Recommender Systems for the Semantic Web

Antonis Loizou [1] and **Srinandan Dasmahapatra** [2]

Abstract.
This paper presents a semantics-based approach to Recommender Systems (RS), to exploit available contextual information about both the items to be recommended and the recommendation process, in an attempt to overcome some of the shortcomings of traditional RS implementations. An ontology is used as a backbone to the system, while multiple web services are orchestrated to compose a suitable recommendation model, matching the current recommendation context at run-time. To achieve such dynamic behaviour the proposed system tackles the recommendation problem by applying existing RS techniques on three different levels: the selection of appropriate sets of features, recommendation model and recommendable items.

1 INTRODUCTION AND PROBLEM

While much work has been done in the Recommender System (RS) domain over the past decade [18] and though such systems have been deployed commercially, eg. [10], recent research in the area seems to have reached a standstill. Intuitively, one may suggest this is due to the fact that the recommendation problem has been solved, deeming further research into the area unnecessary. However, upon deeper inspection and by empirically evaluating such commercially deployed systems, it becomes evident that this is not the case [19].

In the majority of current RS implementations, items are recommended to users through some flavour of Collaborative Filtering (CF) [4, 16, 23], a method that assesses the similarity between user profiles to predict ratings for unseen items. Alternatively, in Content Based (CB) approaches, the items of possible interest are indexed in terms of a set of automatically derived descriptive features, and unseen items with similar attributes to those rated highly by the user are recommended. The two approaches are often combined in Hybrid RS to achieve improvements in the quality of recommendations [1, 3].

The shortcoming of such a method rests in its assumption that active users will respond positively to unseen items rated highly by similar users. As most users are not inclined to rate previously seen items, only a few items will receive ratings. This limited data – the 'cold start' problem [21] – renders similarity metrics not sensitive enough to distinguish between users, particularly new ones introduced to the system. Hence, the most highly rated items from anyone are recommended. Alternatively, in Content Based (CB) approaches, the items of possible interest are indexed in terms of a set of automatically derived descriptive features, and unseen items with similar attributes to those rated highly by the user are recommended. A drawback of the CB method is that it recommends items interchangeable to the ones rated highly by users, ignoring potential user requirements. The two approaches are often combined in Hybrid RS to achieve improvements in the quality of recommendations [1, 3].

These shortcomings reflect the lack of computational support for humans who are interested in items they, or the people who usually share their taste haven't previously come across. In addition, such systems do not allow for shifts of the user's interest over time, since all ratings provided by a user have an equal bearing on the recommendation selection. To clarify this point consider the following conceptualisation: A user X has provided high ratings only for items in some set A, however (s)he is now only interested in items from another set, B. A conventional RS will not be able to recommend items from set B until enough ratings are provided for items in B, in order for them to dominate in the clustering and selection processes. This means that a system shouldn't become stable, and that the classification of the same items to different classes, at different times, may be deemed correct, something that would be unacceptable in most machine learning contexts. To account for this requirement of time dependance on users' preference context, conventional architectures recompute their user clusters periodically, effectively choosing a different training set every time. This can aggravate problems caused by data sparsity, and important modelling decisions about transitions between user needs have to be addressed.

Furthermore, while it is apparent that an artifact's features have a bearing on whether it appears interesting or not, users may not be able to identify its desirable characteristics at the outset. For instance, someone who wants to buy a new car might only specify "I want a black car" to begin with. Instead of buying the first black car available, s/he might look at a variety of black cars and as their knowledge of cars grows in the process, discover other possible features of interest, or even come across an unusual opportunity and end up buying a different coloured car. This would suggest that for a RS to be successful, it needs to be able to identify which of an item's features may potentially be of interest to the user, against a variety of possible modes of generalisation.

