Selecting and Applying Recommendation Technology

Maryam Ramezani[a], Lawrence Bergman[b], Rich Thompson[b], Robin Burke[a], Bamshad Mobasher[a]

[a] DePaul University School of Computer Science, Telecommunications and Information Systems
243 S. Wabash Avenue, Chicago IL 60604
{mramezani,mobasher,rburke}@cti.depaul.edu

[b] IBM T.J Watson Research Center
30 Saw Mill River Rd, Hawthorne, NY 10532
{bergmanl,richt2}@us.ibm.com

ABSTRACT
This paper presents a taxonomy of recommender systems with the goal of assisting in selection and application of these systems. Recommendation methods are usually classified into three main categories: collaborative,content-based, and knowledge-based. We outline a taxonomy of recommender systems based on problem characteristics and the underlying technology. We show how the taxonomy can help researchers and developers select between different kinds of recommender systems by clearly defining the problem characteristics including: problem structure, domain, relationship with the user, user input, background knowledge, and recommendation outputs.

Author Keywords
Recommender Systems, Taxonomy

1. Introduction

With the rapid development of recommender system technology in recent years, it has become difficult for developers to determine which technology is suitable to a particular context. To alleviate this difficulty, we propose a framework that organizes the space of recommendation problems and assists in systematically finding an appropriate recommendation technology for addressing a given problem. We propose six dimensions for organizing the space of recommendation problems: problem structure, domain, user relationship, user input, background knowledge, and recommendation output. We look at three dimensions of recommendation technology: algorithm, user interaction, and user profiling. Finally, we map the problem characteristics to suggested recommendation technologies.

This paper differs from previous similar works in several key ways. First, we consider knowledge-based recommenders as an important part of our taxonomy, a type previously given little attention. Second, the taxonomy is not only a classification of recommendation systems but also a recommender for selecting among recommender technologies based on problem characteristics.

The rest of the paper provides a brief description of different recommendation technologies, followed by a description of problem characteristics with discussion of how to map these characteristics to the technologies.

2. Recommendation Technologies

We examine recommendation technologies along three dimensions; recommendation algorithms, user interactions, and user models.

2.1. Recommendation Algorithms

Collaborative Filtering (CF) Recommenders use social knowledge - typically ratings of items by a community of users – to generate recommendations. Recommenders can be further subdivided into *Memory-based collaborative filtering*, which construct a set of neighbors at runtime and use the neighbors' preferences to generate a recommendation, and *Model-based collaborative filtering*, which learn an aggregated model of user behavior in advance and use this model to generate recommendations.

Collaborative filtering algorithms have the advantage of using only historical data; no knowledge of the items is required. However, they suffer from a "cold-start" problem; a new user cannot receive any recommendations before rating several items and a new item cannot be recommended before being rated by a number of users.

Content-based (CB) Recommenders use item features to recommend items similar to those in which the user has expressed interest. CB has no cold start problem, but is unable to provide the serendipitous recommendations that CF generates. We call content-based systems which use persistent user profiles (described in section 2.3) *learning content-based,* because they learn user interests based on a long-term profile. Content-based systems that use ephemeral user profiles (usually in the form of queries) are typically called *search engines*.

Knowledge-based (KB) Recommenders use domain knowledge to generate recommendations. We discuss three types of KB systems. **Case-based Reasoning (CBR)** systems reuse information and knowledge from previously-encountered situations [1]. One variant of a CBR system is conversational CBR (CCBR). In a CCBR system, a query describing a target problem is incrementally elicited in an

interactive dialog [3]. **Constraint-based Reasoning** systems provide recommendations based on a set of constraints, gathered from the user. **Rule-Based** systems provide recommendations to users using explicit rules to map between the user's need and the available items.

All knowledge based approaches avoid the cold start problem and have the advantage of enhanced reliability as the background knowledge is free of noise. However, knowledge-based systems require considerable knowledge acquisition effort for setup and maintenance during their lifetime [50], which makes them more expensive to develop and maintain. Thus, a knowledge-base approach is not the best option for problems that can be adequately solved with other approaches.

Hybrid Recommender systems combine two or more techniques to gain better results with fewer drawbacks.

