XML and Knowledge Technologies for Semantic-Based Indexing of Paper Documents

Donato Malerba, Michelangelo Ceci, and Margherita Berardi

Dipartimento di Informatica,
Università degli Studi
via Orabona, 4
70126 Bari - Italy
{malerba,ceci,berardi}@di.uniba.it

Abstract. Effective daily processing of large amounts of paper documents in office environments requires the application of semantic-based indexing techniques during the transformation of paper documents to electronic format. For this purpose a combination of both XML and knowledge technologies can be used. XML distinguishes between data, its structure and semantics, allowing the exchange of data elements that carry descriptions of their meaning, usage and relationship. Moreover, the combination with XSLT enables any browser to render the original layout structure of the paper documents accurately. However, an effective transformation of paper documents into XML format is a complex process involving several steps. In this paper we propose the application of knowledge technologies to many document processing steps, namely rule-based systems for semantic indexing of documents and the extraction of the necessary knowledge by means of machine learning techniques. This approach has been implemented in the system Wisdom++, which is currently used in the European project COLLATE (Collaboratory for Annotation, Indexing and Retrieval of Digitized Historical Archive Material) to provide film archivists with a tool for the automated annotation of historical documents in film archives.

1 Introduction

The increasingly large amount of paper documents to be processed daily in office environments requires new document management systems with abilities to catalog and organize these documents automatically on the basis of their contents. Personal document processing systems that can provide functional capabilities like classifying, storing, retrieving, and reproducing documents, as well as extracting, browsing, retrieving and synthesizing information from a variety of documents are in continual demand [5]. However, they operate on electronic documents and not on the more common paper documents. This issue is considered in the area of Document Image Analysis (DIA), which investigates the theory and practice of recovering the symbol structure of digital images scanned from paper or produced by computer.

The representation of extracted information in some common data format is a key issue. Some general data formats (e.g. DAFS [11]) and many ad-hoc formats have been developed for this purpose, but none of them is extensible and general enough to

V. Mařík et al. (Eds.): DEXA 2003, LNCS 2736, pp. 256–265, 2003.

hold for all different situations. This variety of formats prevents the easy exchange of data between different environments. A solution to this problem could lie in XML technology. XML has been proposed as a data representation format in general, but it was originally developed to represent (semi-) structured documents, therefore it is a natural choice for the representation of the output of DIA systems. XML is also an Internet language, a characteristic that can be profitably exploited to make information present on paper more quickly web-accessible and retrievable than distributing the bitmaps of document images on a web server. Moreover, it is possible to define some hypertext structures which improve document reading [16]. Finally, in the XML document, additional information on the semantics of the text can be stored in order to improve the effectiveness of the retrieving. This is a way to reduce the so-called *semantic gap* in document retrieving [17], which corresponds to the mismatch between the user's request and the way automated search engines try to satisfy these requests.

Commercial OCR systems are still far from supporting the XML format generation satisfactorily. Most of them can save scanned documents in HTML format, but generally their appearance on the browser is not similar to the original documents. Rendering problems, such as missing graphical components, wrong reading ordering in two-columned papers, missing indentation and broken text lines, are basically due to poor layout information extracted from the scanned document. In addition, no information on the semantics of some content portions is associated to documents saved in HTML format.

The extraction of information from the document image requires knowledge technologies, which offer various solutions to the knowledge representation problem and automated reasoning, as well as to the knowledge acquisition problem, by means of machine learning techniques. The importance of knowledge technologies has led some distinguished researchers to claim that document image analysis and understanding belongs to a branch of artificial intelligence [12], despite the fact that most of the contributions fall within the area of pattern recognition [10].

