Complex Objects Ranking: A Relational Data Mining Approach

Michelangelo Ceci, Annalisa Appice, Corrado Loglisci, Donato Malerba
Dipartimento di Informatica
Universita' degli Studi di Bari
via Orabona 4, Bari, Italy
{ceci,appice,loglisci,malerba}@di.uniba.it

ABSTRACT

A key task in data mining and information retrieval is learning preference relations. Most of methods reported in the literature learn preference relations between objects which are represented by attribute-value pairs or feature vectors (propositional representation). The growing interest in data mining techniques which are able to directly deal with more sophisticated representations of complex objects, motivates the investigation of relational learning methods for learning preference relations. In this paper, we present a probabilistic relational data mining method which permits to model preference relations between complex objects. Preference relations are then used to rank objects. Experiments on two ranking problems for scientific literature mining prove the effectiveness of the proposed method.

Categories and Subject Descriptors

H.2 [**Data Management**]: Database applications—*Data mining*

General Terms

Design, Algorithms, Experimentations

Keywords

Ranking, Relational Data Mining, Probabilistic approach

1. INTRODUCTION

The problem of learning preference functions has recently received increasing attention due to its many potential applications to information retrieval problems. Studies reported in the literature address either the task of ranking labels associated with objects [12] or the task of ranking objects. In this paper, we focus on the second task, where existing studies are mainly based on two different approaches. The first approach aims at learning a function which assigns a numeric value to each item of a set. This numeric preference is then used to rank items. The second approach asks for less: the learned preference function has to make pairwise comparisons in order to define a relative order (if any) between two objects. In a subsequent step this preference function is used to obtain either a total or a partial ordering of objects in a set.

As regards the first approach, some works reformulate the problem of learning to rank as an ordinal regression problem. For instance, Herbrich et al. [11] propose to learn the mapping of an input vector to a member of an ordered set of numerical ranks. They model ranks as intervals on the real line and consider loss functions that depend on pairs of examples and their target ranks. A similar solution is proposed in [4], where learned functions are modeled by perceptrons.

As regards the second approach, Dekel et al. [6] provide a framework for ranking based on directed graphs, where an arc from A to B means that A has to be ranked higher than B. Arcs are computed according to log-linear models. A drawback of this approach is that it does not quantify the degree of preference. As observed in [2] "ranking algorithms often model preferences, and the ascription of preferences is a much more subjective process than the ascription of, say, classes". To overcome this limitation, Freund et al. [10] propose to exploit a probabilistic approach which permits to compute the probability that A follows B. This probability is computed by exploiting decision stumps as weak learners. The probability is a function of the margin over reweighted examples. Burges et. al [2] propose to estimate probabilities on the basis of a cost function computed according to a logistic regression function. Differently, in [3], a naive Bayesian classifier is used to estimate such probabilities.

Although the first approach appears to be more efficient, it is applicable only when a unique total ordering between objects is admissible. When not all the objects have to be necessarily ranked, or more than one ordering is admissible, the second approach is more suitable.

An important common aspect of all methods reported above is that they work on training data represented in a single relational database table, such that each row (or tuple) represents an object and columns correspond to object properties. This tabular representation of data, also known as *propositional* representation, turns out to be too restrictive for several applications, whose units of analysis have a complex structure involving several objects described by different sets of properties and related by one or more relationships. Some of these objects, called *reference* objects, represent the units of analysis and are the main subject of the analysis. The other objects, called *task-relevant* objects, contribute to defining the units of analysis but are not the

SAC'10 March 22-26, 2010, Sierre, Switzerland.

target of the analysis. The *relational* representation of these units of analysis can be naturally modeled as a set of tables, such that each table describes a specific type of objects involved in the units of analysis, while foreign key constraints model relationships between objects. Examples of units of analysis with a complex structure can be found in document image understanding which denotes the recognition of semantically relevant layout components and relations in the layout extracted from a document image [18]. In this case, units of analysis are semantically relevant layout components that can be implicitly related by locational properties (e.g., a layout component that represents the title of a scientific paper is on top of another layout component which represents the abstract).

At the best of our knowledge, only two methods have been proposed to rank complex objects. The former is presented in [16], where the authors propose to apply an Inductive Logic Programming algorithm to learn a logical theory which defines the predecessor relation. However, learned definitions are "crisp" and do not provide us with a degree of preference. The latter is presented in [14], where the authors propose a probabilistic relational kernel model for preference learning based on relational graph kernels. This method allows us to mine relations between units of analysis (or reference objects) only, hence it suffers from some limits of complex data modeling.

In this paper, we present a different method, named CORA (Complex Objects Ranking Algorithm), which resort to Relational Data Mining [8] in order to deal with complex objects, i.e., units of analysis whose structure is represented by multiple database relations. CORA discovers relational preference patterns which determine when a complex object A precedes (in preference) another object B and then it uses these *preference* patterns to estimate the probability of the preference relation for any pair of complex objects. This probability is finally used to rank the objects.

