Automated Discovery of Dependencies Between Logical Components in Document Image Understanding

Donato Malerba Floriana Esposito Francesca A. Lisi
Oronzo Altamura
Dipartimento di Informatica - Università degli Studi di Bari
via Orabona 4 - 70126 Bari
{malerba, esposito, lisi, altamura}@di.uniba.it

Abstract

Document image understanding denotes the recognition of semantically relevant components in the layout extracted from a document image. This recognition process is based on some visual models, whose manual specification can be a highly demanding task. In order to automatically acquire these models, we propose the application of machine learning techniques. In this paper, problems raised by possible dependencies between concepts to be learned are illustrated and solved with a computational strategy based on the separate-and-parallel-conquer search. The approach is tested on a set of real multi-page documents processed by the system WISDOM++. New results confirm the validity of the proposed strategy and show some limits of the learning system used in this work.

1. Introduction

Recently, many publishing companies have started creating online bibliographic databases of their journal articles. However, a large number of publications is still available solely on paper, and document image analysis tools are essential to support data entry from printed journal articles and proceedings [8]. A straightforward application of OCR technology produces poor results because of the variability of the layout structure of printed documents.

A more advanced solution would be to develop intelligent document processing tools that automatically transform a large variety of printed multi-page documents, especially periodicals, into a web-accessible form such as XML. This transformation requires a solution to several digital image processing problems, such as the separation of textual from graphical components in a document image (document analysis), the recognition of the document (document classification), the identification of logical components (document image understanding), the optical character recognition, and the transformation of the page into HTML/XML format. A large amount of knowledge is required to effectively solve these problems. To support the acquisition of such knowledge, the application of inductive learning techniques has been proposed [3].

In this paper, a new learning issue is investigated in the specific context of document image understanding. Logical components may be related to each other, and such dependence can be reflected by some geometric relationships between the layout components associated to those logical components. For instance, the logical components title and author of a printed paper are generally interrelated sequentially: the author follows the title. In the case of papers published in the IEEE Transactions on Pattern Analysis and Machine Intelligence, such a dependence is conveyed by the following geometric relationship: the layout component labeled as "title" is above the layout component(s) labeled as "author". Thus, when learning rules for document image understanding it is important to capture this typographical convention by generating the following logical clause:

author(X) ← above(Y,X), title(Y)

More generally, rules learned for document image understanding should reflect dependencies between logical components to enable a context-sensitive recognition.

In the above example, it is noteworthy that both author and title are concepts to be learned, and that learning them independently could not lead to the expected result. Most of the studies on inductive learning presented in the machine learning literature make the implicit assumption that concepts are independent (independence assumption). Experimental results of a previous study on inductive learning in the context of document image understanding confirmed that by taking into account concept dependencies it is possible to improve the predictive accuracy of the learned rules [6]. However, in that study multiple dependent concepts could be learned provided the user previously defined

Proceedings of the Sixth International Conference on Document Analysis and Recognition (ICDAR'01)
0-7695-1263-1/01 $10.00 © 2001 IEEE

a graph of possible dependencies among logical components. In this paper, a new approach is presented, which can autonomously discover concept dependencies. This approach is that adopted by ATRE [5], an inductive learning system interfaced by the document processing system WISDOM++ [1]. In Section 2 ATRE is briefly described as regards representation and algorithmic issues. Section 3 illustrates and discusses the experimentation on real-world multi-page documents. Finally, in Section 4 our conclusions are drawn.

2. Learning Multiple Dependent Concepts

The learning problem solved by ATRE can be formulated as follows:

Given

- a set of concepts $C_1, C_2, .., C_r$ to be learned,
- a set of observations *O* described in a language L_O,
- a background knowledge BK described in a language L_{BK},
- a language of hypotheses L_H, a user's preference criterion *PC*,

Find

a (possibly recursive) logical theory *T* for the concepts $C_1, C_2, , C_r$, such that *T* is complete and consistent with respect to O and satisfies the preference criterion *PC*.