In this paper we present the multi-page DIA system WISDOM++ (http://www.di.uniba.it/~malerba/wisdom++/), whose architecture is knowledge-based and supports all the processing steps required for semantic indexing and storing in XML format [1]. More precisely, the transformation process performed by WISDOM++ consists of the preprocessing of the raster image of a scanned paper document, the segmentation of the preprocessed raster image into basic layout components, the classification of basic layout components according to the type of content (e.g., text, graphics, etc.), the identification of a more abstract representation of the document layout (layout analysis), the classification of the document on the basis of its layout and content, the identification of semantically relevant layout components, the application of OCR only to those textual components of interest and the storing in XML format providing additional information on the semantics of the text.

Four of these processing steps are knowledge-based, namely:

1. classification of basic-blocks,
2. layout analysis,
3. automatic global layout analysis correction,
4. semantic indexing (document image classification and understanding).

The knowledge technologies used in these four steps are:

- a knowledge-based system which contains explicitly represented rules and supports inference by resolution (used for document classification and understanding);
- a production-system which operates with a forward-chaining control structure and is used for global layout analysis correction;
- the decision tree learning system ITI [14] (used for block classification);
- the inductive logic programming system ATRE [7] (used to learn rules for layout analysis correction and for semantic indexing);
- the logic programming system for the implementation of several modules of WISDOM++.

In this paper, we briefly describe the current architecture of the WISDOM++ system (next section), and then focus our presentation on the rule-based semantic indexing step (Section 3). The transformation process in XML format is described in Section 4. Finally, in Section 5 a real-world application to censorship documents in film archives is described.

2 System Architecture

The general architecture of WISDOM++, shown in Figure 1, integrates several components to perform all the steps reported in the previous section.

The *System Manager* manages the system by allowing user interaction and by coordinating the activity of all other components. It interfaces the system with the data base module in order to store intermediate information. The *System Manager* is also able to invoke the OCR on textual layout blocks which are relevant for the specific application (e.g., title or authors).

The *Image Processing Module* is in charge of the image preprocessing facilities. Preprocessing consists of a series of image-to-image transformations, which do not increase the system's knowledge of the contents of the document, but may help to extract it. One basic preprocessing step is the detection of the skew angle, which is defined as the orientation angle of the baselines of text blocks. Once the skew angle has been estimated the document image can be rotated to a reference direction to facilitate further format analysis and OCR. Additional preprocessing steps are noise filtering, such as removal of salt-and-pepper noise, and resolution reduction.

The *Layout Analysis Module* supports the separation of text from graphics and the layout analysis. The separation of text from graphics is performed into two steps: the segmentation detects non-overlapping rectangular blocks enclosing content portions, while the block classification identifies the content type (e.g., text, drawings, pictures and horizontal/vertical lines). WISDOM++ segments the reduced document image into rectangular blocks by means of an efficient variant of the Run Length Smoothing Algorithm [15]. The smoothing thresholds used in the segmentation are adaptively defined depending on a spread factor which is computed during the skew evaluation step. The classification of blocks is based on the description of some features of each block. In WISDOM++ only geometrical (e.g., width, height, area, and eccentricity) and textural features are used to describe blocks. The classification of blocks as text, horizontal line, vertical line, picture (i.e., halftone images) and graphics (e.g., line drawings) is performed by means of the decision tree learning system ITI.

