

Proceedings of the

**Workshop on Novelty and Diversity  
in Recommender Systems (DiveRS 2011)**

held at the

**5<sup>th</sup> ACM International Conference  
on Recommender Systems (RecSys 2011)**

23 October 2011

Chicago, Illinois, USA

Edited by

Pablo Castells<sup>1</sup>, Jun Wang<sup>2</sup>, Rubén Lara<sup>3</sup>, Dell Zhang<sup>4</sup>

<sup>1</sup> Universidad Autónoma de Madrid, Spain

<sup>2</sup> University College London, UK

<sup>3</sup> Telefónica, Investigación y Desarrollo, Spain

<sup>4</sup> Birkbeck, University of London, UK



# Preface

## Introduction

Most research and development efforts in the Recommender Systems field have been focused on accuracy in predicting and matching user interests. However there is a growing realization that there is more than accuracy to the practical effectiveness and added-value of recommendation. In particular, novelty and diversity have been identified as key dimensions of recommendation utility in real scenarios, and a fundamental research direction to keep making progress in the field.

Novelty is indeed essential to recommendation: in many, if not most scenarios, the whole point of recommendation is inherently linked to a notion of discovery, as recommendation makes most sense when it exposes the user to a relevant experience that she would not have found, or thought of by herself –obvious, however accurate recommendations are generally of little use.

Not only does a varied recommendation provide in itself for a richer user experience. Given the inherent uncertainty in user interest prediction –since it is based on implicit, incomplete evidence of interests, where the latter are moreover subject to change–, avoiding a too narrow array of choice is generally a good approach to enhance the chances that the user is pleased by at least some recommended item. Sales diversity may enhance businesses as well, leveraging revenues from market niches.

It is easy to increase novelty and diversity by giving up on accuracy; the challenge is to enhance these aspects while still achieving a fair match of the user’s interests. The goal is thus generally to enhance the balance in this trade-off, rather than just a diversity or novelty increase.

Research contributions to this area have addressed the enhancement, evaluation, and understanding of novelty and diversity in recommendation. Businesses are accounting for these aspects in ad-hoc ways when engineering recommendation functionalities, and researchers have started to seek principled foundations for incorporating novelty and diversity in the recommendation models, algorithms, theories, and evaluation methodologies. But large room remains for further research, which motivates the DiveRS 2011 Workshop.

The 1<sup>st</sup> ACM RecSys 2011 International Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011) gathered researchers and practitioners interested in the role of novelty and diversity in recommender systems. The workshop was motivated by the importance of these topics in the field, both in practical terms, for their relevance in the development of recommender systems applications and comprehending real user needs, and for their fundamental implications in recommender systems theory and evaluation methodologies. Novelty and diversity can thus be identified as a rich area for long-haul research. The area has started to be addressed only relatively recently in the field though, and the time thus seems appropriate for an open exchange of ideas, discussion, and reflection in an informal forum.

The workshop sought to advance towards a better understanding of what novelty and diversity are, how they can improve the effectiveness of recommendation methods and the utility of their outputs. DiveRS 2011 pursued the identification of open problems, specific gaps, relevant research directions, and opportunities for innovation in the recommendation business. The workshop sought the formation of common ground and shared perspectives to foster progress in this area.

## Scope and topics

Specific topics of interest for the workshop included, among others, the following:

- Modeling novelty and diversity in recommender systems.
  - Theoretical foundation for novelty and diversity.
  - Recommendation novelty and diversity models.
  - Popularity, risk, surprisal, serendipity, freshness, discovery.
  - Link to diversity models in Information Retrieval.
- Novelty and diversity enhancement.
  - Diversification methods.
  - Recommendation of long-tail and difficult items, cold-start.
  - Individual vs. global diversity.
  - Machine Learning for novelty and diversity.
- Novelty and diversity across recommendations.
  - Novelty and diversity in sequential recommendation.
  - Novelty and diversity in interactive recommendation.
  - Aggregate diversity.
  - Novelty and diversity in time and context.
  - Novelty and trust.
- Novelty and diversity evaluation.
  - Experimental methodologies and design.
  - Novelty and diversity metrics.
  - Datasets.
  - User studies.
- Business perspective on novelty and diversity.

The following questions were, among others, raised and addressed by the workshop:

- What are the different notions and dimensions of novelty and diversity? Is it possible to establish a clear definition and/or taxonomy?
- How are novelty and diversity themselves different and related?
- How can diversity, novelty, and accuracy be enhanced together?
- What important differences arise between the end-user point of view and the system or the business perspective?
- How can novelty and diversity be measured and evaluated?
- What are the potential implications of novelty on user trust, and how can they be properly cared for?
- What are the differences and unexplored connections between diversity as researched in Recommender Systems and Information Retrieval?
- Is there a relevant relation between novelty, diversity and context in recommendation?
- To what extent are novelty and diversity procured by or missing from state-of-the-art technologies?

- Do the different state-of-the-art recommendation algorithms (content-based, nearest-neighbors, matrix factorization, social, hybrid, ensembles, etc.) perform differently to each other in terms of novelty and diversity?
- To what extent are novelty and diversity a concern in the development of real-world recommender system applications, and how are they being addressed? What is the business value in novelty and diversity enhancement?
- What are the scenarios where novelty and diversity are most/least valuable or necessary? Are there situations in which novelty and diversity are not a desirable feature?

### **Submissions and program**

The workshop received 13 submissions, of which 7 were accepted (54%). The first three papers –in the order included in the present proceedings– were selected for long presentation, the next four having a slightly shorter slot in the workshop schedule. The workshop opened with a keynote talk by Neil Hurley, and included an open discussion after the paper presentations. We briefly summarize here the presented works and held discussions.

The keynote talk, entitled “Towards Diverse Recommendation”, provided an overview of the area, its development and main proposals over the past decade, and current perspectives. The papers presented after this covered a wide spectrum of topics, encompassing most of the aspects put forward in the intended workshop scope. The first three papers present approaches to enhance recommendation diversity and novelty, introducing or revising metrics to capture specific aspects of these dimensions. G. Adomavicius and Y. Kwon present a graph-based approach to enhance the global diversity of recommendation, understood as the total set of distinct items that are recommended to the set of all users as a whole. P. Adamopoulos and A. Tuzhilin revise and formalize the notion of unexpectedness as a particular case of user-specific novelty, and propose a method to maximize it. Also in the scope of novelty, K. Oku and F. Hattori propose a method to produce serendipitous recommendations by combining the features of different items of interest.

F. Mourão et al introduce a new angle on novelty by considering the effect that the passing of time may have on known items, which may regain some of their novelty value as past user experience fades away and is to some degree forgotten by users. They explore the positive effect that exploiting such oblivion processes may have on the diversity of recommendation. J. Golbeck and D. L. Hansen explore a new view on set-oriented recommendation by explicitly considering the recommendation of collections of items, where diversity arises as a natural quality dimension. R. Hu and P. Pu address the workshop theme from the point of view of real users and their perceptions. Their paper presents a user study of how user interface aspects influence the practical effectiveness of recommendation diversity and overall user satisfaction. Finally, S. Santini and P. Castells propose new formulations of novelty and diversity models based on fuzzy relevance, as an alternative to probabilistic formalizations based on binary relevance.

Different specific notions of novelty and diversity, distinctions and nuances between them, were identified along the presentations and discussion. The contextual nature of these dimensions –and the need for context-awareness in tackling them– were also underlined: novelty and diversity are relative to users, systems, time, viewpoint (e.g. user vs. business), tasks, session state, and other contextual variables. An issue that received particular attention in the discussion was the elucidation of when, to what extent, and in which scenarios, novelty and diversity are really appropriate in practice, from the understanding that their use should not be indiscriminate. This was a starting point for the open discussion, from which the session progressed towards further workshop topics.

While the usefulness of diversity and novelty is obvious in –or actually inherent to– many well-known applications, examples were mentioned of recommendation functionalities in commercial systems in which novelty and diversity seem to be disregarded, hinting that perhaps navigational recommendation or even the recommendation of known items might be useful in some contexts. Two points of view were distinguished to this respect: users and businesses. Regarding the latter, monetization was pointed out as a main effectiveness metric for commercial applications of recommendation technologies. It was noted to this respect that assessing the business value of novelty and diversity should require a distinction between short vs. longer-term –and direct vs. indirect– benefits. It was also noted that current commercial recommender technologies, such the ones used by Netflix, include novelty and diversity as features in some among their wide array of recommendation algorithms. There was general agreement that business studies in this area would be highly useful in shedding further light on these issues.

Regarding the end-user side, there was also a general call for user studies in order to properly understand and drive the introduction of novelty and diversity dimensions, as well as their precise need. The contribution by R. Hu and P. Pu was highlighted as an example of the studies that would be useful to this respect. User personality and attitude were indicated in this context as key aspects that should be taken into account when procuring novelty and diversity, since the attitude towards new and/or diverse experience varies considerably among users.

## **Conclusion**

The contributions, presentations and discussions held at the workshop provided a good overview of the current progress in the area, where we stand today, and where further work is needed. The importance of the workshop theme was underlined, beyond the DiveRS workshop itself, by the recurrent references to novelty and diversity in the main RecSys conference track. The need for further work and discussion in this area was clear, as well as the interest for future initiatives in line with the present workshop. To this respect, the organizers announced the forthcoming publication of a special issue of the ACM Transactions on Intelligent Systems and Technology in the scope of the workshop.

## **Acknowledgments**

The organizers would like to thank the Program Committee members for their high-quality and timely evaluation of the submissions; the RecSys 2011 organizers and workshop chairs for their support in the organization of this workshop; the keynote speaker, all the authors and presenters, for their contribution to a high-quality workshop program; and all participants for such fruitful discussions and valuable ideas as were exchanged during the workshop. Thanks are due to all such contributions which made DiveRS 2011 a successful venue.

*Pablo Castells*

*Jun Wang*

*Rubén Lara*

*Dell Zhang*

## Organizing Committee

Pablo Castells	Universidad Autónoma de Madrid, Spain
Jun Wang	University College London, UK
Rubén Lara	Telefónica, Investigación y Desarrollo, Spain
Dell Zhang	Birkbeck, University of London, UK

## Program Committee

Xavier Amatriain	Netflix, USA
Leif Azzopardi	University of Glasgow, UK
Iván Cantador	Universidad Autónoma de Madrid, Spain
Licia Capra	University College London, UK
Òscar Celma	Gracenote, USA
Charles Clarke	University of Waterloo, Canada
Sreenivas Gollapudi	Microsoft Research, USA
Neil Hurley	University College Dublin, Ireland
Oren Kurland	Technion, Israel
Neal Lathia	University College London, UK
Hao Ma	Microsoft Research, USA
Qiaozhu Mei	University of Michigan, USA
Jérôme Picault	Bell Labs, Alcatel-Lucent, France
Filip Radlinski	Microsoft, Canada
Davood Rafiei	University of Alberta, Canada
Francesco Ricci	Free University of Bozen-Bolzano, Italy
David Vallet	Universidad Autónoma de Madrid, Spain
Paulo Villegas	Telefónica, Investigación y Desarrollo, Spain
ChengXiang Zhai	University of Illinois at Urbana-Champaign, USA
Tao Zhou	University of Electronic Science and Technology of China, China
Jianhan Zhu	True Knowledge, UK





## Table of Contents

### Keynote talk

Neil J. Hurley <i>Towards Diverse Recommendation</i> .....	1
---	---

### Papers

Gediminas Adomavicius and Youngok Kwon <i>Maximizing Aggregate Recommendation Diversity: A Graph-Theoretic Approach</i> .....	3
Panagiotis Adamopoulos and Alexander Tuzhilin <i>On Unexpectedness in Recommender Systems: Or How to Expect the Unexpected</i> .....	11
Kenta Oku and Fumio Hattori <i>Fusion-based Recommender System for Improving Serendipity</i> .....	19
Fernando Mourão, Claudiane Fonseca, Camila Araújo and Wagner Meira Jr. <i>The Oblivion Problem: Exploiting Forgotten Items to Improve Recommendation Diversity</i> .....	27
Jennifer Golbeck and Derek L. Hansen <i>A Framework for Recommending Collections</i> .....	35
Rong Hu and Pearl Pu <i>Helping Users Perceive Recommendation Diversity</i> .....	43
Simone Santini and Pablo Castells <i>An Evaluation of Novelty and Diversity Based on Fuzzy Logic</i> .....	51



# Towards Diverse Recommendation

Neil J. Hurley  
University College, Dublin  
neil.hurley@ucd.ie

## **ABSTRACT**

In recent years great strides have been made in improving the accuracy of recommender systems from the point-of-view of their ability to predict users' ratings for unrated content given a database of past ratings. In a context where the system should ultimately recommend a list of items to the end-user, such accurate rating predictions can be seen as just one possible input into the decision system that selects the recommended content. It has been recognized for several years now that other qualities of the recommended list are also important in this selection process; it is not simply a matter of recommending those items with highest predicted ratings. In particular, a good system should offer a diverse choice of relevant items, allowing users to select from across their broad range of tastes. It is worth emphasizing that diversifying the recommendation is not simply a matter of selecting a set of highly dissimilar items for recommendation, since relevance is still a primary concern – increasing diversity while maintaining system performance, as measured by a relevance metric is a significant challenge. Research in diverse recommendation is still in an early stage; while a number of algorithms and systems for diverse recommendation have been proposed, many different performance measures and evaluation methodologies are being used making it difficult to compare across different approaches. In this talk, I attempt to summarize the state-of-the-art in diverse recommendation, bringing together the different approaches that have been proposed in recent years and the various performance measures that have been used. The goal is to set the context and to propose some ideas to generate what should be some interesting and controversial discussions during the remainder of the workshop.



# Maximizing Aggregate Recommendation Diversity: A Graph-Theoretic Approach

Gediminas Adomavicius

Department of Information and Decision Sciences  
University of Minnesota

gedas@umn.edu

YoungOk Kwon

Department of Information and Decision Sciences  
University of Minnesota

kwonx052@umn.edu

## ABSTRACT

Recommender systems are being used to help users find relevant items from a large set of alternatives in many online applications. Most existing recommendation techniques have focused on improving recommendation accuracy; however, diversity of recommendations has also been increasingly recognized in research literature as an important aspect of recommendation quality. This paper proposes a graph-theoretic approach for maximizing aggregate recommendation diversity based on maximum flow or maximum bipartite matching computations. The proposed approach is evaluated using real-world movie rating datasets and demonstrates substantial improvements in both diversity and accuracy, as compared to the recommendation re-ranking approaches, which have been introduced in prior literature for the purpose of diversity improvement.

## Keywords

Recommendation diversity, aggregate diversity, collaborative filtering, graph-based algorithms.

## 1. INTRODUCTION AND MOTIVATION

Many recommendation techniques have been developed over the past decade, and major efforts in both academia and industry have been made to improve recommendation accuracy, as exemplified by the recent Netflix Prize competition. However, it has been increasingly noted that it is not sufficient to have accuracy as the sole criteria in measuring recommendation quality, and we should consider other important dimensions, such as diversity, novelty, serendipity, confidence, trust, to generate recommendations that are not only accurate but also useful to users [19,29,34].

In this paper, we focus on the *aggregate diversity* of recommendations, which has recently attracted attention in research literature due to its impact on the shifts in product variety and sales concentration patterns [11,12,15,31]. As observed by Brynjolfsson et al. [12], recommender systems can play a key role in increasing both “long tail” and “superstar” effects in real-world e-commerce applications. In particular, the “long tail” literature argues that recommendations on the Internet help to increase users’ awareness of niche products and create a long tail in the distribution of product sales [6,11,15,31]. For example, one study, using data from online clothing retailer, demonstrates that recommendations would increase sales of the items in the long tail, resulting in the improvement in aggregate diversity [11]. In contrast, the “superstar” literature indicates that recommender systems may promote the so-called “rich get richer” phenomenon, where users are recommended more popular/bestselling items than idiosyncratic/personalized ones. One explanation for this is that the niche products often have limited historical data and, thus, are more difficult to recommend to users, whereas popular products typically have more ratings and, thus, can be recommended to more users [15,26,36].

Copyright is held by the author/owner(s).

Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011), held in conjunction with ACM RecSys 2011. October 23, 2011, Chicago, Illinois, USA.

More diverse recommendations, presumably leading to more sales of long-tail items, could be beneficial for both individual users and some business models [10,11,18]. Exposing individual consumers to more long-tail recommendations can intensify this effect. Thus, more consumers would be attracted to the companies that carry a large selection of long tail items and have long tail strategies, such as providing more diverse recommendations [12]. Also, some business models (e.g., Netflix), can benefit from recommendation diversity, because more diverse recommendations would encourage users to rent more long-tail movies, which are less costly to license and acquire from distributors than new releases or extremely popular movies of big studios [18].

Taking into consideration the potential benefits of aggregate diversity (hereinafter simply diversity) to individual users and businesses, several studies have explored new methods that can increase the diversity of recommendations [2,3,23,27,32]. In particular, considering that recommender systems typically compute recommendations to users in two phases – (Phase 1) estimating ratings of items that the users have not consumed yet and (Phase 2) generating top- $N$  items for each user – the prior work can be divided into two lines of research. One line of research [23,27,32] aims to enhance the estimation phase (mainly for long tail items), and the other focuses on finding the best set of recommendations in the recommendation generation phase [2,3]. The approach proposed in this paper fits within the latter line of research and, therefore, has the flexibility of being used in conjunction with *any* available rating estimation algorithms, as illustrated by our empirical evaluation. In contrast to simple recommendation re-ranking heuristics for diversity improvement proposed in [2,3], we develop a more sophisticated and systematic graph-based approach for direct diversity maximization, while maintaining acceptable levels of accuracy.

Our empirical results, using real-world rating datasets, show that the proposed graph-based approach consistently outperforms the recommendation re-ranking approach from prior literature in terms of both accuracy and diversity. The paper also discusses the scalability of the proposed approach in terms of its theoretical computational complexity as well as its empirical runtime based on real-world rating datasets.

## 2. RELATED WORK

In this section, we briefly discuss two widely used recommendation techniques that are used in conjunction with our proposed approach in our empirical experiments as well as two important dimensions in the evaluation of recommendation quality: accuracy and diversity. We also discuss a simple recommendation re-ranking approach from prior literature, which has been shown to improve the aggregate diversity of recommendations with only a small loss of accuracy, and which

we will use as one of the baseline comparison techniques.

## 2.1 Recommendation Algorithms

Let  $U$  be the set of users and  $I$  be the set of items available in the recommender system. Then, the usefulness or utility of any item  $i$  to any user  $u$  can be denoted as  $R(u,i)$ , which usually is represented by a rating (on a numeric, ordinal, or binary scale) that indicates how much a particular user likes a particular item [1]. Thus, the job of a recommender system in the rating estimation phase (Phase 1) is to use known ratings as well as other information that might be available (e.g., content attributes of items or demographic attributes of users) to estimate ratings for items that the users have not yet consumed. For clarity, we use  $R(u,i)$  to denote the actual rating that user  $u$  gave to item  $i$ , and  $R^*(u,i)$  for the system-estimated rating for item  $i$  that user  $u$  has not rated before. Given all of the unknown item predictions for each user, in generating top- $N$  recommendations (Phase 2) the system selects the most relevant items, i.e., items that maximize a user’s utility, according to a certain ranking criterion. More formally, item  $i_x$  is ranked ahead of item  $i_y$ , if  $rank(i_x) < rank(i_y)$ , where  $rank: I \rightarrow \mathbf{R}$  is a function representing some ranking criterion. Most recommender systems rank the candidate items by their *predicted rating value* and recommend to each user the  $N$  most highly predicted items (where  $N$  is a relatively small positive integer) because users are typically interested in (or have time for) only a limited number of recommendations. We refer to this as the *standard ranking approach* and can formally define the corresponding ranking function as  $rank_{Standard}(i) = R^*(u,i)^{-1}$ . While the standard ranking approach exhibits good recommendation accuracy, its performance in terms of recommendation diversity is poor [2,3], which further emphasizes the need for different recommendation approaches for diversity improvement.

Among a large number of recommendation techniques that have been developed over the past decade, collaborative filtering (CF) techniques represent most widely used and well-performing algorithms; we use two representative CF techniques for Phase 1 (i.e., rating estimation) in this paper: neighborhood-based CF and matrix factorization CF techniques.

**Neighborhood-based CF techniques.** The basic idea of neighborhood-based CF techniques is, given a target user, to find the user’s neighbors who share similar rating patterns, and then to use their ratings to predict the unknown ratings of the target user [1,9]. There are many variations of computational methods to identify a user’s neighbors (i.e., by computing the similarity between users) and aggregate the neighbors’ ratings for the user. In our experiments, we use a popular cosine similarity measure for calculating similarity between users, and the final rating prediction for a specific item to a user is made as an adjusted weighted sum of the ratings of the user’s closest 50 neighbors on this item. The neighborhood CF techniques can be user- or item-based, depending on whether the similarity is computed between users or items [33]; we use both variations in this paper.

**Matrix factorization CF techniques.** Matrix factorization CF techniques have recently gained popularity because of their effectiveness in the Netflix Prize competition in terms of predictive accuracy. In contrast to heuristic-based techniques (such as the neighborhood-based CF techniques mentioned above), the matrix factorization CF techniques use the existing ratings to learn a model with  $k$  latent variables for users and items. In other words, this technique models and estimates each user’s preferences for  $k$  latent features as the user-factors vector and

each item’s importance weights for the  $k$  latent features as the item-factors vector [16,24]. Then, the predicted rating of item  $i$  for user  $u$  can be computed as an inner product of the user-factors vector for user  $u$  and the item-factors vector for item  $i$ . Typically, the model-based techniques have been shown to generate more accurate recommendations than heuristic-based techniques. While a number of variations for the matrix factorization technique have been developed, in this paper we use its basic version, as proposed by Funk [16].

## 2.2 Recommendation Accuracy and Diversity

**Recommendation Accuracy.** The goal of this work is to generate good top- $N$  recommendation lists in terms of accuracy and diversity and, accordingly, we chose to evaluate the accuracy of top- $N$  recommendation lists using one of the most popular decision-support metrics, *precision* [19]. Simply put, precision is measured as a proportion of “relevant” items among the recommended items across all users. Note that the decision-support metrics, such as precision, typically work with binary outcomes; therefore, here the notion of “relevance” is used to convert a numeric rating scale (e.g., 1-5) into binary scale (i.e., relevant vs. irrelevant).

