An Integrated Approach for Automatic Semantic Structure Extraction in Document Images

Margherita Berardi, Michele Lapi, and Donato Malerba

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

Abstract. In this paper we present an integrated approach for semantic structure extraction in document images. Document images are initially processed to extract both their layout and logical structures on the base of geometrical and spatial information. Then, textual content of logical components is employed for automatic semantic labeling of layout structures. To support the whole process different machine learning techniques are applied. Experimental results on a set of biomedical multi-page documents are discussed and future directions are drawn.

1 Introduction

The increasingly large amount of paper documents to be processed daily in office environments requires new document management systems with abilities to automatically catalog and organize these documents 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 [7]. In order to extend these capabilities to paper documents it is necessary to convert them into a suitable electronic format. This can be done by applying knowledge technologies, such as machine learning tools or knowledge representation languages. They are so relevant, that some distinguished researchers claimed that document image analysis and understanding belong to a branch of artificial intelligence [16], despite the fact that most of the contributions fall within the area of pattern recognition [12].

In this paper we present a Document Image Analysis (DIA) framework with a knowledge-base architecture that supports all the processing steps required for the semantic structure extraction from document images. More precisely, this results from a tight integration of the system WISDOM++, which performs document understanding on the basis of geometrical information, with the content-based classification capabilities provided by the system WebClassII [4]. WebClassII is a client-server application that performs the automated classification of Web pages on the basis of their textual content. WebClassII preprocessing and classification modules have been integrated in the proposed framework and applied to the textual content of logical components of interest extracted by WISDOM++.

In WISDOM++, document understanding consists in the mapping of the geometrical structure, extracted during the layout analysis step, in the logical structure, which associates the content of the layout component with a hierarchy of logical components, such as title/authors of a scientific article. The mapping is based on the assumption according to which in many documents the logical and the layout structures are so strongly related that a user is able to "understand" the document without reading the content itself but only considering geometrical criteria. For instance, the title of an article is usually located at the top of the first page of a document and it is written with the largest character set used in the document.

Nevertheless, current results in document management research highlight that the enhancement of information retrieval systems performance is strongly dependent on the use of semantic information about the textual content of documents [14]. In our proposal the upgrade of WISDOM++ that aims to support document understanding with content-based classification functionalities has been investigated.

The paper is organized as follows. In the next section the DIA system WISDOM++ is briefly described and some related works on document understanding are discussed. In Section 3 the layout analysis method is introduced and the document understanding strategy is extensively reported. Section 4 describes the semantic structure extraction approach focusing on the preprocessing and classification methods. Experimental results are shown in Section 5 and future work is presented in Section 6.

2 Background and Related Work

WISDOM++[1] is a document analysis system that can transform textual black and white paper documents into XML format [2]. This is a complex process involving several steps. First, the image is converted in black and white and is segmented into basic layout components (non-overlapping rectangular blocks enclosing content portions) by means of an efficient variant of the Run Length Smoothing Algorithm.

These layout components are classified according to the type of their content: text, horizontal line, vertical line, picture and graphics. This classification is performed by means of a decision tree automatically built from a set of training examples of the five classes. Then, layout analysis is performed in order to detect structures among blocks. The result is a tree-like structure which is a more abstract representation of the document layout. This representation associates the content of a document with a hierarchy of layout components, such as blocks, lines, and paragraphs. Considering the extracted layout, the document image classification is performed. This aims at identifying the membership class (or type) of a document (e.g. business letter, newspaper article, and so on) by means of some first-order rules which can be automatically learned from a set of training examples [10]. In document image understanding, layout components are associated with logical components. This association can theoretically affect layout components at any level in the layout hierarchy. However, in WISDOM++ only the most abstract components (called *frame2*) are associated with components of the logical hierarchy. Moreover, only layout information is used in document understanding.

[1] http://www.di.uniba.it/~malerba/wisdom++/

This approach differs from that proposed by other authors [9] which additionally make use of textual information, font information and universal attributes given by the OCR. This diversity is due to a different conviction on when an OCR should be applied. We believe that only some layout components of interest for the application should be subject to OCR, hence document understanding should precede text reading and cannot be based on textual features. Two assumptions are made: documents belonging to the same class have a set of relevant and invariant layout characteristics; logical components can be identified by using layout information only. Document image understanding also uses first-order rules [10]. Once logical and layout structure have been mapped, OCR can be applied only to those textual components of interest for the application domain, and its content can be stored for future retrieval purposes. The result of the document analysis is an XML document that makes the document image retrievable.