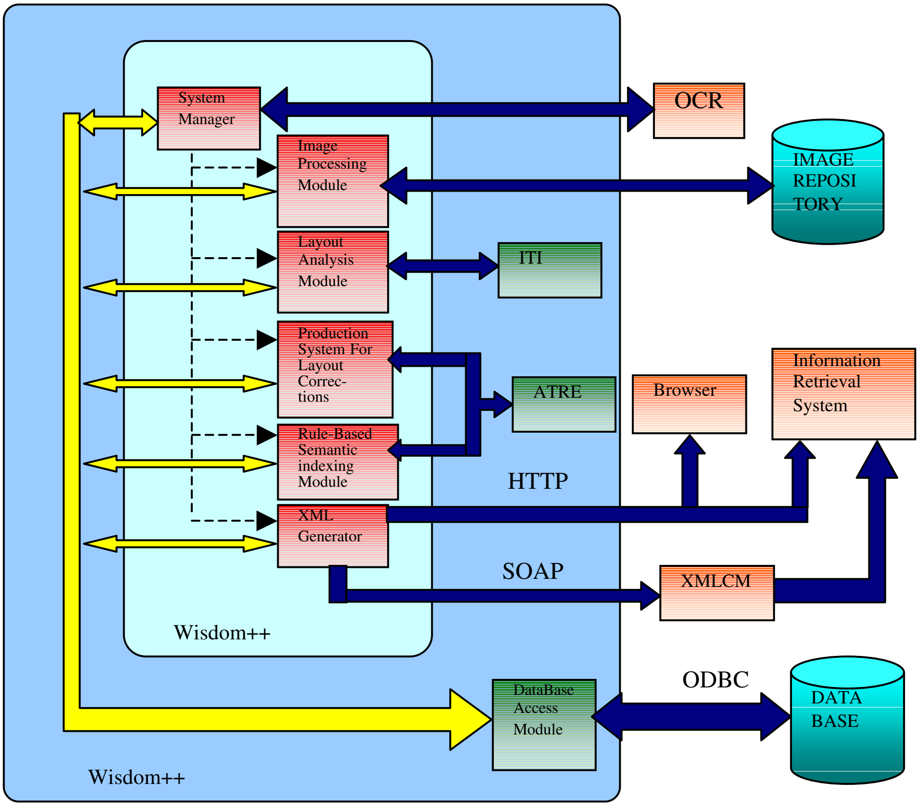


Fig. 1. Wisdom++ architecture

The *layout analysis* detects structures among blocks extracted during the segmentation step. It generates a hierarchy of abstract representations of the document image, the *geometric* (or *layout*) *structure*, which can be modeled by a *layout tree*. It is performed in two steps: firstly, the global analysis determines possible areas containing paragraphs, sections, columns, figures and tables, and secondly, the local analysis groups together blocks that possibly fall within the same area.

Once the layout analysis has been performed and the layout tree has been generated, the user can manually modify the layout tree by performing three different types of actions: vertical or horizontal split of a component in the layout tree, and grouping of two components. WISDOM++ stores both the result of corrective actions and the actions themselves. In this way it is possible to learn corrective layout operations from user interaction [9]. These operations are expressed as a set of "production" rules in the form of an antecedent and a consequent, where the antecedent expresses the precondition to the application of the rule and the consequent expresses the action to be performed in order to modify the layout structure. Production rules are then used by the *Production System for Layout Analysis Module*, which operates with a forward-chaining control structure. The production system is implemented with a theorem prover, using resolution to do forward chaining over a full first-order knowledge base. The system maintains a knowledge base (the working memory) of ground literals describing the layout tree. Ground literals are automatically generated by WISDOM++ after the execution of an operation. In each cycle, the system computes the subset of rules whose condition part is satisfied by the current contents of the working memory (*match phase*). Conflicts are solved by selecting the first rule in the subset.

The *Rule-based Semantic Indexing Module* performs the document classification and the document understanding tasks. Document classification automatically identifies the membership class of a document with respect to a user-defined set of document classes. Document understanding aims at automatically associating some layout components with components of a logical hierarchy. By performing document image classification and understanding, WISDOM++ actually replaces the low-level image feature space (based on geometrical and textural features) with a higher-level semantic space. Query formulation can then be performed using these higher level semantics, which are much more comprehensible to the user than the low level image features [2]. Rules for document classification and understanding are learned by means of the inductive logic programming system ATRE [7], as explained in the next section.

The *XML Generator Module* is used to save the document in XML format. It transforms document images into XML format by integrating textual, graphical, layout and logical information extracted in the document analysis and understanding processes.