.2.2. User Interaction

We classify the forms of user interaction within a recommender systems into conversational and single-shot.

Conversational Recommenders participate in an interactive dialog with the user by asking the user to give feedback or to answer questions. **Question/Answer** systems ask a set of questions of the user and use the responses to formulate a recommendation. **Candidate/Critique** systems display a basic set of recommendations to a user and solicit feedback.

In **Single-shot Recommenders**, each user interaction is used to produce a recommendation independently, with no additional user feedback.

Conversational recommenders have been widely used in knowledge-based systems while collaborative filtering and content-based systems are typically single-shot.

2.3. User Models

A *user model* (or *user profile*) is a set of characteristics that implicitly or explicitly capture user preferences.

Persistent User Models contain user's interests and preferences deduced (using statistical and machine learning techniques) from user inputs accumulated over time. **Ephemeral User Models** contain the user's current interest (or intention), based solely on input from the current user session.

Static User Profiles accumulate information about the user over time, for example, name and address, or shipment history. **Dynamic User Profiles** contain only user information supplied in the current session, for example, the set of items just purchased.

3. Mapping Problem Characteristics to Recommendation Technologies

We introduce six dimensions to characterize and organize recommendation problems. Some of the problem characteristics are fundamental, based on the business model of the system. For example, the question, 'Who are the target users of the system?' should be answered by business strategists.

Problem structure, domain, and user relationship are *fundamental problem characteristics*. When these are clearly defined, the *secondary problem characteristics* - user input, background knowledge, and recommendation output - can be derived or decided by system developers.

For each dimension, we will detail aspects of that dimension, and the appropriate technology choices (algorithm, user interaction, and user model) for each – choices providing the best result with minimum effort and cost. It is important to note that many of the characteristics are not independent of each other and the technology should be determined by considering multiple characteristics.

3.1. Problem Structure

The first step in developing a recommender system is to define the goal of the system. Why is a recommendation needed and what type of problem is it intended to solve? We classify problem structures into four main categories:

(1) Selection problems: The problem is to select from many options where there is a fixed universe of possibilities, but too many for the user to readily scan and compare, for example, selecting a digital camera from the very large set of those commercially available. Most recommender systems help people select from many available options. All of the algorithms described in section 2 are applicable to selection problems, so other problem characteristics should be taken into account.

(2) Configuration problems: The problem is to determine the "best" ways to combine sets of components. One example is building a PC from off-the-shelf components. Configuration problems are typically complex, since there are combinatorially many possible configurations, and the system has to know about interactions between components.

The most appropriate approach to solve large configuration problems is a conversational constraint-based technique; applications are described in [5], [12], [13], and [14]. An example of applying other algorithms, namely structured CBR, to a configuration problem is presented in [4] (selecting life insurance policies by using generalized cases). CBR tends to work less well for configuration problems, since it is difficult to accumulate enough cases to cover all configuration situations. A constraint-based approach, by reasoning on individual logical statements rather than whole cases, allows for easy construction of configuration rules. Rule based systems can also be useful

for small-sized configuration problems.

(3) Planning Problems: The problem is to determine a sequence of activities for accomplishing a particular goal. Examples include developing a military evacuation plan or a travel planning tool. These types of problems can be very complex depending on the domain; each problem might have a large number of parameters, such as the kinds of transport being used, the route taken, time of day, etc.

Due to the complexity, these problems require both domain knowledge and a conversation with the user to produce a plan. Therefore, CCBR is a good choice for solving such problems. A conversational constraint-based approach can also appropriate for planning problems, although it is usually difficult to formally represent all of the constraints that are involved in a real-world planning task.

(4) Exploration Problems: The problem is to support the user in seeking something interesting, where there is no specific goal. For example, a person looking for a birthday present may not have a specific item in mind.

These kinds of problems typically require conducting a conversation with the user, and receiving user feedback on the recommendation; Candidate/Critique conversational systems are the preferred solution. Conversational collaborative recommenders could be appropriate for these kinds of problems, but more research is needed.

3.2. Domain

Factors that characterize the domain space are:

(1) Size and heterogeneity of solution space: The number and type of items that potentially can be recommended must be considered. For example, an e-commerce recommender system such as Amazon.com has a large number of heterogeneous items.