The paper is organized as follows. The problem of learning preference relations between complex objects and an overview of how CORA addresses this learning problem is presented in Section 2. The discovery of preference relational patterns is presented in Section 3, while the probability estimation is described in Section 4. The algorithm to rank complex objects is discussed in Section 5. Section 6 is devoted to the presentation of two applications of CORA in the context of scientific document image understanding.

2. MINING PREFERENCE RELATIONS

The problem of mining preference relations can be formalized as follows:

Given: A database schema S with h relational tables $S = \{T_1, \ldots, T_h\}$. A set PK of primary key constraints on tables in S. A set FK of foreign key constrains on tables in S. A target relation $T \in S$ [1]. A preference relation $PT \in S$ with two attributes[2]. A training database $TrDB$ with schema S and a testing database $TeDB$ represented according to the schema $S - \{PT\}$.

Find: A ranking of reference objects $(a_1, a_2, \ldots, a_n)$, where $a_i \in TeDB.T$.

[1]Objects in T play the role of reference objects, while objects in $S - \{T, PT\}$ play the role of task-relevant objects.

[2]Each tuple in PT represents an ordered pair of reference objects where the first object precedes the second one

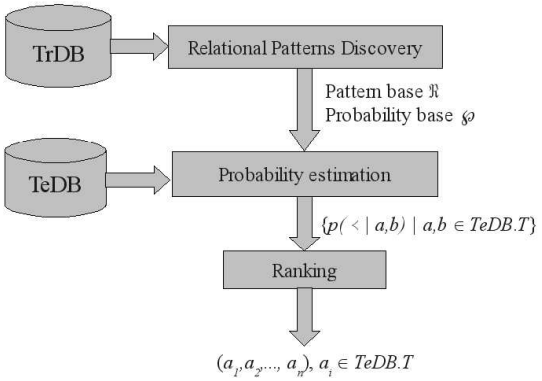

Figure 1: The CORA workflow

The ranking is computed on the basis of the probability $P(\prec |a, b)$, that is, the probability of the relation "precedes" between a and b. By applying the Bayes theorem, this probability can be computed as:

$$P(\prec |a, b) = P(\prec)P(a, b| \prec)/P(a, b). \quad (1)$$

The term $P(\prec)$ in (1) denotes the prior probability that an object precedes another and is computed as:

$$P(\prec) = \frac{|TrDB.PT|}{|TrDB.T| \cdot (|TrDB.T| - 1)} \quad (2)$$

This probability equals 0.5 when training reference objects are totally ordered, while it differs from 0.5 for partial orders.

The term $P(a, b| \prec)$ in (1) denotes the likelihood. In order to simplify its computation, conditional independence is assumed (*naïve Bayes assumption*), according to which:

$$\begin{aligned} P(a, b| \prec) &= P(a_1, \ldots, a_m, b_1, \ldots, b_m| \prec) = \\ &= P(a_1, b_1| \prec) \cdot \ldots \cdot P(a_m, b_m| \prec) \end{aligned} \quad (3)$$

where $a_1, \ldots, a_m$ represent the set of attribute values of a and $b_1, \ldots, b_m$ represent the set of attribute values of b.

Finally, the term $P(a, b)$ in (1) is computed as:

$$P(a, b) = P(\prec)P(a, b| \prec) + (1 - P(\prec))P(b, a| \prec) \quad (4)$$

The formulation in (3) is limited to propositional representations. In the case of complex objects, some extensions are necessary. The basic idea in CORA is that of using a set $\Re$ of particular relational patterns, called *preference relational patterns*, to describe the preference relation between reference objects, and then to define a decomposition of the likelihood *à la* naive Bayesian classifier in order to simplify the probability estimation problem. Probabilities are finally used to rank reference objects. The workflow of CORA is reported in Figure 1.

3. RELATIONAL PATTERNS DISCOVERY

A relational pattern is a set of atoms (atomset) [5]. An atom is a predicate applied to a tuple of terms (variables or constants). Variables denote reference objects in T or some task-relevant objects in $S - \{T, PT\}$, while constants denote attribute values. The set of predicates is automatically defined on the basis of the database schema S. Predicates can be categorized into three classes: *key predicates*, *property predicates* and *structural predicates*. The *key predicates*

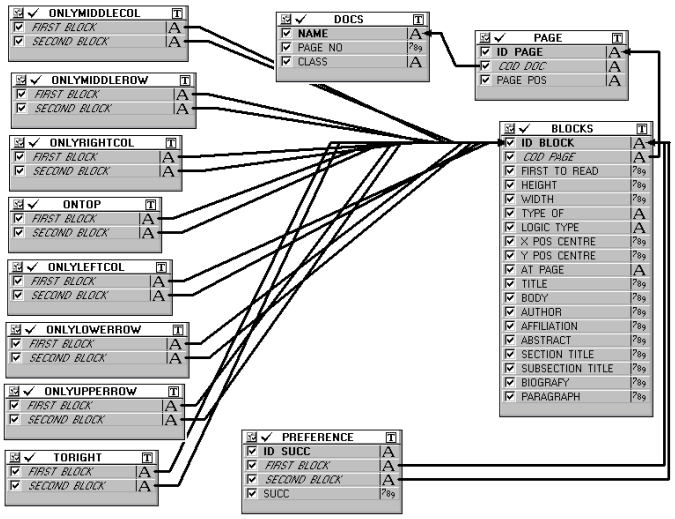

Figure 2: Logical view of the database schema

identify the reference objects. There are two key predicates that represent A and B, respectively, in the $A \prec B$ preference relation. The *property predicates* are binary predicates which define the value taken by an attribute of an object. The *structural predicates* are binary predicates that represent foreign key constraints and relate task-relevant objects with task-relevant objects or reference objects with task-relevant objects. Relational patterns discovered by CORA describe preference relations between two reference objects:

DEFINITION 3.1. *A preference relational pattern P is a set of atoms:*

$$preference(t'_0, t''_0), key1(t'_0), \{p_i(t'_h, t'_k)\}_{i=1,\ldots,s}, key2(t''_0), \{p_i(t''_h, t''_k)\}_{i=s+1,\ldots,s+r}$$

where $preference(\_,\_)$ is a structural predicate that represents the preference relation between two reference objects. $key1(\_)$ and $key2(\_)$ are the key predicates and $p_i(\_,\_)$, $i = 1,\ldots,s+r$, is either a structural or a property predicate.

Preference relational patterns discovered in CORA satisfy the linkedness property, which means that each task-relevant object in a relational pattern P defined as in Definition 3.1 must be transitively linked to the reference objects t'_0 or t''_0 by means of structural predicates.

EXAMPLE 3.1. *Let us consider the database schema S reported in figure 3 where the table PREFERENCE is the preference relation and BLOCK is the target relation. An example of preference relational pattern is:*

$$preference(X,Y), block1(X), block2(Y), to\_right(X,Z), block\_x\_pos\_centre(Y, [435.1,\ldots,478.0])$$

where $preference(\_,\_)$ and $to\_right(\_,\_)$ are structural predicates, $block\_x\_pos\_centre(\_,\_)$ is a property predicate and $block1(\_)$ and $block2(\_)$ are the key predicates.

The *support* of a preference relational pattern P on the preference relation, denoted as $supp_{\prec}(P)$, is the percentage of tuples in $TrDB.PT$ "covered" (i.e., logically entailed) by P. Indeed, it is also possible to compute the support of the same pattern on the complement $\overline{TrDB.PT}$ of the preference relation. The complement is computed as the set of pairs of distinct reference objects that are not present in $TrDB.PT$. In this case, the support, denoted as $supp_{\not\prec}(P)$ is the percentage of the tuples in $\overline{TrDB.PT}$ "covered" by P.

P is frequent if $supp_{\prec}(P) \geq minSup$ or $supp_{\not\prec}(P) \geq minSup$ where $minSup$ is a user-defined threshold.

The *growth rates* [7] of a preference relational pattern P, denoted as $GR_{\prec}(P)$ and $GR_{\not\prec}(P)$, represent the *discriminative power* of P in identifying pairs of reference objects which appear or do not appear in the preference relation:

$$GR_{\prec}(P) = \frac{supp_{\prec}(P)}{supp_{\not\prec}(P)}; \quad GR_{\not\prec}(P) = \frac{supp_{\not\prec}(P)}{supp_{\prec}(P)} \qquad (5)$$

As in [7], we assume that $GR(P) = \frac{0}{0} = 0$ and $GR(P) = \frac{>0}{0} = \infty$. P is discriminative if $GR_{\prec}(P) \geq minGR$ or $GR_{\not\prec}(P) \geq minGR$ where $minGR$ is a user threshold.

CORA discovers frequent and discriminative preference relational patterns by exploring level-by-level the lattice of preference relational patterns ordered according to a generality relation ($\geqslant$) between patterns. This generality order is based on θ-subsumption and is monotonic with respect to support. The search proceeds in a Set Enumerated tree (SE-tree) search framework [21], starting from the most general pattern (the one with only the preference predicate and two key predicates), and iteratively alternating the candidate generation and candidate evaluation [17]. The SE-tree search framework has several advantages. First, the SE-tree enumerates all possible preference relational patterns by allowing a complete search. Second, it prevents the generation and evaluation of candidates which are equivalent under θ-subsumption. Third, it effectively exploits the monotonicity property of $\geq$ to prune the search space.

A node of the SE-tree is associated with a progressive natural index and it is represented by the *head* and the *tail*. The head of the root is the preference relational pattern that contains only the preference predicate and two key predicates. The tail is the ordered set of atoms which may be appended to the head by the downward refinement operator ρ.

