Learning to order basic components of structured complex objects

Donato Malerba and Michelangelo Ceci

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

Abstract. Determining the ordering of basic components of structured complex objects can be a crucial problem for several applications. In this paper, we investigate the problem of discovering partial or total orders among basic components by resorting to a data mining approach which acquires the domain specific knowledge from a set of training examples. The input of the learning method is the description of user-defined "chains" of basic components. The output is a logical theory that defines two predicates, $first/1$ and $succ/2$, useful for consistently reconstructing all chains in new structured complex objects. The proposed method resorts to an ILP approach in order to exploit possible relations among basic components. We describe an application of the proposed method to learning the reading order of layout components extracted from document images. Determining the reading order enables the reconstruction of a single textual element from texts associated to multiple layout components and makes both information extraction and content-based retrieval of documents more effective. Experimental results show the effectiveness of the proposed method.

1 Introduction

Inductive learning has mostly concentrated on learning to classify. However, there are many applications in which it is desirable to order rather than classify instances [6, 8]. They include ordering information retrieval results on the basis of the relevance of retrieved documents with respect to a given query [4, 13] and ranking database query results [5]. Mathematically speaking, the problem of learning to order objects can be seen in two different perspectives [12]: in the first case, the problem is to identify a *total ordering* of a set of objects (ranking). In the second case, the problem is to identify several *partial orderings* of objects.

In the literature, several methods have been proposed for learning total or partial orderings [6, 8, 9, 4, 13, 12]. However, most of them assume that training data are represented in a single relational database table, such that each row (or tuple) represents an independent example (an object) and columns correspond to object properties. This single-table assumption seems to be quite restrictive for some applications. In particular, in the document image understanding domain, an ordering (specifically, a reading order) should be learned from the descriptions of basic objects, the layout components, which are (spatially) correlated. The representation of the different relations between basic objecs requires several tables which cannot be suitably handled under the single-table assumption.

In order to overcome limitations imposed by the single table assumption, in this paper we investigate the problem of learning to identify a partial ordering of basic components of complex objects by resorting to an inductive logic programming (ILP) approach [19, 14, 20]. More precisely, we resort to the application of the ILP system ATRE [16] to learn a logical theory which defines two predicates, $first/1$ and $succ/2$, useful for consistently reconstructing partial orderings (in form of chains) in new structured complex objects.

The paper is organized as follows. The problem of learning to order objects is formally defined in Section 2. The machine learning system ATRE applied to the problem of learning the logical theory is introduced in Section 3, while the application of the learned theory in order to reconstruct a partial order relationship is reported in Section 4. Finally, the application to the document image processing domain is illustrated in Section 5, where experimental results are also reported and commented.

2 Problem Definition

In order to formalize the learning problem, some useful definition are necessary.

Definition 1. *Partial Order [10]*
Let A be a set of basic components of a complex object, a partial order P over A is a relation $P \in A \times A$ such that P is

1. *reflexive* $\forall s \in A \Rightarrow (s, s) \in P$
2. *antisymmetric* $\forall s_1, s_2 \in A$: $(s_1, s_2) \in P \wedge (s_2, s_1) \in P \Leftrightarrow s_1 = s_2$
3. *transitive* $\forall s_1, s_2, s_3 \in A$: $(s_1, s_2) \in P \wedge (s_2, s_3) \in P \Rightarrow (s_1, s_3) \in P$

When P satises the antisymmetric, the transitive and the irreflexive ($\forall s \in A \Rightarrow (s, s) \notin P$) properties, it is called a weak partial order over A.

Definition 2. *Total Order*
Let A be a set of basic components of a complex object, a partial order T over the set A is a total order iff $\forall s_1, s_2 \in A$: $(s_1, s_2) \in T \vee (s_2, s_1) \in T$

Definition 3. *Complete chain, Chain reduction*
Let A be a set of basic components of a complex object, let D be a weak partial order over A, let $B = \{a \in A | (\exists b \in A \ s.t. \ (a, b) \in D \vee (b, a) \in D)\}$ be the subset of elements in A related to any element in A itself. If $D \cup \{(a, a) | a \in B\}$ is a total order over B then D is a complete chain over A.
Furthermore, $C = \{(a, b) \in D | \neg\exists c \in A \ s.t. \ (a, c) \in D \wedge (c, b) \in D\}$ is the reduction of the chain D over A.

