Explicit Passive Analysis in Electronic Catalogs

David Portabella Clotet, Martin Rajman
Artificial Intelligence Laboratory (LIA)
Swiss Federal Institute of Technology (EPFL)

Technical Report IC/1140601063-20467
February 22, 2006

Abstract

We consider example-critiquing systems that help people search for their most preferred item in a large catalog. We compare 6 existing approaches in terms of user or system-centric, implicit or explicit use of preferences, assumptions used and their behavior in underconstrained and overconstrained situations.
We consider several types of explicit passive analysis to guide the users in their search, that is, information offered to the user about his current search but without any action taken by the system. We suggest that such a user-centric system together with the right analysis makes the users feel more confident in their decision and reduces session time and cognitive effort.
We have implemented a prototype to evaluate the impact of explicit passive analysis in (1) a query-building and (2) a preference-based approach.

Introduction

A Multi-Attribute Decision Problem (MADP) is the problem of finding the best outcome or solution based on the user's preferences. However users may not have an accurate idea of their preferences while searching for a product, especially when the product is beyond the user's knowledge of its domain. They may begin the search with some vague set of preferences and refine them as they learn more about the different possibilities [1].

An example-critiquing interaction is an iterative process where the user and the system collaborate to find the best solution. Based on the current user's preference model, the system shows a set of candidate solutions and the user gives some feedback to the system so that it can update the preference model. The loop continues until the user is convinced with one of the candidate solutions. The type of feedback varies from system to system: it can be a direct manipulation of the preference model, such as "a maximum price of 400 Euros", or a more vague feedback such as "a cheaper one (than the presented candidate)".

This iterative process is not only used to refine a preference model, but also to avoid expressing a complex preference one. Imagine for example that the user is interested in an apartment in the center for less than 800 Euros or otherwise an apartment in the suburbs for 600 Euros maximum and close to a bus stop. Instead of entering this complex preference model, he would probably search first for one of the cases entering in an iterative process to refine the preference model and switch afterwards to the other case and repeat the process, to finally take the best option between the two. In this example we cannot say the user was from the beginning to the end refining his preference model, but rather that he studied 2 different types of cases separately.

Example-Critiquing systems

One of the first systems to implement such an approach was the RABBIT system [2], where the preference model is a query explicitly given by the user. The candidates shown to the user are simply the list of all items satisfying the query. The query can then be directly reformulated by using options such as "prohibit" or "specialize" over an attribute. When the solution space is overconstrained (i.e., there is not any solution satisfying the query), the system simply does not show any candidate solution. In contrast, in a preference-based version of the previous system, the user could define a weight to each constraint and this information would be used to provide a list of ranked partial matches. The system is user-centric in the sense that it is the user that guides the interaction and the system just provides the requested information to the user.

The SmartClient [3] interface implements a similar approach to the previous one, but it allows updating the query by adding or changing a constraint for individual or combinations of attributes (e.g., "airport='San Francisco' if departure time > 10am"). Although the user is not allowed to set weights to the constraints, in an overconstrained situation the system displays the k-best partial matches assuming a default equal weight for all of them.

In FindMe [4] there is not a direct mapping between the feedback provided by the user and the preference model constructed by the system. The system limits the type of possible user feedback to a set of options such as "this

Compilation copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

apartment is OK, but make it… bigger" based on one of the candidate solutions and thus, it is system-centric. The advantage of this approach is that the cognitive effort to provide feedback is lower as it is vaguer. As the interaction goes on, the system builds an implicit preference model using the feedback. In an overconstrained situation, the system executes a pre-defined domain-dependent ordered list of operations to relax the preference model. A problem with this approach is that eventually the user may not understand the justification for the proposed candidates as the preference-model built by the system is completely hidden and thus he may feel frustrated.