DEFINITION 3.2 (DOWNWARD REFINEMENT OPERATOR). *Let P be a preference relational pattern. Then $\rho(P) = \{P \cup \{p(\ldots)\} | p$ is either a structural or a property predicate that shares at least one argument with one of the atoms in $P\}$.*

Let $n[head, tail]$ be a node of the SE-tree and $q(\ldots)$ be an atom in $tail(n)$. Then n has a child $n_q[head, tail]$ whose head is defined as follows:

$$head(n_q) = head(n) \cup q(\ldots). \qquad (6)$$

If q is based on a property predicate, its tail is defined as:

$$tail(n_q) = \Pi_{>q} tail(n) \qquad (7)$$

where $\Pi_{>q} tail(n)$ is the order set of atoms stored after q in $tail(n)$. Differently, if q is based on a structural predicate, its tail is defined as follows:

$$tail(n_q) = \Pi_{>q} tail(n) \cup \{r(\ldots)\} \qquad (8)$$

where $\{r(\ldots)\}$ is a set of atoms $r(\ldots)$. Each $r(\ldots)$ is an atom that belongs to one of the refinement $\rho(head(n_q))$ under the conditions that $r(\ldots)$ shares variables with $q(\ldots)$ and $r(\ldots)$ is not included in $tail(n)$. When $r(\ldots)$ is based on a structural predicate, one of its arguments must be a new variable. Heads of the nodes represent the discovered preference relational patterns.

The monotonicity property of $\geq$ with respect to support makes the expansion infrequent patterns useless. In addition, we prevent the expansion of nodes at a depth greater than $MaxD$. A further pruning criterion is based on the growth rate of patterns. This criterion is applied when $P \geq Q$ and $supp_{\prec}(P) > supp_{\not\prec}(P) = 0$. Due to monotonicity of support $supp_{\not\prec}(Q) = 0$ and $GR_{\not\prec}(Q) = 0$. If $supp_{\prec}(Q) = 0$, the node which enumerates Q is pruned due to the fact that Q is infrequent. Otherwise if $supp_{\prec}(Q) \neq 0$, then $GR_{\prec}(P) \to \infty \wedge GR_{\prec}(Q) \to \infty$. In this case, the node which enumerates Q in the head is pruned since Q has the same discriminating ability of P with respect to $GR_{\prec}(.)$ and CORA prefers simpler patterns to more complex patterns under the same growth rate. Analogously, when $P \geq Q$ and $supp_{\not\prec}(P) > supp_{\prec}(P) = 0$, the node Q is pruned.

4. PROBABILITY ESTIMATION

Once the set $\Re$ of preference relational patterns is extracted from $TrDB$, it is used in order to compute the likelihood in (1) for each pair (a, b) of reference objects stored in the target table of the testing database $TeDB.T$.

We denote as $\Re'$ the set of relational patterns defined as:

$$\Re' = \{P | preference(A, B), P \in \Re\} \quad (9)$$

This means that patterns in $\Re'$ do not have the $preference(\_,\_)$ atom. $\Re'$ is used to compute the likelihood in (1) as follows:

$$P(a, b| \prec) = P(\bigwedge_{R_k \in \Re(a,b)} R_k | \prec) \quad (10)$$

where $\Re(a, b)$ is the subset of $\Re'$ that cover the pair (a, b).

The straightforward application of the naïve Bayes independence assumption to all atoms in $\bigwedge_{R_k \in \Re(a,b)} R_k$ is not correct, since it may lead to underestimate the probabilities for the case that the pair (a, b) is covered by several patterns in $\Re'$. For instance, suppose that $\Re' = \{P_1, P_2\}$ such that:

$P1 = block1(X), block2(Y), to\_right(X, Z),$
$block\_x\_pos\_centre(Y, [435.1, \ldots, 478.0])$
$P2 = block1(U), block2(V), on\_top(U, W),$
$block\_x\_pos\_centre(V, [435.1, \ldots, 478.0])$

where the variables X, Y, U and V represent the reference objects. The simple application of the naïve Bayes independence assumption would produce this factorization:

$P(a, b| \prec) = P(P_1 \wedge P_2| \prec) =$
$P(block1(X)| \prec) \times P(block2(Y)| \prec) \times P(to\_right(X, Z)| \prec)$
$\times P(block\_x\_pos\_centre(Y, [435.1, \ldots, 478.0]| \prec)$
$\times P(block1(U)| \prec) \times P(block2(V)| \prec) \times P(on\_top(U, W)| \prec)$
$\times P(block\_x\_pos\_centre(V, [435.1, \ldots, 478.0]| \prec) =$
$P(block1(X)| \prec)^2 \times P(block2(Y)| \prec)^2$
$\times P(to\_right(X, Z)| \prec)$
$\times P(block\_x\_pos\_centre(Y, [435.1, \ldots, 478.0]| \prec)^2$
$\times P(on\_top(X, W)| \prec)$

since the $block1(\_,\_)$ atoms, the $block2(\_,\_)$ atoms and the $block\_x\_pos\_centre(\_, [435.1, \ldots, 478.0])$ atoms can be unified according to the substitution $\theta = \{X \leftarrow U, Y \leftarrow V\}$. Therefore there is a quadratic contribution of some probabilities and if one of them is small, $P(a, b| \prec)$ will approach zero. To prevent this problem we adapt the clause factorization [20] to the notion of relational pattern.