3 Rule-Based Semantic Indexing

Semantic-indexing of a document image is based on a mapping of the *layout structure* into the *logical structure* of a (multi-page) document. The former associates the content of a document with a hierarchy of layout components, such as blocks, lines, and paragraphs. It is related to the presentation of the document on some media. On the other hand, the logical structure associates the content of a document with a hierarchy of logical components, such as sender/receiver of a business letter, title/authors of a scientific article, and so on. It is related to the organization of the content. Luckily, in many documents the two structures are strongly related. This means that layout clues can be profitably used to reconstruct the logical structure of the document without “reading” the document itself.

The general process of defining the mapping is called *document image understanding*[1] (or *interpretation*) [13], while the specific association of the whole document (root of the layout tree) with some class (root of the logical structure) is called *document image classification* [3]. This mapping is usually represented as a labeled layout tree, where each layout component is associated with zero, one or more logical components (the semantics). This association can theoretically affect layout components at any level in the *layout tree*. However, in WISDOM++ only the most abstract components in the *layout tree* are associated with some component of the *logical hierarchy*. Moreover, only layout information is used in document image

[1] This process is distinct from *document understanding* which is related to natural language aspects of one-dimensional text flow.

understanding. This approach differs from that proposed by other authors [6] which also makes use of textual information (e.g. text pattern), font information (e.g. style, size, boldness, etc.) and universal attributes (e.g. number of lines) given by the OCR. This diversity is due to a different opinion on when an OCR should be applied. We believe that only some layout components of interest for the application should be subject to OCR (e.g., title and authors, but not figures and tables of a scientific paper), hence document understanding should precede text reading and cannot be based on textual features.

Procedurally, the mapping is determined by means of a set of rules expressed in a first-order logic language. The antecedent of a rule describes both spatial and aspatial properties that should hold between layout components in a page. The consequent specifies the semantics of some layout components involved in the antecedent part. The matching between the antecedent of a rule and the description of the page layout determines the association between the layout structure and the logical structure.

In order to express spatial relations properly, WISDOM++ resorts to first-order definite clauses, such as rule representation formalism. Therefore, the induction of these rules from a set of labeled layout trees requires the application of an inductive logic programming system that can learn logic theories (i.e., sets of definite clauses). The learning system embedded in WISDOM++ is ATRE and a full description is reported in [7]. We limit ourselves to observing that two important features of ATRE for this specific application domain are its ability to discover concept dependencies [8] and to handle both symbolic and numerical attributes and relations [4].

4 Generating a Document in XML Format

Data concerning the result of document processing can be stored in XML format so that the resulting XML document, which includes semantic information extracted in the document analysis and understanding processes, is accessible via web through queries at a high level of abstraction.

The simplest transformation consists in attaching document images to XML pages, after having converted bitmaps into a format supported by most browsers (e.g. GIF or JPEG). Nevertheless, this approach presents at least four disadvantages. First, compressed raster images are still quite large and their transfer can be unacceptably slow. Second, the original document can only be viewed and not edited. Third, in the case of multi-page documents, pages can be presented only in a sequential order, thus missing the advantages a hypertext structure which supports document browsing. Fourth, additional information about the semantics of the content cannot be represented, hence no semantics-based retrieval facility can be supported. Therefore, it is important to transform document images into XML format by integrating textual, graphical, layout and semantic information extracted in the document analysis and understanding processes. Moreover, the XML specification includes a facility for physically isolating and separately storing any part of a document, for example, storing data without contamination of formatting information.