The incremental-critiquing approach proposed by McCarthy at [5] shows just 1 candidate solution and a set of compound critiques that describe the feature relationship that exist between the remaining cases. The user has the option of directly updating the query by adding or changing a constraint over an attribute, or by accepting one the proposed compound critiques. The system filters the 3 best compound critiques by taking the ones with lower support values, as they have would eliminate many items from consideration if chosen. It has the advantage that if one of the compound critiques turns out to be interesting to the user, selecting it would probably require less cognitive effort than manually inspecting all the matching items, extracting and applying the individual critiques one by one. The system incrementally builds an implicit model of the user, but there is no reason why the compound critiques could not be used also in a query building tool as in Rabbit.

One important feature in ATA [6] and further elaborated in [7] is that the system not only presents the k best ranked items but also a set of alternative suggestions which are significantly different and have the potential to stimulate the expressions of hidden preferences and thus avoiding local optimum solutions. For instance, if the user has not expressed a preference for non-stop flights, presenting the alternative solution that best satisfies the current preferences plus the latter one could potentially induce the user to realize that this new preference is important to him.

Assumption	**Description**
A1	Preferences are additive independent [9]
A2	Fixed type of preference (e.g. the given price by the user is always an upper bound)
A3	Use of parameterized penalty functions that are used to build a numerically precise user preference model
A4	Which attributes are crucial when comparing two items (in AptDecision)
A5	A pre-defined domain-dependent set of retrieval strategies (in FindMe), such as "on 'cheaper' option selected, if the are not such apartments, decrease the 'niceness' or 'neighborhood' constraint"

Table 1. A list of assumptions used by some interfaces.

In Apt Decision [8] the system builds an implicit preference model taking into account the current critique and most of the past critiques. Past critiques are not displayed.

In tables 1 to 4 we summarize the features of the reviewed approaches.

	User-centric	**Use explicit constraints**	**Assumptions**
Rabbit	Yes	Yes	None
SmartClient	Yes	Yes	None
FindMe	No	No	A5
ATA	Yes	Yes	A1
AptDecision	Yes	Yes	A1, A2, A3, A4
Incremental-critiquing	No	No	A1

Table 2. The assumption refer to the list provided at table 1

	In an underconstrained situation, the system shows:
Rabbit	All matching items
SmartClient	Some k matching items
FindMe	The k best ranked items
ATA	The k1 best ranked items (excluding dominated ones) + k2 significantly different items
AptDecision	The k best ranked items
Incremental-critiquing	The best candidate + k best compound critiques

Table 3. Behavior in underconstrained situations

	In an overconstrained situations, the system:
Rabbit	Shows 0 items
SmartClient	Shows partial matching items & allows chronological retraction of constraints
FindMe	The system has a domain-dependent pre-defined ordered list of operations to retrieve items
ATA	Same behavior as in the underconstrained situation
AptDecision	Same behavior as in the underconstrained situation
Incremental-critiquing	Not specified

Table 4. Behavior in overconstrained situations

Explicit Passive Analysis

There are 2 types of approaches revised from the 6 cases presented above: (1) the system-centric, which try to build an implicit preference model or somehow limit the type of user feedback and the (2) user-centric, in where the user can directly update his query or preference model and the system just helps him by providing some kind of passive analysis.

Nonetheless, in the reviewed systems except in the case of incremental-critiquing, the passive analysis is provided in the form of alternative solutions. We suggest that providing the user with the explicit justification that can potentially lead to these alternative solutions will make them feel more confident in their decision process and speed up the interaction. For instance, in the ATA interface the system could inform the user that given the current preference model, he may potentially be interested in adding a preference about non-stop flights.

As users add constraints incrementally, eventually they will end up in an overconstrained situation. More importantly than providing partial matching items, we propose to explicitly provide the user give the information about the minimal conflicting set of constraints. For instance, in a catalog of second-hand cars, given the current constraints stated by the user, the system could inform that “there are not Audi cars & Cheaper than 6000 Euros & Matriculated later than year 2000” nor “cars Cheaper than 6000 Euros & with Blue color & Electric windows”. Rather than just providing partial matching solutions, this information can potentially help the user to characterize the solution space.