Similarly to [1], we propose an integrated approach for document understanding that is based on both geometric and content features and makes extensive use of knowledge. In particular, [1] consider document understanding as the extraction of logical relations on layout objects, where logical relations express reading order between layout components, and extensive use of spatial reasoning and natural language processing techniques is made to support the extraction of reading order relations. Differently from this approach we consider document understanding as logical structure detection combined with a mapping into a set of predefined semantic classes.

As clearly explained in [6], research trend in document understanding turns towards the need of document management frameworks which should be able to handle different typologies of documents and to employ hybrid strategies for knowledge capture in order to handle different dimensions of information (e.g. textual, layout, format, tabular, etc.). In particular, it seems that document management systems will not give answers to real-world needs until they continue to tailor their solutions for the individual sub-problems in which the whole process of document understanding can be articulated. Our proposal fits this scenario and tries to answer some of the mentioned issues.

3 Document Layout Analysis and Understanding in WISDOM++

Before entering into the details of semantic structure extraction, it is important to clarify the document analysis steps performed by WISDOM++, and to briefly explain how machine learning techniques are employed to guarantee a high degree of adaptivity.

Considering the output of the segmentation and blocks classification, it results that this representation is still too detailed for learning document classification and understanding rules. Therefore, the *layout analysis* is performed to detect structures among blocks. The result is a hierarchy of abstract representations of the document image. The leaves of the layout tree are the blocks, while the root represents the whole document. A page may group together several layout components, called *frames*, which are rectangular areas of interest in the document page image. The layout analysis is done in two steps, a *global* analysis of the document image in order to determine possible areas containing paragraphs, sections, columns, figures and tables and a *local analysis* to group together blocks which possibly fall within the same area. Perceptual criteria considered in this last step are: *proximity* (e.g. adjacent components belonging to the same column/area are equally spaced), *continuity* (e.g. overlapping components) and *similarity* (e.g. components of the same type, with an almost equal height). Pairs of layout components that satisfy some of these criteria may be grouped together. The layout structure extracted by WISDOM++ is a hierarchy with six levels: basic blocks, lines, set of lines, frame1, frame2, and pages. The last level is specified by the user, while the first five are automatically extracted. If the user is not satisfied with the result of the layout analysis he/she can act directly on the results of the segmentation process by deleting some blocks or he can modify the result of the global analysis by performing some splitting or grouping operations, which are later used to learn rules for the automated correction of the layout analysis [3].

Some logical components of the document, such as title and authors of a paper, can be identified after having detected the layout structure. They can be arranged in a hierarchical structure, which is called *logical structure*, resulting by a division of the content of a document into increasingly smaller parts. The leaves of the logical structure are the basic logical components, such as authors of a paper. The heading of an article, encompassing the title and the author, is an example of a composite logical component. The root of the logical structure is the document class. The discovery of the logical structure of a document can be cast as the problem of associating some layout components with a correspondent logical component. In WISDOM++ it consists in the association of a page with a document class (*document classification*) and of frame2 layout components with basic logical components (*document understanding*). Classification is performed by matching the layout structure of the first page against *models of classes of documents* that are able to capture the invariant properties of the images/layout structures of documents belonging to the same class. Document understanding of all pages is performed by matching the layout structure of the each page against *models of logical components*. An example of models for the logical components *running_head* and *paragraph* in the case of scientific papers might be:

logic_type(Y)= paragraph← on_top(X,Y)=true,
logic_type(X)=running_head, type(Y)=text

These rule means that a textual layout component below a running-head is a paragraph of the paper. Further details on document image understanding are reported in [11].

4 Semantic Structure Extraction

Due to the limits of a layout-based indexing, WISDOM++ has been integrated in a framework which supports the indexing phase by using relevant terms automatically extracted from logical components of interest. The goal is to select from the ocr-ed text of a logical component those terms that allow the system to assign the logical component to a semantic class of a set of predefined domain-dependent classes.