More specifically, in our empirical data ratings are provided on a 5-point (or 5-star) scale, and the natural assumption is that users provide higher ratings to the items that are more relevant to them. As a consequence, in our experiments, we treat items with ratings 4 and 5 as relevant, and items with ratings 1, 2, 3 as irrelevant, or, more precisely, we choose the threshold between relevant or and irrelevant items as 3.5 (denoted by  $T_H$ ). The list of  $N$  items recommended for user  $u$  should include only items predicted to be relevant and can be formally defined as  $L_N(u) = \{i_1, i_2, \dots, i_N\}$ , where  $R^*(u, i_k) \geq T_H$  for all  $k \in \{1, 2, \dots, N\}$ . The precision of such top- $N$  recommendation lists, often referred to as *precision-in-top- $N$* , is calculated as the percentage of truly “relevant” items, denoted by  $correct(L_N(u)) = \{i \in L_N(u) \mid R(u, i) \geq T_H\}$  among the items recommended across all users, and can be formalized as:

$$precision - in - top - N = \frac{\sum_{u \in U} correct(L_N(u))}{\sum_{u \in U} |L_N(u)|}$$

In real-world settings, obviously a recommender system has to be able to recommend items that users have not yet rated (the ratings for those items typically become available to the system only after item consumption), i.e., the true precision of the generated recommendation lists is not known at the time of recommendation. However, using two popular real-world datasets (details on datasets are provided in Section 4), different popular CF recommendation algorithms discussed above, and standard cross-validation techniques from machine learning and data mining, we show that, not surprisingly, precision is highly correlated with average predicted rating value of recommended items using for all recommendation algorithms, as indicated in Fig. 1. In other words, recommending items with higher predicted rating values results in higher precision (i.e., higher likelihood that the user would actually like the item), which provides further empirical support for using the standard ranking approach if the goal is just to maximize recommendation accuracy. An important consequence of this relationship is that we can use the average predicted rating value of top- $N$  recommendation lists, which can always be computed at the time of recommendation, as a simple proxy for the precision metric. In addition, this metric is extremely simple to compute and easily scales to large-scale real-world applications. We refer to this

metric as *prediction-in-top-N* and formally define it as follows:

$$\text{prediction-in-top-N} = \sum_{u \in U} \sum_{i \in L_N(u)} R^*(u,i) / \left| \sum_{u \in U} L_N(u) \right|$$

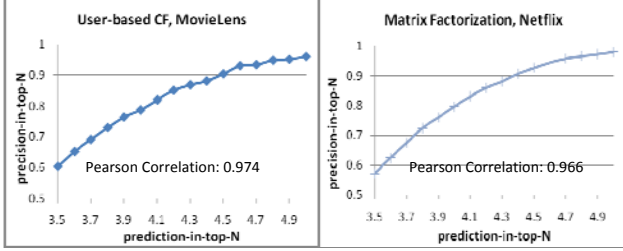


Figure 1. Precision versus Average Predicted Rating Value

**Recommendation Diversity.** As discussed earlier, accurate recommendations are not always useful to users. For example, recommending only popular items (e.g., blockbuster movies that many users tend to like) could obtain high accuracy, but also can lead to a decline of other aspects of recommendations, including recommendation diversity. The inherent tradeoff between accuracy and diversity has been observed in previous studies [2,3,30,38,39], therefore, indicating that maintaining accuracy while improving diversity constitutes a difficult task.

The diversity of recommendations has been assessed at either individual or aggregate level. The majority of previous studies have focused on individual diversity [8,21,30,35,37,38,39]. For example, the individual diversity of recommendations for a user can be measured by calculating an average dissimilarity between all pairs of items recommended to a user (e.g., based on item attributes). In contrast, the aggregate diversity of recommendations across all users has been relatively less studied, and the recent interest in the impact of recommender systems on product variety and sales concentration patterns [11,12,15,31] has sparked a renewed interest in this topic.

Several metrics can be used to evaluate various aspects of aggregate diversity, including absolute long-tail metrics that measure the change in the absolute number of items recommended (e.g., recommendation frequency of items above a certain popularity rank), relative long-tail metrics to measure the relative share of recommendations above or below a certain popularity rank percentile, and the slope of the log-linear relationship between item popularity rank and recommendations (or sales) that can indicate the relative importance of the head versus the tail of the distribution [12]. In the recommender systems literature, both absolute and relative long-tail metrics have been used to measure the aggregate diversity of recommendations [2,3,19,23,27,37]. In this paper, we use a simple absolute long-tail metric which measures aggregate diversity using the total number of distinct items among the top- $N$  items recommended across all users, referred to as the *diversity-in-top-N* [2,3]. More formally:

$$\text{diversity-in-top-N} = \left| \bigcup_{u \in U} L_N(u) \right|$$

and prior research has shown that this simple and easy-to-compute metric exhibits high correlation with more sophisticated, distributional diversity metrics [3], i.e., is able to properly capture the same diversity dynamics as some of the relative long-tail metrics on several real-world rating datasets. This diversity metric could also potentially be viewed as a crude indicator of the system’s level of personalization, because high diversity implies that each user gets very different and unique set of

recommendations (potentially indicating a high level of personalization), whereas low diversity indicates that mostly the same items (possibly bestsellers) are recommended to all users (i.e., low level of personalization).

Although the approach proposed in this paper aims to improve aggregate recommendation diversity, their accuracy is also given the proper attention in the paper, because diverse but inaccurate recommendations may not provide significant value to the users.

### 2.3 Re-Ranking Approaches for Diversity

Several prior studies have explored improving aggregate diversity of recommendations [2,3,23,27,32]. As discussed earlier, one line of research proposes new methods for predicting unknown ratings, mainly for long-tail items. For example, Park and Tuzhilin [32] propose new clustering methods to improve predictive accuracy of long-tail items that have only few ratings, which can also increase the recommendation of long-tail items. In addition, Levy and Bosteels [27] design long-tail music recommender systems, simply by removing popular artists (i.e., with more than 10,000 listeners) in the rating prediction phase. Also, a local scoring model, proposed by Kim et al. [23], was developed to alleviate the scalability and sparsity problems by suggesting a more efficient way to select the best neighbors for neighborhood-based recommendation techniques; however, as a by-product, it is shown to improve aggregate recommendation diversity.

In contrast to these studies, another line of research proposes new approaches for improving top- $N$  item selection *after* the rating estimation is performed. In particular, Adomavicius and Kwon [2,3] propose a heuristic approach for recommendation re-ranking, which has been shown to improve aggregate diversity with a negligible accuracy loss and represents an important baseline for comparison with our proposed diversity maximization approaches. Typical recommender systems recommend to users those items that have the highest predicted ratings, i.e., using the standard recommendation ranking criterion  $rank_{\text{Standard}}$ , as discussed earlier. While the standard ranking approach is used to maximize the accuracy of recommendations, as was illustrated by Fig. 1, Adomavicius and Kwon [2,3] showed that changing the ranking of items (i.e., not following the standard ranking approach) can help with other aspects of recommendation quality, in particular, with recommendation diversity. As a result, they proposed several alternative re-ranking approaches, and showed that all of them can provide substantial improvements in recommendation diversity with only negligible accuracy loss. In our experiments, as a baseline for comparison, we specifically use the ranking approach based on the reverse predicted rating value. This is a personalized yet simple and highly-scalable ranking approach that can be formally defined as  $rank_{\text{RevPred}}(i) = R^*(u,i)$ .

While this re-ranking approach can significantly improve recommendation diversity, as might be expected, this improvement comes at the expense of recommendation accuracy, since not the most highly predicted items are recommended. Adomavicius and Kwon [2,3] demonstrate that the balance between diversity and accuracy can be achieved by parameterizing any ranking function with “ranking threshold”  $T_R \in [T_H, T_{\text{max}}]$  (where  $T_{\text{max}}$  is the largest rating on the rating scale). That is, the ranking threshold enables to specify the level of acceptable accuracy loss while still extracting a significant portion of diversity improvement. The parameterized version  $rank_{\text{RevPred}}(i, T_R)$  of ranking function  $rank_{\text{RevPred}}(i)$  can be implemented as:

$$rank_{RevPred}(i, T_R) = \begin{cases} rank_{RevPred}(i), & \text{if } R^*(u, i) \in [T_R, T_{\max}] \\ \alpha_u + rank_{Standard}(i), & \text{if } R^*(u, i) \in [T_H, T_R) \end{cases},$$

where  $\alpha_u = \max_{i \in I_u^*(T_R)} rank_{RevPred}(i)$ , and  $I_u^*(T_R) = \{i \in I \mid R^*(u, i) \geq T_R\}$ .

In particular, items with predicted ratings from  $[T_R, T_{\max}]$  would be ranked ahead of items with predicted ratings  $[T_H, T_R)$ , as ensured by  $\alpha_u$  in the above definition. Increasing the ranking threshold  $T_R$  towards  $T_{\max}$  would enable choosing the most highly predicted items (i.e., more accuracy and less diversity – similar to the standard ranking approach), while decreasing the ranking threshold  $T_R$  towards  $T_H$  makes  $rank_{RevPred}(i, T_R)$  increasingly more similar to the pure ranking function  $rank_{RevPred}(i)$ , i.e., more diversity with some accuracy loss. Thus, choosing  $T_R \in [T_H, T_{\max}]$  values in-between the two extremes allows setting the desired balance between accuracy and diversity. In our experiments, we are able to explore the accuracy-diversity tradeoff of the re-ranking approach, by varying this ranking threshold  $T_R$ .

In terms of computational complexity, the re-ranking approach is implemented as a simple sorting algorithm. Assuming there are  $m$  users and  $n$  items, the worst case situation for this algorithm occurs when all  $n$  items are available to every user for recommendation. Then, the heuristic-based ranking does the job of sorting  $n$  items,  $O(n \log n)$ , for  $m$  users, and its complexity would be  $O(mn \log n)$ .

### 3. PROPOSED APPROACH

While the recommendation re-ranking approach can obtain a certain level of diversity gains at the expense of a small loss in accuracy, in this section we propose a graph-based approach that can obtain maximum possible diversity.

Graph-based algorithms have been previously used in recommender systems [4,22,28], mostly for the purpose of improving predictive accuracy of CF techniques. We formulate our problem of diversity maximization as a well-known *max-flow problem* in graph theory [5,14]. One simple version of the general maximum flow problem, which has been extensively studied in operations research and combinatorial optimization, can be defined as follows. Assuming that  $V$  is the set of vertices (or nodes), and  $E$  is the set of directed edges, each of which connects two vertices, let  $G = (V, E)$  be a directed graph with a single source node  $s \in V$  and a single sink node  $t \in V$ . Each directed edge  $e \in E$  has capacity  $c(e) \in \mathbf{R}$  associated with it. Also, the amount of actual flow between two vertices is denoted by  $f(e) \in \mathbf{R}$ . The flow of an edge cannot exceed its capacity, and the sum of the flows entering a vertex must equal the sum of the flows exiting a vertex, except for the source and the sink vertices. The maximum flow problem is to find the largest possible amount of flow passing from the source to the sink for a given graph  $G$ .

Translating the top- $N$  recommendation setting into a graph-theoretic framework, let users and items be represented as vertices, and an edge from user  $u$  to item  $i$  exists if and only if item  $i$  is predicted to be relevant for user  $u$ , i.e.,  $R^*(u, i) \geq T_H$  or, in other words, when the item is available to the user for recommendation. Each edge has capacity  $c(e) = 1$  and can be assigned an integer flow of 1 if item  $i$  is actually recommended to user  $u$  as part of top- $N$  recommendations, and the flow of 0 otherwise. As described in the example in Fig. 2a, we augment this directed graph by adding a source node and connecting it by directed edges to each of the user vertices. Let the capacity of each of

these “source” edges be  $N$  and, again, only integer flows of 0, 1, ..., or  $N$  are permitted on each of these edges. Furthermore, we also augment this graph by adding a sink node and connecting each item vertex by a directed edge to this node. Let the capacity of each of these “sink” edges be 1, and again only integer flows (i.e., 0 or 1) are permitted for these edges.

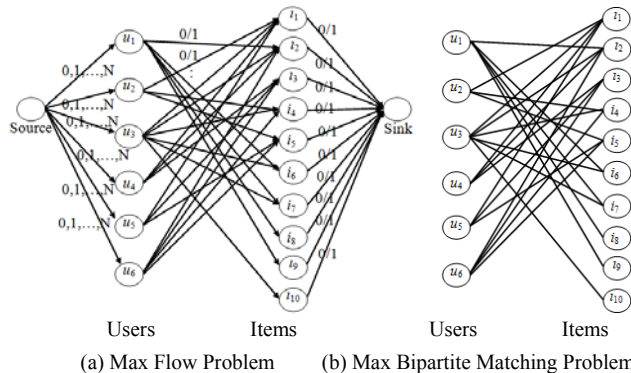


Figure 2. Top- $N$  Recommendation Task as a Graph Theory Problem

As can be easily seen from this construction, because of the specified capacity constraints, i.e., “source” edges not allowing flows larger than  $N$  through each user node and “sink” edges not allowing flows larger than 1 through each item node, the maximum flow value in this graph will be equal to the maximum possible number of edges from users to items that can have flow of 1 assigned to them. In other words, the max-flow value will be equal to the largest possible number of recommendations that can be made from among the available (highly predicted) items, where no user can be recommended more than  $N$  items, and no item can be counted more than once, which is precisely the definition of the *diversity-in-top- $N$*  metric.

Note that, while finding the maximum flow will indeed find the recommendations that yield maximum diversity, since the recommendation of each item is counted only once (i.e., restricted to only one user), as part of the max-flow solution some users may have fewer than  $N$  recommendations. The remaining recommendations for these users can be filled arbitrarily, as they cannot further increase the maximum diversity. We employ the standard ranking approach for the not-yet-recommended items for each user (i.e., the remaining items with the highest predicted ratings), for the purpose of achieving better accuracy.

The maximum flow problem represents a simple and intuitive metaphor for computing top- $N$  recommendations with maximum possible aggregate diversity, and there are many efficient (polynomial-time) algorithms for finding the maximum flow in a given graph [5,14]. Note, however, that the flow graph constructed for the diversity maximization problem is a highly specialized graph, and it may be possible to find even more effective graph-based algorithms for this problem, as compared to general-purpose max-flow algorithms.

To illustrate this, let’s consider the simplest top- $N$  recommendation setting, i.e., where  $N = 1$ . Since each user can be recommended only one item, *all* edges in our max-flow problem would become single-unit capacity edges, implying that the max flow in this graph will correspond to the largest possible set of edges from users to items, where no user and no item can be part of more than one such edge. Because there are no edges between two different users or between two different items (i.e., we have a bipartite user-item graph), for top-1 recommendation



settings the maximum flow problem is, thus, equivalent to the more specialized *maximum bipartite matching* problem which, furthermore, has more efficient algorithmic solutions. Thus, while the max-flow approach represents a general, intuitive approach for achieving maximum diversity by implementing a single-source and single-sink flow network, we follow the equivalent yet more efficient maximum bipartite matching approach (as illustrated in Fig. 2b) and also show how it can be extended from top-1 to the more general top- $N$  settings.

As summarized in Fig. 3, our max-flow/matching optimization approach consists of two steps: (1) find maximum diversity by solving the maximum bipartite matching problem and (2) complete top- $N$  recommendations by applying the standard ranking approach. Since the maximum diversity in Step 1 can be obtained at some expense of accuracy, one can control the balance between accuracy and diversity with the simple parameterization of a “flow-rating threshold”  $T_F \in [T_H, T_{\max}]$ . This allows pre-processing of the data, specifically, to include only higher predicted items (i.e., above  $T_F$ ) among the items that can be recommended for the maximum diversity in Step 1. Similarly to how the ranking threshold was used in re-ranking approaches (Section 2.3), here the lowest  $T_F$  value provides the best diversity but a relatively lower accuracy, whereas higher values of  $T_F$  lower the diversity but provide a certain level of accuracy. Then, in Step 2, the highest predicted remaining items are used to complete top- $N$  recommendation lists.

More formally, let  $G = (U, I; E)$  be a bipartite graph, where vertices represent users  $U$  and items  $I$ , and edges  $E$  represent the possible recommendations of items for users. A subset of edges  $M$  (i.e.,  $M \subseteq E$ ) is a *matching*, if all edges in  $M$  are pairwise non-adjacent, i.e., any two edges in  $M$  share neither a user vertex nor an item vertex. A vertex is *matched* if it is adjacent to an edge that is in the matching (otherwise, the vertex is unmatched). The *maximum matching* of a bipartite graph is a matching with the largest possible number of edges. The maximum bipartite matching algorithm (for top-1 recommendations) in Step 1 starts with matching  $M = \emptyset$  and iteratively adds edges to  $M$ , until all users are matched or no new additional edge can be added. The edges to be iteratively added to  $M$  can be found by finding an *augmenting path* for  $M$ , which is a simple path (i.e., a sequence of alternating user and item vertices with no loops) that starts at an unmatched user and ends at an unmatched item, and its edges belong alternately to  $EM$  and  $M$ . In other words,  $P = (v_1, v_2, \dots, v_{2n-1}, v_{2n})$  is an augmenting path where  $v_{\text{odd}} \in U$ ,  $v_{\text{even}} \in I$ ,  $v_1$  is an unmatched user,  $v_{2n}$  is an unmatched item,  $(v_{2k-1}, v_{2k}) \notin M$  where  $k = \{1, \dots, n\}$ , and  $(v_{2k+1}, v_{2k}) \in M$  where  $k = \{1, 2, \dots, n-1\}$ . Let  $edges(P)$  comprise the set of all edges of the augmenting path  $P$ . The key property of augmenting paths is that the symmetric set difference of  $M$  and  $edges(P)$ , denoted as  $M \Delta edges(P)$ , always results in a matching with cardinality one more than the cardinality of  $M$  [5,14], i.e., if  $M' = M \Delta edges(P)$ , then  $|M'| = |M| + 1$ .

Thus, the notion of augmenting paths allows to find the maximum bipartite matching, by starting with matching  $M = \emptyset$  and iteratively increasing its size one-by-one with each augmenting path, which we use in our algorithm for diversity maximization (Fig. 3). In particular, we adopt Hopcroft-Karp algorithm [20], which finds a maximal set of augmenting paths during every iteration, i.e., multiple augmenting paths in parallel for all unmatched vertices, thereby achieving a significant reduction in time complexity. This is a well-known technique and we

encapsulate it in our algorithm by calling *Find AugmentingPaths* subroutine (lines 6, 15 in Fig. 3); the implementation details for this subroutine can be found in [13].

```

[Step 1] Find Maximum Diversity
// set of edges- items available for recommendation
1   $E := \{(u,i) \mid u \in U, i \in I, R^*(u, i) \in [T_F, T_{\max}]\}$ 
2   $G := (U, I; E)$  // bipartite graph with users, items, and edges
// initialize a set of unmatched users /items
3   $CU := U; CI := \{i \in I \mid u \in U, (u,i) \in E\}$ 
4   $M := \emptyset$  // set of matched edges  $M \subseteq E$ 
Maximum Bipartite Matching (Top-1 Task)
5  // find augmenting paths starting from unmatched user  $v_1$  and ending with
// unmatched item  $v_{2n}$ 
6   $P := \text{Find\_AugmentingPaths}(G, CU, CI, M)$ 
// until all users are matched or no augmenting path exists
7  while ( $CU \neq \emptyset$  and  $P \neq \emptyset$ )
8  for each  $(v_1, v_2, \dots, v_{2n-1}, v_{2n}) \in P$  do
9   $edges := \{(v_{2k-1}, v_{2k}) \mid k = 1..n\} \cup \{(v_{2k+1}, v_{2k}) \mid k = 1..n-1\}$ 
// flip the matched and unmatched edges
10  $M := M \Delta edges$  // symmetric difference
11
12 Remove  $v_1$  from  $CU$  // one matching per user
13 Remove  $v_{2n}$  from  $CI$  // one matching per item
14 end for
15  $P := \text{Find\_AugmentingPaths}(G, CU, CI, M)$ 
16 end while
Extended Version for Top-N Recommendation Task
5  $\forall u \in U, uCnt[u] := 0$  // num. of matches for each user
6  $P := \text{Find\_AugmentingPaths}(G, CU, CI, M)$ 
7 while ( $CU \neq \emptyset$  and  $P \neq \emptyset$ )
8 for each  $(v_1, v_2, \dots, v_{2n-1}, v_{2n}) \in P$  do
9  $edges := \{(v_{2k-1}, v_{2k}) \mid k = 1..n\} \cup \{(v_{2k+1}, v_{2k}) \mid k = 1..n-1\}$ 
10  $M := M \Delta edges$ 
11  $uCnt[v_1] := uCnt[v_1] + 1$  // N matchings per user
12 Remove  $v_1$  from  $CU$  if  $uCnt[v_1] == N$ 
13 Remove  $v_{2n}$  from  $CI$  // one matching per item
14 end for
15  $P := \text{Find\_AugmentingPaths}(G, CU, CI, M)$ 
16 end while
[Step 2] Complete Top-N Recommendations
17 for each  $(u, i) \in M$  do
18 Add  $i$  to  $LN(u)$  // assign matchings as recommendations
19 end for
20 for each  $u \in CU$  do // fill the remaining items according to rankStandard
21 Sort items  $\{i \in I \mid R^*(u, i) \in [T_H, T_{\max}] \text{ and } i \notin LN(u)\}$ 
22 Add top  $(N - |LN(u)|)$  most highly predicted items to  $LN(u)$ 
23 end for

```

Figure 3. Bipartite Matching Approach to Diversity Maximization.

The original bipartite matching algorithm for top-1 recommendations matches a user to only one item and excludes the matched user for the subsequent iterations, i.e., the user is removed from candidate user list  $CU$  (line 12 of Fig. 3). An extended version for top- $N$  recommendations relaxes this rule by waiting to remove the user from  $CU$  until the same user is matched to  $N$  items. We also make the extended algorithm more efficient by allowing a single user to find up to  $N$  item matches in the first iteration (and not just a single match per iteration), which significantly reduces the number of subsequent iterations. However, similarly as with the max-flow approach, since an item can be recommended to only one user, some users may get fewer than  $N$  recommendations. Thus, in Step 2, for accuracy considerations, the most highly predicted items among remaining candidate items are chosen to fill the remaining top- $N$  recommendations for all users. Note that this does not affect diversity (which is already guaranteed to be maximum).

Using the same example in Fig. 2, we illustrate the first step for

top-1 recommendations, how the maximum bipartite matching algorithm can obtain the maximum diversity (Fig. 4a). This algorithm performs two iterations: (Iteration 1) finds all possible 1-edge augmenting paths between unmatched users and unmatched items, i.e., direct paths without any intermediate vertices; and (Iteration 2) finds multi-edge augmenting paths, each of which increases the cardinality of matching by one unit via alternating non-matched and matched edges in the paths. After the first iteration of Fig. 4a, the first five users are matched to one of their candidate items, but user  $u_6$  is still unmatched because all her candidate items ( $i_2, i_3, i_4, i_5$ ) are already matched to other users. The second iteration finds an augmenting path from unmatched user  $u_6$  to unmatched item  $i_6$ , i.e.,  $P=(u_6, i_2, u_1, i_6)$  and  $(u_6, i_2) \notin M, (u_1, i_2) \in M, (u_1, i_6) \notin M$ . As a result, user  $u_6$  is then matched to item  $i_2$ , and user  $u_1$ , previously matched to item  $i_2$ , is now matched to new item  $i_6$ , which leads to the maximum cardinality for this example (i.e., max aggregate diversity of 6 items), and the iterations for searching augmenting paths stop.