A *DTD* is associated to each document class and the XML document refers to the appropriate *DTD*. In the following, an example of a *DTD* generated by WISDOM++ for the class “faa_cen_decision” is reported.

```
<!-- standard DTD file for faa_cen_decision class -->
<!ELEMENT faa_cen_decision (logic-structure?, geometric-structure)>
<!ELEMENT logic-structure (registration-au|undefined|date-
place|department|applicant|reg-number|film-genre|film-length|film-
producer|film-title)*>
<!ELEMENT      registration-au (paragraph)*>
<!ATTLIST      registration-au ID    NMTOKEN        #IMPLIED>
<!ELEMENT      undefined (paragraph)*>
<!ATTLIST      undefined ID          NMTOKEN        #IMPLIED>
<!ELEMENT      date-place (paragraph)*>
<!ATTLIST      date-place ID         NMTOKEN        #IMPLIED>
<!ELEMENT      department (paragraph)*>
<!ATTLIST      department ID         NMTOKEN        #IMPLIED>
…
<!ELEMENT      paragraph (#PCDATA|TAB)*>
<!ELEMENT TAB EMPTY>
<!ELEMENT geometric-structure (image, blocklevels)>
<!ELEMENT image    EMPTY>
<!ATTLIST image    urlimage          CDATA          #REQUIRED
                   length            NMTOKEN        #REQUIRED
                   width             NMTOKEN        #REQUIRED
                   formatimage       NMTOKEN        #REQUIRED
                   resolution        NMTOKEN        #REQUIRED>
<!ELEMENT blocklevels (basic-block, line, setofline, frame1, frame2)>
<!ELEMENT basic-block (block+)>
…
<!ATTLIST      basic-block    numBB  NMTOKEN        #REQUIRED>
<!ATTLIST      line           numL   NMTOKEN        #REQUIRED>
…
<!ELEMENT block    EMPTY>
<!ATTLIST block    indexblock NMTOKEN        #REQUIRED
                   top               NMTOKEN        #REQUIRED
                   bottom            NMTOKEN        #REQUIRED
                   left              NMTOKEN        #REQUIRED
                   right             NMTOKEN        #REQUIRED
                   physical-type     NMTOKEN        #REQUIRED
                   subblockslist     CDATA          #IMPLIED
                   label      (registration-au|undefined|date-
place|department|applicant|reg-number|film-genre|film-length|film-
producer|film-title) "undefined" >
```

The keyword ELEMENT introduces an element declaration which represents the information on the semantics of the content (e.g. registration-au, date-place, department, applicant, reg-number, film-genre, film-length, film-producer, film-title, undefined[2]). An element may have no content at all, may have a content of only text, of only child element, or of a mixture of elements and text. For example, in the DTD presented the content of the element `faa_cen_decision` is a child element, which is structured. An attribute may be associated with a particular element in order to provide refined information on an element. Examples of attributes are the URL, the height, the width, the format and the resolution of a document image. All the attributes are declared separately from the element, but are usually declared together, in the *attribute list declaration*. It is also noteworthy that the DTD generated by WISDOM++ distinguishes the logical structure (`logic-structure`) from the

[2] The element *undefined* refers to all those logical components of no specific interest for the application.

layout structure (geometric-structure). The layout structure is used for storing purposes, in particular it is used to build XSL specifications in order to render the document similar in appearance to the original document, since XML language is not concerned with visualization aspects.

The XML document generated can be stored in an XML-based Content Management System (XMLCM), which is the back-end of WISDOM++. XMLCM uses the XML language to represent/manage documents, structured data and metadata (DTD or XML Schema) and to exchange them over Internet. Because Internet-based applications deal with complex, heterogeneous and worldwide information, the XMLCM is based on basic open communication standards for information processing, such as HTTP, XML and SOAP.

5 Application to Censorship Decisions

Document images processed in the European project COLLATE (http://www.collate.de/) are provided by three national film archives, namely Deutsches Filminstitut (DIF), Filmarchiv Austria (FAA) and Národní Filmový Archiv (NFA). Generally, documents are multi-page, where each page is an RGB 24bit color image representing either a censorship card or, in the case of DIF, a newspaper article. An example of a document is reported in Figure 2.

Table 1. Main features of processed documents

Source	Type	No of documents	Tot No of pages	Size (pixel)	Resolution (dpi)	Image size (mm)
FAA	Censorship cards	29	60	4836×3408	600	204,72 ×144,27
DIF	Censorship cards	6	18	1710×1212	300	144,78 × 102,62
DIF	Censorship cards	30	360	2460×3474	300	208,28 × 294,13
DIF	Newspaper articles	19	57	Not fixed	Not fixed	Not fixed
NFA	Censorship cards	24	72	2528×3988	300	214,05 × 337,66

To investigate the applicability of the solution proposed we considered 108 multi-page documents belonging to 5 classes (see Table 1). We applied WISDOM++ to 567 document images in all.