The larger the solution space, the more difficult it is to encode a knowledge base. Thus, for very large solution spaces, knowledge-base approaches should be avoided unless it is necessary for other reasons (for example, for critical domains) If the items to be recommended are homogeneous, content-based or user-based collaborative filtering can be used. However, for heterogeneous items, item correlations are more appropriate because user opinions may not be consistent across item types.

(2) Life span: The period of time during which a recommendation is valid may be important. Ephemeral media, such as netnews, is of interest for only a short period, so the problem of adding new items should be considered in designing a system. Using a knowledge-based approach is difficult for ephemeral data because the cost of updating the knowledge base is spread over only a short usage period. Collaborative filtering is not a good option because of the new item problem. Thus, content-based recommenders are preferred for ephemeral data.

(3) Criticality: The cost of a wrong recommendation must be considered. An incorrect medical diagnosis is much more costly than an inappropriate movie recommendation.

In critical domains, a knowledge-base approach is needed, as a correct and explainable recommendation is impossible with other approaches.

(4) Importance and duration of user preference: The importance of user preference in creating recommendations is domain dependent. For example, in the medical domain, user preference is insignificant in formulating diagnoses, while in the movie domain, user preference is the most important factor. The duration of user preference can also vary. For example, a person buying a digital camera would typically not be interested in camera recommendations after a purchase, while a person interested in comedy movies likely prefers comedy movie recommendations over a long period of time.

Collaborative filtering systems are the best for recommending serendipitous items that match user preferences. Content-based systems, which find items based on similarity to user interest, are appropriate in cases where user preference is consistent and of long duration.

(5) Type of end user: End-users may be domain experts who use the system in their profession, or ordinary users getting recommendations about everyday items.

Professional users usually need accurate, noise–free, and explainable recommendations, which force the use of knowledge-based approaches. Collaborative and content-based systems are typically appropriate for ordinary users.

In conversational systems, the type of end-user determines the appropriate type of conversation. Ordinary users typically do not like being asked direct and specialized questions during the recommendation process [30]. Furthermore, unless users have a good understanding of the recommendation domain, they may not be in a position to answer a direct feature question reliably. So, the best approach for ordinary users is Candidate/Critique.

3.3. User Relationship

The type of relationship the user has with the recommender system is key to determining the type of user interaction and user profiling technologies to be selected. Based on those choices, the type of background knowledge for the recommender system can then be selected. To describe the user relationship we consider two aspects:

(1) Length of relationship: Systems may have long-term or short-term relationships with users. Netflix has a business model that promotes a long-term relationship with customers. On the other hand, some recommender systems act like an automated consultant in a short-term relationship; for example, a laptop or digital camera recommender.

In systems that maintain long-term relationships with the user, the system will have access to persistent user profiles, which may be dynamic or static. If the system has deep and rich information about the user, knowledge-based system can be used. Rule-based systems with demographic inputs are common in this case. However, if the knowledge about the user is only opinions such as ratings, collaborative filtering [16, 18] and learning content-based systems [28, 36] are appropriate. In systems that have short-term relationships with the user, ephemeral user profiling based on current user inputs is used. In situations where the user does not wish to spend time on a conversation with the system, only current user behavior or a simple query is available as a user profile, resulting in shallow knowledge about user interest. If the user input is a query, the best approach is a search engine. If user input consists of only current behavior or transactions, collaborative filtering with ephemeral profiling is appropriate.

On the other hand, if the user is willing to conduct a conversation with the system to receive recommendations, a rich ephemeral user profile can be built by using conversational techniques. A knowledge-based system such as CCBR or a constraint-based system is preferred in this case. The effort users are willing to expend to receive a recommendation is mostly related to the domain. Users will typically spend more effort to receive an accurate recommendation in domains that are critical or have a high level of expertise than in general domains such as a movie selection.

(2) Depth of relationship: Systems have shallow or deep relationships with users. The shallowest relationship is anonymous access, such as that found in web usage recommenders, which only know about user's clickstreams and navigation. Systems that have a deep relationship are characterized by a rich knowledge about users, such as customer accounts, transactions, and demographics.