The *completeness* property holds when the theory *T* explains all observations in *O* of the *r* concepts C_i, while the *consistency* property holds when the theory *T* explains no counter-example in *O* of any concept C_i. The satisfaction of these properties guarantees the correctness of the induced theory with respect to the given observations *O*. Whether the theory *T* is actually correct, that is whether it classifies correctly all other examples not in *O*, is an extra-logical matter, since no information on the generalization accuracy can be drawn from the training data themselves. In fact, the selection of the "best" theory is always made on the ground of an inductive bias embedded in some heuristic function or expressed by the user of the learning system (preference criterion).

As to the representation languages, the basic component is the literal in the two distinct forms:

$f(t_1, .., t_n)$=*Value* (simple literal)
$f(t_1, .., t_n) \in$ *Range* (set literal),

where *f* and *g* are function symbols called descriptors, t_i's are terms, and *Range* is a closed interval of possible values taken by *f*. Some examples of literals are: color(X1)=red, height(X1)∈ [1.1, .. ,1.2], and ontop(X,Y)=true.

The language of observations L_O allows a more efficient and comprehensible object-centered representation of observations. Indeed, observations are represented by ground multiple-head clauses [4], called *objects*, which have a conjunction of simple literals in the head. An instance of objects taken from the blocks-world is the following:

type(blk1)=lintel ∧ type(blk2)=column ←
pos(blk1)=hor, pos(blk2)=ver, ontop(blk1, blk2),

which is semantically equivalent to the following pair of clauses:

type(blk1)=lintel ←
pos(blk1)=hor, pos(blk2)=ver, ontop(blk1,blk)
type(blk2)= column ←
pos(blk1)=hor, pos(blk2)=ver, ontop(blk1, blk2).

Examples are described as pairs $\langle L, OID \rangle$, where L is a literal in the head of the object indicated by the object identifier OID. Examples can be considered positive or negative, according to the concept to be learned. For instance $\langle type(blk1) = lintel, O1 \rangle$ is a positive example of the concept type(X)=lintel, a negative example of the concept type(X)=column, and it is neither a positive nor a negative example of the concept stable(X)=true.

The language of hypotheses L_H is that of linked, range-restricted definite clauses [2] with simple and set literals in the body and one simple literal in the head. An example of recursive theory expressed in L_H is the following:

logo(X) ← picture(X)
body(X) ← below(Y,X), text(X), logo(Y)
body(X) ← below(Y,X), text(X), body(Y)

It states conditions for recognizing the logo and the body of a letter (all text blocks below the logo). Here X and Y denote variables consistent with Prolog notation. ATRE is also able to deal with numeric descriptors. Given an n-ary function symbol, $f(X_1, .., X_n)$, taking on values in a numerical domain, the system induces hypotheses with set literals $f(X_1, .., X_n) \in [a..b]$, where $[a..b]$ is a numerical interval.

The language of background knowledge L_{BK} has the same constraints as the language of hypotheses.

The high-level learning algorithm in ATRE belongs to the family of *sequential covering*(or *separate-and-conquer*) algorithms [7], since it is based on the strategy of learning one clause at a time (*conquer stage*), removing the covered examples (*separate stage*) and iterating the process on the remaining examples. The most relevant novelties of the learning strategy implemented in ATRE are embedded in the design of the conquer stage. Firstly, the conquer stage of our algorithm aims at generating a clause that covers a specific positive example, called *seed*. Secondly, the search space explored by ATRE is a *forest* of as many search-trees (called *specialization hierarchies*) as the number of chosen seeds, where at least one seed per incomplete concept definition is kept. Each search-tree is rooted with a unit clause and ordered by generalized implication. The forest can be processed in parallel by as many concurrent tasks as the number of search-trees (*parallel-conquer search*). Each task traverses the specialization hierarchies top-down (or general-to-specific), but synchronizes its traversal with the other tasks at each level. Initially, some clauses at depth

one in the forest are examined concurrently. Each task is actually free to adopt its own search strategy and to decide which clauses are worth being tested. If none of the tested clauses is consistent, clauses at depth two are considered. Search proceeds towards deeper and deeper levels of the specialization hierarchies, until at least one consistent clause is found. Task synchronization is performed after all "relevant" clauses at the same depth have been examined. A supervisor task decides whether the search should carry on or not on the basis of the results returned by the concurrent tasks. When the search is stopped, the supervisor selects the "best" consistent clause according to the user's preference criterion PC. This strategy has the advantage that simpler consistent clauses are found first, independently of the concepts to be learned. Moreover, the synchronization allows tasks to save much computational effort when the distribution of consistent clauses in the levels of the different search-trees is uneven. This *separate-and-parallel-conquer* search strategy provides us with a solution to the problem of *interleaving* the induction process for distinct concept definitions.