Example 1. Let $A = \{a, b, c, d, e\}$. $D = \{(a, b), (a, c), (a, d), (b, c), (b, d), (c, d)\}$ is a complete chain over A, then $C = \{(a, b), (b, c), (c, d)\}$ is its reduction.

Indeed, for our purposes it is equivalent to deal with complete chains or their reduction. Henceforth, for the sake of simplicity, the term *chain* will denote the reduction of a complete chain. By resorting to definitions above, it is possible to formalize the ordering induction problem as follows:

Given :

- A description $DesTO_i$ in the language L of the set of n training complex objects $TrainingObjs = \{TP_i \in \Pi | i = 1..n\}$ (where Π is the set of complex objects).
- A description $DesTC_i$ in the language L of the set TC_i of chains (over $TP_i \in TrainingObjs$) for each $TP_i \in TrainingObjs$.

Find :
An intensional definition T in the language L of a chain over a generic compex object $O \in \Pi$ such that T is complete and consistent with respect to all training chains descriptions $DesTC_i$, $i = 1..n$.

In this problem definition, we refer to the intensional definition T as a first order logic theory. The fact that T is complete and consistent with respect to all training chains descriptions can be formally described as follows:

Definition 4 (Completeness and Consistency).
Let:

- *T be a logic theory describing chains instances expressed in the language L.*
- *E^+ be the set of positive examples for the chains instances.*
  ($E^+ = \bigcup_{i=1..n}(\bigcup_{TC \in TC_i} TC)$).
- *E^- be the set of negative examples for the chains instances.*
  ($E^- = \bigcup_{i=1..n}(TP_i \times TP_i)/E^+$).
- *$DesE^+$ be the description of E^+ in L.*
- *$DesE^-$ be the description of E^- in L.*

then T is complete and consistent with respect to all training chains descriptions iff $T \models DesE^+ \ \wedge \ T \not\models DesE^-$

This formalization of the problem permits to represent and identify distinct orderings on the same complex object and allows to avoid to include in the ordering basic components that should not be included.

3 ATRE: The Learning System

ATRE is an ILP system that can learn recursive theories from examples. The learning problem solved by ATRE can be formulated as follows:
Given

- a set of *concepts* $C_1, C_2, \ldots, C_r$ to be learned
- a set of *observations* O described in a language L_O

– a *background theory* BK
– a *language* of hypotheses L_H
– a user's *preference criterion* PC

Find

A logical theory T expressed in the language L_H and defining the concepts $C_1, C_2, \ldots, C_r$, 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 $C_1, C_2, \ldots, C_r$, 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 correct with respect to additional observations 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 grounds of an inductive bias embedded in some heuristic function or expressed by the user of the learning system (preference criterion).

In the context of the ordering induction problem, we identified two concepts to be learned, namely $first/1$ and $succ/2$. The former refers to the the first basic component of a chain, while the latter refers to the relation *successor* between two basic components in a chain. By combining the two concepts it is possible to identify a partial ordering of basic componens of a complex object.

As to the representation languages, literals can be of the two distinct forms: $f(t_1, \ldots, t_n) = \text{Value}$ (simple literal); $f(t_1, \ldots, t_n) \in \text{Range}$ (set literal), where f and g are function symbols called *descriptors*, t_i's are *terms* (constants or variables) and *Range* is a closed interval of possible values taken by f.

4 Application of Learned Rules

Once rules have been learned, they can be applied to new complex objects in order to generate a set of ground atoms such as: $\{first(0) = true, succ(0,1) = true, \ldots, succ(4,3) = true, \ldots\}$ which can be used to reconstruct chains of (possibly logically labelled) layout components. In our approach, we propose two different solutions: 1) Identification of multiple chains of basic components. 2) Identification of a single chain of basic components.

By applying rules learned by ATRE, it is possible to identify:

– A *directed* graph $G = < V, E >$[1] where V is the set of nodes representing all the basic components of a complex object and edges represent the existence of a *succ* relation between two basic components, that is, $E = \{(b_1, b_2) \in V^2 | succ(b_1, b_2) = true\}$
– A list of initial nodes $I = \{b \in V | first(b) = true\}$

Both approaches make use of G and I in order to identify chains.