3.4. User Input

User input describes the information the user must provide to receive recommendations. Some inputs are required to produce anything other than non-personal recommend-ations. Aspects that characterize user input are:

(1) Degree of user effort: The level of user effort or involvement to supply inputs to the system can vary widely.

Implicit inputs or preferences are inferred from the user's behavior without the user's awareness, and require no specific effort of the user. For example, Amazon.com uses the particular item that a user is browsing as an implicit input to recommend additional items. By contrast, *explicit inputs or preferences* are intentionally supplied by users to request recommendations that fit their interests.

E-commerce web sites frequently use implicit inputs. These include the specific items the user is currently viewing, user transaction history, mouse clicks, mouse movement, scrolling and elapsed time. Implicit inputs are usually applicable in CF [34, 35] and CB systems [23, 15].

We note that implicit inputs are naturally noisy because they are inferred from user behavior and are thus not suitable for critical or expert domains which need explainable recommendations. Although it is possible to use implicit inputs in a knowledge-based system, we note that it might not be worth the cost and effort of designing a knowledge-based system when using noisy implicit inputs.

(2) Input type: Implicit input types include the *user behaviors* (enumerated in (1)) and *user transactions* (purchase history, request for information, etc). Explicit input types include opinions (ratings, reviews, tags), demographic information (region, age, sex, etc.) and requirements (queries, constraints, priorities, and contexts). Although most of these types are well-understood, the following discussion clarifies the types of requirements. A query is a request from the user, such as a natural language request, spoken dialog, key words, parameters chosen from a menu, or an example of a similar item. Constraints are restrictions and limitations that the recommender system must take into account, such as looking only for German language movies. A priority is a soft constraint, something preferred but not required in a solution, such a priority for a three bedroom apartment that might be satisfied with a low-priced two bedroom. The user's context consists of the external circumstances associated with the recommendation, such as the user's location when recommending restaurants.

User opinions are widely used in CF systems. User rating is the typical input in CF [18, 24, 41] and learning CB approaches [28, 36, 38]. Social tagging web sites often use tags as user input for recommendations. For example, in del.icio.us users can enter a tag to search for URLs that have been tagged with that tag. Using tags as input for recommender systems is widely applicable in CF systems and is an active research area [46].

Demographic information can be used to form people-to-people correlations , and thus are useful for collaborative recommendation. Krulwich [27] uses demographic groups from marketing research to suggest a range of products and services. In other systems, machine learning is used to train a classifier based on demographic data [39]. Rule-based systems can make recommendations based on demographic inputs. When the rules are extracted from social data (e.g., association rules), we would consider the system a collaborative system, since the background knowledge is social knowledge. When the rules are business rules designed by domain experts, the system would be considered a knowledge-based system. Both kinds of rules could be combined in a hybrid system.

User requirements are applicable in CB and KB systems.

User query is the basic input in search engine recommenders.

CBR systems use different types of user query. For example, Adaptive Place Advisor [11] uses spoken dialog and [8] uses free-text entry of problem descriptions. User priority is usually used as feedback in search engines and KB systems. A survey of preference elicitation methods is presented in [9]. In most recommender systems, more than one type of user input is used. For example, [32] uses users' priorities and constraints in CBR recommender. The use of context in recommender systems is an area of active research [2, 6, 26, 40, 45, 48].

3.5. Background Knowledge

Background knowledge is the information that the system accumulates before the recommendation process begins. We describe three aspects of background knowledge

3.5.1. Social Knowledge

Social knowledge is extracted from community inputs using data mining and pattern recognition techniques. Social knowledge extracted from user inputs is the background knowledge in collaborative recommenders. Thus, if social data is available, CF can be chosen as the recommendation algorithm. Different aspects of social data to be considered in designing collaborative recommenders include:

(1) Data type: The types of data can be any of those discussed in the previous user input section.

Structured user inputs such as ratings are the most straightforward type of data used to model social knowledge. In this case, data is an m × n matrix of ratings where m is number of users and n is the number of items. Modeling of semi-structured user opinions such as tags is an active research area [7, 29, 37, 46, 47]. In this case, the data is a tripartite graph of user, item and tag. Using unstructured user opinions such as reviews to produce background knowledge is a more complicated task. How to model this kind of data is an open research topic.