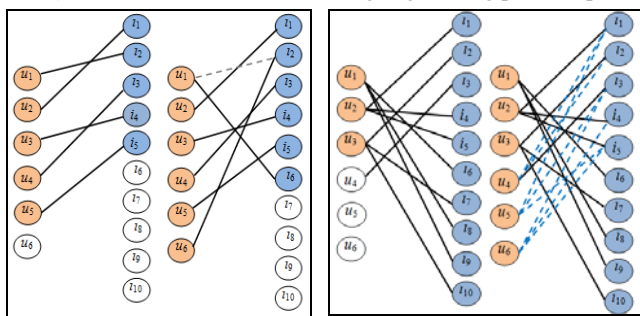


Figure 4. Illustration of Graph-Based Approach.

On the other hand, in case of top-3 recommendations for the same example (Fig. 4b), while maximum diversity (i.e., 10) is reached in Step 1, three users ( $u_4, u_5, u_6$ ) are matched to fewer than 3 items. Thus, as shown in Step 2 of Fig. 4b, the remaining top-3 recommendations are filled with the most highly predicted items among the items available for users.

Note that the sequence in which users and/or items are chosen to be evaluated in Fig. 3 may have implications on the runtime of the algorithm. E.g., finding more augmenting paths (and, therefore, a larger matching) in the first iteration may reduce the total number of iterations needed to reach the maximum matching. We found that applying a simple heuristic of first choosing users for matching who have the smallest number of remaining candidate items leads to substantial runtime improvements, because of the smaller likelihood that the items matched to those users can be replaced by other items, thus, reducing the number of iterations.

As mentioned earlier, for Step 1, we adopt the Hopcroft-Karp algorithm [20], which is known to be among the most efficient algorithms for maximum bipartite matching, having complexity of  $O(E\sqrt{V})$ , where  $E$  is the number of edges in the graph and  $V$  is the number of vertices on the left side of the graph (i.e., the number of users in our case) [13]. In a bipartite graph with  $m$  user vertices,  $n$  item vertices, and a maximum of  $mn$  edges, the complexity of the Hopcroft-Karp algorithm would be  $O(mn\sqrt{m})$ , and by adding the standard ranking approach for Step 2, the total complexity of the max flow based approach for top-1 recommendation tasks would be  $O(mn\log n + mn\sqrt{m})$ . For top- $N$  recommendation tasks, we allow multiple edges from a single

user vertex. We propose an efficient extension of bipartite matching algorithm for top- $N$  recommendations, as discussed earlier; however, in the worst case, the top- $N$  recommendation task can be treated as top-1 task with  $Nm$  users and, correspondingly,  $Nmn$  edges. Even this worst-case extension for top- $N$  recommendations does not change the complexity, i.e.,  $O(Nmn\sqrt{Nm}) = O(mn\sqrt{m})$ , assuming  $N$  (i.e., the number of recommendations provided to each user) is a relatively small, bounded constant. Therefore, this graph-based approach is more complex than the re-ranking heuristic, which had worst case complexity of  $O(mn\log n)$ , as mentioned earlier.

## 4. EMPIRICAL RESULTS

In our experimental evaluation, we used two movie rating datasets: MovieLens (data file available at grouplens.org) and Netflix (used for Netflix Prize competition). Each dataset is pre-processed to include users and movies with significant rating histories, which makes it possible to have a large number of highly predicted items available for recommendations to each user, thus, potentially making the diversity maximization task more challenging. The basic statistical information of the resulting datasets is as follows. MovieLens dataset has 775,176 ratings with 2,830 users and 1,919 items (i.e., 14.27% sparsity), and Netflix dataset has 1,067,999 ratings with 3,333 users and 2,091 items (i.e., 15.32% sparsity). For each dataset, we learn from all of the known ratings and predict the unknown ratings (85.73% of the whole user-item matrix in the MovieLens dataset and 84.68% in the Netflix dataset). As discussed earlier, we use three popular collaborative filtering techniques (user-based, item-based, and matrix factorization CF techniques), and top- $N$  ( $N=1, 5, 10$ ) items are recommended for each user.

We predict unknown ratings based on all known ratings, where a relatively large number of highly-predicted (i.e., with the predicted rating value above 3.5) candidate items are available for all users (typically around 500-800 items for each user). Fig. 5 presents a number of representative results obtained from the empirical evaluation, which shows not only the accuracy and diversity capabilities of the proposed approach in terms of top- $N$  recommendation, but also compares it with two baseline techniques that re-rank the candidate items by their reverse predicted rating values [3] and at random, as well as with the standard recommendation technique. As expected, the standard recommendation technique (i.e., recommending items with highest predicted ratings) represents the most accurate, but very non-diverse set of recommendations. In Fig. 5, the representative accuracy-diversity curves for the baseline random and re-ranking techniques and for graph-based approach were obtained by using different ranking and flow-rating thresholds (3.5, 3.6, ..., 5).

One notable finding is that, while the simple re-ranking technique shows the same or slightly better results than the random approach, the proposed graph-based approach is able to obtain substantial diversity improvements at the given level of accuracy, compared to the two baseline techniques, across all experiments including different datasets, different recommendation techniques, and different number of recommendations ( $N=1, 5, 10$ ).

Another notable result is that, as  $N$  increases, significant diversity improvements can be obtained with increasingly smaller sacrifices to recommendation accuracy. For example, in top-1 recommendation tasks, the graph-based approach was able to obtain the maximum possible diversity with a decrease of about 0.5 (on scale 1-5) in an average prediction. However, for top-5

tasks the accuracy decrease needed for maximum diversity was about 0.1, and for top-10 tasks only about 0.05. Table 1 further illustrates this point by showing the diversity gains of random, re-ranking, and graph-based approaches at three different accuracy loss levels (0.1 for top-1 tasks, 0.05 for top-5 tasks, 0.01 for top-10 tasks). In summary, the proposed graph-based approach was able to consistently provide substantial diversity improvements for all traditional recommendation algorithms (user-based, item-based, and matrix factorization CF) on different real-world recommendation datasets.

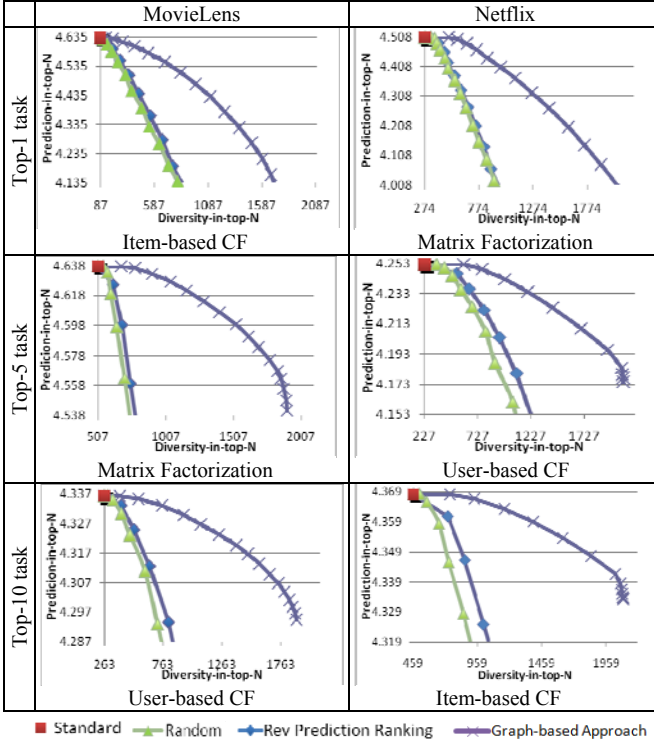


Figure 5. Performance of Ranking and Optimization Approaches

As discussed earlier, the performance improvements for the proposed technique come at the cost of computational complexity, which can become an issue as the data size increases. To complement the earlier discussion on the theoretical computational complexity, here we report how the data size affects the actual runtime. We vary the size of data by changing the number of candidate items that are available for recommendation to each user. For example, for the datasets used in our experiments we treated all items that were predicted above rating threshold  $T_H = 3.5$  as potential candidates for recommendation. By increasing this threshold we can eliminate some candidate items across all users, thus, obtaining smaller datasets. Following this approach, we generated six datasets  $D_1, \dots, D_6$  of increasing size from MovieLens dataset by using different rating thresholds ( $D_1$  for  $T_H = 4.5$ ,  $D_2$  for 4.3,  $D_3$  for 4.1,  $\dots, D_6$  for 3.5), as indicated in Fig. 6a.

We measured the runtime of the two algorithms (i.e., the proposed approach and the re-ranking approach) on the same computer. The obtained results are consistent across different recommendation algorithms, different datasets, and top- $N$  tasks (for different  $N$  values). Fig. 6b illustrates the general trends by presenting the example runtimes of the simple recommendation re-ranking and the graph-based approach on the MovieLens dataset, for generating diverse top-1 recommendations using item-

based CF technique. As expected, as the data size increases, the simple re-ranking heuristic demonstrates good scalability, while the more complex graph-based approach requires increasingly more time (while also generating better recommendation outcomes, as discussed earlier). We do observe that, for our medium-size recommendation setting (with approx. 3,000 users and 2,000 items), the proposed approach demonstrated good computational performance; even running it on the largest dataset ( $D_6$ ) took less than 1.5 minutes.

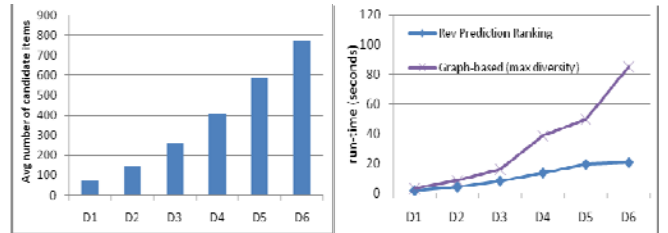
Table 1. Diversity Gains at the Given Accuracy Level

	User-based CF	Item-based CF	MF
Top-1 recommendation task (Accuracy level: Standard - 0.1)			
Standard	98, acc=4.40	87, acc=4.63	247, acc=4.73
Random	368.7 (276.2%)	272.6 (213.3%)	390.3 (58.0%)
Re-Ranking	409.1 (317.4%)	300.8 (245.7%)	412.3 (66.9%)
Graph-Based	826.0 (742.9%)	748.6 (760.4%)	927.5 (275.5%)
Top-5 recommendation task (Accuracy level: Standard - 0.05)			
Standard	190, acc=4.36	200, acc=4.57	507, acc=4.64
Random	581.6 (206.1%)	385.1 (92.6%)	659.3 (30.0%)
Re-Ranking	648.1 (241.1%)	424.5 (112.3%)	698.1 (37.7%)
Graph-Based	1562.9 (722.6%)	1415.9 (607.9%)	1647.7 (225.0%)
Top-10 recommendation task (Accuracy level: Standard - 0.01)			
Standard	263, acc=4.34	279, acc=4.53	667, acc=4.58
Random	448.6 (70.6%)	354.9 (27.2%)	745.0 (11.7%)
Re-Ranking	497.4 (89.1%)	385.7 (38.3%)	794.3 (19.1%)
Graph-Based	1107.0 (320.9%)	978.7 (250.8%)	1408.2 (111.1%)

(a) MovieLens data

	User-based CF	Item-based CF	MF
Top-1 recommendation task (Accuracy level: Standard - 0.1)			
Standard	67, acc=4.31	142, acc=4.51	274, acc=4.51
Random	416.6 (521.8%)	379.1 (167.0%)	484.1 (76.7%)
Re-Ranking	417.5 (523.1%)	420.2 (195.9%)	505.4 (84.5%)
Graph-Based	764.5 (1041.1%)	842.6 (493.4%)	967.8 (253.2%)
Top-5 recommendation task (Accuracy level: Standard - 0.05)			
Standard	227, acc=4.25	335, acc=4.42	561, acc=4.43
Random	822.5 (262.3%)	671.5 (100.4%)	904.8 (61.3%)
Re-Ranking	943.4 (315.6%)	754.4 (125.2%)	966.8 (72.3%)
Graph-Based	1829.1 (705.8%)	1795.8 (436.1%)	2000.9 (256.7%)
Top-10 recommendation task (Accuracy level: Standard - 0.01)			
Standard	341, acc=4.21	459, acc=4.37	771, acc=4.39
Random	736.1 (115.9%)	655.7 (42.9%)	1046.8 (35.8%)
Re-Ranking	876.6 (157.1%)	750.0 (63.4%)	1193.2 (54.8%)
Graph-Based	1528.9 (348.3%)	1426.3 (210.7%)	2456.1 (218.6%)

(b) Netflix data



(a) Avg Number of Candidate Items Per User (b) Runtime  
MovieLens dataset, Item-based CF, Top-1 recommendation task

Figure 6. Different Datasets and Algorithmic Runtime

## 5. CONCLUSIONS AND FUTURE WORK

Recommendation diversity recently has attracted attention as an important aspect in evaluating the quality of recommendations. Traditional recommender systems typically recommend the top- $N$  most highly predicted items for each user, thereby providing good predictive accuracy, but performing poorly with respect to

recommendation diversity. This paper extends prior work by developing a more sophisticated graph-theoretic approach that models the diversity maximization problem as a network flow maximization or bipartite matching maximization problems and provides significant advantages over the recommendation re-ranking approaches in terms of the accuracy/diversity tradeoff.

The proposed optimization approaches have been designed specifically for the diversity-in-top- $N$  metric, i.e., the number of distinct items among top- $N$  recommendations. The extension of these approaches to more sophisticated diversity metrics, including relative long-tail metrics such as Gini coefficient [17] and the long-tail shape parameter such as the slope of the log-linear relationship between popularity and recommendations, represent a promising direction for future research. Another interesting and important direction would be to investigate whether the use of the diversity-maximizing recommendation algorithms can truly lead to an increase in sales diversity and user satisfaction. In particular, as discussed in recent research [12,25], it would be valuable to examine the impact of recommendations on long-tail phenomena in different categories of users and products and possibly propose different algorithms based on the appropriate categorization. We believe that this work provides insights into developing new recommendation techniques that can consider multiple aspects of recommendation quality, going beyond using just the accuracy measures.

## 6. ACKNOWLEDGMENTS

The research reported in this paper was supported in part by the US National Science Foundation CAREER Grant IIS-0546443.

## 7. REFERENCES

- [1] Adomavicius, G., A. Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE TKDE* 17:6 734-749.
- [2] Adomavicius, G., Y. Kwon. 2009. Toward More Diverse Recommendations: Item Re-Ranking Methods for Recommender Systems. *Proc. of the 19th Workshop on Information Technologies and Systems*.
- [3] Adomavicius, G., Y. Kwon. 2011. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering*. Forthcoming.
- [4] Aggarwal, C.C., J.L. Wolf, K.L. Wu, P.S. Yu. 1999. Horting Hatches An Egg: A New Graph-Theoretic Approach to Collaborative Filtering. *Proc. of the 5th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD '99)*. 201-212.
- [5] Ahuja, R. K., T. L. Magnanti, J. B. Orlin, 1993. *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice-Hall.
- [6] Anderson, C. 2006. *The Long Tail*. New York: Hyperion.
- [7] Balabanovic, M., Y. Shoham. 1997. Fab: Content-Based, Collaborative Recommendation. *Comm. of the ACM* 40:3 66-72.
- [8] Bradley, K., B. Smyth. 2001. Improving Recommendation Diversity. *Proc. of the 12th Irish Conf. on Artif. Intelligence and Cognitive Sci.*
- [9] Breese, S., D. Heckerman, C. Kadie. 1998. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proc. of the 14th Conf. on Uncertainty in Artificial Intelligence*.
- [10] Brynjolfsson, E., M.D. Smith, Y.J. Hu. 2003. Consumer Surplus in the Digital Economy: Estimating the value of increased product variety at online booksellers. *Management Sci.* 49:11 1580-1596.
- [11] Brynjolfsson, E., Y.J. Hu, D. Simester. 2007. Goodbye Pareto Principle, Hello Long Tail: The Effect of Search Costs on the Concentration of Product Sales. *NET Institute*.
- [12] Brynjolfsson, E., Y.J. Hu, M.D. Smith. 2010. Long Tails vs. Superstars: The Effect of Information Technology on Product Variety and Sales Concentration Patterns. *Information Systems Research* 21:4 736-347.
- [13] Burkard R., M. Dell'Amico, S. Martello. 2009. Assignment Problems. *Society for Industrial and Applied Mathematics (SIAM)*.
- [14] Cormen, T.H., C.E. Leiserson, R.L. Rivest, C. Stein. 2001. *Introduction to Algorithms*, MIT Press.
- [15] Fleder, D., K. Hosanagar. 2009. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity. *Management Science* 55:5 697-712.
- [16] Funk, S. 2006. Netflix Update: Try This At Home. <http://sifter.org/~simon/journal/20061211.html>.
- [17] Gini, C. 1921. Measurement of Inequality and Incomes. *Economic Journal* 31 124-126.
- [18] Goldstein, D.G., D.C. Goldstein. 2006. Profiting from the Long Tail. *Harvard Business Review*, Jun 2010.
- [19] Herlocker, J.L., J.A. Konstan, L.G. Terveen, J. Riedl. 2004. Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems* 22:1 5-53.
- [20] Hopcroft, J.E., R.M. Karp. 1973. An  $n^{5/2}$  Algorithm for Maximum Matchings in Bipartite Graphs, *SIAM J. on Computing* 2:4 225-231.
- [21] Hu, R., P. Pu. 2011. Enhancing Recommendation Diversity with Organization Interfaces. *Proc. of the 16th Int'l Conf. on Intelligent User Interfaces (IUI '11)*. 347-350.
- [22] Huang, Z., D. Zeng, H. Chen. 2007. Analyzing Consumer-product Graphs: Empirical Findings and Applications in Recommender Systems. *Management Science* 53:7 1146-1164.
- [23] Kim, H.K., J.K. Kim, Y. Ryu. 2010. A Local Scoring Model for Recommendation. *Proc. of the 20th Workshop on Information Technologies and Systems (WITS'10)*.
- [24] Koren, Y., R. Bell, C. Volinsky. 2009. Matrix Factorization Techniques For Recommender Systems. *IEEE Computer Society*, 42 30-37.
- [25] Lee, J., J.N. Lee, H. Shin. 2011. The Long Tail or the Short Tail: The Category-Specific Impact of eWOM on Sales Distributions. *Decision Support Systems* 51:3 466-479.
- [26] Leonard, D. 2010. Tech Entrepreneur Peter Gabriel Knows What You Want. *Business Week*, April.
- [27] Levy, M., K. Bosteels. 2010. Music Recommendation and the Long Tail. Workshop on Music Recommendation and Discovery, *ACM Intl. Conf. on Recommender Systems*.
- [28] Liu, J., M. Shang, D. Chen. 2009. Personal Recommendation Based on Weighted Bipartite Networks. *Proc. of the 6th Intl. Conf. on Fuzzy Systems and Knowledge Discovery* 134-137.
- [29] McNee, S.M., J. Riedl, J.A. Konstan. 2006. Being Accurate is Not Enough: How Accuracy Metrics have hurt Recommender Systems. *Conf. on Human Factors in Computing Systems* 1097-1101.
- [30] McSherry, D. 2002. Diversity-Conscious Retrieval. *Proc. of the 6th European Conf. on Advances in Case-Based Reasoning* 219-233.
- [31] Oestreicher-Singer, G., A. Sundararajan. 2011. Recommendation Networks and the Long Tail of Electronic. *MIS Quarterly*. Forthcoming.
- [32] Park, Y.J., A. Tuzhilin. 2008. The Long Tail of Recommender Systems and How to Leverage It. *Proc. of the 2nd ACM Conf. on Recommender Systems* 11-18.
- [33] Sarwar, B., G. Karypis, J.A. Konstan, J. Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. *Proc. of the 10th Intl. World Wide Web Conf.*
- [34] Shani G., A. Gunawardana. 2011. Evaluating Recommendation Systems, in P. B. Kantor, F. Ricci, L. Rokach, B. Shapira (Eds.), *Recommender Systems Handbook: A Complete Guide for Research Scientists and Practitioners*, Chapter 8, Springer.
- [35] Smyth, B., P. McClave. 2001. Similarity vs. Diversity. *Proc. of the 4th Intl. Conf. on Case-Based Reasoning*.
- [36] Thompson, C. 2008. If You Liked This, You're Sure to Love That. *The New York Times*. Nov 2008. <http://www.nytimes.com/2008/11/23/magazine/23Netflix-t.html>.
- [37] Zhang, M. 2009. Enhancing Diversity in Top- $N$  Recommendation. *Proc. of the 3rd ACM Conf. on Recommender Systems* 397-400.
- [38] Zhang, M., N. Hurley. 2008. Avoiding monotony: improving the diversity of recommendation lists. *Proc. of the 2nd ACM Conf. on Recommender Systems* 123-130.
- [39] Ziegler, C.N., S.M. McNee, J.A. Konstan, G. Lausen. 2005. Improving Recommendation Lists Through Topic Diversification, *Proc. of the 14th Intl. World Wide Web Conf.* 22-32.

# On Unexpectedness in Recommender Systems: Or How to Expect the Unexpected

Panagiotis Adamopoulos and Alexander Tuzhilin  
Department of Information, Operations and Management Sciences  
Leonard N. Stern School of Business, New York University  
{padamopo, atuzhili}@stern.nyu.edu

## ABSTRACT

Although the broad social and business success of recommender systems has been achieved across several domains, there is still a long way to go in terms of user satisfaction. One of the key dimensions for improvement is the concept of *unexpectedness*. In this paper, we propose a model to improve user satisfaction by generating unexpected recommendations based on the utility theory of economics. In particular, we propose a new concept of unexpectedness as recommending to users those items that depart from what they expect from the system. We define and formalize the concept of unexpectedness and discuss how it differs from the related notions of novelty, serendipity and diversity. We also measure the quality of recommendations using specific metrics under certain utility functions. Finally, we provide unexpected recommendations of high quality and conduct several experiments on a “real-world” dataset to compare our recommendation results with some other standard baseline methods. Our proposed approach outperforms these baseline methods in terms of unexpectedness while avoiding accuracy loss.

## Keywords

Recommender Systems, Unexpectedness, Utility Theory

*“If you do not expect it, you will not find the unexpected, for it is hard to find and difficult”.*

- Heraclitus of Ephesus, 544 - 484 B.C.

## 1. INTRODUCTION

Over the last decade, a wide variety of different types of recommender systems (RSs) has been developed and used across several domains [4]. Although the broad-based social and business acceptance of RSs has been achieved and the recommendations of the latest class of systems are significantly more accurate than they used to be a decade ago [6], there is still a long way to go in terms of satisfaction of the users’ actual needs. This is due, primarily, to the fact that many existing RSs focus on providing more accurate rather than more novel, serendipitous, diverse and useful recommendations. Some of the main problems pertaining to the narrow accuracy-based focus of many existing RSs and the ways to broaden the current approaches have been discussed in [19].

One of the key dimensions for improvement in RSs that can significantly contribute to the overall performance and usefulness of recommendations and that is still under-explored is the notion of “unexpectedness”. RSs often recommend items that the users are already familiar with and, thus, they are of little interest to them. For example, a shopping RS may recommend to customers

products such as milk and bread. Although being an accurate recommendation in the sense that the customer will indeed buy these two products, this recommendation is of little interest to the shopper because it is an obvious one: the shopper will, most likely, buy these products even without this recommendation. Therefore, motivated by the potential of higher user satisfaction, the difficulty of the problem and its implications, we try to resolve this problem of recommending items with which the users are already familiar, by recommending *unexpected* items of significant usefulness to them.