A second type of explicit passive analysis is to provide concise relaxations propositions to overcome the minimal conflicting set. While the information about the minimal conflicting set may be useful to characterize the solution space and thus get more confidence, finding which constraints to relax to overcome the situation is not evident and it may involve trying several relaxing combinations. For instance, in the previous example the system would also inform that there are cars “if he relaxes the maximum price to 6500 Euros” or “if he relaxes the maximum price to 6200 Euros and without Electric windows”. Providing such a list of possible relaxations may deal to greatly speed up the interaction.

Pilot Study

We have implemented a prototype to evaluate the impact of explicit passive analysis in (1) a query-building and (2) a preference-based approach. We have used 4 databases: (1) a list of apartments provided by the University of UNIL at Lausanne, Switzerland, (2) a list of movies from the imdb[1], (3) a list of optional courses at EPFL and (4) a list of 7000 second-hand cars provided by Comparis[2]. We present at table 5 with the number of items and attributes for each database.

We have tuned the prototype to compute the minimal conflicting set for combinations of maximum 4 constraints. The justification is that it takes too much cognitive effort to interpret a conflicting set with large number of constraints.

How to decide the appropriate number for this maximum deserves further investigation. In the same spirit, we have limited the maximum number of attributes in a relaxation proposition to 3.

Database	Source	No. items	No. attributes
Apartments	Unil	186	16
Movies	imdb.com	2897	7-13
Optional courses	EPFL	50	11
Cars	comparis.ch	7924	25-35

Table 5. Databases used

Although the pilot study involved only 3 users, the general feeling is that in both models, query-building and preference-based, the use of this explicit passive analysis helps the user up to some extent to speed up the interaction and to better characterize the solution space and thus provides confidence and trust in the final decision.

Conclusions

We have compared several existing example-critiquing systems for electronic catalogs in terms of user or system-centric, implicit or explicit use of preferences, assumptions used and their behavior in underconstrained and overconstrained situations.
Based on this analysis, we suggest that providing analysis in explicit form rather than hidden in proposed alternative suggestions may greatly help the user to better characterize the solution space, feel more confident about the final decision and speed up the interaction.
A prototype and a small pilot study have been realized to evaluate our approach with positive expectations, and we are currently preparing a complete user study with 30 live users.

[1] The Internet Movies Database, http://www.imdb.com
[2] http://www.comparis.ch

References

1. J.W. Payne, J.R. Bettman, and E.j. Johnson. The Adaptive Decision Maker. Cambridge University Press, 1993.
2. M.D. Williams. What Makes RABBIT Run? International Journal of Man-Machine Studies, 21:333-352, 1984.
3. B. Faltings, P. Pu, M. Torrens, and P. Viappiani. Designing Example-Critiquing Interaction. In Proceedings of the International Conference on Intelligent User Interfaces (IRI-2004), pages 22-29. ACM Press, 2004. Funchal, Madeira, Portugal.
4. R. Burke, K. Hammond, and B. Young. Knowledge-based Navigation of Complex Information Spaces. In Proceedings of the Thirteen National Conference on Artificial Intelligence, pages 462-468. AAAI Press/MIT Press, 1996. Portland, OR.
5. K. McCarthy, J. Reilly, L. McGinty, and B. Smyth. An analysis of Critique Diversity in Case-Based Recommendation. In proceedings of the Eighteenth International FLAIS Conference (FLAIRS-05). 2005. Clearwater Beach, FL, USA.
6. G. Linden, S. Hanks, and N. Lesh. Interactive assessment of user preference models: Te Automated Travel Assistant. In C. Paris A. Jameson and C. Tasso, editors. User Modeling: Proceedings of the Sixth International Conference, pges 67-78.Springer Wien, 1997.
7. Boi Faltings, Pearl Pu, Marc Torrens, Paolo Viappiani: Designing example-critiquing interaction. Intelligent User Interfaces 2004: 22-29
8. S. Sherin and H. Lieberman. Intelligent Profiling by Example. In Proceedings of the International Conference on Intelligent User Interfaces (IUI 2001), pages 145-152. ACM Press, 2001. Santa Fe, NM, USA.
9. Keeney, R., and Raiffa, H. Decisions With Multiple Objectives: Preferences and Value Tradeoofs. Wiley. 1976