DEFINITION 4.1. *Let P be a relational pattern, which has a non-empty subset $Q \subseteq P$ of unifiable atoms with most general unifier (*mgu*) θ. Then $P\theta$ is called a* factor *of P.*

A factor of a pattern P is obtained by applying a substitution θ to P which unifies one or more atoms in P, and then deleting all but one copy of these unified atoms. In our context, we are interested in particular factors, namely those that are obtained by substitutions θ which satisfy three conditions: *i)* $Domain(\theta) = \bigcup_{R_k \in \Re(a,b)} Vars(R_k)$, that is, the domain of θ includes all variables in the pattern $R_k \in \Re(a, b)$; *ii)* $Domain(\theta) \cap Range(\theta) = \oslash$, that is, θ renames all variables in the pattern $R_k \in \Re(a, b)$ with new variable names; *iii)* $\theta_{|Vars(R_k)}$ is injective, that is, the restriction of θ on the variables in R_k is injective.

In the previous example, $\theta = \{A \leftarrow X, B \leftarrow Y, A \leftarrow U, B \leftarrow V, C \leftarrow Z, D \leftarrow W\}$ satisfies all these conditions, therefore, the factor of interest is:

$block1(A), block2(B), to\_right(A, C),$
$block\_x\_pos\_centre(B, [435.1, \ldots, 478.0]), on\_top(A, D)$

For each pattern P, a factor always exists. In the trivial case, it coincides with P up to a renomination of variables in P. A factor $P\theta$ is minimal, when there are no other factors of P with less literals than $P\theta$. >From a logic point of view, $\bigwedge_{R_k \in \Re(a,b)} R_k$ is equivalent to one of its factors since only redundant atoms are removed in the factorization. However, working with factors permits to avoid that the probability will approach zero. For this reason, for any minimal factor F of $\bigwedge_{R_k \in \Re(a,b)} R_k$, we compute $P(a, b| \prec)$ as $P(F| \prec)$ in (10).

By separating in F the contribution of the conjunctions of atoms corresponding to structural predicates ($str(F) = \{rel_{i_1}(A, B), \ldots, rel_{i_s}(A_s, B_s)\}$) from the contribution of the conjunction of atoms corresponding to property predicates ($prp(F) = \{attr_{i_1}(A, B), \ldots, attr_{i_t}(A_t, B_t)\}$) we have:

$$P(a, b| \prec) = P(str(F)| \prec) \cdot P(prp(F)|str(F) \wedge \prec) \quad (11)$$

Under the naïve Bayes independence assumption:

$$P(str(F)| \prec) = P(rel_{i_1}(A_1, B_1)| \prec) \cdot \ldots \cdot P(rel_{i_s}(A_s, B_s)| \prec) \quad (12)$$

where $P(rel_{i_j}(A_j, B_j)| \prec)$ is the relative frequency that two objects, denoted as A_j and B_j respectively, are related in $TrDB$ by the foreign key constraint associated to the structural predicate rel_{i_j} given the event $\prec$.

The naïve Bayes conditional independence is also assumed to compute $P(prp(F)|str(F) \wedge \prec)$ as follows:

$$P(prp(F) \mid str(F) \wedge \prec) = P(attr_{i_1}(A_1, v_{i_1})|str(F) \wedge \prec) \cdot \ldots \cdot P(attr_{i_t}(A_t, v_{i_t})|str(F) \wedge \prec) \quad (13)$$

where $P(attr_{i_j}(A_j, v_{i_j})|str(F) \wedge \prec)$ is computed on the training set $TrDB$ as the relative frequency that the attribute A_j assumes the value v_{i_j} in $TrDB$ given $str(F)$ and $\prec$.

The probabilities in the Equations (12-13) are apriori computed on the training database and are stored in the probability base $\wp$. This means that both the pattern base $\Re$ and the probability base $\wp$ are the output of the training phase. For each testing database $TeDB$, the probability estimation phase is then in charge of computing the probability $P(\prec |a, b)$ for each pair of testing reference object (a, b). This probability is computed as reported in Equation (1). In particular, the likelihood $P(a, b| \prec)$ is obtained by combining the probabilities stored in $\wp$ for the patterns falling in $\Re(a, b)$ according to the schema reported in Equation (12).

5. RANKING

The ranking algorithm in CORA allows us to identify a total order of reference objects stored in the testing database. The ranking algorithm follows the proposal in [13]. The basic idea is that of using a directed graph, where nodes represent reference objects to be ranked and edges express the preference relation between them, and to iteratively evaluating the most promising object to be appended to the resulting rank. Let $G = \langle V, E \rangle$ be a *weighted* directed graph where: $V = \{b \in TeDB.T\}$ is the node set and $E = \{(a, b, w_{a,b}) \in V^2 \times [0, 1] | w_{a,b}$ is the set of weighted edges where weights $w_{a,b}$ are the probabilities $P(\prec |a, b)$ computed according to (1).

We denote as $SUMPREF_G : V \rightarrow [0, \#V]$ the function:

$$SUMPREF_G(a) = \sum_{b \in V, b \neq a} w_{a,b} \qquad (14)$$

that expresses the *degree of preference* of an object a.