As shown in Fig.1, in this framework the output of WISDOM++, that is logical labeling and ocr-ed text of logical components, is used as input for WebClassII. In particular, the feature extractor module selects relevant terms from training textual components through which the classifier is learned. It is noteworthy that output of the automatic logical labeling performed by WISDOM++ might be eventually subject to users' correction and used as WebClassII input. Users corrections are useful both to generate new training examples for logical labeling learning and to preserve WebClassII classifier learning. The output of WebClassII modules is the tagging of those semantic components that are not classifiable through geometrical criteria (e.g. the section about the *Methods* of a scientific paper) but only supporting the process with a semantic-based classification step.

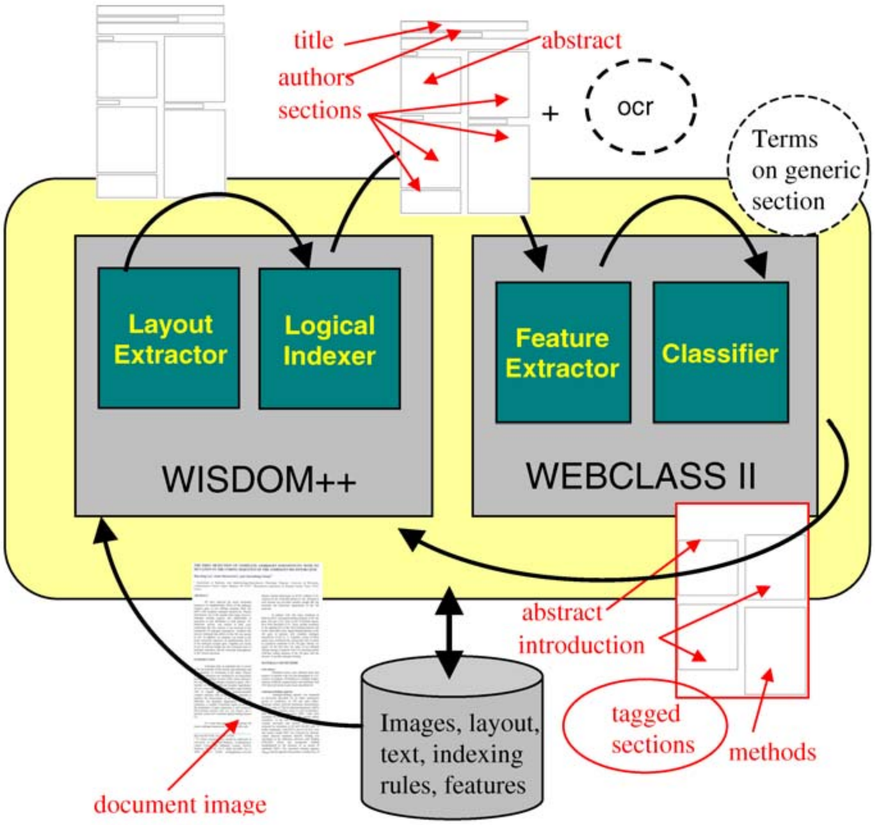

Fig. 1. DIA Framework Architecture.

WebClassII is a client-server application that performs the automated classification of Web pages on the basis of their textual content. The automated classification of documents, in general, requires the solution of the problem of the representation language definition and of classifier construction tailored on representation language.

WebClassII adopts a feature vector representation of documents, where each single feature corresponds to a distinct term extracted from the training documents. Features are determined by means of a complex preprocessing phase, which includes both a term extraction and a term selection process. As regards the classifier construction problem, WebClassII has two alternative ways of assigning a Web page to a class, a centroid-based method and a naïve Bayesian method, and both of them require a training phase by means of which the classifier is learned.

In our specific context, we use the preprocessing and the classification modules as common text document processing modules without taking into account WebClassII Web document management capabilities.

4.1 The Feature Extractor Module

Initially, all training documents are tokenized, and the set of tokens (words) is filtered in order to remove punctuation marks, numbers and tokens of less than three characters. The basic idea is to select relevant tokens to be used in the bag-of-words representation. These tokens will be called features.

The text pre-processing procedure implemented in WebClassII is:

1. Removal of words with high frequency, or stopwords, such as articles, prepositions, and conjunctions.
2. Removal of suffixes, such as those used in plurals (e.g., -s, -es and -ies), gerund (-ing), simple past (-ed), and so on.
3. Determination of equivalent stems (stemming), such as "analy" in the words "analysis", "analyses", "analyze", "analyzing" and "analyzer".