[1] G is not a direct acyclic graph (dag) since it could also contain cycles.

Multiple chains identification This approach aims at identifying a (possibly empty) set of chains over the set of basic components of a complex object. It is based on two steps, the first of which aims at identifying the heads (first elements) of the possible chains, that is the set

$$Heads = I \cup \{b_1 \in V| \ \exists b_2 \in V \ (b_1, b_2) \in E \ \wedge \forall b_0 \in V \ (b_0, b_1) \notin E\}$$

This set contains both nodes for which $first$ is true and nodes which occur as a first argument in a true *succ* atom and do not occur as a second argument in any true *succ* atom.

Once the set $Heads$ has been identified, it is necessary to reconstruct the distinct chains. Intuitively, each chain is the list of nodes forming a path in G which begins with a node in $Heads$ and ends with a node without outgoing edges. Formally, an extracted chain $C \subseteq E$ is defined as follows:

$C = \{(b_1, b_2), (b_2, b_3), \ldots, (b_k, b_{k+1})\}$, such that $b_1 \in Heads$, $\forall i = 1..k$: $(b_i, b_{i+1}) \in E$ and $\forall b \in V \ (b_{k+1}, b) \notin E$.
In order to avoid cyclic paths, we impose that the same node cannot appear more than once in the same chain. The motivation for this constraint is that the same layout component is generally not read more than once by the reader.

Single chain identification The result of the second approach is a single chain. Following the proposal reported in [6], we aim at iteratively evaluating the most promising node to be appended to the resulting chain. More formally, let $PREF_G : V \times V \rightarrow \{0, 1\}$ be a preference function defined as follows:

$$PREF_G(b_1, b_2) = \begin{cases} 1 \text{ if } b_1 = b_2 \text{ or a path connecting } b_1 \text{ and } b_2 \text{ exists in } G \\ 0 \text{ otherwise} \end{cases}$$

Let $\mu : V \rightarrow \mathbb{N}$ be the function defined as follows:

$$\begin{aligned} \mu(L, G, I, b) = & \ countConnections(L, G, I, b) + outGoing(V/L, b) \\ & -inComing(V/L, b) \end{aligned}$$

where

- $G =< V, E >$ is the ordered graph
- L is a list of *distinct* nodes in G
- $b \in V/L$ is a candidate node
- $countConnections(L, G, I, b) = |\{d \in L \cup I | PREF_G(d, b) = 1\}|$ counts the number of nodes in $L \cup I$ from which b is reachable.
- $outGoing(V/L, b) = |\{d \in V/L | PREF_G(b, d) = 1\}|$ counts the number of nodes in V/L reachable from b.
- $inComing(V/L, b) = |\{d \in V/L | PREF_G(d, b) = 1\}|$ counts the number of nodes in V/L from which b is reachable.

Algorithm 1 fully specifies the method for the single chain identification. The rationale is that at each step a node is added to the final chain. Such a node is that for which μ is the highest. Higher values of μ are given to nodes which can be reached from I, as well as from other nodes already added to the chain, and have a high (low) number of outgoing (incoming) paths to (from) nodes in V/L. Indeed, the algorithm returns an ordered list of nodes which could be straightforwardly transformed into a chain.

Algorithm 1 Single chain identification algorithm

```
1:  findChain ($G =< V, E >, I$) Output: L: chain of nodes
2:  L← $\emptyset$;
3:  repeat
4:     $best\_mu \leftarrow -\infty$;
5:     for all $b \in V/L$ do
6:        $cc \leftarrow countConnections(L, G, I, b)$;
7:        $inC \leftarrow incoming(V/L, b)$; $outG \leftarrow outGoing(V/L, b)$;
8:        if $((cc \neq 0)$ AND $(inC \neq 0)$ AND $(outG \neq 0))$ then
9:           $\mu \leftarrow cc + outG - inC$;
10:          if $best\_mu < \mu$ then
11:             $best\_b \leftarrow b$; $best\_mu \leftarrow \mu$;
12:          end if
13:       end if
14:    end for
15:    if $(best\_mu <> -\infty)$ then
16:       $L.add(best\_b)$;
17:    end if
18: until $best\_mu = -\infty$
19: return L
```