To overcome such issues, a system should be able to consider the semantics of both the recommendation context and those of the items at hand to constrain the recommendation process. Information specific to the recommendation context for both user clustering and content-based comparisons have been shown to improve overall recommendation performance [3, 17, 21]. By incorporating relevant contextual information into a recommendation model, we enable the system to evaluate the appropriateness of a given recommendation based on some heuristics, for example the time of recommendation or the utility of the recommended item to the user [1, 6, 9]. The system proposed in this paper is one designed to choose appropriate input and output spaces dynamically, in a manner that will allow for real time recommending, matching the variable temporal and contextual recommendation requirements while still performing the bulk of the

[1] IAM, School of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK, email: al05r@ecs.soton.ac.uk
[2] SENSE, School of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK, email: sd@ecs.soton.ac.uk

computation off-line. The system architecture has been designed to allow this flexibility, which we describe next. We have experimented with various evaluation models to identify the sources of potential improvement in this dynamic architecture.

2 PROPOSED SYSTEM ARCHITECTURE

Assuming that the human selection process is best modelled through a dynamic function that operates on some subset of an artifact's attributes and other contextual variables, the proposed architecture employs a variety of Web Services (WS), each capable of performing a subroutine of the recommendation process. The recommendation process in this architecture is split into three distinct phases:

Knowledge acquisition: Knowledge acquisition is continuously carried out in the background by specialised WS, in order to import new recommendable items into the system. This consists of discovering new items from trusted sources, and acquiring (partial) translations of their descriptions to the system ontology, in a manner that allows feature vectors to be extracted from such descriptions in order to apply any available distance measures between newly acquired items and existing ones.

Recommendation subspace selection: The recommendation space is conceptualised as a multidimensional space where each unique feature in all items known is represented by a dimension. Our aim is to identify which 'slices' of this space contain items that can be recommended under the current recommendation context. To carry this out each unique feature will be assigned a weight based on how strong the relationship between items with the same feature value is, that can be determined through Ontology Network Analysis (ONA) [2, 14]. The recommendation context will be determined through the users' recent behaviour as logged by the system, inferred restrictions from a long-term observation of the users' preferences, additional restrictions provided explicitly by the user and global trends. As the user provides ratings for the recommended items the weights these factors receive will be adjusted to produce recommendations more likely to receive high ratings. Having split the dataset into clusters of similar items and users, the set of recommendable items is determined based on the current context. A suitable user cluster is then identified by selecting users with experience of the recommendable items but also with sufficent overlaps between their profiles and that of the active user in order to meaningfully assess similiraty. These users are viewed as the group of domain experts who are able to communicate best (in terms of their personal ontologies) with the active user. Furthermore, this problem is similar to that of making a recommendation and can be tackled by obtaining a ranked list of all possible combinations of item and user clusters, based on the current contextual setting.

System configuration composition: Having selected an appropriate recommendation subspace, we can now choose components that have performed well in the past with the domain experts as well as with the active user and similar recommendation contexts, to compose an appropriate system configuration. The unseen items of the highest relevance to the current context in the selected subspace, that have received high aggregate ratings by the domain experts are recommended to the active user.

Even on a conceptual level, the immense computational requirements of such a venture quickly become apparent. Pairwise comparisons need to be evaluated between all items, users and available components, thus prohibiting a centralised design. Therefore we intend to deploy the proposed system in a large scale peer-to-peer network, allowing the computation to take place in a distributed fashion through the exchange and reuse of data processed by different peers.

2.1 ONTOLOGY DESIGN

In order to encode and process contextual information, ontologies are used to record and reason over similarity between artifacts and identify indirect relationships between entities. Referrals to classes in the system ontology will appear in inverted commas for the remainder of this paper, for ease of reference. It is assumed that user preferences are correlated across different domains. Therefore by making a system aware of the different 'Types' of 'Items' available for recommendation allows these correlations to be discovered by indicating which features are comparable, or equivalent under a set of criteria and can be used to compute predicted ratings for unseen items.