For the first step, stopwords used by WebClassII have been taken from Glimpse (glimpse.cs.arizona.edu), a tool used to index files by means of words, while for the last two steps, the algorithm proposed by Porter [13] has been implemented.

Many approaches have been proposed in the literature on information retrieval for the identification of relevant words to be used as index terms of documents [15]. Most of them simply score words according to some measure and select the best firsts. However, techniques proposed for information retrieval purposes are not always appropriate for the task of document classification, also known as text categorization. Indeed, we are not interested in words characterizing each single document, but we look for words that distinguish a class of documents from other classes. Generally speaking, the set of words required for classification purposes is much smaller than the set of words required for indexing purposes.

The feature selection algorithm implemented in WebClassII is based on a variant of TF-IDF. Given the training document *d* of the *i-th* class, for each token *t* the frequency *TF(i,d,t)* of the token in the document is computed. Then, for each class *i* and token *t*, the following statistics are computed:

– *MaxTF(i,t)*, the maximum value of *TF(i,d,t)* on all training documents *d* of class *i*;
– *PF(i,t)*, the page frequency, that is, the percentage of documents of class *i* in which the token *t* occurs.

The union of sets of tokens extracted from documents in one class defines an "empirical" class dictionary used by documents on the topic specified by the class. By sorting the dictionary with respect to *MaxTF(i,t)*, words occurring frequently only in one document might be favored. By sorting each class dictionary according to the product $MaxTF(i,t)*PF(i,t)^2$, briefly denoted as $MaxTF\text{-}PF^2$ (Max Term Frequency - Square Page Frequency) measure, the effect of this phenomenon is kept under control. Moreover, common words used in documents of a given class will appear in the first entries of the corresponding class dictionary. Some of these words are actually specific to that class, while others are simply common English words (e.g., "information", "unique", "suggestion", "time" and "people") and should be considered as quasi-stopwords. In order to move quasi-stopwords down in the sorted dictionary, the $MaxTF\text{-}PF^2$ of each term is multiplied by a factor *ICF=1/CF(t),* where *CF(t)* (category frequency) is the number of class dictionaries in which the word t occurs. In this way, the sorted dictionary will have the most representative words of each class in the first entries, so that it will be sufficient to choose the first N words per dictionary, in order to define the set of attributes. Once the class dictionaries are determined, a unique set of features is selected to represent documents of all classes. Once the set of features has been determined, training documents can be represented as feature vectors of term frequencies.

4.2 The Classification Method

Currently, WebClassII has two alternative ways of assigning a Web page to a category:

1. By computing the similarity between the document and the centroid of that category.
2. By estimating the Bayesian posterior probability for that category (naïve Bayes).

Therefore, a training phase is necessary either to compute the centroids of the categories or to estimate the posterior probability distributions. In the following, only the naïve Bayes method is illustrated because it outperforms the centroid method and it has been used for experiments.

Let *d* be a document temporarily assigned to category *c*. We intend to classify *d* into one of the subcategories of *c*. According to the Bayesian theory, the optimal classification of *d* assigns *d* to the category $c_i \in SubCategories(c)$ maximizing the posterior probability $P_c(c_i|d)$. Under the assumption that each word in *d* occurs independently of other words, as well as independently of the text length, it is possible to estimate the posterior probability as follows:

$$P_c(c_i \mid d) = \frac{P_c(c_i) \cdot \prod_{w \in FeatSet_c} P_c(w \mid c_i)^{TF(w,d)}}{\sum_{c' \in SubCategories(c)} P_c(c') \cdot \prod_{w \in FeatSet_c} P_c(w \mid c')^{TF(w,d)}} \tag{1}$$

where the prior probability $P_c(c_i)$ is estimated as follows:

$$P_c(c_i) = \frac{|Training(c_i)|}{\sum_{c' \in SubCategories(c)} |Training(c')|} = \frac{|Training(c_i)|}{|Training(c)|} \tag{2}$$

and the likelihood $P_c(w \mid c_i)$ is estimated according to Laplace's law of succession:

$$P_c(w \mid c_i) = \frac{1 + PF(w, c_i)}{|FeatSet_c| + \sum_{w' \in FeatSet_c} PF(w', c_i)} \tag{3}$$