(2) Data volume: Data volume is determined by the number of users and the size of each user's profile. In cases where the number of users is very large, model-based algorithms must be used because it would be very time consuming and computationally expensive to find neighbors of the target user on the fly.

(3) Data distribution: Some commercial recommender systems have very large item sets that can be evaluated by users. In practice, even active users rate only a small fraction of the available items, resulting in very sparse data. The sparsity of a data set is sometimes measured by the average percentage of items rated per user. However, since popular items may be rated by many users, the data will be dense in some places and sparse in others. Distribution properties of the data are the most important factor in selecting an algorithm for CF systems.

The relationship between the number of users and number of items determines whether to build correlations among users or among items. In cases where the number of items is much larger than the number of users, model-based algorithms should be used because data sparsity makes finding neighbors difficult. For example, the large number of items on Amazon.com makes it difficult to find users who have rated or purchased more than a limited number of items.

The rating distribution of items and users also may affect both algorithm and evaluation metric choices. Systems with dense subregions of agreement can use that agreement to recommend in the sparse subregions [21]. Systems with an even ratings distribution may be more challenged to cope with sparsity unless they use dimensionality reduction techniques such as SVD [43].

(4) Data dimensionality: Most available data sets contain two dimensions, users and items, with a single implicit or explicit rating that associates them. However, both research and commercial systems are exploring schemes that allow users to enter several ratings for a single item [20], such as separate ratings for the story, acting, and special effects of a movie. Timestamps can add an additional dimension to data, and are particularly important when user tastes are ephemeral.

In some cases, the data may have more than two dimensions (the two typical dimensions are users and items). Such data can be handled using a multi-dimensional (MD) approach [2], or by mapping the data into multiple sets of two dimensions each. For example, a tripartite data set of user, item and tags can be mapped to three bipartite models: user-item, user-tag, and tag-item.

3.5.2. Content Knowledge

Sources of content knowledge include text documents, metadata, and databases. They are usually acquired by information Retrieval (IR) and Information Extraction (IE) techniques. Content knowledge is usually composed of item profiles - a flat set of attributes or features characterizing each item. This kind of knowledge does not rely on deep structured relationships between items.

Content knowledge is the background data for CB algorithms. Aspects characterizing content knowledge are:

(1) Data type: Data types include unstructured, semi-structured or structured documents; multimedia; and item attributes. Different types require the use of different techniques for extracting features for CB algorithms. Unstructured documents containing free text (e.g., news stories), are represented as bags of words. Item features can

be Boolean (a word either occurs or does not occur in a document), or frequency-based (frequency of the word in a document). The feature volume can be reduced by applying feature selection techniques, such as information gain, mutual information, cross entropy, or odds ratio [33]. Different approaches to using IR and IE techniques to extract content knowledge from unstructured and semi-structured documents are discussed in [25]. If the data type is multimedia, it is more difficult to extract the content knowledge. Multimedia techniques are discussed in [17, 49].

(2) Data quality: Items may not be directly comparable due to factors such as inconsistent naming schemes. For example, manufacturers of similar products may use different labels for the features of their product.

(3) Data availability: The ease of accessing and gathering content data is a determining factor in choosing the kind of background knowledge for recommendation. Can we gather the content data by crawling HTML web pages or do we need additional resources? For example, music content-based systems like [22] or Pandora.com need to use music information filtering/retrieval systems to extract music features; this is more compute-intensive than text retrieval. In cases where it is difficult to extract quality content knowledge, other approaches such as CF must be used.

3.5.3 Domain knowledge

Domain knowledge is any general knowledge used for recommending, as opposed to the association of specific features with specific items. For example, that "Prairie Joe's" restaurant serves "American" food is a content feature of that restaurant, while "American" cuisine being similar to "Diner" cuisine and different from "Thai" cuisine is domain knowledge. Domain knowledge is typically acquired through interviews with experts or by knowledge discovery techniques.

(1) Data type: Domain knowledge can take many forms, but without loss of generality we can describe it as an ontology in which there are relations among attributes, objects, and concepts, especially related to item features. It may also contain means-ends knowledge about how items may meet potential user goals. The presence of domain knowledge may permit items to be described with structure representations as opposed to simple vectors of features.