Anything that can be recommended by the system is represented as a member of the concept 'Item' with an association to a valid 'Type' and is described through a number of 'Features'. The 'Features' associated with each 'Item' are restricted to a subset of all the 'Features' of the 'Type' the item belongs to. This scheme will allow the system to derive inferences on how different 'Type's' features are correlated with respect to each user, based on the profiling information available for that user.

Instances of the concept 'User' represent users in this ontology and each one is assigned an 'ID' and a 'Profile'. General information about the user is recorded through relationships like has-name, has-address, is-employed-by, etc. and is also associated with a number of 'History' instances:

QueryHistory: The user will be allowed to form explicit queries, imposing added restrictions on the 'Item's that can be recommended and to provide a finer grained recommendation context. These queries are recorded in instances of 'QueryHistory' and are used later, together with the corresponding 'RecommendationHistory' and 'RatingHistory' objects, in order to determine how apt 'Recommender' web services are for handling specific types of query.

RecommendationHistory: Any items recommended by the system to a user are recorded in their 'RecommendationHistory' together with a log of which web services were used to produce a recommendation. The future selection of web services will be based on the ratings assigned by the user to the recommendations.

RatingHistory: Instances of this class contain records of all the 'Rating's provided by a user and are used by 'ClusterDetector', 'Aggregator' and 'Classifier' web services as described below.

By implementing the various components used in the recommendation process as web services, an arbitrary number of runtime configurations can be achieved by using alternative instances from each class. Furthermore, by defining the concept 'WebService' as a subclass of the concept 'Item' allows for an added level of abstraction in the recommendation process, transparent to the user, where specific instances of 'WebService' are recommended in order to compose a system configuration. The various subclasses of 'WebService' are briefly described below.

ClusterDetector: Instances from this class, are able to detect clusters of users or items through the use of clustering algorithms and methods. The clustering for users is achieved by exploiting the information available in the 'History' and 'Profile' instances associated with them, while items are clustered together based on the

'Feature' instances collected by the system to describe them. Each time clusters need to be computed a 'ClusterDetector' instance is chosen based on its past performance in similar contexts.

Aggregator: An 'Aggregator' web service is responsible for computing the aggregate ratings of a user cluster, for some specified items. As in 'ClusterDetector' the choice of 'Aggregator' depends on its past performance and also on the current recommendation context.

Classifier: Classifier' web services are used to assign predicted ratings to unseen items, by training various machine learning algorithms on the user's 'RatingHistory'. Again, past performance in similar contexts determines the bias in choosing an instance.

Recommender: Web services of type 'Recommender' are responsible for evaluating the context of a recommendation need and for selecting the web services that will be used to produce that recommendation. 'Recommender's also receive predicted ratings computed by 'Classifier's and rank them according to the recommendation context. Different 'Recommender's may use different component selection and ranking strategies to improve performance in specific contexts.

Aligner: 'Aligner' web services are capable of computing partial alignments between two ontologies, in order to import knowledge from heterogeneous sources.

Gatherer: Instances of the 'Gatherer' web service are able to discover entities described in external ontologies and import them into the system, using an 'Aligner' to compute a translation of the gathered instances to the internal ontology.

3 CASE STUDY: MUSIC PREFERENCE PREDICTION

The key evaluation of our approach lies not in recommendation performance per se (although that is a goal), but rather in assessing how different representations of the same dataset affect the performance of recommendation schemes and how these may be applied to the same dataset at different times, predicated on the current system state and requirements. To quantify this, the magnitude of any improvement over conventional RS approaches where a single recommendation strategy is always used has to be evaluated. Therefore, we require a second fully-fledged RS (and not a 'random recommmender' as is usually assumed), already primed for the same dataset and known to produce high quality recommendations to meaningfully benchmark the proposed approach. Furthermore, the problem domain selected would ideally have clear links to other subjects, since we aim to be able to also recommend items from multiple domains, even those with no previous user ratings.

