On discovering user‘s needs in the ontology-based portals using implicit relevance feedback

Nenad Stojanovic
Institute AIFB, University of Karlsruhe, Germany
nst@aifb.uni-karlsruhe.de

Abstract. In this paper we present a novel approach for discovering preferences of a user who searches for information in an ontology-based portal. The approach is based on analysing the user’s searching behaviour regarding the given ontology and the content of the underlying information repository. In that way we develop so called short-term user’s profile, that can be used for recommending relevant information to the user. The approach is realized as a method for ranking query refinements in the so-called step-by-step query refinement process. It is very suitable for modelling interactive searching in digital libraries or product catalogues.

1. Introduction

By introducing an ontology as the conceptual model of the domain instantiated in an information portal, some substantial improvements in querying can be achieved, especially regarding the precision of the retrieval process [GMV99]. Moreover, the ontology can be used as the conceptual backbone for the query refinement and the ranking process. In our previous work we propose the so-called step-by-step Query refinement [St03], as a method in which a user can tailor an (ontology-based) query to his information need incrementally. He is provided with a list of possible refinements so that he can choose the most relevant one. In this paper we propose a method for discovering preferences of a user in the absence of his explicit feedback information. Indeed, by analysing a user’s behaviour during the searching it is possible to get some implicit indicators about what the user is currently searching for. Moreover, these indicators can be combined with the given ontology and the underlying repository in order to tailor them to the content of the portal. Therefore, the so-called implicit relevance feedback can be used for developing the user’s profile for his current searching.

2. Background

The research we present in this paper relies on the Librarian Agent, a system for the collection- and query- management of an ontology-based information portal. In this section we present basic information about the structure of that system, which is needed for understanding the rest of the paper. For more details see [St03].

Definition1: Annotated Knowledge Repository *aKR* is the structure *(O, M, ann)*, where:

- *O* is a set the elements of which are called information objects

- *M* is a set the elements of which are called metadata. Metadata can be organized in the vocabulary *V* (see Definition 2). In that case we say that the repository is annotated with the vocabulary *V*;

- *ann* is a binary relation between a set of objects and a set of metadata, $ann \subseteq O \times M$.

We write *ann(o,m)*, meaning that object *o* has the metadata *m*.

Definition2: A vocabulary on a set *M* of metadata is a structure *V:= (M, H)*, where:

- *M* is a set of the metadata (terms);

- *H* is a set of ordered pairs on *M*, defining a partial order on *M*.

For the relations from *H* the following equation holds:

$\forall o \in O \forall m_1, m_2 \in M \wedge \forall h \in H, ann(o,m_1) \wedge h(m_1,m_2) \rightarrow ann(o,m_2)$

Definition3: A query-answering pair in a repository *aKR=(O, M, ann)*, which is annotated using the vocabulary *V=(M, H)*, is a tuple $Q = (M_x, O_y)$, where:

- $M_x \subseteq M$, M_x is called a set of query_terms;

- $O_y \subseteq O$, O_y is called a set of query_constraints. $O_y = \{o \in O | (\forall m \in M_x) \rightarrow ann(o,m)\}$.

In order to simplify the notation, instead of the term *query-answering pair* we will use in the rest of the text only the term *query*. The set of all queries *Q* is denoted by *Q(aKR)*. We define two relations on this set: structural equivalence and structural subsumption.

Definition 4: Structural equivalence (=) between queries by:

$(M_{x1}, O_{y1}) = (M_{x2}, O_{y2}) \leftrightarrow O_{y1} = O_{y2}$, which can be written as $Q_1 = Q_2 \leftrightarrow O_{y1} = O_{y2}$ (1)

Two queries are structurally equivalent if their result sets are the same. Note that this relation is reflexive, symmetric and transitive.

Definition 5: Structural subsumption (parent-child) between queries (<) by:

$(M_{x1}, O_{y1}) < (M_{x2}, O_{y2}) \leftrightarrow O_{y1} \subset O_{y2}$. (2)

A query (M_{x2}, O_{y2}) subsumes another query (M_{x1}, O_{y1}) if the result set of Q_2 subsumes the results of the first query. Note that this relation is irreflexive, anti-symmetric and transitive. We define two special subsumption relations on the set Q:

- *DirectParents*: Q→Q*, as $Q_1 <_{dir} Q_2 \leftrightarrow Q_1 < Q_2 \wedge \neg\exists Q_i, Q_1 < Q_i < Q_2$, (3)
  then we call Q_2 direct_parent of the Q_1;

-*DirectChildern*: Q→Q*, as $Q_1 >_{dir} Q_2 \leftrightarrow Q_2 < Q_1 \wedge \neg\exists Q_i, Q_2 < Q_i < Q_1$, (4)
  then we call Q_2 direct_child of the Q_1.