As regards the extraction of semantic information on the class of the document, the rule learned by ATRE for the faa_cen_decision class is the following:

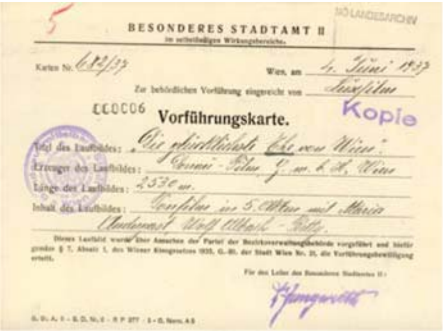
class(X1)=faa_cen_decision←image_width(X1)∈[4832..4992].

where X1 denotes the whole page. This rule is simple and its interpretation is straightforward. The paper document is considered as a faa_cen_decision if the image width is between 4832 and 4992 pixels. The simplicity of the rule depends on the standard dimension of the images. In particular, the learning system is able to classify the document without ambiguity by considering only dimensions, rather than additional information on the internal layout structure.

As regards the extraction of semantic information on the logical components of the document, some examples of rules learned by ATRE for the faa_cen_decision class are reported:

```
film_genre(X2) ←y_pos_centre(X2)∈[452..472],
                   to_right(X2,X1), width(X2)∈[182..881]
```

This rule expresses the condition that a possibly large layout component (width between 182 and 881) with its baricentre at a point between 452 and 472 on the y-axis and to the left of another block (X1) is the genre of the film. An example of the mapping of the layout structure into the logical structure is reported in Figure 3.



5

BESONDERES STADTAMT II
im selbständigen Wirkungsbereiche.

NÖ LANDESARCHIV

Karten Nr. 682/37

Wien, am 4. Juni 1937

Zur behördlichen Vorführung eingereicht von Luxfilm

Kopie

CCOC06 Vorführungskarte.

Titel des Laufbildes: „Die glücklichste Ehe von Wien."

Erzeuger des Laufbildes: [illegible]-Film G. m. b. H., Wien

Länge des Laufbildes: 2530 m.

Inhalt des Laufbildes: Tonfilm in 5 Akten mit Maria Andergast Wolf Albach-Retty.

Dieses Laufbild wurde über Ansuchen der Partei der Bezirksverwaltungsbehörde vorgeführt und hiefür gemäss § 7, Absatz 1, des Wiener Kinogesetzes 1935, G.-Bl. der Stadt Wien Nr. 21, die Vorführungsbewilligung erteilt.

Für den Leiter des Besonderen Stadtamtes II:

[illegible]

Fig. 2. The original scanned document

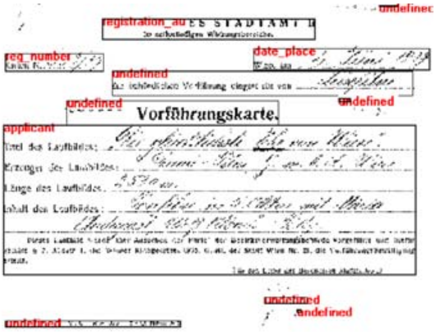


Fig. 3. The labeled image. This is the result of the document unterstanding process.

6 Conclusions and Challenging Problems

This work presents the integration of knowledge and XML technologies for semantic indexing of paper documents. Semantic-indexing is the result of a complex process, involving, among other things, the document classification and understanding steps and the application of machine learning techniques. The proposed approach has been investigated in the context of a European project on annotation, indexing and retrieval of digitized historical archive material.