A recommender system may also rely on knowledge that is external to the question of product suitability or "best match". The owners of a system may prefer certain recommendations for business reasons. For example, a video rental service may prefer to recommend items from its less-popular back catalog over in-demand new releases. Such knowledge we refer to as business rules knowledge.

(2) Knowledge availability: Domain knowledge must be gathered, verified, represented in appropriate form, and maintained over time. There may be significant cost to each of these steps.

If cold-start or criticality considerations render collaborative and content-based approaches ineffective, assembling background knowledge in the form of a domain ontology and using knowledge-based techniques is appropriate. Such knowledge-bases typically do not preexist. The effort involved in engineering a KB system can vary widely, depending on the domain and the recommendation context. Note that a KB recommender requires content features about which it can reason, so all of the considerations associated with acquiring content knowledge also apply to KB systems.

3.6. Recommendation Output

(1) Type of output: Recommendation output can be classified into *suggestions* and *predictions*. The most common type of output suggests one or a list of items. Recommender systems may also present users with predictions of item ratings, helping the user to understand the strength of a recommendation [44].

CF and learning CB systems are appropriate for producing personalized predictions, provided that the user has entered sufficient ratings in their profile. Cosley and colleagues [10] show that displaying predictions while users rate items affect their opinions.

KB systems are able to produce suggestions based on user requirements. If suggestions are the desired output, any kind of recommender system can be used, since prediction outputs can be treated as a ranked list of items, allowing suggestion of items with the highest predicted rating.

(2) Degree of explanation: Explaining how recommendations are produced may be an important component in establishing trust in a recommendation system. Explaining recommendations is simplest and more common in knowledge-based systems, in particular, CBR systems [31, 42]. Explaining recommendations is difficult in a collaborative system, since the output is based on a statistical model. The challenges are to extract meaningful explanations from computational models that are more ad hoc than knowledge-based systems, and to provide a usable interface to the explanations. Herlocker and colleagues [19] hypothesized that adding explanation interfaces to collaborative systems would increase their acceptance as filtering systems but were unable to prove or disprove the hypothesis. How to explain CF and to what degree that would be helpful is an open research question.

5. Conclusion

We have identified three dimensions for recommendation technologies: algorithms, user interaction, and user models or profiles. We have further identified dimensions that describe the space of decision problems: problem type,

domain, user relationship, user input, background knowledge and recommendation output. We have developed a taxonomy of recommender systems based on problem characteristics and mapped each dimension of the problem characteristics to dimensions in recommendation technology.

The approach described here suggests future developments that could assist researchers and developers in selecting components of recommendation technology. We envision development of automated or semi-automated tools to assist with the complexity of making technology choices, thereby pushing forward the concept of "recommending recommenders."

ACKNOWLEDGEMENTS

We wish to thank Ravi Konuru for his support and valuable input. We also wish to thank Markus Stolze for discussions that helped to focus and direct this work.