3. Experimental results

The proposed approach to multiple predicate learning has been applied to the problem of understanding multi-page printed documents. Experiments have been conducted by interfacing ATRE with an intelligent document processing system named WISDOM++ (http://www.di.uniba.it/∼malerba/wisdom++) [1]. 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 ATRE each training page is represented as an *object*, where different constants represent distinct layout components within a page (see Figure 1). All descriptors for the page layout of multi-page documents are listed in Table 1. The following clauses are used as background knowledge, in order to automatically associate information on page order to layout components:

at_page(X)=first ← part_of(Y,X), page(Y)=first
at_page(X)=intermediate ←
    part_of(Y,X), page(Y)=intermediate
at_page(X)=last_but_one←
    part_of(Y,X), page(Y)=last_but_one
at_page(X)=last ← part_of(Y,X), page(Y)=last

Three long papers that appeared in the January 1996 issue of the IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) have been considered. The papers contain thirty_seven pages, each of which has a variable number of layout components (about ten on average). Layout components can be associated with at most one of the following eleven logical labels: *abstract, affiliation, author, biography, caption, figure, index_term, page_number, references, running_head, title*.

As already pointed out, learning rules for document image understanding raises issues concerning both the discovery of concept dependencies and the induction of recursive definitions. For instance, in the case of papers published in journals, the following dependent clauses:,

running_head(X)←
    top_left(X), text(X), even_page_number(X)
running_head(X)←
    top_right (X), text(X), odd_page_number(X)
paragraph(Y)← on_top(X,Y), running_head(X), text(Y),

express the fact that a textual layout component at the top left (right) corner of an even (odd) page is a running head, while a textual layout component below a running_head is a paragraph of the paper. Moreover, the recursive clause,

paragraph(Y)← on_top(X,Y), paragraph(X),text(Y),

is useful to classify all textual layout components below the upper_most paragraph.

By running ATRE on the training set described above, the following theory is returned:

1. logic_type(X)=page_number←
    width(X)∈[2 .. 8],y_pos_centre(X)∈[19 ..40]
2. logic_type(X)=figure ←
    type_of(X)=image,at_page(X)=intermediate
3. logic_type(X)=figure ← type_of(X)=graphic
4. logic_type(X)=running _head ←
    width(X)∈ [388..544],y_pos_centre(X) ∈ [22 .. 39]
5. logic_type(X)=caption ←
    alignment(Y,X)=only_middle_col, logic_type(Y)=figure,
    height(X)∈ [18..75],type_of(X)=text
6. logic_type(X)=running_head ←
    height(X) ∈ [7 ..9],y_pos_centre(X)∈[18 ..39]
7. logic_type(X)=references ←
    height(X)∈[332 .. 355], x_pos_centre(X)∈[153 .. 435]
8. logic_type(X)=abstract←
    at_page(X)=first, width(X)∈[487 .. 488]
9. logic_type(X)= running_head ←
    height(X)∈[6 .. 9], width(X)∈[77 .. 398],
    y_pos_centre(X)∈[18 .. 39]
10. logic_type(X)=title ←
    at_page(X)=first, height(X)∈[18.. 53]
11. logic_type(X)=affiliation ←
    at_page(X)=first, y_pos_centre(X)∈[720 .. 745]
12. logic_type(X)=author ←
    at_page(X)=first, y_pos_centre(X)∈[128 .. 158]