This work can be extended in several directions. In particular, the project requires more complex document preprocessing and layout analysis techniques. Moreover, text extracted by the OCR enables the investigation of the integration of DIA techniques with both text mining and information extraction techniques. Finally, we intend to extend the system WISDOM++ presented in the paper with information retrieval facilities, based both on semantic annotations and OCRed text and on graphical (layout) information.

Acknowledgements. This work partially fulfills the research objectives set by the IST-1999-20882 project COLLATE (Collaboratory for Automation, Indexing and Retrieval of Digitized Historical Archive Material) funded by the European Union (http://www.collate.de). The authors also wish to thank Lynn Rudd for her help in reading the manuscript.

References

1. Altamura O., Esposito F., & Malerba D.: Transforming paper documents into XML format with WISDOM++, *International Journal on Document Analysis and Recognition*, 4(1), (2001), pp. 2–17.
2. Bradshaw, B.: Semantic based image retrieval: a probabilistic approach. *ACM Multimedia* 2000, (2000), pp. 167–176.
3. Esposito F., Malerba D., Semeraro G., Annese E., and Scafuro G.: An experimental page layout recognition system for office document automatic classitication: An integrated approach for inductive generalization. *In Proc. of the 10th Int. Conf on Pattern Recognition*, (1990), pp. 557–562.
4. Esposito, F.; Malerba, D. & Lisi, F.A.: Machine Learning for intelligent processing of printed documents. *Journal of Intelligent Information Systems* 14(2/3), (2000), pp. 175–198.
5. Fan X., Sheng F., Ng P.A.: DOCPROS: A Knowledge-Based Personal Document Management System. *Proc. of the 10th International Workshop on Database &amp; Expert Systems Applications*, DEXA Workshop. (1999), pp. 527–531.
6. Klink S., Dengel A., and Kieninger T.: Document structure analysis based on layout and textual features. In *Proc. of Fourth IAPR International Workshop on Document Analysis Systems, DAS2000*, Rio de Janeiro, Brazil, (2000), pp. 99–111.
7. Malerba D., Esposito F., and Lisi F.A.: Learning recursive theories with ATRE, in H. Prade (Ed.), *Proceedings of the 13th European Conference on Artificial Intelligence*, John Wiley & Sons, Chichester, England, (1998), pp. 435–439.
8. Malerba D., Esposito F., Lisi F.A. and Altamura O.: Automated Discovery of Dependencies Between Logical Components in Document Image Understanding. *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, Seattle (WA), (2001), pp. 174–178.
9. Malerba D., Esposito F., Altamura O., Ceci M., and Berardi M.: Correcting the Document Layout: A Machine Learning Approach. *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, Edinburgh (UK), (2003), to appear.
10. Nagy, G.: Twenty Years of Document Image Analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 22 (1), (2000), pp. 38–62.
11. RAF Technology, Inc. *DAFS Library, Programmer's Guide and Referenc*e, August (1995).
12. Tang Y.Y., Yan C. D., Suen C. Y.: Document Processing for Automatic Knowledge Acquisition, *in IEEE Trans. on Knowledge and Data Engineering*, 6(1), (1994), pp.3–21.
13. Tsujimoto S., Asada H.: Understanding Multi-articled Documents, *in Proceedings of the 10th International Conference on Pattern Recognition*, Atlantic City, N.J., (1990), pp. 551–556.
14. Utgoff P.E.: An improved algorithm for incremental induction of decision trees. *Proc. of the Eleventh Int. Conf. on Machine Learning*, San Francisco, CA: Morgan Kaufmann, (1994).
15. Wong K.Y., Casey R.G., and Wahl F.M.: Document analysis system. *IBM Journal of Research Development* 26(6), (1982), pp. 647–656.
16. Worring M., Smeulders A.W.M.: Content based Internet access to scanned documents. *Int J. Doc. Anal. Recognition* 1(4), (1999).
17. Zhao R., Grosky W. I.: Narrowing the Semantic Gap Improved Text-Based Web Document Retrieval Using Visual Features, *in IEEE Trans. on Multimedia*, 4(2), (2002), pp. 189–200.