5 The Application: Learning Reading Order of Layout Components

In this paper, we investigate an application to the document image understanding problem. More specifically, we are interested in determining the reading order of layout components in each page of a multi-page document. Indeed, the spatial order in which the information appears in a paper document may have more to do with optimizing the print process than with reflecting the logical order of the information contained. Determining the correct reading order can be a crucial problem for several applications. By following the reading order recognized in a document image, it is possible to cluster together text regions labeled with the same logical label into the same textual component (e.g., “introduction”, “results”, “method” of a scientific paper). In this way, the reconstruction of a single textual component is supported and advanced techniques for text processing can be subsequently applied. For instance, information extraction methods may be applied locally to reconstructed textual components. Moreover, retrieval of document images on the basis of their textual contents is more effectively supported.

Several works on reading order detection have already been reported in the literature [22][11][18][21][1] [3]. A common aspect of all methods is that they strongly depend on the specific domain and are not "reusable" when the classes of documents or the task at hand change. There is no work, to the best of our knowledge, that handles the reading order problem by resorting to machine learning techniques, which can generate the required knowledge from a set of training layout structures whose correct reading order has been provided by the user. In previous works on document image understanding, we investigated the application of machine learning techniques to several knowledge-based document image processing tasks, such as classification of blocks [2], automatic global layout analysis correction [17], classification of documents into a set of pre-defined classes and logical labelling. Following this mainstream of research, herein we consider the problem of learning the reading order.

In this context the limitations posed by the single table assumption are quite restrictive for at least two reasons. First, layout components cannot be realistically considered independent observations, because their spatial arrangement is mutually constrained by formatting rules typically used in document editing. Second, spatial relationships between a layout component and a variable number of other components in its neighborhood cannot be properly represented by a fixed number of attributes in a table. Even more so, the representation of properties of the other components in the neighborhood, because different layout components may have different properties (e.g., the property "brightness" is appropriate for half-tone images, but not for textual components). Since the single-table assumption limits the representation of relationships (spatial or non) between examples, it also prevents the discovery of this kind of pattern, which can be very useful in document image mining.

For these reasons, the ILP approach proposed in this paper seems to be appropriate for the task at hand. In ATRE, training observations are represented by ground multiple-head clauses [15], called *objects*, which have a conjunction of simple literals in the head. The head of an object contains positive and negative examples for the concepts to be learned, while the body contains the description of layout components on the basis of geometrical features (e.g. width, height) and topological relations (e.g. vertical and horizontal alignments) existing among blocks, the type of the content (e.g. text, horizontal line, image) and the logic type of a block (e.g. title or authors of a scientific paper). Terms of literals in objects can only be constants, where different constants represent distinct layout components within a page. An example of object description generated for the document page in Figure 1 is the following:

$object('tpami17\_1\text{-}13', [class(p) = tpami,$
$first(0) = true, first(1) = false, ...$
$succ(0,1) = true, succ(1,2) = true, ..., succ(7,8) = true, succ(2,10) = false, ...],$
$[part\_of(p,0) = true, ...,$

$height(0) = 83, height(1) = 11, ..., width(0) = 514, width(1) = 207, ...,$
$type\_of(0) = text, ..., type\_of(11) = hor\_line,$
$title(0) = true, author(1) = true, affiliation(2) = true, .., undefined(16) = true,$
$x\_pos\_centre(0) = 300, x\_pos\_centre(1) = 299, ...,$
$y\_pos\_centre(0) = 132, y\_pos\_centre(1) = 192, ...,$
$on\_top(9, 0) = true, on\_top(15, 0) = true, ..., to\_right(6, 8) = true, ...$
$alignment(16, 8) = only\_right\_col, alignment(17, 5) = only\_left\_col, ...$
$class(p) = tpami, page(p) = first]).$

The constant p denotes the whole page while the remaining integer constants (0, 1, . . ., 17) identify distinct layout components. In this example, the block number 0 corresponds to the first block to read ($first(0) = true$), it is a textual component ($type\_of(0) = text$) and it is logically labelled as 'title' ($title(0) = true$). Block number 1 (immediately) follows block 0 in the reading order ($succ(0, 1) = true$), it is a textual component and it includes information on the authors of the paper ($author(1) = true$).