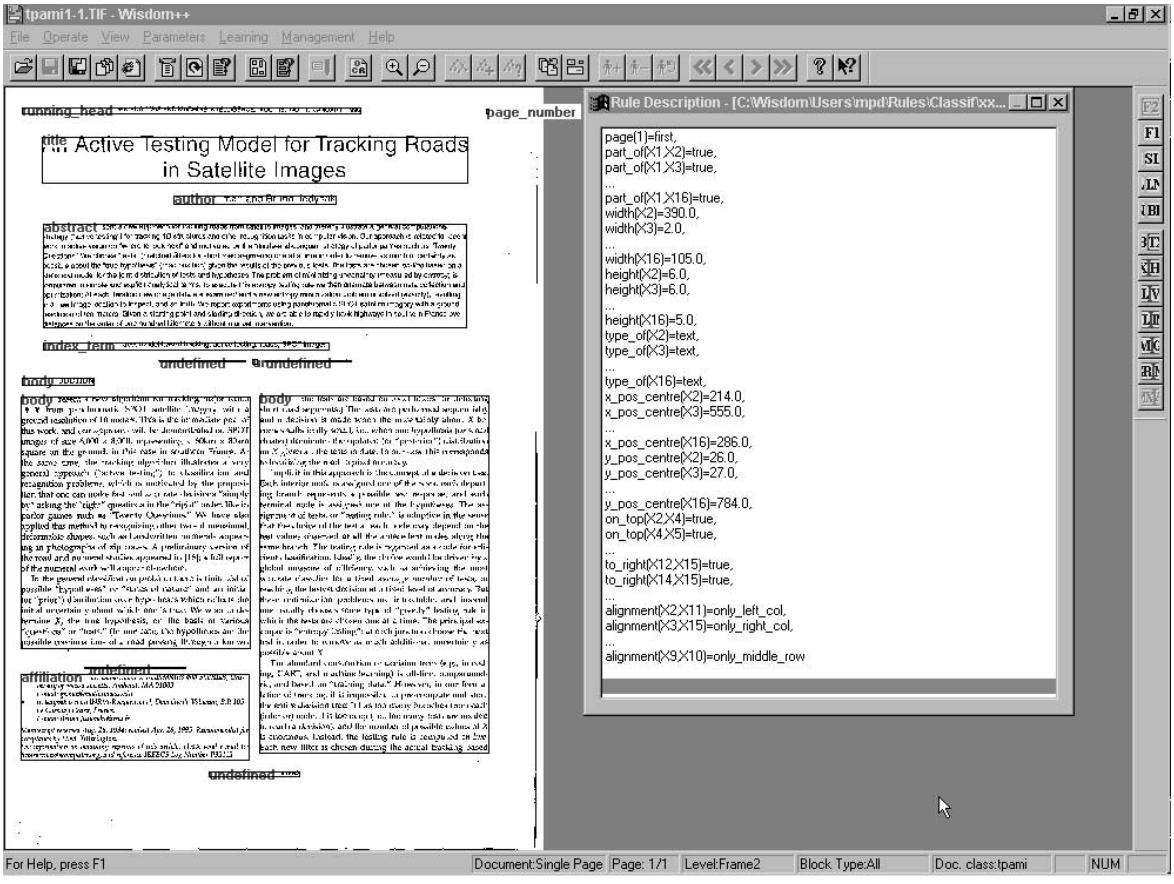

Figure 1. Layout extracted by WISDOM++ for the first page of a multi-page document (*left*) and its logical description (*right*).

Table 1. Descriptors used by WISDOM++ to represent the layout of multi-page documents.

Descriptor	Domain
page(page)	Nominal domain:first,intermediate,last_but_one,last
width(block)	Integer domain: (1..640)
height(block)	Integer domain: (1..875)
x_pos_centre(block)	Integer domain: (1..640)
y_pos_centre(block)	Integer domain: (1..875)
type_of(block)	Nominal domain: text, hor_line, image, ver_line, graphic, mixed
part_of(page,block)	Boolean domain: true if page contains block
on_top(block1,block2)	Boolean domain: true if block1 is above block2
to_rigth(block1,block2)	Boolean domain: true if block3 is to right of block1
alignment(block1,block2)	Nominal domain: only_left_col, only_rigth_col, only_middle_col, both_columns, only_upper_row, only_lower_row, onlymiddle_row, both_rows