Definition 6: Query Neighbourhood (Map) of a query Q consists of its equivalent-, direct_parent and direct_child queries.

3. Ranking of Query Refinements

As we mentioned in the introduction, in the searching for information a user is often provided with a large list of refinements, so that a method for ranking these refinements Is needed in order to make that process more efficient. Obviously, these ranking has to be aligned to the preferences of the current user. However, a problem arises from the fact that users are usually reluctant to give an explicit information about their preferences in searching. Therefore, these preferences have to be discovered in a process that looks over a user's shoulder and "intelligently" interprets his behaviour. In that way the system takes so-called implicit relevance feedback of the user as a criterion for determining the relevance of query refinements. Moreover, since the structure of the portal is ontology-based, the implicit relevance indicators should be combined with the information about the content information in order to make more relevant suggestions for the refinement. In the rest of the text we present several such parameters.

3.1 The relevance parameters

By determining the relevance of a refinement for the given user's query we take into account the characteristics of (i) the used vocabulary, (ii) underlying repository and (iii) the searching process.

3.1.1The characteristics of the used vocabulary

Clarity. The clarity of the interpretation of a constraint is defined regarding the properties of the used vocabulary. A constraint can be interpreted in various ways depending on the context of its appearance. For example, if in an ontology the concept "Car" is modelled through three subconcepts "SportsCar", "FamilyCar" and "MiniCar", then the query for a car, i.e. $\forall$ `x` $\leftarrow$ `Car(x)` can be (mis)interpreted as a user's need to a (i) Sport-, (ii) Family- or (iii) Mini-car. Formally, the clarity of a constraint *c* is calculated as: $Clarity(c,O) = k * \frac{1}{|\{A_x \mid H(A_x,c)\}|}$, (5)

where *k* is a parameter that reflects the homogeneity of that hierarchy. If *Clarity* of an attribute is low, then its meaning can be further refined and vice versa.

3.1.2 The characteristic of the underlying repository

Specificity. The traditional approach in IR to select a term to be added to a user's query in the query refinement process, is based on comparing the effects that expansion terms could have on the results [Ef95]. We adapt this experience for the case of the searching in an ontology-based repository by introducing the parameter *Specificity* as follows:

$Specificity(c,Q) = \frac{|Q|}{|Q_{xc}|}$, where Q is the query that corresponds to the user's request and Q_{xc} is a direct_child cluster that contains constraint *c*. |a| depicts the number of objects contained in the set a.

3.1.3 The characteristics of the searching process

Since the goal of a refinement process is to satisfy a user's need, in order to determine the relevance of a refinement we have to take into account not only the last query a user made, but rather the whole process of creating a query. We define two types of such an implicit relevance feedback [SG90]: *Actuality* and *ImplicitRelevance*.

Actuality. The *Actuality* parameter reflects the phenomena, accounted in the IR research that a user may change the criteria about the relevance of a query term, when encountering newly retrieved results. In other words, the constraints most recently introduced in a user's query are more indicative of what the user currently finds relevant for his need. We model it using the analogy to the ostensive relevance proposed in [Ca00]: $Actuality(c,S_q) = \frac{1}{num\_session(c,S_q)+1}$, where c is a query_constraint , S_q is the current query session and $num\_session(c, S_q)$ is the number of refinement steps, which the constraint c is involved in.

ImplicitRelevance. If a user selects a resource from the list of retrieved results, then this resource corresponds, to some extent, to the user's information need. For example, it is possible that, for a query, the user selected for viewing several results that have a common characteristic. Obviously, such a constraint should be proposed as a very relevant query refinement. We call such a constraint *preferred constraint*.

We model such a relevance: $ImplicitRelevance(c,S_q) = \{ \begin{matrix} 0, & c \notin Preferred(S_q) \\ 1, & c \in Preferred(S_q) \end{matrix}$, where $Preferred(S_q)$ is the set of preferred constraints in the query session S_q. The task of finding this set is out of scope of this paper.