The rationale of the ranking identification is that a testing reference object is iteratively added to the final ranking. Such object is that for which $SUMPREF_G(\_)$ is the highest. Higher values of $SUMPREF_G(\_)$ are given to objects which have a high sum of probabilities to precede others.

6. EXPERIMENTS

We evaluate CORA in scientific literature mining and, in particular, in its application to scientific document image understanding. Document image understanding denotes the recognition of semantically relevant components and relations in the layout extracted from a document image. This recognition process is based on domain-specific knowledge that can be acquired automatically by applying data mining techniques. In particular, we focus our attention on two specific tasks: reading order detection [3] and document summarization [1] applied to scientific document images.

6.1 Reading order detection

Reading order detection is the task that permits to determine the reading order for layout components extracted from a document image. This is a crucial problem for several applications since it 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.

We argue that the spatial dimension of a page layout makes a multi-relational data mining approach the most suitable candidate for this specific task.

Reference objects correspond to descriptions of the layout components extracted from document images and are described according to the database schema provided in Figure 3 (where the target table is *blocks*). Properties or attributes of layout components are:

•*Locational*: (*x_pos_centre*, *y_pos_centre*): position of the centroid of the layout component.

•*Geometrical*: *height* (*width*): the height (width) in pixels of a layout component.

•*Logical*: "logical label" associated to a layout component.

•*Topological*: *on_top*: a layout component is on top/above another layout component. *to_right*: a layout component is to the right of another layout component. *alignment*: defines the type of vertical (col) or horizontal (row) alignment between two layout components. Possible alignments are: right_col, left_col, middle_col, both_columns, middle_row, lower_row,upper_row, both_rows.

•*Content type*: *type_of*: content type of a layout component. Possible values are: {image, text, horizontal line, vertical line, graphic, mixed}.

•*Page position*: Position of the page in the document. Possible values are: {first, intermediate, last_but_one, last}.

For the experiments we considered 24 papers, published as either regular or short articles, in the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) in two issues of 1996. Each paper is a multi-page document, therefore, we processed 211 document images. Initially, document images were pre-processed by WISDOM++[3] in order to segment them, perform layout analysis, identify the membership class and identify the logical label of a layout component. In all, 206 reading orderings were manually specified and 1,629 layout components were involved in such orderings. Possible logical labels for each layout component, in this class of documents, are: {*abstract*, *affiliation*, *author*, *biography*, *formulae*, *index_term*, *reference*, *section_title*, *paragraph*, *subsection_title*, *title*, *caption*, *figure*, *table*, *page_no*, *running_head*}. In this work, reading ordering is identified only on {*abstract*, *affiliation*, *author*, *biography*, *formulae*, *index_term*, *reference*, *section_title*, *paragraph*, *subsection_title*, *title*}. Remaining components are not considered to be relevant for the reading order.

We evaluated the performance of the proposed approach by means of a 6-fold cross-validation, that is, the dataset of 24 documents was divided into six *folds* and then, for every fold, training is performed on the remaining folds, while testing is performed on the current fold. Parameters of training are $minSup = 0.1$, $minGR = 1.5$ and $MaxD = 4$.

For each fold, statistics on precision and recall were recorded. Such measures refer to the $\prec$ relation. To evaluate these measures, we considered a reference object a to precede another object b when $P(\prec |a, b) > P(\prec |b, a)$. This permits us to evaluate decision capabilities of probabilities computed by CORA. To *globally* evaluate CORA, we resorted to metrics used in information retrieval for the evaluation of the returned rankings [9]. In particular, we considered the *normalized Spearman footrule distance* which, given two complete lists L and L_1 on a set S (L and L_1 are two different permutations without repetition of all the elements in S), is defined as $F(L, L_1) = 2/|S|^2 \sum_{b \in S} abs(pos(L, b) - pos(L_1, b))$ where the $pos(L, b)$ returns the position of the element b in L.

Results reported in Table 1 permit to compare CORA with the multi-relational approach proposed in [16] that is applied to the same dataset with equivalent representation. It is noteworthy that although the proposed approach shows comparable results in terms of precision to those obtained in [16], results in terms of recall are significantly in favour of the present approach. This can be explained by the high degree of adaptivity to noise of probabilistic approaches.

Experimental results concerning the reconstruction of the ranking (reading order) are reported in Table 2. We recall that lower the Spearman distance value, the better the reconstruction of the original ranking. Also in this case, CORA outperforms competitors. In particular, since the algorithm proposed in [3] is probabilistic and, as in our case, exploits the naive Bayesian learner, it is possible to say that the multi-relational approach is beneficial since it permits

[3]http:/www.di.uniba.it/%7Emalerba/wisdom++/

Concept	$\prec$ relation		$\prec$ relation in [16]	
	Precision %	Recall%	Precision %	Recall%
FOLD1	76.32	81.44	76.90	64.10
FOLD2	77.24	79.19	74.10	65.20
FOLD3	81.69	83.29	81.00	66.10
FOLD4	77.97	87.63	67.80	56.30
FOLD5	77.26	84.75	78.40	68.70
FOLD6	80.46	85.38	79.40	62.90
AVG	78.49%	83.61%	76.27%	63.88%

Table 1: Precision and Recall results.