In the above formulas, *TF(w,d)* and *PF(w,c)* denote the absolute frequency of *w* in *d* and the absolute frequency of *w* in documents of category *c*, respectively. The likelihood $P_c(w \mid c_i)$ could be estimated according to the relative frequency, that is:

$$P_c(w \mid c_i) = \frac{PF(w, c_i)}{\sum_{w' \in FeatSet_c} PF(w', c_i)} \tag{4}$$

The above formulation of the naïve Bayes classifier assigns a document *d* to the most probable or the most similar class, independently of the absolute value of the posterior probability. By assuming that documents to be rejected have a low posterior probability for all categories, the problem can be reformulated in a different way, namely, how to define a threshold for the value taken by a naïve classifier. Details on the thresholding algorithm are reported in [5].

The classification process of a new document is performed by searching the hierarchy of categories. The system starts from the root and selects the nodes to be expanded such that the score returned by the classifier is higher than a threshold determined by the system. At the end of the process, all explored categories (either internal or leaf) are considered for the final selection. The winner is the explored category with the highest score. If the document is assigned to the root, then it is considered rejected.

5 Experimental Results

A user/trainer of WISDOM++ is asked to label some layout components of a set of training documents according to their logical meaning. Those layout components with no clear logical meaning are not labeled. Therefore, each document generates as many training examples as the number of layout components. Classes of training examples correspond to the distinct logical components to be recognized in a document. The unlabelled layout components play the role of counterexamples for all the classes to be learned. In particular, we have considered a set of nineteen full text multi-page documents, which are scientific papers on biomedical topics. This kind of text corpus is particularly suited for layout based document understanding and from the other hand highlights the need to support it with content information. Indeed, they are characterized by a regular section structure both from the geometrical viewpoint and from the content distribution viewpoint. In particular, the text of each article is distributed on five different and recurrent types of sections: abstract, introduction, methods, results and discussion. We processed 103 document images in all, each of which has a variable number of layout components. In particular, we use 51 document images for training the learning system integrated in WISDOM++, that is ATRE [10], on logical labelling and 24 for testing the induced set of rules. Layout components can be associated with at most one of the following logical components: *title, authors, abstract, section* and *section_title*. In Table 1 logical components distribution on the processed documents is shown. In particular, the total number of logical components is 790 (583 of which are *undefined*) about 49 logical descriptors for each page document.

Table 1. Training set. Distribution of pages and examples per document.

Name of the multi-page document	*No. of pages*	*No. of title labels*	*No. of authors labels*	*No. of abstract labels*	*No. of section labels*	*No. of section-title labels*	*Total no. of examples*
Article_1	6	1	1	1	21	3	94
Article_2	4	1	0	1	10	3	82
Article_3	6	1	1	1	18	6	81
Article_4	3	1	1	1	8	4	46
Article_5	3	1	1	1	5	4	35
Article_6	7	1	1	1	14	5	69
Article_7	7	1	0	1	16	4	88
Article_8	4	1	1	1	11	3	55
Article_9	4	1	1	1	9	3	156
Article_10	2	1	1	1	4	3	23
Article_11	5	1	1	1	17	5	61
Total (training)	51	11	9	11	133	43	790

In order to test the predictive accuracy of the learned theory, we considered 3 articles whose distribution is reported in Table 2. WISDOM++ segmented the 24 pages in 482 layout components. To WISDOM++ is asked to use the learned theory to label layout components as generic *sections* or *section_title* as well as *title*, *authors* and *abstract* sections.

Table 2. Testing set. Distribution of pages and examples per document.

Name of the multi-page document	*No. of pages*	*No. of title labels*	*No. of authors labels*	*No. of abstract labels*	*No. of section labels v*	*No. of section-title labels*	*Total no. of examples*
Article_12	10	1	1	1	41	4	208
Article_13	5	1	1	1	15	4	108
Article_14	9	1	1	1	33	3	200
Total (testing)	24	3	3	3	89	11	516

In Table 3 commission and omission errors performed on the set of testing documents are showed. A commission error occurs when a wrong labelling of logical components is “recommended” by a rule, while an omission error occurs when a “correct” labelling is missed.

Table 3. Commission and omission errors performed by learned rules.