3.2 The ranking function

We transform the search space into a transition network, allowing the use of Markov chains theory. The transition between states are defined as follows: if constraints x and y are connected regarding the ontology structure, then they are connected in the transition network. We assume for each transaction e a probability $q_e > 0$ of occurring. In a transition network, for each state a: $\sum_{b:a\rightarrow b} q_{a\rightarrow b} = 1$. The transition matrix T is defined as $T(x,y) = \{ \begin{matrix} q_s & if \quad x \in Related(y) \\ 0 & otherwise \end{matrix}$, where q_s is uniformly distributed over all related constraints and $Related(c)$ is a function that retrieves constraints that are related to c regarding the underlying ontology. For example, `Car` and `Motorbike` are related to each other through the superconcept `Vehicle`. Further, $T'(x,y)$ is the probability of reaching y starting from x in i transitions. $T^0 = I$, the identity matrix. Thus, the sum $\sum_{i=0}^{\infty} T^i(e,d)$ is the probability of reaching d from e in any number of steps. Next we focus on the probability $Pr(d \mid e)$ of a constraint d being the search target of a navigation path. The destination probability for d after traversing a path from e is defined as follows: $Pr(d \mid e) = \sum_{i=0}^{\infty} T^i(e,d)$. The infinite sum converges to $(I-T)^{-1}(e,d)$ (see [Tri82]).

However, in a step-by-step refinement a user is navigating through queries that contain several constraints. In that case we expand the probability to include this information as follows: $Pr(Q_d \mid Q_e) = \frac{1}{|Q_d|} \cdot \sum_{d \in Q_d} \frac{1}{|Q_e|} \cdot \sum_{e \in Q_e} Pr(d \mid e)$, where d and e are constraints from Q_d and Q_e respectively.| Q_d | depicts the number of constraints in Q_d .

Finally, the relevance is formalized by assigning the coefficient *Rank* to each query Q_d that belongs to the lattice of the refinement for the query Q_e:

$$Rank(Q_d \mid Q_e) = \frac{1}{|Q_d|} \cdot \sum_{d \in Q_d} \frac{1}{|Q_e|} \cdot$$

$$\left(\sum_{e \in M_{xe}} Pr(d \mid e) \cdot \left(\frac{\lambda \cdot Implicit\,Relevance(d) \cdot Actuality(e) + Specificty(e, Q_d) \cdot Clarity(e)}{\lambda + 1} \right) \right)$$ where λ is a forgetfulness coefficient that model the impact on the past user's behaviour on the ranking process: $\lambda = 0$ - the past is forgotten, $\lambda < 1$ - the past carries less weight then the present, usually $\lambda = 1/2$.

Therefore, our approach prioritises highly relevant refinements, i.e. the refinements that are related to the very characteristic (regarding a query) constraints and tailored to the user's need.

4. Conclusion

In this paper we present an approach for ranking query refinements in an ontology-based searching for information. The approach is based on analysing the user's searching behaviour regarding the given ontology and the content of the underlying information repository. In that way we develop so called short-term user's profile, that can be used for recommending relevant information to the user. The approach is realized as a method for ranking query refinements in a step-by-step query refinement process, which enables a user to tailor his query to his information need incrementaly.

References

[Ca00] Campbell, I. The ostensive model of developing information needs, Ph.D. Thesis, University of Glasgow, 2000

[Ef95] Efthimiadis, E.N. User choices: A new yardstick for the evaluation of ranking algorithms for interactive query expansion. Information Processing and Management, 31(4), 605-620, 1995.

[GMV99] Guarino, N., Masolo, C. and Vetere, G. OntoSeek: Content-Based Access to the Web, *IEEE Intelligent Systems*, 14(3), pp. 70-80, May 1999

[SB90] Salton G., Buckley C. Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science. 41(4): 288-297, 1990.

[St03] Stojanovic, N. An Approach for Using Query Ambiguity for Query Refinement: The Librarian Agent Approach, 22nd International Conference on Conceptual Modeling (ER 2003), Chicago, Illinois, USA, Springer

[Tri82] Trividi K.S. Probability and Statistics with Reliability, Queuing and Computer Science Applications. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.