Algorithm	Average	Standard deviation
CORA	0.180	0.03
In [16]	0.491	0.03
In [3]	0.240	0.07

Table 2: CV Results: normalized Spearman footrule distance.

to capture the spatial dimension of the document layout.

>From a qualitative point of view, 6,229 preference relational patterns have been automatically extracted in average for each learning task. Two examples of patterns with high growth rate are reported in the following:

$preference(X,Y), block1(X), block2(Y), to\_right(X,Z),$
$block\_x\_pos\_centre(Y, [435.1, \ldots, 478.0]).$
$(supp_{\prec} : 0.1 \quad GR_{\prec} : +\infty)$

This pattern states that a block X that appears to the left of another block (Z) is preferred to a block Y that is approximately located close to the right margin of the document page. The second example of pattern is:

$preference(X,Y), block1(X), block2(Y),$
$only\_lower\_row(X,Z), block\_author(X, false),$
$block\_biography(X, false), block\_section\_title(X, false).$
$(supp_{\prec} : 0.12 \quad GR_{\prec} : 6.5)$

This pattern includes information on the logical label associated to a block and states that a block X (which represents neither the biography block nor an author block nor a section title block) which is aligned on the bottom margin to another block (Z) is preferred to a block Y.

Preference patterns with lower growth rate are less interesting from a qualitative point of view, but are still useful for computing probabilities.

6.2 Document summarization

Document summarization refers to the task of reducing a text document into a short set of sentences that conveys the main meaning of the text. A strategy widely investigated for document summarization is that of selecting the more salient sentences of a document to be included in the summary [1]. Several proposed approaches are based on ranking algorithms that rank sentences according to their properties and choose the best M to be included in the summary [23].

Reference objects correspond to descriptions of sentences extracted from document images. The representation of the sentences is obtained through natural language processing techniques such as tokenization, sentence splitting, part-of-speech (POS) tagging, stop-word removing and stemming. The execution in sequence of these techniques permits to represent sentences in terms of the following features:

• ADJ_POS_FREQ, V_F_POS_FREQ, NOUN_POS_FREQ that express the percentage of the words of POS categories (adjectives,verbal forms and nouns) included in the sentence w.r.t. the total set of words in the same sentence.

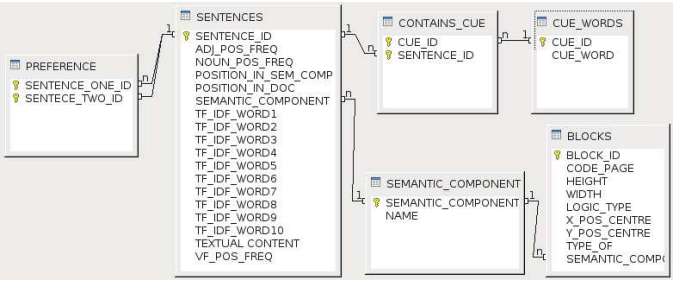

Figure 3: Logical view of the summarization DB.

• TF_IDF_WORD1, ..., TF_IDF_WORDN that denote the presence in the sentence of the N words having highest $tf - idf$ [22] values over the training corpus.

• POSITION_IN_DOC, POSITION_IN_SEM_COMP that represent the normalized position of the sentence in the document and in the semantic component.

In addition, we also consider the presence of indicator phrases (CUE_WORDs), used in discourse analysis, that give information about the discourse structure[19].

The database schema is reported in Figure 3 which includes the tables to describe the semantic components (table SEMANTIC_COMPONENTS), layout components (table BLOCKS) as well as the preference table (PREFERENCE). Where semantic components are components at a higher level of abstraction composed by several logic components possibly belonging to different document pages (e.g., motivations and experiments of a scientific paper).

Layout components are described according to features that are classified as:

• *Locational*: *x_pos_centre* (*y_pos_centre*): position of the centroid of the logical component w.r.t. the x (y) axis.

• *Geometrical*: *height* (*width*): the height (width) in pixels of a logical component.

• *Logical*: "logical label" associated to a logical component.

• *Content type*: *type_of*: content type of a logical component. Possible values are: {image, text, horizontal line, vertical line, graphic, mixed}.

CORA is used in the domain of document image understanding in order to generate summaries in terms of phrases contained in the semantic components. The corpus of training documents is the same used in Section 6.1. Documents are processed in order to perform layout analysis and identify logical and semantic components. Admissible semantic components are *abstract, method, motivations* and *experimental results*. The relevant semantic components used for summarization in this work are *method* and *motivations*.

Since the PREFERENCE table is not populated in each training database, we populate it according to the value of cosine similarity computed between sentences wa_j occurring in the *abstract* of the document and sentences w_k occurring in *method* and *motivations*:

$$sim(wa_j, w_k) = \frac{wa_j \cdot w_k}{||wa_j|| \cdot ||w_k||} \tag{15}$$

where each sentence (wa_j or w_k) is represented in form of a $tf - idf$ vector of N elements. The score $score(w_k) = max_j\ sim(wa_j, w_k)$ is used for ranking (and, then for defining tuples of the training preference relation).

