standard Generalized Markup Language (SGML, ISO 8879)/Extensible Markup Language (XML) enable the researchers to fully explore the results of their document analysis systems. As a markup language, XML has numerous benefits and applications [84]. In the field of document analysis, XML can be used as a tool for transmitting and rendering the recognition results. Using the XML representation, the recognition results also can be used in information retrieval easily.

2.2.2 DAFS File Hierarchy

The Document Attribute Format Specification (DAFS) [36] is a file format specification suitable for document decomposition applications, such as the document layout analysis, OCR and the logical analysis. The DAFS utilizes several existing document interchange standards, such as the standard generalized markup language (SGML, ISO 8879), the standard universal character set (UNICODE, ISO 10606), and the CCITT group IV standard for bi-level image compression.

DAFS provides a format for breaking down documents into entities, e.g. glyphs and words, defining entity boundaries and attributes, and labeling their content (text values and/or bitmap images). The DAFS entities are conveniently defined as the objects of interest within a document such as a paragraph, or text line, or word.

DAFS permits the creation of “parent”, “child” and “sibling” relationship among entities, forming the basis for specifying any hierarchical relationships desired. As an example, consider a paragraph entity made up of text lines, and the text lines composed of words. The words are the child entities of each text line, while the paragraph is the parent entity of each text line. The other text lines in the same paragraph forms a given text line’s siblings (see Figure 2.2).

In addition, DAFS permits the creation of an unlimited number of user defined properties. A property is used to describe or classify an entity and its contents, and exists only in association with the entity to which it refers.

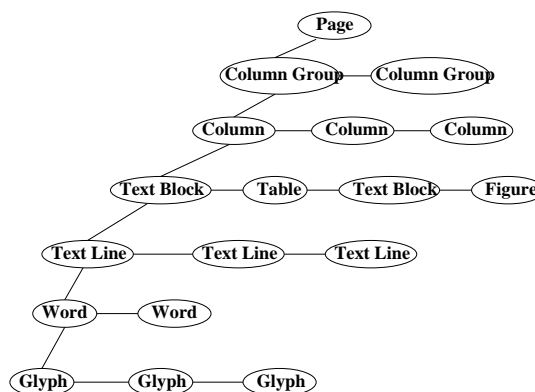


Figure 2.2: An example of a DAFS hierarchical structure.

2.2.3 The Document Layout Analysis Module

The input to the Document Layout Analysis module (DLA) is the glyph information, the location and the ASCII value of the glyph, from the OCR module (or other

source). From the glyph information, the DLA module first constructs a one-level DAFS structure where the entities are glyphs. The output of DLA is a hierarchical DAFS structure. The hierarchical structure (from top-to-bottom) consists of column-groups, columns, zones (paragraphs, tables, figures, etc.), text lines, words, and glyphs (see Figure 2.2 for an example.)

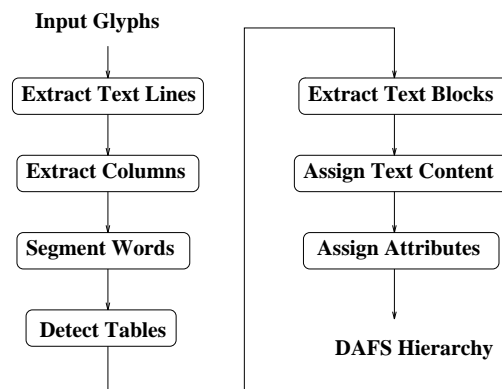


Figure 2.3: The DLA system architecture

There are seven sub-modules within the DLA module. (see Figure 2.3.) The DLA works as follows. The spatial configuration of glyph bounding boxes are analyzed to extract text lines ([59, 58]). The extracted text-lines are further segmented into words. From the extracted text-line, the text-columns and text-regions are formed to extract text-blocks (i.e. paragraphs). Within each text region, the neighboring text-lines are merged to form text-block (paragraph) based upon the changes in the inter-text-line spacing and the text-line justification [60]. Each tabular structure is further decomposed into row and column items for further analysis. For each text line, words and word spaces are formed into a text string. The formatting attributes are assigned to each text string. The result of this module is represented as a hierarchical document structure, a DAFS structure (see Figure 2.2.), which is comparable to the GPML syntax.

2.2.4 Markup Module

The input to the markup module is a hierarchical DAFS structure and the output of this module is a well partitioned SGML/XML representation, an GPML file. The elements defined within an GPML file are page, column-group, column, text-block, table, figure, and text-line. The GPML hierarchy is similar to that of DAFS hierarchy (see Figure 2.2). A column-group can have one or more column elements. A column element can have one or more text-block elements and non-text elements (such as tables, figures, etc.). A text-block element can have one or more text-line elements. The markup module directly translates the input DAFS structure to an GPML file according to the rules of the GPML definition.

Within a DAFS hierarchy, besides the content, each entity is associated with a list of attributes. The attributes we use are similar to those used in the UW databases [76]. We add a few formatting attributes to text-line elements.

The translation process works as follows. We visit all nodes in the input DAFS hierarchy in a depth-first fashion. At each node, we translate the contents and the attributes of the node into an GPML element statement, according to the node type and the GPML's syntax rules. For example, if the input DAFS hierarchy has a column-group entity that contains two columns. And within one of the columns there is a table, a figure, and two text-blocks, and the other column has five text-blocks, the translation result of the GPML file would have 14 elements – one page, one column-group, two columns, one table, one figure and eight text-block elements. (Here, we do not count the text-line elements.)

Within a GPML, the content of a text-block is stored as a sequence of text-line elements. A text-line element is represented as a character string with the formatting attributes attached to each character in the string. A GPML statement for a table element is a sequence of matrix elements, caption element and legend element. And each matrix element can have one or more cell elements. The caption and legend

elements are defined as text-blocks. A figure element in an GPML is given by the name of the bit-map file of the figure.

2.3 Summary

In this chapter, we review some of the document layout analysis systems. We discuss different techniques used in the existing systems and state several open problems. We also give a brief introduction to a real document layout analysis system scenario: *PDF-XML Conversion System*, in which most of my research work was conducted. Currently, this system is supplying automatic document conversion service to the public in a commercial server.