Following the Greek philosopher Heraclitus, we approach this hard and difficult problem of finding and recommending the unexpected items by first capturing expectations of the user. The challenge is not only to identify the set of items expected by the user and then derive the unexpected ones but also to enhance the concept of unexpectedness while still delivering recommendations of high quality and achieving a fair match of user's interests.

In this paper, we formalize this concept by providing a new formal definition of unexpected recommendations and differentiating it from various related concepts. We also suggest specific metrics to measure both unexpectedness and quality of recommendations. Finally, we propose a method for generating unexpected recommendations and evaluate the results of the proposed approach.

## 2. RELATED WORK AND CONCEPTS

In the past, several researchers tried to provide alternative definitions of unexpectedness and various related but still different concepts, such as recommendations of novel, diverse and serendipitous items. In particular, *novel* recommendations are recommendations of those items that the user did not know about [17]. Hijikata et al. in their work [14] use collaborative filtering to derive novel recommendations by explicitly asking users what items they already know and Weng et al. [28] suggest a taxonomy-based RS that utilizes hot topic detection using association rules to improve novelty and quality of recommendations. However, comparing novelty to unexpectedness, a novel recommendation might be unexpected but novelty is strictly defined in terms of previously unknown non-redundant items without allowing for known but unexpected ones. Also, novelty does not include positive reactions of the user to recommendations. Illustrating these differences in the movie context, assume that user John Doe is mainly interested in Action & Adventure films. Recommending the latest popular Children & Family film to this user is definitely a novel recommendation but probably of low utility for him since Children & Family films are not included in his preferences and will be likely considered “irrelevant” because they depart too much from his expectations.

Moreover, *serendipity*, the most closely related concept to unexpectedness, involves a positive emotional response of the user about a previously unknown (novel) item and measures how

surprising these recommendations are [24]; serendipitous recommendations are by definition also novel. Iaquinta et al. propose in [15] to enhance serendipity by recommending novel items whose description is semantically far from users’ profiles and Kawamae et al. [16] suggest an algorithm for recommending novel items based on the assumption that users follow the earlier adopters who have demonstrated similar preferences but purchased items earlier. Nevertheless, even though both serendipity and unexpectedness involve positive surprise of the user, serendipity is restricted just to novel items without taking consideration of users’ expectations and relevance of the items. To further illustrate the differences of these two concepts, let’s assume that we recommend to John Doe the newly released production of his favorite Action & Adventure film director. Although John will probably like the recommended item, such a serendipitous recommendation does not maximize his utility because John was probably expecting the release of this film or he could easily find out about it.

Furthermore, *diversification* is defined as the process of maximizing the variety of items in our recommendation lists. Most of the literature in RSs and Information Retrieval including [2], [3], [26] and [27] studies the principle of diversity to improve user satisfaction. Typical approaches replace items in the derived recommendation lists to minimize similarity between all items or remove “obvious” items from them as in [8]. Adomavicius and Kwon [2], [3] address the concept of aggregated diversity as the ability of a system to recommend across all users as many different items as possible over the whole population while keeping accuracy loss to a minimum, by a controlled promotion of less popular items towards the top of the recommendation lists. Even though avoiding a too narrow set of choices is generally a good approach to increase the usefulness of the final list, since it enhances the chances that the user is pleased by at least some recommended items, diversity is a very different concept from unexpectedness and constitutes an ex-post process that can actually be combined with our model of unexpectedness.

Pertaining to *unexpectedness*, in the field of knowledge discovery, [22] and [23] proposed a characterization of unexpectedness relative to the system of prior domain beliefs and developed efficient algorithms for the discovery of unexpected patterns, which combined the independent concepts of unexpectedness and minimality of patterns. In the field of recommender systems, Murakami et al. [20] and Ge et al. [11] suggested both a definition of unexpectedness as the deviation from the results obtained from a primitive prediction model and metrics for evaluating unexpectedness and serendipity. Also, Akiyama et al. [5] proposed unexpectedness as a general metric that does not depend on a user’s record and involves an unlikely combination of features. However, all these approaches do not fully capture the multi-faceted concept of unexpectedness since they do not truly take into account the actual *expectations of the users*, which is crucial according to philosophers, such as Heraclitus, and some modern researchers [22], [23]. Hence an alternative definition of unexpectedness, taking into account prior expectations of the user, and methods for providing unexpected recommendations are still needed. In this paper, we deviate from the previous definitions of unexpectedness and propose a new formal definition as recommending to users those items that depart from what they expect from the RS.

Based on the previous definitions and the discussed similarities and differences, the concepts of novelty, serendipity and unexpectedness are overlapping. Obviously, all these entities are linked to a notion of discovery, as a recommendation makes more

sense when it exposes the user to a relevant experience that he/she has not thought of or found yet. However, the part of novelty and serendipity that adds to the usefulness of recommending a specific product can be captured by unexpectedness. This is because unexpectedness includes the positive reaction of a user to recommendations about previously unknown items but without being strictly restricted only to novel items and also because unexpectedness avoids recommendations of items that are obvious, irrelevant and expected to the user.

### 3. DEFINITION OF UNEXPECTEDNESS

In this section, we formally model and define the concept of unexpected recommendations as those recommendations that significantly depart from the user’s expectations. However, unexpectedness alone is not enough for providing truly useful recommendations since it is possible to deliver unexpected recommendations but of low quality. Therefore, after defining *unexpectedness*, we introduce *utility* of a recommendation as a function of recommendation *quality* (specified by item’s rating) and its *unexpectedness*. We maintain that this utility of a recommended item is the concept on which we should focus (vis-à-vis “pure” unexpectedness) by recommending items with the highest levels of utility to the user. Finally, we propose measures for evaluating the generated recommendations. We define unexpectedness in Section 3.1, the utility of recommendations in Section 3.2 and metrics for their evaluation in Section 3.3.

#### 3.1 Unexpectedness

To define unexpectedness, we start with user expectations. The *expected items* for each user  $u$  can be defined as a collection of items that the user is thinking of as serving his/her own current needs or fulfilling his/her intentions indicated by visiting the recommender system. This set of expected items  $E_u$  for a user can be specified in various ways, such as the set of past transactions performed by the user, or as a set of “typical” recommendations that he/she expects to receive. For example, in case of a movie RS, this set of expected items may include all the movies seen by the user and all their related and similar movies, where “relatedness” and “similarity” are formally defined in Section 4.

Intuitively, an item included in the set of expected movies derives “zero unexpectedness” for the user, whereas the more an item departs from the set of expectations, the more unexpected it is until it starts being perceived as irrelevant by the user. Unexpectedness should thus be a positive, unbounded function of the distance of this item from the set of expected items. More formally, we define *unexpectedness* in recommender systems as follows. First, we define:

$$\delta_{u,i} = d(i; E_u) \quad (1)$$

where  $d(i; E_u)$  is the distance of item  $i$  from the set of expected items  $E_u$  for user  $u$ . Then, *unexpectedness* of item  $i$  with respect to user expectations  $E_u$  is defined as some unimodal function  $\Delta$  of this distance:

$$\Delta(\delta_{u,i}; \delta_u^*) \quad (2)$$

where  $\delta_u^*$  is the best (most preferred) unexpected distance from the set of expected items  $E_u$  for user  $u$  (the mode of distribution  $\Delta$ ). Intuitively, unimodality of this function  $\Delta$  indicates that (a) there is only one *most preferred unexpected* distance, (b) an item that greatly departs from user’s expectations, even though results in a big departure from expectations, will be probably perceived as irrelevant by the user and, hence, it is not truly unexpected, and (c) items that are close to the expected set are not truly unexpected but rather obvious to the user.

However, recommending the items that result in the highest possible level of unexpectedness would be unreasonable and problematic since recommendations should be of high quality and fairly match users' preferences; otherwise the users might be dissatisfied with the RS. In order to generate recommendations of high quality that would maximize the users' satisfaction, we use certain concepts from the utility theory in economics [18].

### 3.2 Utility of Recommendations

In the context of recommender systems, we specify the utility of a recommendation of an item to a user in terms of two components: the utility of quality that the user will gain from using the product (as defined by its rating) and the utility of unexpectedness of the recommended item, as defined in Section 3.1. Our proposed model assumes that the users are engaging into optimal utility maximizing behavior [18]. Additionally to the assumptions made in Section 3.1, we further assume that, given the unexpectedness of an item, the greater the rating of this item, the greater the utility of the recommendation to the user.

Consequently, without loss of generality, we propose that we can estimate this overall utility of a recommendation using the previously mentioned utility of quality and the loss in utility by the departure from the preferred level of unexpectedness  $\delta_u^*$ . This will allow the utility function to have the required characteristics described so far. Note that the distribution of utility as a function of unexpectedness and rating is non-linear, bounded and experiences a global maximum.

Formalizing these concepts, we assume that each user  $u$  values the quality of an item by a constant  $q_u$  and that the quality of the item  $i$  is represented by the corresponding rating  $r_{u,i}$ . Then, we define utility derived from the quality of the recommended item  $i$  to the user  $u$  as:

$$U_{q_{u,i}} = q_u * r_{u,i} + \varepsilon_{u,i}^q \quad (3)$$

where  $\varepsilon_{u,i}^q$  is the error term defined as a random variable capturing the stochastic aspect of recommending item  $i$  to user  $u$ .

Correspondingly, we assume that each user values the unexpectedness of an item by a factor  $\lambda_u$ ;  $\lambda_u$  being interpreted as user's tolerance to redundancy and irrelevance. The user losses in utility by departing from the preferred level of unexpectedness  $\delta_u^*$ . Then, the utility of the unexpectedness of a recommendation can be represented as follows:

$$U_{\delta_{u,i}} = -\lambda_u * \varphi(\delta_{u,i}; \delta_u^*) + \varepsilon_{u,i}^\delta \quad (4)$$

where function  $\varphi$  captures the departure of unexpectedness of item  $i$  from the preferred level of unexpectedness  $\delta_u^*$  for the user  $u$  and  $\varepsilon_{u,i}^\delta$  is the error term of the specific user and item.

Thus, the utility of recommending an item to a user can be computed as the sum of functions (3) and (4):

$$U_{u,i} = U_{q_{u,i}} + U_{\delta_{u,i}} \quad (5)$$

$$U_{u,i} = q_u * r_{u,i} - \lambda_u * \varphi(\delta_{u,i}; \delta_u^*) + \varepsilon \quad (6)$$

where  $\varepsilon$  is the stochastic error term.

Function  $\varphi$  can be defined in various ways. For example, using popular location models for horizontal and vertical differentiation of products in economics [10], [21] and [25], the departure of the preferred level of unexpectedness can be defined as the linear distance:

$$U_{u,i} = q_u * r_{u,i} - \lambda_u * |\delta_{u,i} - \delta_u^*| \quad (7)$$

or the quadratic one:

$$U_{u,i} = q_u * r_{u,i} - \lambda_u * (\delta_{u,i} - \delta_u^*)^2. \quad (8)$$

Note that the usefulness of a recommendation is linearly increasing with the ratings for these distances. Whereas, given the

rating of the product, the usefulness of a recommendation increases with unexpectedness up to the threshold of the preferred level of unexpectedness  $\delta_u^*$ . This threshold  $\delta_u^*$  is specific for each user and context. It should also be obvious by now, that two recommended items with different ratings and distances from the set of expected items may derive the same levels of usefulness.

Once the utility function  $U_{u,i}$  is defined, we can then make recommendations to user  $u$  by selecting items  $i$  having the highest values of utility  $U_{u,i}$ .

### 3.3 Evaluation of Recommendations

[4], [13] and [19] suggest that recommender systems should be evaluated not only by their accuracy, but also by other important metrics such as coverage, novelty, serendipity, unexpectedness and usefulness. Hence, we suggest specific measures to evaluate the candidate items and the generated recommendation lists.

#### 3.3.1 Measures of Unexpectedness

Our approach regards unexpectedness of the recommended item as a component of the overall user satisfaction. Therefore, we should evaluate the proposed method for the resulting unexpectedness of the derived recommendation lists.

In order to measure unexpectedness, we follow the approach proposed by Murakami et al. [20] and Ge et al. [11], and adapt their measures to our method. In particular, [11] defines an unexpected set of recommendations (UNEXP) as:

$$UNEXP = RS \setminus PM \quad (9)$$

where PM is a set of recommendations generated by a primitive prediction model, such as predicting items based on users' favorite categories or items' number of ratings, and RS denotes the recommendations generated by a recommender system. When an element of RS does not belong to PM, they consider this element to be unexpected.

As the authors maintain, based on their definition of unexpectedness, unexpected recommendations may not be always useful and, thus, they also introduce serendipity measure as:

$$SRDP = \frac{|UNEXP \cap USEFUL|}{|N|} \quad (10)$$

where USEFUL denotes the set of "useful" items and N the length of the recommendation list. For instance, the usefulness of an item can be judged by the users or approximated by the items' ratings as described in Section 4.2.5.

However, these measures do not fully capture our definition of unexpectedness since PM contains the most popular items and does not actually take into account the expectations of the user. Consequently, we revise their definition and introduce our own metrics to measure unexpectedness as follows.

First of all, we define expectedness (EXPECTED) as the mean ratio of the movies which are included in both the set of expected movies for a user and the generated recommendation list:

$$EXPECTED = \sum_u \frac{|RS_u \cap E_u|}{|N|}. \quad (11)$$

Furthermore, we propose a metric of unexpectedness (UNEXPECTED) as the mean ratio of the movies that are not included in the set of expected movies for the user and are included in the generated recommendation list:

$$UNEXPECTED = \sum_u \frac{|RS_u \setminus E_u|}{|N|}. \quad (12)$$

Correspondingly, we can also derive a new metric for serendipity as in (10) based on the proposed metric of unexpectedness (12).

Finally, recommendation lists should also be evaluated for the catalog coverage. The catalog coverage of a recommender describes the area of choices for the users and measures the domain of items over which the system can make recommendations [13].

### 3.3.2 Measures of Accuracy

The recommendation lists should also be evaluated for the accuracy of rating and item prediction.

(i) *Rating prediction*: The Root Mean Square Error (RMSE) is perhaps the most popular measure of evaluating the accuracy of predicted ratings:

$$\text{RMSE} = \sqrt{\frac{1}{|R|} \sum_{(u,i) \in R} (\hat{r}_{u,i} - r_{u,i})^2} \quad (13)$$

where  $\hat{r}_{u,i}$  is the estimated rating and  $R$  is the set of user-item pairs  $(u, i)$  for which the true ratings  $r_{u,i}$  are known.

Another popular alternative is the Mean Absolute Error (MAE):

$$\text{MAE} = \sqrt{\frac{1}{|R|} \sum_{(u,i) \in R} |\hat{r}_{u,i} - r_{u,i}|} \quad (14)$$

(ii) *Item prediction*: We can classify all the possible results of a recommendation of an item to a user as in Table 1:

**Table 1. Classification of the possible result of a recommendation.**

	Recommended	Not Recommended
Used	True-Positive (tp)	False-Negative (fn)
Not Used	False-Positive (fp)	True-Negative (tn)

and compute the following popular quantities for item prediction:

$$\text{Precision} = \frac{\# tp}{\# tp + \# fp} \quad (15)$$

$$\text{Recall (True Positive Rate)} = \frac{\# tp}{\# tp + \# fn} \quad (16)$$

$$\text{False Positive Rate (1 - Specificity)} = \frac{\# fp}{\# fp + \# tn} \quad (17)$$

## 4. EXPERIMENTS

To empirically validate the method presented in Section 3 and evaluate unexpectedness of recommendations generated by this method, we conduct experiments on a “real-world” dataset and compare our results to popular Collaborative Filtering methods.

Unfortunately, we could not compare our results with other methods for deriving unexpected recommendations for the following reasons. Most of the existing methods are based on related but different principles such as diversity and novelty. Since these concepts are different from our definition, they cannot be directly compared with our approach. Further, among the previously proposed methods of unexpectedness that are consistent with our approach, as explained in Section 2, authors of these methods do not provide any clear computational algorithm for unexpected recommendations but metrics, thus making the comparison impossible. Consequently, we selected a number of standard collaborative filtering (CF) algorithms as baseline methods to compare with the proposed approach. In particular, we selected the  $k$ -nearest neighborhood approach (kNN), the Slope One (SO) algorithm and a matrix factorization (MF) approach.<sup>1</sup> We would like to point out that, although the selected CF methods do not explicitly support the notion of unexpectedness, they constitute fairly reasonable baselines because, as was pointed out in [9], CF methods perform reasonably well in terms of some

other performance measures besides classical accuracy measures, and indeed our empirical results reported in Section 5 confirm this general observation of [9] for unexpected recommendations.

### 4.1 Dataset

The basic dataset we used is the RecSys HetRec 2011 [1] MovieLens dataset. This is an extension of a dataset published by GroupLens research group [12], which contains personal ratings and tags about movies. This dataset consists of 855,598 ratings (0.5 - 5) from 2,113 users on 10,197 movies (on average about 405 ratings per user and 85 ratings per movie). In the dataset, the movies are linked to the Internet Movie Database (IMDb) and RottenTomatoes (RT) movie review systems. Each movie has its IMDb and RT identifiers, English and Spanish titles, picture URLs, genres, directors, actors (ordered by “popularity” per movie), RT audience’ and experts’ ratings and scores, countries, and filming locations. It also contains the tag assignments of the movies provided by each user. However, this dataset does not contain any demographic information about the users.

The selected dataset is relatively dense (3.97%) compared to other frequently used datasets (e.g. the original Netflix Prize dataset [7]) but we believe that this specific characteristic is a virtue that will let us better evaluate our methods since it allows us to better approximate the set of expected movies for each user.

In addition, we used information and further details from Wikipedia and the database of IMDb. Joining these datasets we were able to enhance the information included in our basic dataset by finding any missing values of the movie attributes that were mentioned above and, also, identifying whether a movie is an episode or sequel of another movie included in our dataset. We succeeded to identify *related* movies (i.e. episodes, sequels, movies with exact the same title) for 2,443 of our movies (23.95% of the movies with 2.18 related movies on average and a maximum of 22 “related” movies). We used this information about related movies to identify sets of expected movies, as described in Section 4.2.3.

### 4.2 Experimental Setup

We conducted in total 2160 experiments. In the one half of the experiments we explore the simpler case where the users are homogeneous (Hom) and have exactly the same preferences. In the other half, we investigate the more realistic case (Het) where users have different preferences that depend on their previous interactions with the system. Furthermore, we use two different sets of expected movies for each user, and different utilities functions. Also, we conducted experiments using different rating prediction algorithms, various measures of distance between movies and between a movie and the set of expected movies for each user. Finally, we derived recommendation lists of different sizes ( $k = \{10, 20, \dots, 100\}$ ). In conclusion, we used 2 sets of expected movies  $\times$  3 algorithms for rating prediction  $\times$  3 correlation metrics  $\times$  3 distance metrics  $\times$  2 utility functions  $\times$  2 assumptions about users preferences  $\times$  10 different lengths of recommendation lists, resulting in 2160 experiments in total.

#### 4.2.1 Utility of Recommendation

In our experiments, we considered the following utility functions:

(a1) *Homogeneous users with linear distance* (Hom-Lin): This is the simpler case where users are homogeneous and have similar preferences (i.e.  $q, \lambda, \delta^*$ ) and the departure of the preferred level of unexpectedness is linear as in function (7).

(a2) *Homogeneous users with quadratic distance* (Hom-Quad): The users are assumed to be homogeneous but the departure of the preferred level of unexpectedness is quadratic as in function (8).

<sup>1</sup> Various algorithms including baseline methods for rating prediction and matrix factorization with explicit user and item bias were tested with similar results.



(b1) *Heterogeneous users with linear distance* (Het-Lin): Here, the users are heterogeneous and have different preferences (i.e.  $q_u, \lambda_u, \delta_u^*$ ) and the departure of the preferred level of unexpectedness is linear. This case corresponds to function (7)

(b2) *Heterogeneous users with quadratic distance* (Het-Quad): This is the more realistic case. Users have different preferences and the departure of the preferred level of unexpectedness is quadratic. This case corresponds to function (8).

#### 4.2.2 Item Similarity

To build the set of expected movies, the system calculates the distance  $d$  between two movies by measuring the relevance of these movies. In our experiments, we use both collaborative-based and content-based similarity for the item distance.<sup>2</sup>

(i) The collaborative filtering similarity can be defined using (a) the Pearson correlation coefficient:

$$\rho_{i,j} = \frac{\sum_{u \in U(i,j)} ((r_{u,i} - \bar{r}_{u,i})(r_{u,j} - \bar{r}_{u,j}))}{\sqrt{\sum_{u \in U(i,j)} (r_{u,i} - \bar{r}_{u,i})^2 \sum_{u \in U(i,j)} (r_{u,j} - \bar{r}_{u,j})^2}}, \quad (18)$$

(b) the Cosine similarity:

$$\text{sim}(i,j) = \cos(\theta) = \frac{i \cdot j}{\|i\| \|j\|} = \frac{\sum_u r_{u,i} r_{u,j}}{\sqrt{\sum_u r_{u,i}^2} \sqrt{\sum_u r_{u,j}^2}} \quad (19)$$

and (c) the Jaccard coefficient:

$$J(i,j) = \frac{|A \cap B|}{|A \cup B|} \quad (20)$$

where A is the set of users who rated movie  $i$  and B the set of users who rated movie  $j$ .

(ii) The content based similarity of movies  $i$  and  $j$  is defined as:

$$\text{sim}(i,j) = \frac{\sum_{k=1}^n w_k * \sigma_{k,i,j}}{\sum_{k=1}^n w_k} \quad (21)$$

where movie  $i$  is represented by a vector of its attributes:

$$i = (a_1, a_2, \dots, a_n)$$

and  $\sigma_{k,i,j}$  is the similarity of the value of attribute  $k$  of the movie  $i$  with the corresponding value of this attribute for movie  $j$  and  $w_k$  the weight of this attribute.

#### 4.2.3 Expected Movies

We use the following two examples of definitions of *expected* movies in our study. The first set of expected movies ( $E_{u,Short}$ ) for user  $u$  follows a very strict user-specific definition of unexpectedness, as defined in Section 3. The profile of user  $u$  consists of the set of movies that he/she has already rated. In particular movie  $i$  is *expected* for user  $u$  if the user has already rated some movie  $j$  such that  $i$  has the same title or is an episode or sequel of movie  $j$ , where episode or sequel is identified as explained in Section 4.1. In our dataset, on average a user rated 405 movies and the number of expected movies per user is 586; augmenting the number of rated movies by 44.75%.

The second set of expected movies ( $E_{u,Long}$ ) follows a broader definition. It includes the first set plus a number of closely “related” movies ( $E_{u,Long} \supseteq E_{u,Short}$ ). In order to form the second set of expected movies we, also, use content-based similarity between movies. We first compute the attribute-specific distance between the values of each attribute (e.g. distance between the Comedy and Adventure genres) based on the similarity metrics and, then, use the weighted distance described

<sup>2</sup> Other measures such as the set correlation and conditional probabilities were tested with no significant differences.

in Section 4.2.2 for the attributes of each movie (i.e. language, genre, director, actor, country of filming and year of release) in order to compute the final distance between two movies.