REFERENCES

1. Agnar Aamodt and Enric Plaza. Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Commun.*, 7(1):39–59, 1994.
2. Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans. Inf. Syst.*, 23(1):103–145, 2005.
3. David W. Aha, Len Breslow, and Hector Mu Avila. Conversational case-based reasoning. *Applied Intelligence*, 14(1):9–32, 2001.
4. Martin Schaaf, Alexander Tartakovski and Ralph Bergmann. Retrieval and configuration of life insurance policies. *ICCBR 2005*, pp. 552–565, 2005.
5. Liliana Ardissono, et al. A framework for the development of personalized, distributed web-based configuration systems. *AI Mag.*, 24(3):93–108, 2003.
6. Shlomo Berkovsky, et al. Providing context-aware personalization through cross-context reasoning of user modeling data. In S. Berkovsky, et al, editors, *UbiDeUM'2007 – Int. Workshop on Ubiquitous and Decentralized User Modeling, at User Modeling 2007, 11th Int. Conf., UM 2007, Corfu, Greece, June 26, 2007, Proceedings*, 2007.
7. S Veeramachaneni, C Hayes, P Avesani. An analysis of the use of tags in a blog recommender system. *IJCAI '07,* pp. 2772-2777, 2007.
8. Christina Carrick and Qiang Yang. Activating CBR systems through autonomous information gathering. In *ICCBR '99: Proc of the 3rd Int. Conf. on Case-Based Reasoning and Development*, pp. 74–88, London, UK, 1999. Springer- Verlag.
9. Li Chen and Pearl Pu. Survey of preference elicitation methods. Technical report, *Swiss Federal Institute of Technology in Lausanne (EPFL),* Lausanne, Switzerland, August 2004.
10. Dan Cosley, et al. Is seeing believing?: how recommender system interfaces affect users' opinions. In *CHI '03: Proc. of the SIGCHI Conf. on Human factors in computing systems*, pp. 585–592, New York, NY, USA, 2003. ACM Press.
11. Mehmet H. Gker, Cynthia A. Thompson and Pat Langley. Automatic personalization based on web usage mining. *Journal of Artificial Intelligence Research*, 21:393–428, 2004.
12. Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Stumptner. Consistency-based diagnosis of configuration knowledge bases. *Artif. Intell.*, 152(2):213–234, 2004.
13. Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Zanker. An integrated environment for the development of knowledge-based recommender applications. *Int. J. Electron. Commerce*, 11(2):11–34, 06-7.
14. Alexander Felfernig and A. Kiener. Knowledge-based interactive selling of financial services with FSAdvisor. 17th Innovative Applications of Artificial Intelligence Conference (IAAI05), pp. 1475–1482, 2005. AAAI Press.
15. V. Moustakis G. Potamias and M. van Someren. Using content-based filtering for recommendation. In *CML/MLNET Workshop on Machine Learning and the New Information Age, Barcelona*, pages 47–56, 2000.
16. Nathaniel Good, et al. Combining collaborative filtering with personal agents for better recommendations., *AAAI: Conference on Artificial Intelligence*, pp. 439–446, 1999.
17. Alexander G. Hauptmann. Integrating and using large databases of text, images, video, and audio. *Intelligent Systems and Their Applications, IEEE*, 14(5):34 – 35, 1999.
18. Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proc. of the 22nd annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 230–237, New York, NY, USA, 1999. ACM Press.
19. Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *CSCW '00: Proc. of the 2000 ACM Conf. on Comp. Supp. Coop. Work*, pp. 241–250, New York, NY, USA, 2000. ACM Press.
20. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems.