Given these requirements, a suitable domain had to be chosen with a sparse dataset, identified as the major source of shortcomings of RS, and have the characteristics identified above. We selected the data made available by the Audioscrobbler system [11], a music engine based on a massive collection of music profiles. These profiles are built through the widespread use of the company's flagship product, Last.fm [12], a system that provides personalised radio stations for its users and updates their profiles using they music they listen to, in order to improve the station's content, and also makes personalised artist recommendations. In addition Audioscrobbler exposes large portions of data through their web services API. Moreover, this dataset is extremely sparse (we only obtained values for 0.22% of all possible (User,Artist) pairs) since the number of available artists greatly exceeds the number of users and more than one track is typically available for each artist. Finally, the music domain is considered fertile ground for making cross-domain recommendation since some songs will be influenced by other art forms, associated with local traditions or socio political events, or even convey the ideological convictions of the creator.

The first task is to assess in a qualitative fashion the clusterings of the dataset that can be obtained without taking into account any collaborative information or content descriptors, but rather by considering the contextual metadata available from trusted on-line resources about the data points (in this case artists). Wikipedia [8] was identified as one such source, due to its wide coverage of subjects, maintained by their respective interest groups, which may be deemed expert knowledge and used as a source of contextual information for multiple domains [5]. For each resource all the hyper-links to and from the resource's Wikipedia page are recorded as items related to it. This relationship is then weighted by the relative importance of the discovered resources, a quantity thought to be proportional to the number of other pages linking to it. The rich, highly interconnected structure of wikis is considered ideal for this purpose.

Data was collected for 5 964 UK users of Last.fm and 17 882 artists, by implementing instances of Gatherer and Aligner WS, to interact with the Audioscrobbler WS and assign the instances acquired to the internal system ontology. Similarly, in total 158 273 resources from Wikipedia were discovered to be related to this set of artists. This was stored in a flat RDF file through Jena [13]. However the data has to be extracted from the RDF model and loaded into matrices each time a clustering algorithm is applied, as off the shelf algorithms typically use matrices as inputs, but mainly because of the much lower cost of accessing a cell in matrix than submitting a query to an RDF model.

As such the data was summarised in two matrices: a $[Users \times Artists]$ matrix where each row represents a user, each column an artist and each cell contains the number of times a user listened to a track by an artist, and a $[Artists \times Context]$, with each row corresponding to an artist and columns representing each one of the 158 273 resources gathered from Wikipedia, with boolean entries indicating whether an artist is related to the context described by a particular resource. Note that the dimensionality of these matrices strictly limits the use of computationally intensive machine learning algorithms on the raw data.

3.1 PROCESSING AND RESULTS

3.1.1 SINGULAR VALUE DECOMPOSITION (SVD) AND k NEAREST NEIGHBOURS (kNN)

To circumvent the computational limitations posed by the dimensionality of the dataset, the original input spcae was projected in 200 latent dimensions, by computing through SVD [20] the best $rank$ 200 approximation of the highly sparse (99.9832% zeros) $[Artist \times Context]$ matrix.

In order to assess whether the features harvested from Wikipedia contain enough information to describe similarity between artists as it is perceived by listeners, the cosine distance between each pair of artists in the vector space spanned in the 200 latent dimensions the dataset was reduced to, was evaluated and lists of 100, 50, 35, 20, 10 and 5 Nearest Neighbours were recorded. These lists were then compared to real user play count statistics, made available through the Audioscrobbler WS API, in the form the 100 artists with the highest number of appearances in the same playlist. If the artist has only appeared in the same playlist with less than 100 artists, only those are recorded. We used the following statistics to evaluate the quality of the clustering achieved:

Artists	**17882**
Artists with no Last.fm cluster	1120
Artists not found in Last.fm	528
Artists with no links to Wikipedia pages	13309
Working set	**4495**

k	***100***	***50***
Mean precision	**0.0655**	**0.0890**
Mean recall	**0.1025**	**0.0710**
Mean hits	6.5497	4.4523
Mean Kendall's τ	0.0868	0.0665
Corresponding Z-score	**1.9643***	**1.0760**
Critical Z value at the 10% level (two-sided) or 5% level(right-sided): ***1.64***		
Artists with > 0 hits	**3180**	**2937**
% of working set	70.75	65.34
Mean precision	**0.0926**	**0.1363**
Mean recall	**0.1449**	**0.1087**
Mean Kendall's τ	0.1226	0.1017
Corresponding Z-score	**2.7765***	**1.6468***
Random recommender		
Expected mean precision	0.000056	
Expected mean recall	0.000056	0.000028
Expected mean Kendall's τ	0	

Table 1. Precision/Recall and Kendall's correlation coefficient analysis of the results achieved by applying 100-NN and 50-NN on feature vectors reduced via SVD.

Precision and recall: While precision and recall are typically used to assess the quality of query results using labelled test data, their use in this context is not as well defined. Precision is defined as the number of hits retrieved by a search divided by the number of search results (1), while recall is the number of hits retrieved by the search divided by the number of all hits (2). This gives:

$$Precision = \frac{|kNN \cap Last.fm|}{k} \qquad (1)$$

$$Recall = \frac{|kNN \cap Last.fm|}{100} \qquad (2)$$

where $|k - NN \cap Last.fm|$ is the number of artists that appear in both lists. Since the number of all hits is unknown, we are forced to assume that the list provided by Last.fm is exhaustive, which is untrue. In addition, our choice of clustering algorithm defines explicitly the number of results that will be retrieved for any artist. However, training an unsupervised algorithm with a number of classes of the same order of magnitude as the number of artists in the working set is simply impractical. We observe that the obtained values are not large enough to suggest that the features collected are sufficient to reproduce the clusterings as those that emerge through recording real users' listening behaviours. However the order of improvement over the 'random recommender', and the fact that reducing the number of neighbours causes recall to reduce, while increasing precision and vice versa, as expected, provide motivation for further evaluating the utility of the contextual features gathered. In addition, it can be shown that both precision and recall monotonically increase as functions of the number of features available for each artist.

Kendall's τ: Kendall's correlation coefficient, τ, is a widely used statistic to compare two different rankings of the same variables and thus it was used to measure whether the 'hits' produced by k-NN are ranked in a similar manner as in Last.fm's lists of artists commonly played together. The τ value is obtained by dividing the difference between the number of concordant (n_c) and discordant (n_d) pairs in the ranked lists, by the total number of pairs. A concordant pair is defined as a two variables that appear in the same order in both rankings, while otherwise the pair is considered discordant. More formally:

$$\tau = \frac{n_c - n_d}{\frac{1}{2}n(n-1)} \qquad (3)$$

The results showed statistically significant evidence at the 5% level (right-sided) of correlation between the lists of 100-NN and those provided by Last.fm, and also for 50-NN when artists with 0 hits were excluded from the analysis, reinforcing our beliefs about the quality of the collected features. The full results are again presented in Table 1.

3.1.2 *NAIVE BAYES CLASSIFIER*

Having performed the analysis described in the previous subsection, the need for more meaningful metrics to qualitatively assess the harvested features arises. As such, a probabilistic evaluation scheme was decided to be carried out. Last.fm also makes available lists of the 50 most played artists by each user. These lists were randomly sampled to obtain train and test sets of various sizes. For each artist in the dataset we evaluate:

$$P(artist_i|top50_j) = \frac{P(top50_j|artist_i)P(artist_i)}{P(top50_j)} \qquad (4)$$

, where:

$$P(top50_j|artist_i) = \frac{N_j}{N_i} \qquad (5)$$

$$P(artist_i) \propto \prod_k (P(f_k)^{f_k^i} \quad , \quad P(f_k) = \frac{\sum_i f_k^i}{\sum_i \sum_k f_k^i} \qquad (6)$$

$$P(top50_j) \propto \prod_{artist_i \in top50_j} P(artist_i) \qquad (7)$$

Artists are treated as 'bags of features' and N_i denotes the number of users with $artist_i$ in their top 50 list, N_j the number of users with exactly the same artists in their list as $top50_j$ and f_k^i is the value of the k^{th} feature of $artist_i$. The analysis shows that on average $P(artist_i|top50_j)$ is consistently higher for artists in the test set, as shown in Figure 1. In particular, randomly sampling the top 50 lists 15 times to obtain test sets and recommending the 15 artists with the largest $P(artist_i|top50_j)$ gives $Precision = 0.4841$ and $Recall = 0.7333$ averaged over all users.