More specifically for this second case, two movies are *related* if at least one of the following conditions holds: (i) they were produced by the same director, belong to the same genre and are released within an interval of 5 years, (ii) the same set of protagonists appear in both of them (where protagonist defined as actor with ranking in our dataset = {1, 2, 3}) and they belong to the same genre, (iii) the two movies share more than twenty common tags, are in the same language and their correlation metric is above a certain threshold  $\theta$  (Jaccard Coefficient ( $J$ ) > 0.50), (iv) there is a link from the Wikipedia article for movie  $i$  to the article for movie  $j$  and the two movies are sufficiently correlated ( $J > 0.50$ ) and (v) the content-based distance metric defined in this subsection is below a threshold  $\theta$  ( $d < 0.50$ ). The average size of the extended set of expected movies per user is 1127, thus increasing the size of rated movies by 178% (7% of the total number of movies).

#### 4.2.4 Distance from the Set of Expected Movies

We can then define the distance of movie  $i$  from the set of expected movies  $E_u$  for user  $u$  in various ways. For example, it can be determined by averaging the distances between the candidate item  $i$  and all the items included in set  $E_u$ :

$$d(i, E_u) = \frac{\sum_{j=1}^{|E_u|} d(i,j)}{|E_u|} \quad (22)$$

where  $d$  is defined as in Section 4.2.2. Another approach is based on the Hausdorff distance:

$$d(i, E_u) = \inf\{d(i,j) : j \in E_u\}. \quad (23)$$

Additionally, we also use the Centroid distance that is defined as the distance of an item  $i$  from the centroid point of the set of expected movies  $E_u$  for the user  $u$ .

#### 4.2.5 Measures of Unexpectedness and Accuracy

To evaluate our approach in terms of unexpectedness, we use the measures described in Section 3.3.1. For the primitive prediction model of (9) we used the top-N items with the highest average rating and the top-N items with the largest number of ratings in order to form the list of top-K items (where  $K=100$ ) which form our PM recommendation list.

Additionally, we introduce *expectedness*’ (EXPECTED’) as the mean ratio of the movies that are either included in the set of expected movies for a user or in the primitive prediction model and are also included in the generated recommendation list:

$$\text{EXPECTED}' = \sum_u \frac{|RS_u \cap (E_u \cup PM)|}{|N|}. \quad (24)$$

Correspondingly, we define *unexpectedness*’ (UNEXPECTED’) as the mean ratio of the movies that are neither included in the set of expected movies for users nor in the primitive prediction model and are included in the generated recommendation list:

$$\text{UNEXPECTED}' = \sum_u \frac{|RS_u \setminus (E_u \cup PM)|}{|N|}. \quad (25)$$

Based on the ratio of Ge et al. (10), we also use the metrics SERENDIPITY and SERENDIPITY’ to evaluate serendipitous recommendations in conjunction with the proposed measures of unexpectedness in (12) and (25), respectively. In our experiments, we consider an item to be useful if its average rating is greater than 3.0 ( $\text{USEFUL} = \{i : \hat{r}_i > 3.0\}$ ).

Finally, we evaluate the generated recommendations lists based on the coverage of our product base and accuracy of rating and item prediction using the metrics discussed in Section 3.3.

## 5. RESULTS

In order to estimate the parameters of preferences (i.e.  $q_u, \lambda_u$ ) we used models of multiple linear regression. In our experiments, the average  $q_u$  was 1.005. For the experiments with the first set of expected movies the average  $\lambda_u$  was 0.158 for the linear distance and 0.578 for the quadratic one. For the extended set of expected movies the average  $\lambda_u$  was 0.218 and 0.591, respectively. Furthermore, to estimate the preferred level of unexpectedness  $\delta_u^*$  for each user and distance metric, we used the average distance of rated movies from the set of expected movies; for the case of homogeneous users, we used the average value over all users.

The experiments conducted using the Hausdorff distance indicate inconsistent performance and sometimes, except for the metric of coverage, under-performed the standard CF methods. Henceforth we present the results only for the rest of the experiments.<sup>3</sup> We have to note that the experiments using heterogeneous users uniformly outperform those conducted under the assumption of homogeneous users. The most realistic case of heterogeneous users for the extended set of expectations outperformed all the other approaches including the standard CF methods in 99.08% of the conducted experiments. Also, it was observed that smaller sizes of recommendation lists resulted in constantly greater improvements.

### 5.1 Comparison of Coverage

For the first set of expected movies, in the case of homogeneous users (Hom-Short), the average coverage was increased by 36.569% and, in the case of heterogeneous users (Het-Short), by 108.406%. For the second set of expected movies, the average coverage was increased by 61.898% and 80.294% in the cases of homogeneous users (Hom-Long) and heterogeneous users (Het-Long), respectively (figure 1).

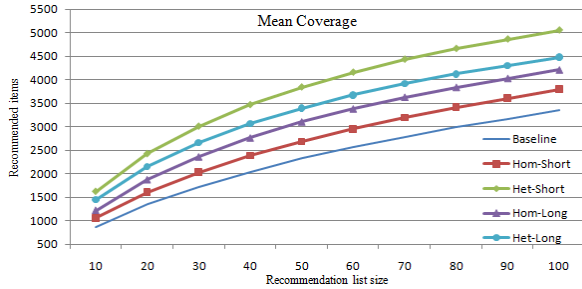


Figure 1. Comparison of mean coverage.

Coverage was increased in 100% of the experiments with a maximum of 7982 recommending items (78.278%). No differences were observed between the linear and quadratic distances whereas the average distance performed better than the centroid one. The biggest average increase occurred for the Slope One algorithm and the smallest for the Matrix Factorization.

### 5.2 Comparison of Unexpectedness

For the first set of expected movies, the EXPECTED metric was decreased by 6.138% in the case of homogeneous users and by 75.186% for the heterogeneous users. For the second set of expected items, the metric was decreased by 61.220% on average for the homogeneous users and by 78.751% for the heterogeneous. Similar results were also observed for the EXPECTED' metric. For the short set of expected movies, the

<sup>3</sup> Due to space limitations and the large number of experiments, only aggregated results are presented. For non-significant differences we plot the necessary dimensions or mean values.

metric was decreased by 3.848% for the homogeneous users and by 26.988% for the heterogeneous. For the long set of expected movies, the ratio was decreased by 39.197% and 47.078%, respectively. Our approach outperformed the standard methods in 94.93% of the experiments (100% for heterogeneous users).

Furthermore, the UNEXPECTED metric increased by 0.091% and 1.171% in the first set of experiments for the homogeneous and heterogeneous users, respectively. For the second set of expected movies, the metric was improved by 4.417% for the homogeneous users and by 5.516% for the heterogeneous (figure 3).

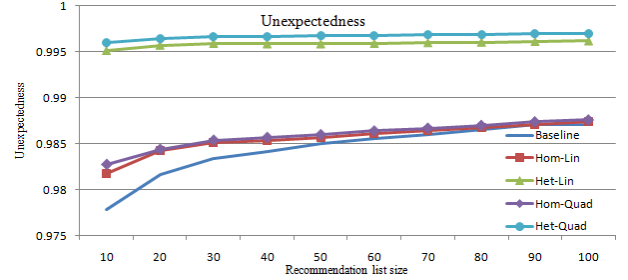


Figure 2. Comparison of Unexpectedness for the 1<sup>st</sup> set of expectations.

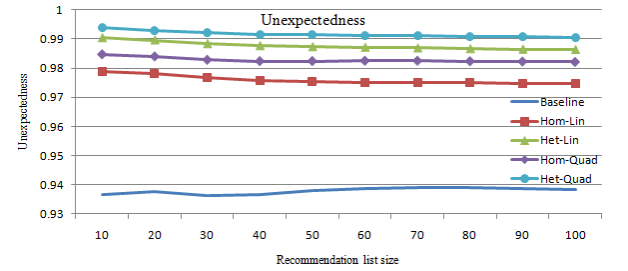


Figure 3. Comparison of Unexpectedness for the 2<sup>nd</sup> set of expectations.

The worst performance of our algorithm was observed in the experiments using the Matrix Factorization algorithm, the first set of expected movies and the linear function of distance under the assumption of homogeneous users (figure 4).

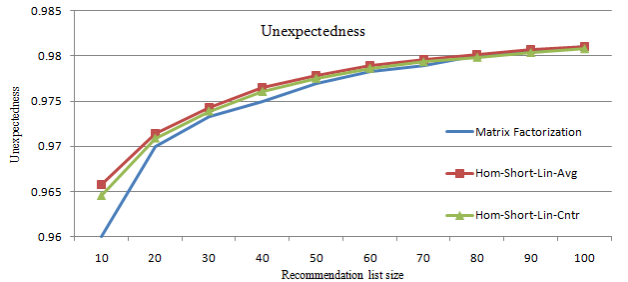


Figure 4. Worst case scenario of Unexpectedness.

As it was expected based on the previous metrics, for the first set of expected movies, the UNEXPECTED' metric was increased by 3.366% and 8.672% in the cases of homogeneous and heterogeneous users, respectively. For the second set of expected movies, in the case of homogeneous users the ratio increased by 8.245% and for the heterogeneous users by 11.980%. It was also observed that using the quadratic distance resulted in more unexpected recommendations. The greatest improvements were observed for the case of Slope One algorithm. Correspondingly, for the metric of unexpectedness given by (9), for the first set of expected movies, the ratios increased by 3.491% and 7.867%. For the second set of expected movies, in the case of homogeneous users, the metric was improved by 4.649% and in the case of

heterogeneous users by 7.660%. Our approach outperformed the standard CF methods in 92.83% of the experiments (97.55% for the case of heterogeneous users).

Moreover, considering an item to be useful if its average rating is greater than 3.0, the SERENDIPITY metric increased, in the first set of experiments, by 2.513% and 3.418% for the homogeneous and heterogeneous users, respectively. For the second set of expected movies (figure 6), the metric was improved by 5.888% for the homogeneous users and by 9.392% for the heterogeneous.

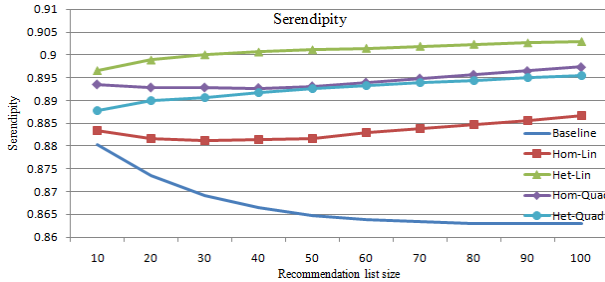


Figure 5. Comparison of Serendipity for the 1<sup>st</sup> set of expectations.

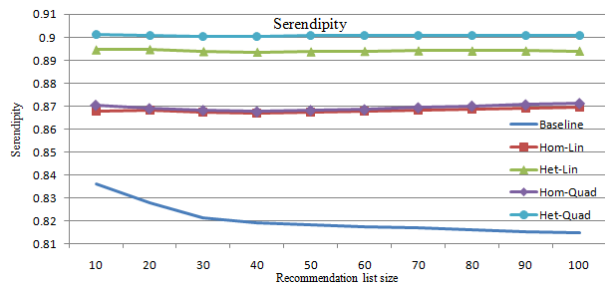


Figure 6. Comparison of Serendipity for the 2<sup>nd</sup> set of expectations.

The worst performance of our algorithm was observed again using the assumption of homogeneous users with the first set of expected movies and the linear function of distance (figure 7).

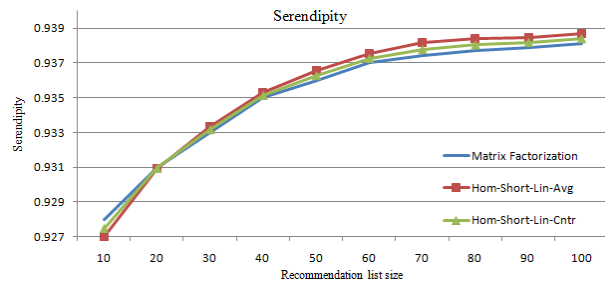


Figure 7. Worst case scenario of Serendipity.

In the first set of experiments, the metric SERENDIPITY<sup>7</sup> increased by 6.284% and 11.451% for the homogeneous and heterogeneous users, respectively. For the second set of expected movies, the metric was improved by 10.267% for the homogeneous users and by 16.669% for the heterogeneous. As expected, the metric of serendipity given by (10) increased by 6.488% in the case of homogeneous users and by 10.625% in the case of heterogeneous, for the short set of expected items. For the case of homogeneous users and the second set of expected movies the ratio was improved by 6.399% and by 12.043% for the heterogeneous users. Our approach outperformed the standard methods in 85.03% of the experiments.

Additionally, qualitatively evaluating the experimental results, our approach, unlike to many popular websites, avoids anecdotal

recommendations such as recommending to a user the movies “The Lord of the Rings: The Return of the King”, “The Bourne Identity” and “The Dark Knight” because the user had already highly rated all the sequels / prequels of these movies (MF, k = 10, user id = 11244).

### 5.3 Comparison of Rating Prediction

The accuracy of rating prediction for the first set of expected movies and the case of the homogeneous users resulted in 0.058% higher RMSE and 0.015% lower MAE on average. Respectively, in the case of heterogeneous customers the RMSE was improved by 1.906% and the MAE by 0.988% on average. For the second set of expected movies, in the case of homogeneous users, the RMSE was reduced by 1.403% and the MAE by 0.735%. For heterogeneous users, the RMSE was improved by 1.548% and the MAE by 0.821% on average with an overall minimum of 0.680 RMSE and 0.719 MAE. The differences between linear and quadratic utility functions are not statistically significant.

Table 2. Mean % improvement of accuracy.

% improvement	Method	Hom-Short		Het-Short		Hom-Long		Het-Long	
		RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
KNN	Avg	0.11	0.01	0.67	4.17	8.30	4.00	8.23	4.03
	Cnt	-0.5	-0.2	8.59	0.33	0.10	0.00	0.18	0.07
MF	Avg	0.02	0.04	0.32	0.23	0.00	0.10	0.03	0.09
	Cnt	0.00	0.10	0.30	0.22	0.00	0.10	0.10	0.14
Slope One	Avg	0.01	0.08	0.80	0.50	0.01	0.12	0.32	0.23
	Cnt	0.01	0.06	0.76	0.48	0.01	0.09	0.43	0.36

### 5.4 Comparison of Item Prediction

For the case of the first set of expected movies, the precision was improved by 25.417% on average for homogeneous users and by 65.436% for heterogeneous users (figure 8). For the extended set (figure 9), the figures are -58.158% and 65.437%, respectively. Similar results were observed for other metrics such AUC and F1.

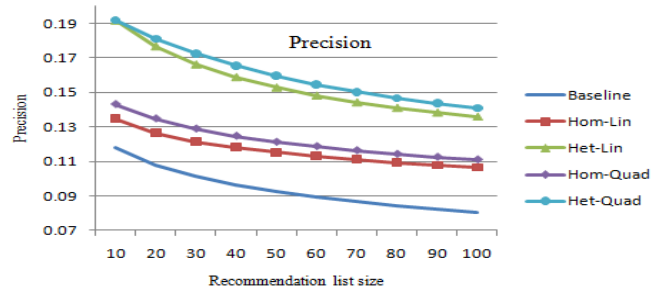


Figure 8. Comparison of Precision for the 1<sup>st</sup> set of expectations.

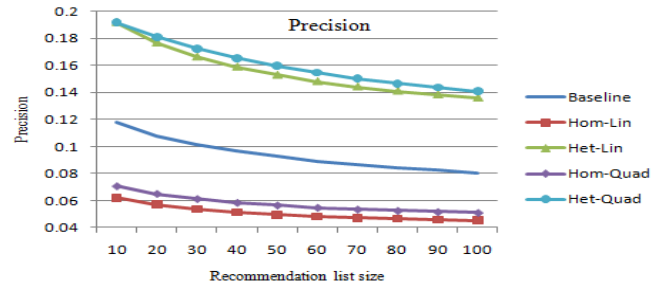


Figure 9. Comparison of Precision for the 2<sup>nd</sup> set of expectations.

## 6. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed and studied a concept of unexpected recommendations as recommending to a user those items that depart from what the specific user expects from the recommender system. After formally defining and formulating theoretically this concept, we discussed how it differs from the related notions of novelty, serendipity and diversity. We presented a method for deriving recommendations based on their utility for the user and compared the quality of the generated unexpected recommendations with some baseline methods using the proposed performance metrics.

Our experimental results demonstrate that our proposed method improves performance in terms of both unexpectedness and accuracy. As discussed in Section 5, all the examined variations of the proposed method, including homogeneous and heterogeneous users with different departure functions, significantly outperformed the standard Collaborative Filtering algorithms, such as k-Nearest Neighbors, Matrix Factorization and Slope One, in terms of measures of unexpectedness. This demonstrates that the proposed method is indeed effectively capturing the concept of unexpectedness since in principle it should do better than unexpectedness-agnostic classical CF methods. Furthermore, the proposed unexpected recommendation method performed at least as well as, and in most of the cases even better than, the baseline CF algorithms in terms of the classical rating prediction accuracy-based measures, such as RMSE and MAE. In the case of heterogeneous users our method also outperforms the CF methods in terms of usage prediction measures such as precision and recall. Thus, the proposed method performed well in terms of both the classical accuracy and the unexpectedness performance measures.

The greatest improvements both in terms of unexpectedness and accuracy vis-à-vis all other approaches were observed in the most realistic case of the extended set of expected movies under the assumption of heterogeneous users. The assumption of heterogeneous users allowed for better approximation of users' preferences at the individual level, while the extended set of expected movies allowed us to better estimate the expectations of each user through a more realistic and natural definition of closely "related" movies.

As a part of the future work, we are going to conduct experiments with real users for evaluating unexpectedness and analyze both qualitative and quantitative aspects in order to enhance the proposed method and explore other ideas as well. Moreover, we plan to introduce and study additional metrics of unexpectedness and compare recommendation performance across these different metrics. We also aim to use different datasets from other domains with users' demographics so as to better estimate the required parameters and derive a customer theory. Overall, the field of unexpectedness in recommending systems constitutes a relatively new and underexplored area of research where much more work should be done to solve this important, interesting and practical problem.

## 7. REFERENCES

- [1] 2<sup>nd</sup> Workshop on Information Heterogeneity and Fusion in Recommender Systems, 5<sup>th</sup> ACM Conf. on Recommender Systems (RecSys 2011) <http://ir.ii.uam.es/hetrec2011>
- [2] Adomavicius, G., & Kwon, Y. Toward more diverse recommendations: Item re-ranking methods for recommender systems. In *WITS* (2009).
- [3] Adomavicius, G., & Kwon, Y. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE TKDE* (2011), pp. 1-15.
- [4] Adomavicius, G., & Tuzhilin, A. Toward the next generation of recommender systems: A Survey of the state-of-the-art and possible extensions. *IEEE TKDE* (2005), pp. 734-749.
- [5] Akiyama, T., Obara, T., & Tanizaki, M. Proposal and evaluation of serendipitous recommendation method using general unexpectedness. In *PRSAT RecSys* (2010).
- [6] Bell, R., Bennett, J., Koren, J., & Volinsky, C. The million dollar programming prize. *IEEE Spectrum* 46, 5 (2009).
- [7] Bennett, J., & Lanning, S. The Netflix Prize. In *KDD* (2007).
- [8] Billsus, D., & Pazzani, M. User modeling for adaptive news access. *UMUAI* 10, 2-3 (2000), pp. 147-180.
- [9] Burke, R. Hybrid recommender systems: Survey and experiments. *UMUAI* 12, 4 (2002), pp. 331-370.
- [10] Cremer, H. & Thisse, J.F. Location models of horizontal differentiation: A special case of vertical differentiation models. *The Journal of Industrial Economics* 39, 4 (1991).
- [11] Ge, M., Delgado-Battenfeld, C., & Jannach, D. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In *RecSys* (2010).
- [12] GroupLens Research. <http://www.grouplens.org>
- [13] Herlocker, J., Konstan, J., Terveen, L., & Riedl, J. Evaluating collaborative filtering recommender systems. *ACM TOIS* 22, 1 (2004), pp. 5-53.
- [14] Hijikata, Y., Shimizu, T., & Nishida, S. Discovery-oriented collaborative filtering for improving user satisfaction. *IUI* (2009), pp. 67-76.
- [15] Iaquinata, L., Gemmis, M. D., Lops, P., Semeraro, G., Filannino, M., & Molino, P. Introducing serendipity in a content-based recommender system. In *HIS* (2008).
- [16] Kawamae, N., Sakano, H., & Yamada, T. Personalized recommendation based on the personal innovator degree. In *RecSys* (2009).
- [17] Konstan, J., McNee, S., Ziegler, C.N., Torres, R., Kapoor, N., & Riedl, J. Lessons on applying automated recommender systems to information-seeking tasks. In *AAAI* (2006).
- [18] Marshall, A. *Principles of Economics*, 8<sup>th</sup> ed. Macmillan and Co., London, UK, (1926).
- [19] McNee, S., Riedl, J., & Konstan, J. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *CHI* (2006).
- [20] Murakami, T., Mori, K., & Orihara, R. Metrics for evaluating serendipity of recommendation lists. In *JSAI* (2007).
- [21] Neven, D. Two stage equilibrium in Hotelling's model. *The Journal of Industrial Economics* 33, 3 (1985), pp. 317-325.
- [22] Padmanabhan, B., & Tuzhilin, A. A belief-driven method for discovering unexpected patterns. *KDD* (1998), pp. 94-100.
- [23] Padmanabhan, B., & Tuzhilin, A. Unexpectedness as a measure of interestingness in knowledge discovery. *Decision Support Systems* 27, 3 (1999), pp. 303-318.
- [24] Shani, G., & Gunawardana, A. Evaluating recommendation systems, In *Recommender Systems Handbook*, Springer-Verlag, New York, NY, USA, (2011), pp. 257-297.
- [25] Tirole, J. Product differentiation: Price competition and non-price competition. *The Theory of Industrial Organization*. The MIT Press, Cambridge, USA, (1988).
- [26] Zhang, M., & Hurley, N. Avoiding monotony: Improving the diversity of recommendation lists. In *RecSys* (2008).
- [27] Ziegler, C.N., McNee, S., Konstan, J., & Lausen, G. Improving recommendation lists through topic diversification. In *WWW* (2005).
- [28] Weng, L.T., Xu, Y., Li, Y., & Nayak, R. Improving recommendation novelty based on topic taxonomy. *WIC* (2007), pp. 115-118.

# Fusion-based Recommender System for Improving Serendipity

Kenta Oku

College of Information Science and Engineering,  
Ritsumeikan University,  
1-1-1 Nojihigashi, Kusatsu-city,  
Shiga, Japan  
oku@fc.ritsumei.ac.jp

Fumio Hattori

College of Information Science and Engineering,  
Ritsumeikan University,  
1-1-1 Nojihigashi, Kusatsu-city,  
Shiga, Japan  
fhattori@is.ritsumei.ac.jp

## ABSTRACT

Recent work has focused on new measures that are beyond the accuracy of recommender systems. Serendipity, which is one of these measures, is defined as a measure that indicates how the recommender system can find unexpected and useful items for users. In this paper, we propose a Fusion-based Recommender System that aims to improve the serendipity of recommender systems. The system is based on the novel notion that the system finds new items, which have the mixed features of two user-input items, produced by mixing the two items together. The system consists of item-fusion methods and scoring methods. The item-fusion methods generate a recommendation list based on mixed features of two user-input items. Scoring methods are used to rank the recommendation list. This paper describes these methods and gives experimental results.