As explained in the previous sections, ATRE learns a logical theory T defining the concepts $first/1$ and $succ/2$ such that T is complete and consistent with respect to the examples. This means that it is necessary to represent both positive and negative examples and the representation of negative examples for the concept $succ/2$ poses some feasibility problems due to their quadratic growth. In order to reduce the number of negative examples, we resort to sampling techniques. In our case, we sampled negative examples by limiting their number to 1000% of the number of positive examples. This way, it is possible to simplify the learning stage and to have rules that are less specialized and avoid overfitting.

In order to evaluate the applicability of the proposed approach to reading order identification, we considered a set of multi-page articles published in an international journal. In particular, we considered twenty-four papers, published as either regular or short articles, in the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), in the January and February issues of 1996. Each paper is a multi-page document; therefore, we processed 211 document images. Each document page corresponds to a 24bit TIFF color image.

Initially, document images are pre-processed in order to segment them, perform layout analysis, identify the membership class and map the layout structure of each page into the logical structure. Training examples are then generated by manually specifying the reading order. In all, 211 positive examples and 3,263 negative examples for the concept $first/1$ and 1,418 positive examples and 15,518 negative examples for the concept $succ/2$ are generated.

We evaluated the performance of the proposed approach by means of a 6-fold cross-validation: the dataset is first divided into 6 *folds* of equal size and then, for every fold, the learner is trained on the remaining folds and tested on it.

For each learning problem, statistics on precision and recall of the learned logical theory are recorded. In order to evaluate the ordering returned by the proposed approach, we resort to metrics used in information retrieval to evaluate the returned ranking of results [7]. Herein we consider the metrics valid for partial orders evaluation.

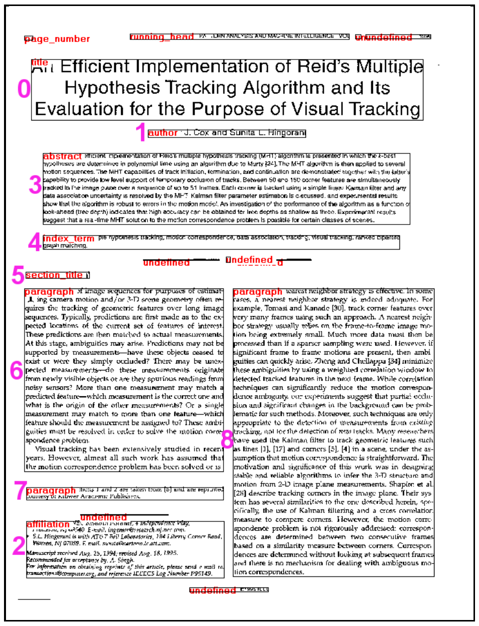
page_number
running_head
undefined
title
An Efficient Implementation of Reid's Multiple Hypothesis Tracking Algorithm and Its Evaluation for the Purpose of Visual Tracking
0
author
I. J. Cox and Sunita L. Hingorani
1
abstract
3
index_term
4
undefined
undefined
5
section_title
paragraph
6
paragraph
8
7
paragraph
undefined
affiliation
2
undefined

Fig. 1. A document page: the input reading order chain. Sequential numbers indicate the reading order.

In particular, we consider the *normalized Spearman footrule distance* which, given two complete lists L and L_1 on a set S (that is, L and L_1 are two different permutations without repetition of all the elements in S), is defined as follows:

$$F(L, L_1) = \frac{\sum_{b \in S} abs(pos(L, b) - pos(L_1, b))}{|S|^2/2} \tag{1}$$

where the function $pos(L, b)$ returns the position of the element b in the ordered list L. This measure can be straightforwardly generalized to the case of several lists: $F(L, L_1, \ldots, L_k) = 1/k \sum_{i=1 \ldots k} F(L, L_i)$.

Indeed, this measure is specifically designed for total orders and not for partial ones. In order to consider partial orders, we resorted to a variant of this measure (*induced normalized footrule distance*):
$F(L, L_1, \ldots, L_k) = 1/k \sum_{i=1 \ldots k} F(L|_{L_i}, L_i)$
where $L|_{L_i}$ is the projection of L on L_i. Since this measure does not take into account the length of single lists, we also adopted the *normalized scaled footrule distance*:

$$F'(L, L_1) = \frac{\sum_{b \in S} abs(pos(L, b)/|L| - pos(L_1, b)/|L_1|)}{|L_1|/2} \tag{2}$$

Also in this case it is possible to extend the measure to the case of multiple lists: $F'(L, L_1, \ldots, L_k) = 1/k \sum_{i=1 \ldots k} F'(L|_{L_i}, L_i)$.