4 DISCUSSION AND FUTURE WORK

The analysis carried out to this point has shown that contextual relationships between artists and arbitrary resources can be successfully used to build feature vectors and to produce clusterings reflective of real users' listening preferences. It is intended that the collaborative filtering information, available from Last.fm will be imported into the system in order to assess the circumstances under which selection and combination of appropriate sub-spaces of the full high-dimensional recommendation space is beneficial, with respect to the predictive ability of the system. Mappings from these to possible recommendation contexts will then be drawn, to gain insight into both how recommendation contexts can be formed and articulated, as well

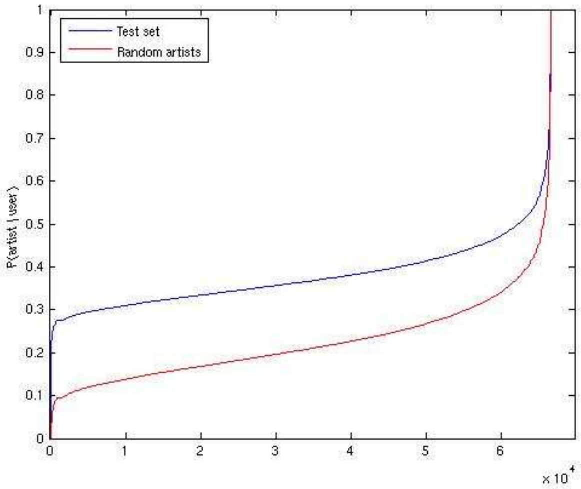

Figure 1. $P(artist_i|top50_j)$ for test and random sets, sorted in order of magnitude.

as how these can then be translated to restrictions on the sub-space selection process.

In addition to the fact that meaning can be more naturally be attributed to probabilistic classifications of any dataset, we found the naive Bayes classifier to greatly outperform k-NN in terms of precision and recall. This provides motivation for further investigating probabilistic dimensionality reduction and clustering techniques, and in particular Probabilistic Latent Semantic Analysis, [7]. PLSA is of particular interest, since its computational requirements can be drastically reduced, using multi-resolution kd-trees (mrkd-trees) as shown in [15], without compromising the optimality of the resluting classifications [22]. By using mrkd-trees to summarise and compartmentalise the dataset we expect to gain insight into how the computiations required can be carried out in a large scale distributed p2p system, where each pear is responsible for performing the computations in an encolosing hyperectangle.

Furthermore, we aim to assess which of the resources (as described by their Wikipedia pages) that have been extracted as contextual features for artists, can be regarded as recommendable items in their own right. This will be achieved by assessing the relative importance of these resources in Wikipedia and also by evaluating the probability they can be meaningfully matched to users, based on how big the overlap of the respective feature spaces is. The retrieval of conceptual descriptors of the newly found instances will also be attempted, through the use of ontology alignment techniques. It is intended that the resources discovered in this manner will be matched with concepts from arbitrary ontologies, thus indicating possible links and partial translations from one ontology to the other making possible the recommendation of items described in these ontologies that the system was previously unaware of. An RDF Knowledge Base described by the system ontology will then be populated and reasoning will be carried out in order to assess which features can be associated with or classify a 'Type' and to identify overlaps between different 'Types's' features in order to evaluate similarity.

REFERENCES

[1] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, 'Incorporating contextual information in recommender systems using a multidimensional approach', *j-TOIS*, **23**(1), 103–145, (January 2005).