13. logic_type(X)=biography ←
    at_page(X)=last, height(X)∈[65 .. 234]
14. logic_type(X)=index_term ←
    height(X)∈[8 .. 8], y_pos_centre(X)∈[263 .. 295]
15. logic_type(X)=running_head ←
    to_right(X,Y),logic_type(Y)= running_head
16. logic_type(X)=caption ←
    on_top(Y,X), logic_type(Y)=figure, type_of(X)=text,
    height(Y)∈[74..313],height(X)∈[9..75]
17. logic_type(X)=caption ←
    height(X)∈[9 .. 40], width(X)∈[263 .. 546], on_top(Y,X),
    height(Y)∈[4 .. 7],at_page(X)=intermediate
18. logic_type(X)=caption ←
    height(X)∈[9 .. 9],width(X)∈[77 .. 214],
    y_pos_centre(X)∈[417 .. 605]
19. logic_type(X)=caption ←
    width(X)∈[501 .. 546], on_top(Y,X),
    logic_type(Y)=running_head

Clauses are reported in the order in which they are learned. The theory contains some concept dependencies (see clauses 5, 16 and 19), as well as some kind of recursion (see clause 15). In particular, the unusual dependency between caption and running head (clause 19) is due to the fact that figure captions cannot be distinguished from table headings in our training set. Furthermore, some expected concept dependencies were not discovered by the system, such as that relating the running head to the page number:

logic_type(X)=page_number←
    to_right(X,Y),logic_type(Y)=running_head
logic_type(X)= page_number←
    to_right(Y,X),logic_type(Y)= running_head

This is due to the semantics of the descriptor to_right, which is generated by WISDOM++ only when two layout components are at a maximum distance of 100 points, which is not the case of articles published on the PAMI transactions. The same consideration applies to other possible concept dependencies (e.g.,title-authors-abstract).

In order to test the predictive accuracy of the learned theory, we considered the fourth long article published in the same issue of the transactions used for training. WISDOM++ segmented the fourteen pages of the article into 169 layout components, twelve of which (about 7%) could not be properly labeled by the learned theory (omission errors). Only two commission errors were observed due to clause 19. Finally, it is noteworthy that many omission errors are due to near misses. For instance, the running head of the first page is not recognized simply because its centroid is located at point 40 along the vertical axis, while none of the ranges for y_pos_center, determined by ATRE during training, includes the value 40 (see clauses 4, 6, and 9). Significant recovery of omission errors can be obtained by relaxing the definition of matching definite clauses.

4. Conclusions

This paper illustrates the problem of learning rules for document image understanding. To carry out the task in hand, it is necessary to establish models, i.e. general descriptions of each logical component to be recognized. These descriptions are expressed in a first-order logic formalism, such that layout components correspond to variables, properties are expressed by means of either unary predicates or function symbols, while spatial relations among layout components are represented by either predicates or function symbols of arity n>1. The main issue in learning models for document image understanding is concept dependency: mutual relations often occur between logical components and it would be sensible to learn rules that express such relations. Discovering concept dependencies is not easy, so in this work we have presented a solution based on a separate-and-parallel-conquer search strategy. The proposed strategy has been implemented in ATRE, a learning system that induces logical theories used by the document processing system WISDOM++ when the document image understanding task is carried out.

References

[1] O. Altamura, F. Esposito, and D. Malerba. Trasforming paper documents into xml format with wisdom++. *International Journal of Document Analysis and Recognition*, 4, 2001.

[2] L. De Raedt. *Interactive Theory Revision*. Academic Press, London, 1992.

[3] F. Esposito, D. Malerba, and F. Lisi. Machine learning for intelligent processing of printed documents. *Journal of Intelligent Information Systems*, 14((2/3)):175–198, 2000.

[4] G. Levi and F. Sirovich. Generalized and-or graphs. *Artificial Intelligence*, 7:243–259, 1976.

[5] D. Malerba, F. Esposito, and F. A. Lisi. Learning recursive theories with atre. In *Proceedings of Thirteenth European Conference on Artificial Intelligence*, Chichester, 1998. John Wiley.

[6] D. Malerba, G. Semeraro, and F. Esposito. A multistrategy approach to learning multiple dependent concepts. In G. Nakhaeizadeh and C. Taylor, editors, *Machine Learning and Statistics:The interface*. Wiley, New York, 1997.

[7] T. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[8] G. Thoma. Automating data entry for an online biomedical database:a document image analysis application. In *Proceedings of Fifth International Conference on Document Analysis and Recognition*, Los Vaqueros,CA, 1999. IEEE Computer Society Press.