## Categories and Subject Descriptors

H.3.3 [Information Search and Retrieval]: Information filtering

## General Terms

Algorithms, Experimentation

## Keywords

Recommender system, Fusion-based recommender system, Serendipity

## 1. INTRODUCTION

Various recommender systems have been proposed and developed since collaborative filtering was first introduced in the mid-1990s [8][6][1]. In the early years, most recommender systems focused on recommendation accuracy, based on the notion that providing items suitable for users' preferences contributes to an improvement in user satisfaction [8][9]. In contrast, in recent years, several researchers have indicated that recommender systems with high accuracy do not always satisfy users [4][7][5]. They say that recommender systems should be evaluated not only by accuracy, but also by various other metrics such as diversity, novelty, and serendipity.

Copyright is held by the authors. Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011), held in conjunction with ACM RecSys 2011. October 23, 2011, Chicago, Illinois, USA.

Suppose that Alice likes "Harry Potter Part I." To recommend "Harry Potter Part II" or "Harry Potter Part III" to her is obvious and not surprising. Although, from the viewpoint of accuracy, this recommendation is good, it is hard to say that the recommendation satisfies her. Recommender systems should surprise users by providing them with unexpected and useful items.

We focus on serendipity, which is one of the measures beyond accuracy. Although the definition of serendipity has not yet been fixed, Herlocker et al. [4] define serendipity as a measure of the degree to which recommendations are both attractive and surprising to users.

Ge et al. [3] also mention two aspects related to serendipity. The first one is that a serendipitous item should not yet have been discovered by the user and should not be expected by the user. The second one is that the item should also be interesting, relevant, and useful to the user. Although several researchers have tried to improve serendipity, fixing the definition of serendipity and designing recommender systems that improve serendipity are still open problems.

In order to improve serendipity, we believe that recommender systems should have a mechanism that enables users to accidentally discover novel values from unexpected results caused by the user's active actions. "The Three Princes of Serendip" (by Horace Walpole), which is the origin of the term serendipity, tells the story of three princes. They discovered a series of novel things from various and unexpected events on their journeys. Then, they connected these things with their luck. Serendipity is also often involved in making new discoveries. Researchers notice unexpected results in their experiments by trial and error, and then, they connect these results with new discoveries.

Based on this notion, we propose a Fusion-based Recommender System that aims to improve the serendipity of recommender systems. The system recommends new items, which have the mixed features of two user-input items, produced by mixing the two items together.

Such acts of "mixing together", for example, "mixing colors," "mixing ingredients", and "mixing sounds", is intuitive, familiar to people, and has the following characteristics.

- (a) New substances are created from existing ones.
- (b) We can intuitively imagine the mixed results from a combination of input substances. However, some combinations yield unexpected results.
- (c) Because our curiosity may be aroused by characteristic

(b), we might feel like trying to mix various combinations.

Consider the case of mixing colors. If there is no existing color that we want to use, we can create a new color by mixing the existing colors. We can easily imagine that we can create sky-blue from blue and white. On the other hand, some combinations of colors yield unexpected colors. Therefore, our curiosity may be aroused, and we may feel like mixing various combinations by trial and error, for example, “What kind of colors can we create by mixing a certain color and another one?”

The Fusion-based Recommender System adopts an item-fusion approach that produces serendipitous items. The system consists of item-fusion methods and scoring methods. The item-fusion methods generate a recommendation list based on the mixed features of two user-input items. Scoring methods are used to rank the recommendation list.

The contributions of this paper include:

- providing a novel Fusion-based Recommender System that adopts a fusion-based approach to improving serendipity;
- providing three item-fusion methods depending on item representation, and several scoring methods for each item-fusion method;
- a proposed system that can be applied to any dataset that consists of at least an item table, a user table, and a rating table, which are traditional structures in the area of recommendation research;
- an evaluation of the recommender system from the viewpoint of unexpectedness and serendipity.

This paper is organized as follows. In Section 2, we discuss related work that mentions serendipity. In Section 3, we present our proposed system, i.e., a Fusion-based Recommender System. Specifically, we describe item-fusion methods and scoring methods. In Section 4, we evaluate the system from the viewpoint of unexpectedness and serendipity. Finally, we conclude the paper and show future directions in Section 5.

## 2. RELATED WORK

Herlocker et al. [4] suggest that recommender systems with high accuracy do not always satisfy users. They say that recommender systems should be evaluated not only by their accuracy, but also by various other metrics such as diversity, novelty, and serendipity.

Several researchers mention serendipity in the context of recommendation. Ziegler et al. [11][12] assume that diversifying recommendation lists improves user satisfaction. They proposed topic diversification, which diversifies recommendation lists, based on an intra-list similarity metric. Sarwar et al. [10] mention that serendipity might be improved by removing obvious items from recommendation lists. Berkovsky et al. [2] proposed group-based recipe recommendations. They suggest that recipes loved by a group member are likely to be recommended to others, which may increase serendipity.

Hijikata et al. [5] and Murakami et al. [7] proposed recommendation methods that predict novelty or unexpectedness. Hijikata et al. [5] proposed collaborative filtering, which

aims to improve novelty. Collaborative filtering predicts unknown items for a target user, based on known/unknown profiles explicitly acquired from the user. They showed that such filtering can improve novelty by providing unknown items to the user. Murakami et al. [7] proposed a method that implicitly predicts unexpectedness based on a user’s action history. They introduced a preference model, which predicts items the user likes, and a habit model, which predicts items habitually selected by the user. The method estimates the unexpectedness of recommended items by considering differences between the models. They need to accumulate models or profiles for an individual user, but our proposed system does not need these. Our system can instantly recommend serendipitous items based on items the user has just selected.

Murakami et al. [7] and Ge et al. [3] introduced measures for evaluating the unexpectedness and serendipity of recommender systems.

Murakami et al. [7] assume that unexpectedness is the distance between the results produced by the system to be evaluated and those produced by primitive prediction methods. Here, primitive prediction methods mean naive methods such as recommendation methods based on user profiles or action histories. Based on this notion, they proposed *unexpectedness* for measuring the unexpectedness of recommendation lists. They also proposed *unexpectedness\_r*, which takes into account the rankings in the lists.

Ge et al. [3] mention two aspects related to serendipity. The first one is that a serendipitous item should not yet have been discovered and should not be expected by the user. The second one is that the item should also be interesting, relevant, and useful to the user. Although several researchers have tried to improve serendipity, fixing the definition of serendipity and designing recommender systems that improve serendipity are still an open problem. With respect to unexpectedness, they follow the notion of Murakami et al. [7]. They defined an unexpected set of recommendations as follows:

$$UNEXP = RS \setminus PM \quad (1)$$

Here,  $PM$  denotes a set of recommendations generated by primitive prediction models and  $RS$  denotes the recommendations generated by a recommender system to be evaluated. In addition, by using  $u(RS \setminus PM)$ , which denotes the usefulness of the unexpected recommendations, they defined serendipity as follows:

$$SRDP = \frac{\sum_i u(UNEXP_i)}{|UNEXP|} \quad (2)$$

Here,  $UNEXP_i$  denotes an element of  $UNEXP$ . When  $u(UNEXP_i) = 1$ ,  $UNEXP_i$  is useful to the user, and when  $u(UNEXP_i) = 0$ ,  $UNEXP_i$  is useless to the user. The usefulness of  $UNEXP_i$  is given by the user. In Section 4, we evaluate our system using Ge’s measures.

## 3. FUSION-BASED RECOMMENDER SYSTEM

In this section, we describe our proposed system, a Fusion-based Recommender System.

First of all, a user of this system selects two arbitrary items as input items to the system. Then the system finds new items that have the mixed features of both items, using

the item-fusion methods described in Section 3.3. After that, the system makes a recommendation list from the item set, and ranks the list by scoring methods described in Section 3.4. Finally, the system provides the top- $N$  items to the user. The user can then repeatedly use the system by using items in the ranking results in order to find more satisfactory items.

This section is organized as follows. In Section 3.1, we describe the database structure that the system assumes. In addition, we define item similarity and supporting user, as used in this paper. In Section 3.2, we explain the feature representations of items used to apply the system. In Section 3.3, we describe item-fusion methods for generating a recommendation list, and in Section 3.4, we describe scoring methods for ranking the list.

## 3.1 Preliminary

### 3.1.1 Database structure

First of all, in this section, we describe the database structure that the system assumes.

The system assumes that a database consists of the following tables:

- (a) Item table (Item ID, Feature 1, Feature 2, ...)
- (b) User table (User ID, Profile 1, Profile 2, ...)
- (c) Rating table (User ID, Item ID, Rating)

Public datasets such as MovieLens Data Sets and Book-Crossing Data Sets<sup>1</sup> already include the above tables. Other datasets can also be applied to the system by relating them to the above tables.

### 3.1.2 Item similarity

The system calculates item similarity by measuring the similarity between items. This paper defines the following two types of item similarity:

- (a) content-based similarity,
- (b) collaborative-based similarity

Consider two items  $a$  and  $b$ .

(a) Content-based similarity is calculated based on features of items in the item table. Although the features used for the calculation of similarity depend on the datasets, for each item, the system generates a feature vector whose elements correspond to feature values. Then the system calculates item similarity by the cosine similarity between the feature vectors. Consider items  $a$  and  $b$  represented as follows:

$$\mathbf{a} = (a_1, a_2, \dots, a_n) \quad (3)$$

$$\mathbf{b} = (b_1, b_2, \dots, b_n) \quad (4)$$

Here,  $n$  is the number of dimensions of the vector. Then the similarity between the items  $\text{sim}(\mathbf{a}, \mathbf{b})$  is calculated by the following equation:

$$\text{sim}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = \frac{\sum_i a_i b_i}{\sqrt{\sum_i a_i^2} \sqrt{\sum_i b_i^2}} \quad (5)$$

(b) Collaborative-based similarity is calculated based on ratings given to items in the rating table. The system finds a

<sup>1</sup><http://www.grouplens.org/node/74>

common set of users,  $U = \{u_1, u_2, \dots, u_m\}$  ( $m$  is the number of common users), who gave ratings to both items  $a$  and  $b$ . Let  $\text{rating}(u_i, j)$  be a rating given by a user  $u_i$  to an item  $j$ . Consider items  $a$  and  $b$ , represented as follows:

$$\mathbf{a} = (\text{rating}(u_1, a), \text{rating}(u_2, a), \dots, \text{rating}(u_m, a)) \quad (6)$$

$$\mathbf{b} = (\text{rating}(u_1, b), \text{rating}(u_2, b), \dots, \text{rating}(u_m, b)) \quad (7)$$

Then the similarity between the items  $\text{sim}(\mathbf{a}, \mathbf{b})$  is calculated by Equation (5) in the same way.

We define a similar-item set  $S_a$  for an item  $a$  as an item set that consists of items whose similarity to item  $a$  is greater than or equal to a threshold  $\theta$ , i.e., the similar-item set  $S_a$  is expressed by the following equation:

$$S_a = \{x | \text{sim}(x, a) \geq \theta\} \quad (8)$$

### 3.1.3 Supporting user

We define a supporting-user set  $V_a$  for an item  $a$  as a user set that gave ratings equal to or greater than a threshold  $\tau$  to the item  $a$ , i.e., the supporting-user set  $V_a$  is expressed by the following equation:

$$V_a = \{x | \text{rating}(x, a) \geq \tau\} \quad (9)$$

## 3.2 Feature representation of an item

We define the feature representation of items on the basis of the database described in Section 3.1. In this study, we define the following naive representation.

### (1) Bit-string representation.

This representation represents an item as a bit string. Suppose that five elements, {"Action," "Adventure," "Comedy," "Horror," "Romance"}, are defined as attributes that denote item genres. If an item  $a$  corresponds to the genres {"Action," "Horror"}, the item  $a$  is represented as  $\text{bit}(a) = [10010]$ .

### (2) Set representation.

This representation represents an item as a set of related elements. In this paper, we define the following two types of representation, depending on the types of elements:

- (2a) representation by a similar-item set,
- (2b) representation by a supporting-user set.

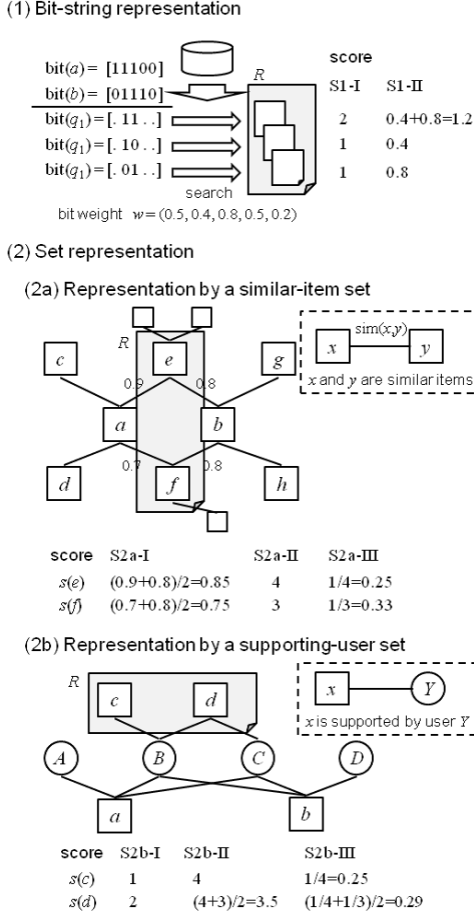
Consider an item  $a$ .

In case (a), we extract a similar-item set  $S_a$  for item  $a$ , based on item similarity as defined in Section 3.1.2. If the similar-item set is given as  $S_a = \{b, c, d, e, f\}$ , the item  $a$  is expressed by  $\text{set}(a) = S_a = \{b, c, d, e, f\}$ .

In case (b), we extract a supporting-user set  $V_a$  for item  $a$ , based on the rating table. If the supporting-user set is given as  $V_a = \{\text{"Alice"}, \text{"Bob"}, \text{"Carol"}\}$ , item  $a$  is expressed by  $\text{set}(a) = V_a = \{\text{"Alice"}, \text{"Bob"}, \text{"Carol"}\}$ .

## 3.3 Item-fusion method for generating recommendation list

A user can find novel items by mixing two items  $a$  and  $b$  that the user selected at will. The system defines criteria for searching novel items, related to mixing features of items  $a$  and  $b$ . Then the system finds an item set that matches the



**Figure 1: Item-fusion methods and scoring methods in case that items  $a$  and  $b$  have common features**

criteria from the database and adds them to a recommendation list. Here, we consider the following assumptions, depending on the features of the input items  $a$  and  $b$ .

- (i) If there are common features between items  $a$  and  $b$ , the user requests items that have the common features.
- (ii) Otherwise, the user requests diverse items that relate to either item  $a$  or item  $b$ .

Based on the above assumptions, we propose item-fusion methods for each feature representation defined in Section 3.2. Figure 1 illustrates examples of the item-fusion methods and scoring methods, which are described in Section 3.4, in case that items  $a$  and  $b$  have common features.

### (1) Bit-string representation.

Based on the notion of a bitwise AND and a bitwise OR, which are the primitive bitwise operations, the system generates a recommendation list  $R$ .

Suppose that a user inputs two items  $a$  and  $b$ , represented as  $\text{bit}(a) = [11100]$ ,  $\text{bit}(b) = [01110]$ , respectively. Here, since the values of the second and the third bit are all 1 in both items  $a$  and  $b$ , the items  $a$  and  $b$  have common features. In order to emphasize these common features, the system generates a representative query bit-string  $\text{bit}(q_1)$ ,

based on the notion of the bitwise AND, i.e., in the case of bits whose values are 1 in both items  $a$  and  $b$ , let the values of the query bits be 1. In the other case, let the values of the query bits be “.”. In this example, the query is expressed by  $\text{bit}(q_1) = [.11..]$ . Here, “.” matches either of  $\{0, 1\}$  while searching.

However, if the number of bits is large, there is little possibility of finding items that match the query. In order to avoid such cases, we consider a query set that considers all combinations of the values  $\{0, 1\}$  except the bits whose values are “.”, i.e., in this example, the generated query set is expressed by  $Q = \{[.11..], [.10..], [.01..]\}$  (see Figure 1 (1)).

Now consider two items  $a$  and  $b$  that are represented as  $\text{bit}(a) = [10000]$ ,  $\text{bit}(b) = [00110]$ , respectively. In this case, the items  $a$  and  $b$  have no common feature. In order to diversify the recommendation list, the system generates a representative query bit-string  $\text{bit}(q_1)$ , based on the notion of the bitwise OR, i.e., in the case of bits whose values are 1 in either item  $a$  or  $b$ , let the values of the query bits be 1. In the other case, let the values of the query bits be “.”. In this example, the query is expressed by  $\text{bit}(q_1) = [1.11..]$ .

In the same way, in this example, the generated query set is expressed by  $Q = \{[1.11..], [1.10..], [1.01..], [1.00..], [0.11..], [0.10..], [0.01..]\}$ .

Finally, the system finds an item set that matches each query bit-string from the item table and then adds the item set to the recommendation list  $R$ .

Now, we generalize the above examples. Consider items  $a$  and  $b$  represented by  $n$ -digit bit-strings. Let  $\text{bit}(a)_i$  be the  $i$ th bit of item  $a$ . We define items  $a$  and  $b$  as having common features if there is at least one  $i$  that satisfies  $\text{bit}(a)_i = \text{bit}(b)_i = 1$  ( $i = 1, 2, \dots, n$ ).

(i) If items  $a$  and  $b$  have common features, each bit of the representative query bit-string  $\text{bit}(q_1)$  is as follows:

$$\text{bit}(q_1)_i = \begin{cases} 1 & \text{if } \text{bit}(a)_i = 1 \wedge \text{bit}(b)_i = 1 \\ \text{“.”} & \text{otherwise} \end{cases} \quad (10)$$

(ii) Otherwise, each bit of the string is as follows:

$$\text{bit}(q_1)_i = \begin{cases} 1 & \text{if } \text{bit}(a)_i = 1 \vee \text{bit}(b)_i = 1 \\ \text{“.”} & \text{otherwise} \end{cases} \quad (11)$$

As we stated in the above examples, the system generates a query set  $Q$ , which considers all combinations of query bit-strings. Finally, the system generates a recommendation list  $R$ , based on the query set  $Q$ .

### (2) Set representation.

Based on the notions of intersection and union, which are the primitive set operations, the system generates a recommendation list  $R$ .

Consider items  $a$  and  $b$  represented as  $\text{set}(a) = \{a_1, a_2, \dots, a_n\}$ ,  $\text{set}(b) = \{b_1, b_2, \dots, b_m\}$ , respectively. We define items  $a$  and  $b$  as having common features if  $\text{set}(a) \cap \text{set}(b) \neq \phi$ . We explain how to generate the recommendation list  $R$  in cases of representation by (2a) a similar-item set, (2b) a supporting-user set.

#### (2a) Representation by a similar-item set.

(i) If the items  $a$  and  $b$  have common features, the system regards the intersection of similar-item sets of items  $a$  and  $b$  as the recommendation list  $R$ . (ii) Otherwise, the system



regards the union of the sets as the recommendation list  $R$ , i.e., the recommendation list  $R$  in each case is as follows:

$$R = \begin{cases} \text{set}(a) \cap \text{set}(b) & \text{if } \text{set}(a) \cap \text{set}(b) \neq \phi \\ \text{set}(a) \cup \text{set}(b) & \text{otherwise} \end{cases} \quad (12)$$

For example, given an item  $a = \{c, d, e, f\}$  and an item  $b = \{e, f, g, h\}$ , the recommendation list is  $R = \text{set}(a) \cap \text{set}(b) = \{e, f\}$  (see Figure 1 (2a)). On the other hand, given an item  $a = \{c, d\}$  and an item  $b = \{e, f\}$ , the recommendation list is  $R = \text{set}(a) \cup \text{set}(b) = \{c, d, e, f\}$ .

### (2b) Representation by a supporting-user set.

(i) If the items  $a$  and  $b$  have common features, consider the intersection  $V$  of the supporting-user sets of items  $a$  and  $b$ . (ii) Otherwise, consider their union  $V$ , i.e., the user set in each case is as follows:

$$V = \begin{cases} \text{set}(a) \cap \text{set}(b) & \text{if } \text{set}(a) \cap \text{set}(b) \neq \phi \\ \text{set}(a) \cup \text{set}(b) & \text{otherwise} \end{cases} \quad (13)$$

Then the system regards an item set supported by each user  $v_i \in V$  (who gave ratings equal to or greater than a threshold  $\tau$  to the item), i.e., the recommendation list is as follows:

$$R = \bigcup_{v_i \in V} \{x | \text{rating}(v_i, x) \geq \tau\} \quad (14)$$

For example, given an item set  $(a) = \{\text{“Alice,” “Bob,” “Carol”}\}$  and an item set  $(b) = \{\text{“Bob,” “Carol,” “Dave”}\}$ , the user set is  $V = \text{set}(a) \cap \text{set}(b) = \{\text{“Bob,” “Carol”}\}$ . Furthermore, given an item set  $\{c, d\}$  supported by “Bob”, and an item set  $\{d\}$  supported by “Carol”, the recommendation list is  $R = \{c, d\}$  (see Figure 1 (2b)).

On the other hand, given an item set  $(a) = \{\text{“Alice,” “Bob”}\}$  and an item set  $(b) = \{\text{“Carol”}\}$ , the user set is  $V = \text{set}(a) \cup \text{set}(b) = \{\text{“Alice,” “Bob,” “Carol”}\}$ . Furthermore, given an item set  $\{a\}$  supported by “Alice”, an item set  $\{c, d\}$  supported by “Bob”, and an item set  $\{d, e\}$  supported by “Carol”, the recommendation list is  $R = \{a, c, d, e\}$ .

## 3.4 Scoring method for ranking recommendation list

Some combinations of items produce recommendation lists that consist of an enormous number of items. In such cases, the recommendation list should be ranked according to some criteria in order to narrow the list of recommended items shown to the user. In this section, we define scoring methods for each item-fusion method provided in Section 3.3.

### (1) Bit-string representation.

We define the following two scoring methods, S1-I and S1-II, for bit-string representation. Examples of these scoring methods are illustrated in Figure 1 (1).

#### (S1-I) Score based on the number of common bits.

As we stated in Section 3.3, the system generates a query set  $Q$ , which consists of all combinations of query bit-strings. We also explained that the value of each bit  $\text{bit}(q_i)_j$  can take  $\{1, 0, \text{“.”}\}$ . We assume that the larger the number of bits that satisfy  $\text{bit}(q_i)_j = 1$ , the more strongly the query bit-string reflects common features of items  $a$  and  $b$ . Therefore, we define the following score  $s(r_k)$  for a recommended item  $r_k$ :

$$s(r_k) = |\{x | \text{bit}(q_k)_x = 1\}| \quad (15)$$

Here,  $q_k$  denotes a query used for searching the item  $r_k$ .