[2] H Alani, S. Dasmahapatra, K. O'Hara, and N. R. Shadbolt, 'Identifying communities of practice through ontology network analysis.', *IEEE Intelligent Systems*, **18**(2), 18–25, (2003).

[3] A. Berenzweig, B. Logan, D. Ellis, and B. Whitman, 'A large-scale evaluation of acoustic and subjective music similarity measures', in *4th International Symposium on Music Information Retrieval*, (2003).

[4] D. Billsus and M. J. Pazzani, 'Learning collaborative information filters', in *15th International Conference on Machine Learning*, pp. 46–54. Morgan Kaufmann, San Francisco, CA, (1998).

[5] J. Giles, 'Internet encyclopaedias go head to head', *Nature*, **438**(7070), 900–901, (December 2005).

[6] T. Heath, E. Motta, and M. Dzbor, 'Use of contextual information to support online tasks', in *1st AKT Doctoral Symposium*, pp. 107–113.

[7] T. Hofmann, 'Probabilistic latent semantic indexing', in *22nd Annual ACM Conference on Research and Development in Information Retrieval*, pp. 50–57, Berkeley, California, (1999).

[8] The Wikimedia Foundation Inc., 'Wikipedia: The free encyclopedia', *http://wikipedia.org*, (2006).

[9] S Lawrence, 'Context in web search', *IEEE Data Engineering Bulletin*, **23**(3), 25–32, (2000).

[10] G. Linden, B. Smith, and J. York, 'Amazon.com recommendations: Item-to-item collaborative filtering.', *IEEE Internet Computing*, **7**(1), 76–80, (2003).

[11] Audioscrobbler Ltd., 'Audioscrobbler', *http://www.audioscrobbler.net*, (2006).

[12] Last.fm Ltd., 'Last.fm', *http://www.last.fm*, (2006).

[13] B. McBride, 'Jena: Implementing the rdf model and syntax specification.', in *SemWeb*, (2001).

[14] S. E. Middleton, D. C. De Roure, and N. R. Shadbolt, 'Capturing knowledge of user preferences: ontologies in recommender systems', in *1st international conference on Knowledge Capture*, pp. 100–107, New York, NY, USA, (2001). ACM Press.

[15] A. Moore, 'Very fast EM-based mixture model clustering using multiresolution kd-trees', in *Advances in Neural Information Processing Systems*, eds., M. Kearns and D. Cohn, pp. 543–549, 340 Pine Street, 6th Fl., San Francisco, CA 94104, (April 1999). Morgan Kaufman.

[16] D. M. Pennock, E. Horvitz, and C. L. Giles, 'Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering', in *AAAI/IAAI*, pp. 729–734, (2000).

[17] A. Popescul, L Ungar, D. M. Pennock, and S. Lawrence, 'Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments', in *17th Conference on Uncertainty in Artificial Intelligence*, pp. 437–444, Seattle, Washington, (2001).

[18] P. Resnick and H. R. Varian, 'Recommender systems', *Communications of the ACM*, **40**(3), 56–58, (1997).

[19] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, 'Analysis of recommendation algorithms for e-commerce', in *ACM Conference on Electronic Commerce*, pp. 158–167, (2000).

[20] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, 'Application of dimensionality reduction in recommender systems - a case study', in *ACM WebKDD2000, Web Mining for E-Commerce - Challenges and Opportunities*, (2000).

[21] A. I. Schein, A Popescul, L. H. Ungar, and D. M. Pennock, 'Methods and metrics for cold-start recommendations', in *25th International ACM Conference on Research and Development in Information Retreival*, (2002).

[22] J. J. Verbeek, J. R. J. Nunnink, and N. Vlassis, 'Accelerated EM-based clustering of large data sets', The Netherlands, (2005). Kluwer Academic Publishers.

[23] W. Yang, Z. Wang, and M. You, 'An improved collaborative filtering method for recommendations' generation.', in *IEEE International Conference on Systems, Man & Cybernetics*, pp. 4135–4139, (2004).