ACM Trans. Inf. Syst., 22(1):5–53, 2004.
21. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.
22. Yoshinori Hijikata, Kazuhiro Iwahama, and Shogo Nishida. Content-based music filtering system with editable user profile. In *SAC '06: Proc. of the 2006 ACM Symp. on Appl. computing*, pp. 1050–1057, New York, NY, 2006. ACM.
23. Chen Jian, Yin Jian, and Huang Jin. Automatic content-based recommendation in e-commerce. In *EEE '05: Proceedings of the 2005 IEEE International Conf. on e-Technology, e-Commerce and e-Service (EEE'05)*, pp. 748–753,Washington, DC, 2005. IEEE Comp. Soc.
24. Joseph A. Konstan, et al. Grouplens: applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, 1997.
25. Raymond Kosala and Hendrik Blockeel. Web mining research: a survey. *SIGKDD Explor. Newsl.*, 2(1):1–15, 2000.
26. Alexander I. Kovacs and Haruki Ueno. Recommending in context: A spreading activation model that is independent of the type of recommender system and its contents. In Gulden Uchyigit, editor, *Proc. of Workshop on Web Personalisation, Recommender Systems and Intelligent User Interfaces*, Dublin, June 20, 2006.
27. B. Krulwich. Lifestyle finder: Intelligent user profiling using large-scale demographic data. *Artificial Intelligence Magazine*, 18(2), 1997.
28. Ken Lang. Newsweeder: Learning to filter netnews. In *Proc. of the 12th Int. Conf. on Machine Learning*, pp. 331–339, 1995.
29. Ben Markines, Lubomira Stoilova, and Filippo Menczer. Bookmark hierarchies and collaborative recommendation. In *AAAI*. AAAI Press, 2006.
30. Lorraine McGinty and Barry Smyth. Comparison-based recommendation. In *ECCBR '02: Proc. of the 6th Euro. Conf. on Adv. in Case-Based Reasoning*, pp. 575–589. Springer- Verlag, 2002.
31. David McSherry. Explaining the pros and cons of conclusions in CBR. In *Adv in Case-Based Reasoning*, pp. 317–330. Springer Verlag, 2004.
32. David McSherry and David W. Aha. Mixed-initiative relaxation of constraints in critiquing dialogues. In *ICCBR*, pp. 107–121, 2007.
33. Dunja Mladenic and Marko Grobelnik. Feature selection for unbalanced class distribution and naive bayes. In *ICML '99: Proc. of the 16th Int, Conf. on Machine Learning*, pp. 258– 267, San Francisco, CA, 1999. Morgan Kaufmann Pub. Inc.
34. Bamshad Mobasher, Robert Cooley, and Jaideep Srivastava. Automatic personalization based on web usage mining. *Comm. of the ACM*, 43(8):142–151, 2000.
35. Bamshad Mobasher, Honghua Dai, Tao Luo, and Miki Nakagawa. Discovery and evaluation of aggregate usage profiles for web personalization. *Data Min. Knowl. Discov.*, 6(1):61–82, 2002.
36. Raymond J. Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *DL '00: Proc. of the 5th ACM Conf. on Digital Libraries*, pp. 195–204, New York, NY, USA, 2000. ACM Press.
37. S. Niwa, Takuo Doi, and S. Honiden. Web page recommender system based on folksonomy mining for ITNG 06. In 3rd Int. Conf. On Information Technology: New Generations. pp. 388–393, 2006.
38. Michael Pazzani and Daniel Billsus. Learning and revising user profiles: The identification of interesting web sites. *Mach. Learn.*, 27(3):313–331, 1997.
39. Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13(5-6):393–408, 1999.
40. C. Rack, S. Arbanowski, and S. Steglich. A generic multipurpose recommender system for contextual recommendations. *ISADS*, 00:445–450, 2007.
41. Paul Resnick, et al. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proc. of the 1994 ACM Conf. on Computer Supported Cooperative Work*, pp. 175–186, New York, NY, 1994. ACM Press.
42. Thomas R. Roth-Berghofer. Explanations and case-based reasoning: Foundational issues. In *Advances in Case-Based Reasoning*, pp. 389–403. Springer Verlag, 2004.
43. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems-a case study. In ACM WebKDD Workshop, 2000.
44. J. Ben Schafer, Joseph A. Konstan, and John Riedl. Ecommerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1-2):115–153, 2001.
45. Mark van Setten, Stanislav Pokraev, and Johan Koolwaaij. Context-aware recommendations in the mobile tourist application compass. In W. Nejdl and P. De Bra, editors, *Adaptive Hypermedia 2004*, pp. 235–244. Springer Verlag, 26-29 August 2004, Eindhoven, The Netherlands 2004.
46. Harris Wu, Mohammad Zubair, and Kurt Maly. Harvesting social knowledge from folksonomies. In *HYPERTEXT '06: Proceedings of the seventeenth conference on Hypertext and hypermedia*, pages 111–114, New York, NY, USA, 2006. ACM Press.
47. Zhichen Xu, Yun Fu, Jianchang Mao, and Difu Su.

Towards the semantic web: Collaborative tag suggestions. In *Proceedings of the Collaborative Web Tagging Workshop at the WWW 2006*, Edinburgh, Scotland, 2006.

48. Zhiwen Yu, Xingshe Zhou, Daqing Zhang, Chung-Yau Chin, Xiaohang Wang, and Ji Men. Supporting context-aware media recommendations for smart phones. *Pervasive Computing*, 5(3):68–75, 2006.
49. Osmar R. Za¨ıane, et al. Multimediaminer: a system prototype for multimedia data mining. In *SIGMOD '98: Proc. of the 1998 ACM SIGMOD Int. Conf. on Management of data*, pp. 581–583, New York, NY, USA, 1998. ACM.
50. Markus Zanker, Markus Jessenitschnig, Dietmar Jannach, and Sergiu Gordea. Comparing recommendation strategies in a commercial context. *IEEE Intelligent Systems*, 22(3):69–73, 2007.