#### (S1-II) Weighted score based on the number of common bits.

Some datasets have different weights for each bit. If the weight  $w_j$  for each bit  $j$  is assigned in advance, we define the following weighted score  $s(r_k)$ :

$$s(r_k) = \sum_{j \in \{x | \text{bit}(q_k)_x = 1\}} w_j \quad (16)$$

How the weight for each bit is calculated depends on the datasets, but, for a simple example, we can employ the bit variance in the dataset.

### (2) Set representation.

#### (2a) Representation by a similar-item set.

We define the following three scoring methods, S2a-I, S2a-II, and S2a-III, for representation by a similar-item set. Examples of these scoring methods are illustrated in Figure 1 (2a).

#### (S2a-I) Score based on item similarity to input items.

We define the following score  $s(r_k)$ , based on the collaborative-based similarities  $\text{sim}(r_k, a)$  and  $\text{sim}(r_k, b)$  between the recommended item  $r_k$  and the input items  $a$  and  $b$ :

$$s(r_k) = \begin{cases} \frac{1}{2}(\text{sim}(r_k, a) + \text{sim}(r_k, b)) & \text{if } \text{set}(a) \cap \text{set}(b) \neq \phi \\ \max(\text{sim}(r_k, a), \text{sim}(r_k, b)) & \text{otherwise} \end{cases} \quad (17)$$

#### (S2a-II) Score based on the number of items similar to the recommended item.

We define the following score  $s(r_k)$ , based on the number of items similar to the recommended item  $r_k$ :

$$s(r_k) = |\{x | \text{sim}(r_k, x) \geq \theta\}| \quad (18)$$

Here,  $\theta$  denotes the similarity threshold.

#### (S2a-III) Score based on the reciprocal of the number of items similar to the recommended item.

This score is in contrast to that in S2a-II. It is based on the assumption that the more the recommended item  $r_k$  is restricted to only items similar to the input items  $a$  and  $b$ , the more strongly the recommended item  $r_k$  is related to the input items  $a$  and  $b$ . Thus, we define the following score  $s(r_k)$ :

$$s(r_k) = \frac{1}{|\{x | \text{sim}(r_k, x) \geq \theta\}|} \quad (19)$$

Here,  $\theta$  denotes the similarity threshold.

#### (2b) Representation by a supporting-user set.

We define the following three scoring methods, S2b-I, S2b-II, and S2b-III, for representation by a supporting-user set. Examples of these scoring methods are illustrated in Figure 1 (2b).

#### (S2b-I) Score based on the number of common users.

Among users who support the recommended item  $r_k$ , we assume that the larger the number of users who support both

input items  $a$  and  $b$ , the more strongly the recommended item  $r_k$  is related to the input items  $a$  and  $b$ . Therefore, we define the following score  $s(r_k)$ :

$$s(r_k) = |V_{r_k} \cap (V_a \cup V_b)| \quad (20)$$

Here,  $V_{r_k}$ ,  $V_a$ , and  $V_b$  denote the supporting-user set for item  $r_k$ , and the input items  $a$  and  $b$ , respectively.

*(S2b-II) Score based on the mean of the number of items supported by supporting users.*

We define a score based on the mean of the number of items supported by the supporting-user set  $V_{r_k}$ . We assume that the larger the number of items the user supports, the more reliable the user is. Thus, we define the following score  $s(r_k)$ :

$$s(r_k) = \frac{1}{|V_{r_k}|} \sum_{v_i \in V_{r_k}} |\{x | \text{rating}(v_i, x) \geq \tau\}| \quad (21)$$

Here,  $\tau$  denotes the threshold for whether the user supports the item.

*(S2b-III) Score based on the mean of the reciprocal of the number of items supported by supporting users.*

It is based on the assumption that the greater the extent to which the user restricts support to only the recommended item  $r_k$ , the better the user supports item  $r_k$ . Thus, we define the following score  $s(r_k)$ :

$$s(r_k) = \frac{1}{|V_{r_k}|} \sum_{v_i \in V_{r_k}} \frac{1}{|\{x | \text{rating}(v_i, x) \geq \tau\}|} \quad (22)$$

Based on each scoring method, the system calculates the score for each recommended item. Then the system orders the recommendation list  $R$  according to the scores.

## 4. EXPERIMENTS

We conducted experiments for evaluating serendipity of recommendation lists generated by the item-fusion methods and scoring methods described in Section 3. In Section 4.1, we explain the dataset used for evaluation, feature representation of items, measures and baseline methods for comparison, respectively. After we describe experimental steps in Section 4.2, we show experimental results and discuss them in Section 4.3.

### 4.1 Experimental setup

#### 4.1.1 Dataset and feature representation of items

We used MovieLens Data Set in the experiments. This dataset consists of 100,000 ratings (1-5) from 943 users on 1682 movies. This dataset has the principal tables shown in Table 1.

Followed the tables described in Section 3.1.1, the u.item, u.user and u.data correspond to an item table, user table, and rating table, respectively. Attributes, “movie title” to “Western” of u.item, correspond to item features. Particularly, 18 attributes, “Action” to “Western,” represent item genres, which are given by either  $\{0, 1\}$  according to contents of the movie.

According to Section 3.2, we define feature representation of items based on the Table 1 as follows.

Table 1: Contents of MovieLens Data Set

Table type	Table name	Attributes
Item table	u.item	movie id, movie title, release date, video release date, IMDb URL, unknown, Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western
User table	u.user	user id, age, gender, occupation, zip code
Rating table	u.data	user id, item id, rating, timestamp

#### (1) Bit-string representation.

We represent an item as a bit string based on the genres of the item. For example, since the movie whose movie id = 1, which is “Toy Story,” corresponds to the third, fourth and fifth genres, {“Animation,” “Children,” “Comedy”}, respectively, it is represented as  $\text{bit}(1) = [001110000000000000]$ . We also employed the bit variance in the dataset as the weight for each bit  $w_j$  in the scoring method S1-II in the experiments.

#### (2a) Representation by a similar-item set.

We obtain a similar-item set for each item based on the item similarity described in Section 3.1.2. In the experiments, we calculated the item similarity by the collaborative-based similarity. Here, we let the similarity threshold be  $\theta = 0.95$ .

#### (2b) Representation by a supporting-user set.

We obtain a supporting-user set for each item based on the calculation shown in Section 3.1.3. We let the threshold be  $\tau = 3.0$ .

#### 4.1.2 Measures

Our proposed system regards serendipity as important rather than recommendation accuracy. Therefore, we evaluate our system from the point of view of how the system can generate serendipitous items.

In the experiments, we introduce  $r$ -*unexpectedness* and  $r$ -*serendipity* based on the notions of Murakami et al. [7] and Ge et al. [3] presented in Section 2.

First of all, we present the Equation (1) again.

$$UNEXP = RS \setminus PM \quad (23)$$

Here,  $RS$  denotes a recommendation list generated by the proposed system. We also employed the following two methods as the primitive prediction methods:  $PM_{mean}$ , a prediction method based on the mean ratings, and  $PM_{num}$ , a prediction method based on the number of ratings.

The  $PM_{mean}$  regards the top- $N$  items with the highest mean ratings as a recommendation item set. The  $PM_{num}$  regards the top- $N$  items with the largest number of ratings as a recommendation item set. Finally, we utilize the union of the  $PM_{mean}$  and  $PM_{num}$  as  $PM$ . Thus, the  $PM$  includes items whose total number is  $2N$ .

We introduce  $r$ -*unexpectedness* that denotes a ratio of the number of the unexpected items of the top- $r$  ranked recommendation list, and represent it as follows:

$$r\text{-unexpectedness} = \frac{|UNEXP(r)|}{r} \quad (24)$$

Here,  $UNEXP(r)$  denotes  $UNEXP$  when the top- $r$  items are provided by the  $RS$ .

We also introduce serendipitous item set as follows (note

that this is different from the Equation (1)):

$$SERENDIP = UNEXP \cap USEFUL \quad (25)$$

Here, *USEFUL* denotes useful item set given separately. In the experiments, we gave the usefulness of items based on their mean ratings. Then we regard items equal to or greater than a threshold  $v$  as useful items. The *USEFUL* consists of the useful item set.

In the same way, we introduce *r-serendipity* that denotes a ratio of the number of the serendipitous items of the top- $r$  ranked recommendation list, and represent it as follows:

$$r\text{-serendipity} = \frac{|SERENDIP(r)|}{r} \quad (26)$$

Here,  $SERENDIP(r)$  denotes *SERENDIP* when the top- $r$  items are provided by the *RS*.

### 4.1.3 Baseline methods

In order to evaluate serendipity of the proposed system, we compare the system with the following three types of baseline methods:  $CB_a$  and  $CB_b$ , content-based filtering for input items  $a$  and  $b$ ,  $CF_a$  and  $CF_b$ , collaborative filtering for input items  $a$  and  $b$ , and *RAND*, random method.

Here, the  $CB_a$  and  $CB_b$  provide items in order of content-based similarity to items  $a$  and  $b$ , respectively. In the experiments, in order to calculate the similarity, we used cosine similarity between feature vectors whose elements denote 18 item genres. The  $CF_a$  and  $CF_b$  provide items in order of collaborative-based similarity to items  $a$  and  $b$ , respectively. The *RAND* provides items selected and ordered at random from the item table.

## 4.2 Experimental steps

We conducted the experiments by using 1000 pairs of items selected from the item table at random. Given an item pair  $(a, b)$ , the experimental steps are as follows:

- step 1 Generate a recommendation list  $R$  by each item-fusion method (see Section 3.3) for the item pair  $(a, b)$ .
- step 2 Make a ranking list  $R'$  for the recommendation list  $R$  by each scoring method, i.e., S1-I, S1-II, S2a-I, S2a-II, S2a-III, S2b-I, S2b-II, and S2b-III (see Section 3.4), and by each baseline method.
- step 3 Obtain *r-unexpectedness* and *r-serendipity* for each  $R'$ .

## 4.3 Results and discussion

In this section, we show experimental results and discuss them. In the experiments, we used  $N = 50$ , which is the number of items provided by each primitive prediction method *PM*, and  $r = 20$  for *r-unexpectedness* and *r-serendipity*. Before the experiments, we conducted preliminary experiments under conditions of  $N = \{10, 20, \dots, 100\}$  and  $r = \{10, 20, \dots, 100\}$ . Note that we found that relative relationship between scoring methods did not significantly depend on the conditions. We also used  $v = 3.0$  that is the threshold for whether the item is useful.

### 4.3.1 Comparison with baseline methods

Figure 2 shows the mean *r-unexpectedness* and *r-serendipity* by the scoring methods and baseline methods. The horizontal axis denotes *r-unexpectedness* and the vertical axis denotes *r-serendipity*. Note that, on the baseline

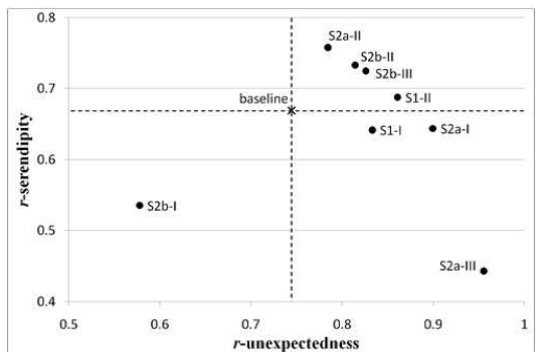


Figure 2: Mean *r-unexpectedness* and *r-serendipity* by scoring methods and baseline methods

methods, this figure shows the mean of them obtained by three types of baseline methods described in Section 4.1.3.

We found that S1-II, S2a-II, S2b-II, and S2b-III produced significantly higher *r-unexpectedness* and *r-serendipity* than the baseline methods produced ( $p < 0.01$ ). Notably, since S1-II, S2a-II, S2b-II, or S2b-III can yield high serendipity for each item-fusion method, (1), (2a), and (2b), we believe that the proposed system works effectively by using any feature representation of items. We discuss these cases in detail in the next section.

On the other hand, *r-unexpectedness* by S1-I, S2a-I, and S2a-III are less than ones by the baseline methods while *r-serendipity* by them are higher than ones by the baseline methods. Particularly, the *r-unexpectedness* by S2a-III is much less than one by the baseline methods. In addition, both *r-unexpectedness* and *r-serendipity* by S2b-I are less than ones by the baseline methods.

As we described in Section 3.4 (2b), the S2b-I calculates scores based on the number of users who support both the recommended item  $r_k$  and input item  $a$  or  $b$ . In the experiments, we employed the  $PM_{num}$ , which is based on the number of ratings, as one of the primitive prediction methods. Since the items rated by many users can be easy to be predicted, the S2b-I yields low *r-unexpectedness*.

As we described in Section 3.4 (2a), we assume that the more the recommended item  $r_k$  is restricted to only items similar to the input items  $a$  and  $b$ , the more strongly the recommended item  $r_k$  is related to the input items  $a$  and  $b$ . However, this result shows that the assumption is not correct. Since S2a-II, which is opposite to the S2a-III, showed higher *r-serendipity*, we found that we should employ S2a-II for the purpose of improving serendipity.

### 4.3.2 Comparison of serendipity by different relationship between input items

We conducted an additional experiment to analyze difference between S1-II, S2a-II, S2b-I, and S2b-II, which yielded higher serendipity. We analyzed the difference of *r-serendipity* depending on the relationship between input items. We focus on the relationship between input items shown in Table 2. We grouped 1000 item pairs used in the experiments by the relationship between input items. As shown in Figure 3, we obtained *r-serendipity* for each group.

We found that S2a-II could produce significant high serendipity ( $p < 0.01$ ) in all cases except for mean-HH and num-HH. We also found that S2b-II and S2b-III in case of

Table 2: Relationship between input items

Symbol	Relationship between input items
sim-H	items with high content-based similarity
sim-L	items with low content-based similarity
mean-HH	items with high mean ratings
mean-HL	an item with high mean ratings and an item with low mean ratings
mean-LL	items with low mean ratings
num-HH	items with many ratings
num-HL	an item with many ratings and an item with few ratings
num-LL	items with few ratings

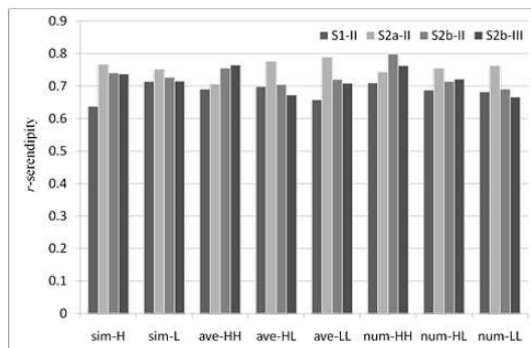


Figure 3: Mean  $r$ -serendipity by scoring methods depending on combinations of input items

mean-HH, and S2b-II in case of num-HH could produce significant high serendipity ( $p < 0.01$ ), respectively. On the basis of this result, we can expect that effectiveness of the system can be improved by introducing a switching method that dynamically switches scoring methods depending on relationship between user-input items.

Furthermore, we are interested in that S2a-II can produce high serendipitous items by using unpopular items, i.e., items with low or few ratings, as materials for item-fusion. We believe that we can expect that system usage can be broadened by using such items effectively.

## 5. CONCLUSION

In this paper, we proposed a Fusion-based Recommender System that aims to improve the serendipity of recommender systems. The system is based on the novel notion that the system finds new items, which have the mixed features of two user-input items, produced by mixing the two items together. The system consists of item-fusion methods and scoring methods. We proposed three item-fusion methods and eight scoring methods on the basis of the item-fusion methods.

Experimental results showed that S1-II, S2a-II, S2b-II, and S2b-III produced higher serendipitous items than baseline methods produced. This paper describes these methods and gives experimental results. The results also showed that S2a-II could produce high serendipity in most cases. We also found that S2b-II and S2b-III in case of using input items with high mean ratings, and S2b-II in case of using input items with high ratings could produce high serendipity, respectively. These results suggest that effectiveness of the system can be improved by introducing a switching method that dynamically switches scoring methods depending on relationship between user-input items.

In the future, we would like to analyze the results qualitatively. We want to know what kind of item pair yields what

kind of recommendations. Although we used static ratings for judging useful items, we will conduct experiments with real users for evaluating serendipity. We plan to implement other feature representation of items, e.g., by a tag set and feature vectors. We also plan to design user interface of the system that makes the system usage more effective.

## 6. ACKNOWLEDGEMENT

This work was supported by Grant-in-Aid for Young Scientists (B) (23700132).

## 7. REFERENCES

- [1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.
- [2] Shlomo Berkovsky and Jill Freyne. Group-based recipe recommendations: analysis of data aggregation strategies. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 111–118. ACM, 2010.
- [3] Mouzhi Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 257–260. ACM, 2010.
- [4] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.
- [5] Y. Hijikata, T. Shimizu, and S. Nishida. Discovery-oriented collaborative filtering for improving user satisfaction. In *Proceedings of the 13th international conference on Intelligent user interfaces*, pages 67–76. ACM, 2009.
- [6] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, January 2003.
- [7] Tomoko Murakami, Koichiro Mori, and Ryohei Orihara. Metrics for evaluating the serendipity of recommendation lists. *New Frontiers in Artificial Intelligence*, pages 40–46, 2008.
- [8] Paul Resnick, N. Iacovou, M. Suchak, Peter Bergstrom, and John Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.
- [9] Paul Resnick and H.R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.
- [10] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, New York, USA, 2001. ACM.
- [11] Cai-Nicolas Ziegler, Georg Lausen, and Lars Schmidt-Thieme. Taxonomy-driven computation of product recommendations. In *Proceedings of the Thirteenth ACM conference on Information and knowledge management - CIKM '04*, page 406, New York, New York, USA, 2004. ACM Press.
- [12] C.N. Ziegler, S.M. McNee, J.A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pages 22–32, New York, New York, USA, 2005. ACM.

# The Oblivion Problem: Exploiting forgotten items to improve recommendation diversity

Fernando Mourão, Claudiane Fonseca, Camila Araújo, Wagner Meira Jr.  
Department of Computer Science - Universidade Federal de Minas Gerais (UFMG)  
Av. Antônio Carlos 6627 - ICEx - 31270-010, Belo Horizonte, Brazil  
(fhmourao,fonseca,camilaaraujo,meira)@dcc.ufmg.br

## ABSTRACT

Recommender Systems (RSs) have become a crucial tool to assist users in their choices on various commercial applications. Despite recent advances, there is still room for more effective techniques that are applicable to a larger range of domains. A major challenge recurrently researched is the lack of diversity in the recommendation lists provided by current RSs. That is, besides being effective to suggest interesting items to users, a good RS should provide useful and diversified items. In order to address this problem, we evaluate the use of forgotten items in recommendation. By forgotten items, we mean items that have been very relevant to users in the past but are not anymore. Therefore, we formally define the **Oblivion Problem**, which is the problem of recommending forgotten items, propose a methodology for verifying it in real scenarios, and perform a deep characterization of this problem in a relevant music domain, the *Last.fm* system. Applying our methodology to Last.fm has demonstrated the existence of the oblivion problem in practice, as well as showed the utility of this methodology. Further, the behavior exhibited by forgotten items in *Last.fm* suggests that defining techniques that incorporate such items into RSs consists in a promising research direction.

## Categories and Subject Descriptors

H.4.m [Information System Applications]: Miscellaneous; H.m [Information System]: Miscellaneous

## General Terms

Recommendation, Formalization, Characterization

## 1. INTRODUCTION

Recommender Systems (RSs) are becoming increasingly important tools for many commercial applications due to their ability to filter a huge and growing volume of options, showing only what may be interesting to users [8]. We define RSs as any system that is designed to produce individualized recommendations as output, or to guide users through a huge variety of options [3, 7]. Intuitively, the growing demand for such tools may be explained by the so-called **Paradox of Choice** [15], which states that, as the number

of options grows, the effort required to make a wise decision also increases, making the possibility of choosing a burden, instead of an advantage.

Despite the numerous strategies proposed for RSs, current systems still lack effectiveness in terms of identifying not only accurate but also diversified lists of recommended items [19]. The problem of such lack of diversity is that recommending over and over again the same items, even being relevant ones, to the same users is likely to annoy them, decreasing their interest in interacting with the RS over time. It was observed that, although the domains where RSs operate present a wide diversity of items, the recommendations are, in general, poor in terms of diversity [19], as a consequence of a huge concentration of users around few popular products followed by a much smaller demand around the other products, a phenomenon known as *Long Tail* [2]. As a result, only a small portion of items obtain enough ratings and, therefore, is considered suitable for recommendation [13]. In addition, an important factor that affects the items popularity, and consequently the lack of diversity on recommendations, is their aging, since the probability of an item to be recommended is inversely proportional to its age [2]. In summary, diversity means accuracy loss in most application scenarios, and achieving both in a wide range of real-world scenarios such as travel and financial services, among others, is a constant challenge.

Traditionally, diversity in RSs increases with the arrival of new items in the system. Nevertheless, new items have few evaluations and do not contribute immediately to improve recommendations [1]. Another source of diversity that is assessed in this paper is to use forgotten items. We define as forgotten items any item that used to be relevant and of frequent interest to a particular user in the past, and now it is not. The main hypothesis of this work is that forgotten items, that appear in the tail of the popularity distribution of items, may increase the recommendation diversity while keeping its accuracy. The main premise here is that user recommendation profiles are defined as a function of their most consumed items in a given period of time, and despite the relevance loss associated with aging and competition associated with new items, their importance in the past may guarantee a good recommendation. Further, given the amount of information associated with forgotten items, they represent a richer source of information, compared to new ones, becoming more suitable for RSs. In addition, rescuing these items may be surprising and bring back good memories in scenarios where old items may remain interesting for the users over time.

There are two key issues related to use forgotten items that are addressed in this work. The first one consists of validating the main hypothesis, that is, the utility of forgotten items, since both users and the domain as a whole evolve over time. For instance, considering the music domain, someone who used to like *Cindy Lauper* in the past not necessarily would like her nowadays. Further, forgotten items may have a very long past, but no recent information, since they are no longer consumed. The second key issue is to as-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright is held by the author/owner(s). Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011), held in conjunction with ACM RecSys 2011. October 23, 2011, Chicago, Illinois, USA.

sess the utility of a forgotten item, which is difficult because items are forgotten at different ages by different users.

One interesting observation is that the problem of recommending forgotten items, which we call the **Oblivion Problem**, is similar to the *Cold Start* problem in RSs [14], once the difficulty of recommending new items also lies in the absence information, but now due to lack of past information. Such similarity may allow the use of current solutions for the *Cold Start* in the task of recommending new and forgotten items, in order to appropriately improve the diversity in traditional RSs.

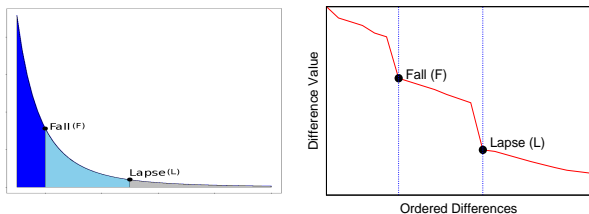
We restrict the evaluation of such problem to the music recommendation domain, due to an intrinsic feature of this domain: the high repetition rate in music consumption over time. That is, users tend to listen to a given song repeatedly more often than watch a given movie, reinforcing this work hypothesis. Further, this behavior shows that the requirement that old items may remain interesting for the users over time could be expected in such scenario.