In this study, we apply such distance measures to chains. In particular FD= $F(L|_{L_1}, L_1)$ and SFD=$F'(L|_{L_1}, L_1)$ are used in the evaluation of single chain identification. While IFD=$F(L, L_1, \ldots, L_k)$ and ISFD=$F'(L, L_1, \ldots, L_k)$ are used in the evaluation of multiple chains identification.

Results reported in Table 1 show that the system has a precision of about 65% and a recall greater than 75%. Moreover, there is no significant difference in terms of recall between the two concepts, while precision is higher for rules concerning the *succ* concept. This is mainly due to the specificity of rules learned for the concept $first$: rules learned for the concept $first$ cover (on average) fewer positive examples than rules learned for the concept *succ* (statistics are not reported here beacusse of space contraints). We can conclude that the concept $first$ appears to be more complex to learn than the concept *succ*, probably because of the lower number of training examples (one per page).

Experimental results concerning the reconstruction of single/multiple chains are reported in Table 2. We recall that the lower the distance value the better the reconstruction of the original chain(s). By comparing results in terms of the *footrule distance* measure (IFD vs FD), we note that the reconstruction of multiple chains shows better results than the reconstruction of single chains. Indeed, this result does not take into account the length of the lists. When considering the length of the lists (ISFD vs. SFD) the situation is completely different and the reconstruction of single chains outperforms the reconstruction of multiple chains.

Some examples of rules learned by ATRE are reported below:

1. $first(X1) = true \leftarrow x\_pos\_centre(X1) \in [55..177]$,
   $y\_pos\_centre(X1) \in [60..121], height(X1) \in [98..138]$.
2. $first(X1) = true \leftarrow title(X1) = true,\ x\_pos\_centre(X1) \in [293..341]$,
   $succ(X1, X2) = true$.
3. $succ(X2, X1) = true \leftarrow affiliation(X1) = true,\ author(X2) = true$,
   $height(X1) \in [45..124]$.
4. $succ(X2, X1) = true \leftarrow alignment(X1, X3) = both\_columns$,
   $on\_top(X2, X3) = true, succ(X1, X3) = true,\ height(X1) \in [10..15]$

They show that ATRE is particularly indicated for the task at hand since it is able to identify dependencies among concepts to be learned or even recursion.

Concept	$first/1$		$succ/2$		*Overall*	
	Precision %	Recall%	Precision%	Recall%	Precision%	Recall%
FOLD1	75.00	50.00	76.90	64.10	76.60	61.80
FOLD2	66.70	63.20	74.10	65.20	73.00	64.90
FOLD3	74.30	78.80	81.00	66.10	80.10	67.40
FOLD4	69.40	71.40	67.80	56.30	68.00	58.20
FOLD5	66.70	66.70	78.40	68.70	76.80	68.40
FOLD6	71.00	61.10	79.40	62.90	78.20	62.60
AVG	70.52%	65.20%	76.27%	63.88%	75.45%	63.88%

Table 1. Precision and Recall results shown per concept to be learned

6 Conclusions

In this paper, we present an ILP approach to the problem of inducing a partial ordering between basic components of complex objects. The proposed solution is based on learning a logical theory which defines two predicates $first/1$ and $succ/2$. The learned theory should be able to express dependencies between the two target predicates. For this reason we used the learning system ATRE which is able to learn mutually recursive predicate definitions. In the recognition phase, learned predicate definitions are used to reconstruct reading order chains according two different modalities: single vs. multiple chains identification.

The proposed approach can be applied to several application domains. In this paper, it has been applied to a real-world problem, namely detecting the reading order between layout components extracted from images of multi-page documents. Results prove that learned rules are quite accurate and that the reconstruction phase significantly depends on the application at hand. In particular, if the user is interested in reconstructing the actual chain (e.g. text reconstruction for rendering purposes), the best solution is in the identification of single chains. On the contrary, when the user is interested in recomposing text such that sequential components are correctly linked (e.g. in information extraction), the most promising solution is the identification of multiple chains.