Evaluation of automatically generated summaries on test-

	CORA	GREEDYexp	GREEDYunif	SVD	FURTHEST
fold #1	81.1	80.52	80.52	82.51	83.53
fold #2	77.17	76.50	80.29	77.64	75.25
fold #3	77.77	78.31	75.01	83.81	86.37
fold #4	85.77	77.88	77.75	75.53	79.25
fold #5	80.40	81.95	81.63	86.77	83.30
fold #6	87.84	81.69	81.41	78.21	86.87
Average	82.19	79.52	79.63	80.6	82.26

Table 3: CV Results: average cosine similarity between the abstract and the generated summaries.

ing documents is performed by means of a six fold cross validation. Obtained summaries have been compared with original abstracts and the cosine similarity between them has been recorded. This similarity is similar to the ROUGE-1 metric [1] typically used in document summarization. The main difference is that cosine similarity considers weights of terms in the document instead of their presence/absence. Parameters of the experiments are $minSup = 0.05$, $minGR = 1.1$, $MaxD = 3$, $N = 10$ and $M = 10$. Where M is the number of best ranked sentences to be included in the summary. Evaluation has been performed by comparing CORA with the algorithms of document summarization implemented in the ManyAspects system [15]. This comparison allows us to prove the applicability of CORA on this task.

Results are reported in Table 3 and show that summaries obtained by CORA are generally better than those obtained by other techniques. The only competitor is Furthest, whose performances, however, depend on a chosen seed [15].

7. CONCLUSIONS

In this paper we have motivated, presented and evaluated a probabilistic, relational algorithm for preference learning. The algorithm determines the probability that an object can be preferred to another. The probability is computed by extending the naive Bayes assumption to relational representations. We used estimated probabilities to rank a set of complex objects. In particular, we applied the algorithm to the domain of document image understanding for reading order detection and for document summarization. In both cases, experimental results prove the advantages of the proposed algorithm with respect to other algorithms reported in the literature. As future work, we intend to evaluate CORA with new document corpora with different layout.

Acknowledgment

This work is partial fulfillment of the research objective of ATENEO-2009 project "Estrazione, Rappresentazione e Analisi di Dati Complessi".

8. REFERENCES

[1] L. Antiqueira, O. N. Oliveira, L. da Fontoura Costa, and M. das Graças Volpe Nunes. A complex network approach to text summarization. *Inf. Sci.*, 179(5):584, 2009.

[2] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender. Learning to rank using gradient descent. In L. D. Raedt and S. Wrobel, editors, *ICML*, volume 119 of *ACM International Conference Proceeding Series*, pages 89–96. ACM, 2005.

[3] M. Ceci, M. Berardi, G. Porcelli, and D. Malerba. A data mining approach to reading order detection. In *ICDAR*, pages 924–928. IEEE, 2007.

[4] K. Crammer and Y. Singer. Pranking with ranking. In *NIPS*, pages 641–647. MIT Press, 2001.

[5] L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *Journal of Data Mining and Knowledge Discovery*, 3(1):7–36, 1999.

[6] O. Dekel, C. D. Manning, and Y. Singer. Log-linear models for label ranking. In *NIPS*. MIT Press, 2003.

[7] G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *KDD*, pages 43–52. ACM Press, 1999.

[8] S. Džeroski and N. Lavrač. *Relational Data Mining*. Springer, 2001.

[9] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW '01*, pages 613–622. ACM Press, 2001.

[10] Y. Freund, R. D. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[11] R. Herbrich, T. Graepel, and K. Obermayer. *Large margin rank boundaries for ordinal regression*. MIT Press, 2000.

[12] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. Label ranking by learning pairwise preferences. *Artif. Intell.*, 172(16-17):1897–1916, 2008.

[13] T. Kamishima and S. Akaho. Learning from order examples. In *ICDM*, pages 645–648. IEEE, 2002.

[14] K. Kersting and Z. Xu. Learning preferences with hidden common cause relations. In *ECML PKDD 09*, LNAI. Springer, 2009.

[15] K. Liu, E. Terzi, and T. Grandison. Manyaspects: a system for highlighting diverse concepts in documents. *PVLDB*, 1(2):1444–1447, 2008.

[16] D. Malerba and M. Ceci. Learning to order: A relational approach. In *MCD*, volume 4944 of *LNCS*, pages 209–223, 2008.

[17] H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.

[18] G. Nagy. Twenty years of document image analysis in PAMI. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):38–62, 2000.

[19] C. D. Paice. Constructing literature abstracts by computer: Techniques and prospects. *Inf. Process. Manage.*, 26(1):171–186, 1990.

[20] J. A. Robinson. A machine oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.

[21] R. Rymon. An SE-tree based characterization of the induction problem. In *ICML*, pages 268–275. Morgan Kaufmann, 1993.

[22] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[23] C. Wang, F. Jing, L. Zhang, and H.-J. Zhang. Learning query-biased web page summarization. In *CIKM*, pages 555–562. ACM, 2007.