Rule for	***No. omission errors***	***No. commission errors***
logic_type(X)=title	0	5
logic_type(X)=authors	0	2
logic_type(X)=abstract	0	2
logic_type(X)=section	0	39
logic_type(X)=section_title	0	1

As showed no omission errors are performed while some commissions are found especially for sections components. This is due to the heterogeneous layout of generic section components that produces an heterogeneous set of examples and so a set of rules that is quite specific.

Using layout-based labelling, to WebClassII is asked to classify the text of a generic section as either *introduction*, or *materials and methods*, or *results*, or *discussion*, and also as *abstract* even if generally the layout based classification it is sufficient to correctly classify abstract components on geometrical and spatial criteria. Therefore, the date source used in this experimental study is an ontology composed of five categories (i.e. *Abstract*, *Introduction*, *Materials & Methods*, *Discussion* and *Results*), (see Fig. 2). These categories correspond to the five recurrent types of sections according to which biomedical corpora are generally structured. For each category, there are 19 sections one for each of the 19 documents (e.g. the *Abstract* category contains 19 abstract sections, the *Introduction* category contains 19 introductions and so on). The documents have been partitioned into five subsets, four of which compose the training set while the fifth subset represents the testing set.

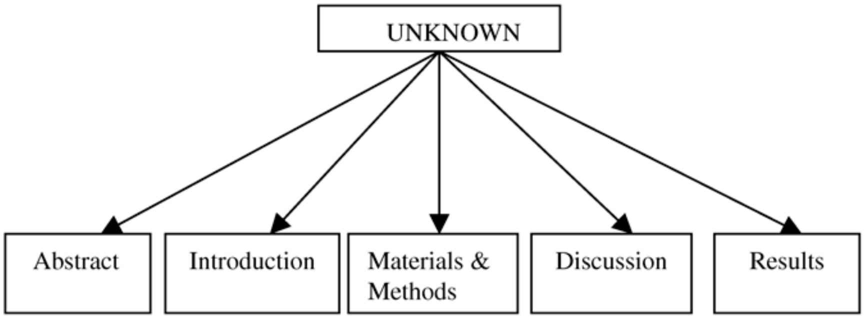

Fig. 2. Documents Ontology.

Classification accuracy has been evaluated by means of precision and recall. The *precision* for a category *c* measures the percentage of correct assignments among all the documents assigned to *c*, while the *recall* gives the percentage of correct assignments in *c* among all the documents that should be assigned to *c*. Also the *misclassification error (ncErr)* has been computed, it computes the percentage of documents in *c* misclassified into a category *c'* not related to *c* in the hierarchy.

Results of the classification process are reported into Fig.3. The graphs show that precision and recall increase when the features number is variable into the range from 15 to 30. Instead, the system reaches the best precision values for the *Methods & Materials* and *Results* classes. Nevertheless, *Abstract*, *Introduction* and *Discussion* sections are much more difficult to classify, because generally these sections are devoted to general topic of the document.

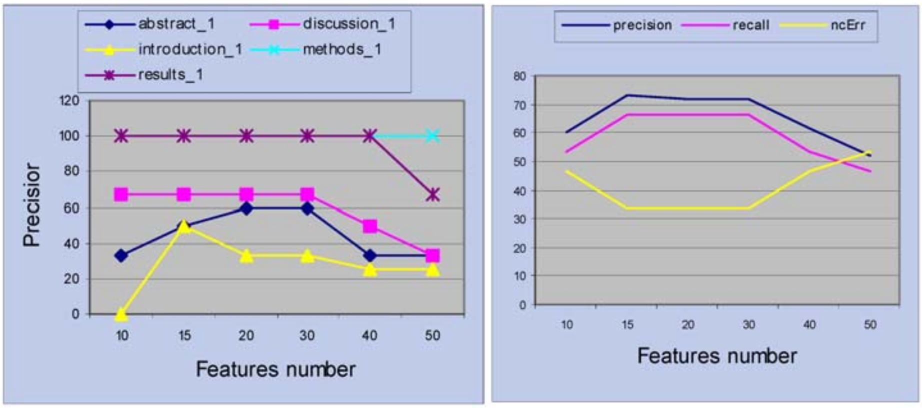

Fig. 3. Experimental results.