For future work we intend to extend our empirical investigation to other application domains as well as to synthetically generated datasets.

Acknowledgments

This work is partial fulfillment of the research objectives of the project "D.A.M.A." (Document Acquisition, Management and Archiving).

References

1. M. Aiello, C. Monz, L. Todoran, and M. Worring. Document understanding for a broad class of documents. *International Journal on Document Analysis and Recognition-IJDAR*, 5(1):1–16, 2002.
2. O. Altamura, F. Esposito, and D. Malerba. Transforming paper documents into XML format with WISDOM++. *IJDAR*, 4(1):2–17, 2001.

Concept	Multiple chains		Single chain	
	AVG. IFD%	AVG. ISFD%	AVG. FD%	AVG. SFD%
FOLD1	13.18	21.12	47.33	10.17
FOLD2	10.98	18.51	46.32	8.13
FOLD3	1.31	26.91	47.32	17.63
FOLD4	1.32	24.00	49.96	14.51
FOLD5	0.90	22.50	49.31	10.60
FOLD6	0.90	27.65	54.38	12.97
AVG	4.76%	23.45%	49.10%	12.33%

Table 2. Reading order reconstruction results

3. T. M. Breuel. High performance document layout analysis. In *Proceedings of the 2003 Symposium on Document Image Understanding (SDIUT '03)*, 2003.
4. S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW7: Proceedings of the seventh international conference on World Wide Web 7*, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.
5. S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum. Probabilistic information retrieval approach for ranking of database query results. *ACM Trans. Database Syst.*, 31(3):1134–1168, 2006.
6. W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *J. Artif. Intell. Res. (JAIR)*, 10:243–270, 1999.
7. C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 613–622, New York, NY, USA, 2001. ACM Press.
8. A. Gionis, T. Kujala, and H. Mannila. Fragments of order. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 129–136, New York, NY, USA, 2003. ACM Press.
9. A. Gionis, H. Mannila, K. Puolamäki, and A. Ukkonen. Algorithms for discovering bucket orders from data. In *Proceedings 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 561–566. ACM, 2006.
10. R. P. Grimaldi. *Discrete and Combinatorial Mathematics, an Applied Introduction.* Addison Wesley, terza edizione, 1994.
11. Y. Ishitani. Document transformation system from papers to xml data based on pivot xml document method. In *ICDAR '03: 7th International Conference on Document Analysis and Recognition*, page 250. IEEE Computer Society, 2003.
12. S. Kambhampati and J. Chen. Relative utility of EBG based plan reuse in partial ordering vs. total ordering planning. In *AAAI-93*, pages 514–519, Washington, D.C., USA, 1993. AAAI Press/MIT Press.
13. J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.
14. N. Lavrač and S. Džeroski. *Inductive Logic Programming: techniques and applications.* Ellis Horwood, Chichester, 1994.
15. G. Levi and F. Sirovich. Generalized and/or graphs. *Artif. Intell.*, 7(3):243–259, 1976.
16. D. Malerba. Learning recursive theories in the normal ilp setting. *Fundamenta Informaticae*, 57(1):39–77, 2003.
17. D. Malerba, F. Esposito, O. Altamura, M. Ceci, and M. Berardi. Correcting the document layout: A machine learning approach. In *ICDAR*, page 97, 2003.
18. J.-L. Meunier. Optimized xy-cut for determining a page reading order. In *ICDAR '05: Proceedings of the Eighth International Conference on Document Analysis and Recognition*, pages 347–351, Washington, DC, USA, 2005. IEEE Computer Society.
19. S. Muggleton. *Inductive Logic Programming.* Academic Press, London, 1992.
20. S.-W. Nienhuys-Cheng and R. de Wolf. *Foundations of inductive logic programming.* Springer, Heidelberg, 1997.
21. S. L. Taylor, D. A. Dahl, M. Lipshutz, C. Weir, L. M. Norton, R. Nilson, and M. Linebarger. Integrated text and image understanding for document understanding. In *HLT '94: Proceedings of the workshop on Human Language Technology*, pages 421–426, 1994.
22. S. Tsujimoto and H. Asada. Understanding multi-articled documents. In *in Proceedings of the 10th International Conference on Pattern Recognition*, pages 551–556, 1990.