6 Conclusions and Future Work

This work presents a starting step towards a document management framework that can take advantage of different types of knowledge (layout and textual features of different level of abstraction) to solve the problem of the mismatch between user requests (based on high-level abstract concepts) and the way in which these requests are satisfied (low-level features for indexing) during an information retrieval process. Indeed, in further extension of our DIA framework we plan to explore the application of information extraction techniques in order to extract useful information from the text. In particular, nowadays there are several methods for information extraction of biological information from scientific articles and generally domain experts recognize without ambiguity in which part of a paper relevant data are located [8]. A formalization of the domain expert knowledge promises to lead to a new generation of document management systems that could learn about distribution of semantics in papers and build indexing models based on really relevant keywords.

Moreover, we plan to conduct more extensive evaluations of the "semantic" classifier. We are also interested in investigating the application of machine learning techniques to reading order detection as further knowledge acquisition step in the workflow from the logical to the semantic structure extraction.

References

1. Aiello, M., Monz, C., Todoran, L., Worring, M.: Document understanding for a Broad class of documents. International Journal of Document Analysis and Recognition, Springer Berlin Heidelberg, Germany (2002).
2. Altamura, O., Esposito, F., Malerba, D.: Transforming paper documents into XML format with WISDOM++, Int. Journal on Document Analysis and Recognition, 4(1), (2001) 2-17.
3. Berardi, M., Ceci, M., Esposito, F., Malerba, D.: Learning Logic Programs for Layout Analysis Correction. In: Proc. of the Twentieth International Conference on Machine Learning, Washington, DC, (2003).
4. Ceci, M., Malerba, D., Lapi, M., Esposito, F.: Automated Classification of Web Documents into a Hierarchy of Categories. In: Ö M.A. Klopotek, S.T. Wierzchon, K. Trojanowski (Eds.), Intelligent Information Processing and Web Mining, Series: Advances in Soft Computing. Berlin: Springer, (2003) 59-68.
5. Ceci, M., Malerba, D.: Web-pages Classification into a Hierarchy of Categories. In: Proc. of the BCS-IRSG 25th European Conference on Information Retrieval Research (ECIR '03), Pisa, Italy, (2003).
6. Dengel, A. R.: Making Documents Work: Challenges for Document Understanding. In: Proc. of the Seventh Int. Conf. on Document Analysis and Recognition (ICDAR '03), IEEE Computer Society Press, Edinburgh, Scotland, (2003) 1026-1036.
7. Fan, X., Sheng, F., Ng, P.A.: DOCPROS: A Knowledge-Based Personal Document Management System. In: Proc. of the 10th International Workshop on Database and Expert Systems Applications (DEXA Workshop), (1999) 527-531.
8. Shah, K. P., Perez-Iratxeta, C., Bork, P., Andrade, M. A.: Information extraction from full text scientific articles: where are the keywords?. BMC Bioinformatics. 4(1):20, (2003).
9. Klink, S., Dengel, A., Kieninger, T.: Document structure analysis based on layout and textual features. In Proc. of Fourth IAPR International Workshop on Document Analysis Systems (DAS '00), (2000) 99-111.
10. Malerba D., Esposito F., and Lisi F.A.: Learning recursive theories with ATRE, in H. Prade (Ed.), Proceedings of the Thirteenth European Conference on Artificial Intelligence, John Wiley & Sons, Chichester, England, (1998), 435-439.
11. Malerba, D., Esposito, F., Lisi, F.A., Altamura, O.: Automated Discovery of Dependencies Between Logical Components in Document Image Understanding. In: Proc. of the Sixth Int. Conference on Document Analysis and Recognition, Seattle (WA), (2001) 174-178.
12. Nagy, G.: Twenty Years of Document Image Analysis in PAMI. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(1), (2000) 38-62.
13. Porter, M.F.: An algorithm for suffix stripping. Program, 14(3), (1980) 130-137.
14. Rindflesch, T., Aronson, A.: Semantic processing in information retrieval. In C. Safran, (ed.), Seventeenth Annual Symposium on Computer Applications in Medical Care (SCAMC '93). McGraw-Hill Inc., New York, (1993) 611-615.
15. Salton, G.: Automatic text processing: The transformation, analysis, and retrieval of information by computer. Reading, Ma: Addison-Wesley, (1989).
16. 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) 3-21.
17. Tsujimoto, S., Asada, H.: Understanding Multi-articled Documents. In: Proc. of the Tenth Int. Conf. on Pattern Recognition, Atlantic City, N.J., (1990) 551-556.