In summary, the aims of this work are restricted to define and to characterize the Oblivion problem, rather than proposing solutions for it. In this sense, we highlight the following contributions: (1) the formalization of a new problem in RSs, namely the **Oblivion Problem**; (2) the proposal of a methodology to verify whether this problem occurs in real domains; and (3) a deep characterization of the Oblivion Problem in a real and relevant scenario, *Last.fm*. Our analyses use *Last.fm* as workload, since it represents one of the largest musical community in the world, comprising more than 12 million distinct artists and 30 million active users at the gathering moment. Besides this huge size, the availability of most data on the WEB make this system a promising data source for studies on music recommendation. Results from our methodology on a *Last.fm* sample demonstrate the existence of the Oblivion Problem in this scenario, as well as show the practical utility of this methodology. Further, the behavior exhibited by forgotten items in this domain suggests that defining techniques that incorporate such items into RSs represents a promising research direction.

## 2. BACKGROUND CONCEPTS

In this section we present some key concepts for formalizing the Oblivion Problem. We start by discussing the role of long tail distribution on the recommendation domain, since it is one of the main motivations for the Oblivion Problem. After, we introduce some definitions derived from the analysis of this distribution.

Recently, the so-called *Long Tail* distribution [13] is regarded as one of the most recurrent data models for various commercial applications. It is defined as a distribution of items ordered decreasingly by the number of distinct users who have consumed each item in a given period of time. This distribution indicates a sharp and strong interest for a restricted set of popular products, followed by fast demand decrease that extends for a long and low tail, associated with increasingly unpopular products [2]. Figure 1 (a) presents an example of this distribution.



(a) Example Distribution (b) Example of Distribution Partition

Figure 1: Long Tail

The relevance of this distribution to RSs stems from a property discussed by Anderson [2], which states that the long tail defines a market of a large number of non-popular items that rivals with the popular ones. In fact, the emergence of very specific users' niches, made explicit by the long tail, increases the need for RSs that are more capable of providing diverse and accurate recommendations, since recommending just popular items is not enough to reach a large portion of those niches. However, long-tailed distributions pose some challenges for RSs and the main one refers to the scarcity of information about many of the items, since most of them are consumed infrequently.

As illustrated in Figure 1 (a), we divide long-tail distributions into three parts. The first part represents the head of distribution and is composed of the most popular items of the system, and, controversially, the smallest number of unique items. The second part is the body of the distribution, which includes a larger number of distinct items but with a lower popularity compared to the items in the first part. Finally, we have the tail, which represents the vast majority of existing items in commercial domains. Each of these items, however, is consumed only by a negligible portion of distinct users of the domain. We define the *Fall Point* ( $F$ ) as the point in the distribution that separates the head from the body, and the *Lapse Point* ( $L$ ) as the point that splits the body from the tail. In general, the identification of such points is performed through some transformation functions on the distribution. For example, we can plot the absolute values of the differences between adjacent points in the distribution. In this plot, those points are identified by possible existing elbows, as illustrated in Figure 1 (b).

For recommendation purposes, we determine the distribution for a delimited period of time. That is, for a given period of time we evaluate which items are more or less popular, obtaining different distributions for distinct periods. Thus, the dynamics of real scenarios is captured and items that are no longer relevant for the users over time are not considered in the distribution. The peculiarity of recommendation scenarios is that, regardless the period of analysis, the generated distributions are long-tail as a consequence of the following facts. First, most of the novel items that appear during each observation period remain at the tail of the distribution. Second, there are always "migration" movements of items along the distribution. That is, some items become more popular, migrating toward the head, while other items that are no longer relevant become less popular, moving toward the tail of the distribution. By considering both the three parts and possibility of popularity changes over time, we denote the items from the head as *successful* items, items from the body as *transition* ones, and items belonging to the tail as *unpopular* ones. In fact, item migration along the long tail distribution over time is the starting point for the Oblivion Problem, which is described in the next section.

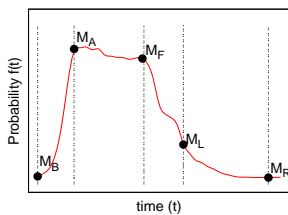
## 3. THE OBLIVION PROBLEM

In the last section we discussed as the popularity distribution of an item stems from the consumption habits of all its users. Conversely, we may adopt a similar concept when modeling the popularity of products for a given user, that is, we may define individual relevance distributions for each user. That is, let  $S$  be the set of all items consumed by a particular user  $U_i$  in a given period of time  $M_c$ . Given a non-increasing frequency distribution of the consumption of those items by  $U_i$  during  $M_c$ , the most consumed items, thus the most relevant ones, are on the distribution head while the remaining are distributed along the tail, following a long-tail shape. This distribution is a simple model for determining "successful" items for user  $U_i$  during  $M_c$ .

Considering the age of each item  $I_j$  as a relevant factor that affects how often  $I_j$  is consumed by  $U_i$ , the older  $I_j$  is for  $U_i$  the

less relevant it becomes, moving gradually toward the tail of the distribution defined for  $U_i$ . Thus, the temporal dynamics, the variety of items that arise, and the competition among them for the user preference results in successful items being forgotten, which makes them less and less visible over time. Therefore, we define as **oblivion** this natural shift of successful items from the head of individualized distributions toward the tail over time.

More formally, we define the Oblivion Problem based on the items movement on the long tail distribution of items relevant to each user. As discussed in Section 2, we divide this type of distribution into three distinct regions, delimited by the *Fall point* ( $F$ ) and the *Lapse point* ( $L$ ). From these definitions, we may better model the movement of items along this distribution over time. In this sense, we can define a probability density function  $f_{i,j}(t)$  of each item  $I_j$  be consumed by a user  $U_i$  at a given time  $t$ , as illustrated in Figure 2. Such function is hypothetically impossible to be exactly defined, since it may include many unmeasured or even unknown variables, as well as subjective aspects inherent to the users.



**Figure 2: Hypothetical Probability Density Function for a user  $U_i$  listening to an item  $I_j$  over time.**

It is interesting to notice that, despite the possible approximations to function  $f_{i,j}(t)$ , we can define five distinct critical time moments for this function. The first one refers to the *Birth Moment*  $M_B$  of  $I_j$  for  $U_i$ , and represents the first moment at which  $I_j$  was consumed by  $U_i$ . The second moment, called *Ascension Moment*  $M_A$ , comprises the moment at which the frequency that  $U_i$  consumes  $I_j$  exceeds the value found at the *Fall point* on the items relevance distribution of  $U_i$ , as illustrated in Figure 1 (a). The third moment  $M_F$  refers to the *Fall Moment* and it is related to when  $I_j$  cross back the *Fall point* towards the tail, losing its relevance. The fourth moment represents the *Lapse Moment*  $M_L$  and it is associated with the moment that  $I_j$  has crossed over the *Lapse Point*  $L$  on the relevance distribution towards the tail. Finally, we have the *Rescue Moment*  $M_R$ , which refers to the moment at which the probability  $f_{i,j}(t)$  becomes again greater than the probability found at the moment  $M_L$ . It is important to mention that  $f_{i,j}(t)$  may present distinct behaviors along these five moments. Between  $M_B$  and  $M_A$ , we have the displacement period of the item toward the head, showing that  $I_j$  is becoming relevant to  $U_i$ . In the period between  $M_A$  and  $M_F$ ,  $I_j$  is among the most frequently consumed items by  $U_i$ . Subsequently, we have between  $M_F$  and  $M_L$ , the period in which  $I_j$  is consumed much less frequently. After, the period between  $M_L$  and  $M_R$  is defined as the period during which  $I_j$  becomes unpopular to  $U_i$ , presenting a very low probability of being consumed. Thus, this represents the period during which the items become forgotten. In fact, even though, theoretically, the probability of  $I_j$  be consumed during this period is greater than zero, in practice  $I_j$  is not consumed, becoming effectively forgotten. Finally, we have the period after  $M_R$  when  $I_j$  again becomes potentially relevant to  $U_i$ .

It should be highlighted that not all items  $I_j$  will necessarily present the five defined moments. In fact, the vast majority will not reach the *Ascension Moment* and some of them will not have a *Rescue Moment*. Probably a significant part of the items that exceed

the *Lapse Moment* may never be rescued simply by representing a mismatch w.r.t. the user taste, which changes dynamically. Further, we are not assuming that the function  $f_{i,j}(t)$  exhibits a monotonic behavior. In fact, some items may have more than one *Ascension Moment*, for instance, presenting a periodic behavior. At this way, we can define the Oblivion Problem as the problem of determining the *Rescue Moment*  $M_R$  for each item  $I_j$  that has achieved, at least once, the *Ascension Moment*  $M_A$  in the past.

Clearly this problem represents a challenging task, since it consists of predicting **which** items and **when** they must be recommended to users again. Since not all items in the tail are likely to be rescued, it is also important to know the exact moment to recommend them. Premature recommendations may be ineffective, since users may still be “tired of” the recommended items. On the other hand, late recommendations may no longer be of users interest given the evolution of their tastes. Moreover, the task of identifying a subset of relevant items from the tail is a challenge by itself. In addition to choose among a huge range of items, the recommendations domain is inherently dynamic, since both environment and users evolve over time. Thus, in a music scenario, for example, people who used to like *Cindy Lauper* long ago no longer like listening to her today. New songs come out and remixed versions of old songs may become more attractive to a given user. Finally, it is important to consider the trade-of between recommending forgotten items and new ones, since the overuse of forgotten items may make the recommendation even less diversified.

A relevant aspect to point out is that the Oblivion Problem may be defined considering two perspectives: individual and global. From an individual perspective, we are interested in finding the *Rescue Moment* of items for each user, considering the individualized relevance distributions. Thus, the same item might have different rescue moments to distinct users, or even not be relevant to others. The rescue of forgotten items, in this case, improves the ability of personalizing recommendation services. In turn, the global perspective aims to identify, at each moment  $t$ , what the best items, forgotten by the system as whole, to be rescued are, given the traditional popularity distribution of items in the system at the moment  $t$ . That is, we are interested in identifying the items that were collectively successful in the past and exhibit a high probability of being successful again. In the music scenario, for instance, this perspective has a great utility for record labels, assisting them in the following question: Which music to be remixed at the time  $t$  is likely to become a hit?

In this work, we evaluate specifically the Oblivion Problem in music recommendation domains, given an important characteristic inherent to this sort of domain. Considering the scenario in which the focus is on the individualized preferences, it is believed that, in general, the frequency that a user might listen to a particular song is significantly higher than his or her willingness to watch a given movie or to read a given book again. Therefore, rescuing a particular song seems to have a different impact, probably more promising, than retrieving an old movie or book. Probably, the reason for such difference is related to the time required to listen to a song compared to watch a movie or to read a book.

## 4. METHODOLOGY OF CHARACTERIZATION

In this section, we present a characterization methodology, for recommendation domains, that assesses empirically the two main issues related to the Oblivion Problem: its existence and utility in real scenarios. Therefore, we divide our methodology into 2 main steps, namely: **Problem Verification** and **Utility Analysis**. It is noteworthy that, since the Oblivion problem is stated regarding a global and an individualized perspective, such methodology can be

applied for evaluating both of them. All metric descriptions were designed considering the global perspective, but for individualized analyses all we need to do is to ignore the summarization process executed by mean or median calculations and distributions.

## 4.1 Problem Verification

Our first task consists in verifying the existence of the Oblivion Problem in real scenarios. Thus, measuring and understanding how this problem manifests itself is a primary goal. To this end, we define five metrics that, together, provide evidences of the natural process of forgetfulness that occurs in recommendation scenarios. Table 4.1 defines each of these metrics and we discuss them next.

The *Mean Individualized Relevance* allows us to verify whether, in fact, items become less frequently consumed by users over time. The *Mean Inter-consumption Interval*, in turn, aims to identify how the interval between consecutive consumption of an item behaves over time. If this interval increases, it means that users tend to consume the item less and less frequently, until they effectively stop consuming it. Considering now, the *Mean Items popularity*, we aim to verify whether items become less popular or not over time in the system. That is, in addition to a given item becoming less relevant to each user individually, we also investigate whether it becomes globally forgotten by the system, since fewer users continue to consume it over time. In a complementary way, the *User Mean Age of Items* verify whether there is an increasing, stable or decreasing consumption behavior of the “age” of the consumed items over time, considering the moment at which each user first consumes each item. If such behavior is stable, decreasing or increasing at a rate lower than the actual aging rate of items and users in the system, we have a scenario in which items known for a long time by the users become forgotten. Similarly, the *System Mean Age of Items* analyzes this aging rate of the consumed items, but now considering the first time that each item appeared in the system. In this case, a decreasing behavior demonstrates that users also tend to focus their consumption on recently released items. Finally, the *Age Correlation* aims to verify whether items consumed by the first time by a given user are new or old in the system, showing up their interest in finding old items.

For sake of analysis, if a domain presents a decreasing relevance of items over time; an increasing inter-consumption time interval between consecutive consumptions of an item for a same user; such items still exhibit a descending popularity over time; and users present a stable consumption pattern focused on recent items in the system; we have a clear picture of oblivion. In this case, the rescue of forgotten items might be an important strategy to diversify the recommendation.

## 4.2 Utility Analysis

Once we verify that the Oblivion Problem happens in a given domain, the next step consists in checking the relevance of the forgotten items to RSs. In fact, may forgotten items be useful to recommendation in this sort of domain? This methodology step is concerned with answering this question. We define in Table 4.2 the main metrics related to the Utility Analysis and discuss them next.

Through the *Percentage of Successful Items*, we aim to analyze the diversity of items regarded as relevant for each user at each moment, based on the premise that a moderate diversity is better for popular items. It should be verified since, for a large diversity of successful items, users may behave dynamically, and, as a consequence, most of such items may not stay relevant for many users. In this case, recommending these items might worsen the accuracy of the recommendations. On the other hand, a very restricted range of successful items helps very little in diversifying recommendations, given the small number of distinct items. Considering the *Probability of Continuous Return*, we aim to determine whether,

Metric	Description
Mean Individualized Relevance (PV-1)	First, we define for each item $I_j$ its “birth” moment in the system as the first moment at which $I_j$ has been consumed by any user. Later, for each age $A_t$ of $I_j$ we count how many times, on average, $I_j$ was consumed by the subset of users who have consumed it on age $A_t$ . Then, we normalize the frequency found for each item $I_j$ , on each age, by the highest value found for $I_j$ throughout the period of analysis. Finally, we define the mean individualized relevance on each item age $A_t$ as the mean of the normalized frequencies of all items analyzed at $A_t$ .
Mean Inter-consumption Interval (PV-2)	Let $I$ be the set of all items consumed by a given user $U_i$ . For each item $I_j \in I$ , we define its “birth” to the user $U_i$ as the first moment at which $I_j$ has been consumed by $U_i$ . Then, we define the Inter-consumption Interval for each age $A_t$ of $I_j$ (i.e., $A_t$ is seen as the user age of $I_j$ ) as the number of time units between the age $A_t$ and the last moment at which $I_j$ was consumed by $U_i$ . We repeat this process for all users of the domain and, finally, we calculate a mean time interval between consecutive consumptions at each age $A_t$ considering all items $I_j$ analyzed for all users.
Mean Items popularity (PV-3)	For each item $I_j$ , first, we define its system age at each moment $M_t$ of analysis, considering as its “birth” moment in the system the first moment at which $I_j$ was consumed by any user. Thereafter, we identify the distinct number of users who have consumed $I_j$ at each distinct age $A_t$ . Then, we normalize the value found on each age for each item $I_j$ by the highest value obtained for $I_j$ in a single age. Finally, we define the mean for all items at each age $A_t$ as the Mean Items popularity at $A_t$ .
User Mean Age of Items (PV-4)	For each user $U_i$ , we define his age in the system since the first time he or she has consumed an item. Considering the items, we set its age since the first time it was consumed by $U_i$ in the system. Therefore, for each user age $A_t$ we calculate the mean age of all items consumed by $U_i$ at $A_t$ . Finally, we define the mean age among all users at each age $A_t$ as the User Mean Age of Items at $A_t$ .
System Mean Age of Items (PV-5)	Again, we assign to each user his or her age in the system since the first time he or she has consumed an item. However, we define the age of each item considering the first time it has been consumed in the system by any user. Therefore, for each user age $A_t$ we calculate the mean age of all items consumed by $U_i$ at $A_t$ . Finally, we define the mean age among all users on each age $A_t$ as the System Mean Age of Items at $A_t$ .
Age Correlation (PV-6)	For each item $I_j$ , we define its system birth as described before. Then, for each user $U_i$ , we define the user birth of $I_j$ as the first moment at which $I_j$ has been consumed by $U_i$ . Later, we calculate the differences between both moments of birth of each item $I_j$ for each user $U_i$ . Finally, we generate a probability distribution of these values, which we call Age Correlation.

Table 1: Metrics for Problem Verification (PV)

compared to the behavior presented by new items, old items still consumed by the users are likely to be consumed for a longer time. Although these old items do not consist of forgotten ones, this analysis demonstrates the relevance of old items compared to new ones for each user individually. If an RS rescues forgotten items with similar relevance, it would be expected a similar behavior for the probability of these items be listened continuously as well. We may say the same about the analysis of the *Probability of Continuous Frequency*. We aim to verify whether old items are consumed more often than new ones, assuming that the same might occur with forgotten items properly selected.

The Analysis of *User Age Distribution of Items* aims to verify whether a set of old items, consistently, belongs to the consumption set of the users in different periods. If so, in fact, rescuing old items not only diversifies the items consumed by users, but also this particular subset of old items. In a complementary way, the analysis of *System Age Distribution of Items* aims to verify whether items considered new for each individual user are also new to the system, or if they are actually old items just discovered by the users. This analysis makes possible to contrast individual behaviors to global trends in terms of consumption of new and old items. Therefore, from this set of metrics we determine a relevance measure of forgotten items for each scenario, allowing us to check the potential of techniques focused on rescuing forgotten items.

It should be noted that our methodology is based on comparative analyses between distinct periods. Thus, for verifying if a given scenario suffers from the Oblivion Problem it is necessary some assessments over time, in order to identify behavioral trends. Absolute values of these metrics, by themselves, are not enough for such purpose. In order to demonstrate the applicability of the proposed methodology, we evaluate it on a relevant music scenario in the next section.

## 5. LAST.FM: A CASE STUDY

### 5.1 Dataset



Metric	Description
Percentage of Successful Items (UA-1)	For each user $U_i$ , at each age $A_t$ in the system, we define the ratio between the total number of distinct items recognized as successful ones for $U_i$ at $A_t$ and the total number of distinct items consumed by $U_i$ at $A_t$ . An item $I_j$ is considered successful for a user $U_i$ , at a given moment $A_t$ , if their frequency of consumption is $X$ standard deviations greater than the mean frequency of consumption of all items consumed by $U_i$ at $A_t$ .
Probability of Continuous Return (UA-2)	First, we define two distinct sets of items for each user $U_i$ at each moment $M_a$ of analysis. The first set comprises the new items for $U_i$ , since they were consumed by $U_i$ for the first time on less than $X$ time units before the moment $M_a$ . In the second set, we have the old items, consumed by $U_i$ for the first time on more than $Y$ time units before $M_a$ , with $Y > X$ . In each set, we define for each item $I_j$ the largest continuous time interval, considering all distinct moments $M_b$ (such that $a \geq b$ ), that $I_j$ has been consumed by $U_i$ . Then, we generate a probability distribution of these interval values found in each set for all users, obtaining two distinct curves.
Probability of Continuous Frequency (UA-3)	As defined for the metric UA-2, we assign to each user a set of new items and other of old ones. In each set, we define, for each item $I_j$ , the mean frequency of consumption at the time interval identified as the largest continuous one for $I_j$ , starting from the moment of analysis $M_a$ . Then, we generate a probability distribution for the frequency values found in each set for all users, obtaining two distinct curves.
User Age Distribution of Items (UA-4)	Initially, we select, for each user $U_i$ and time moment $M_a$ , a set $S$ of the $K$ items most consumed at $M_a$ . After, for each item $I_j \in S$ , we define the age of $I_j$ at $M_a$ , considering as "birth" moment the first moment at which $I_j$ has been consumed by $U_i$ . Finally, we plot the percentage of occurrence of each age in $S$ at each analyzed moment $M_a$ .
System Age Distribution of Items (UA-5)	This analysis is exactly the same as described for the metric UA-4, except that we consider, to calculate the age of each item $I_j \in S$ , its "birth" as the moment that $I_j$ has been first consumed in the system.

Table 2: Metrics for Utility Analysis (UA)

In this section we present the dataset used in our analysis. We use a dataset from the *Last.Fm* system<sup>1</sup>, which is a UK-based Internet radio and music community website, founded in 2002. It has claimed over 30 million active users when we collected the dataset. It was also estimated that *Last.fm* has more than 27 million different tracks and 12 million distinct artists in its database<sup>2</sup>. As *Last.Fm* represents one of the largest musical community in the world, and since all data are readily available in the WEB, it is a good data source for music recommender systems.

Our analyses were performed on a data sample from *Last.Fm*. These data were collected through an API provided by *Last.fm*<sup>3</sup>. This API allows us to collect information related to several data entities such as artists, albums, tracks, and users, among others. We consider as relevant to our analysis only information related to users, artists, and tracks. Such information was collected for a set of 104,770 distinct users, randomly selected, 217,774 different artists and about 2 millions distinct tracks listened to by the collected users, spanning the period from 11/12/2008 to 04/26/2009.

## 5.2 Characterization of the Oblivion Problem

In this section, we present the results from the application of the methodology presented in Section 4 to our *Last.fm* data sample.

### 5.2.1 Problem Verification

Starting our analysis by the Verification Problem step, we obtained the results shown in Figure 3. Analyzing, first, the measure **PV-1**, Figure 3 (a), we observe that *Last.fm* appears as an environment with decreasing frequency of song consumption over time. That is, the longer the songs are in the system, the lower the frequency users listen to them. Furthermore, by analyzing the measure **PV-2**, Figure 3 (b), we note that the interval between consecutive consumptions becomes larger as the song ages for each user. Therefore, the longer a user knows a song, in general, the higher is the interval between consecutive consumptions, until the user stops listening to these songs. Clearly, these results show that, over time, old songs loose relevance for each user at an individual basis.

Considering the *Mean Items Popularity* (**PV-3**), as shown in Figure 3 (c), we note that the songs also become less popular globally

<sup>1</sup>Available at <http://www.last.fm/>

<sup>2</sup>This information were retrieved from the *Last.Fm Radio Announcement*, on 03/25/2009, available at <http://blog.last.fm/2009/03/24/lastfm-radio-announcement>

<sup>3</sup><http://www.last.fm/api>

in our *Last.fm* sample. That is, besides becoming less relevant over time for each user individually, in fact, the songs become less relevant globally for the system. Thus, over time, most of these songs may become forgotten, both by users and by the system as a whole, and, in general, even by the RSs. It is also noteworthy in this analysis the growth in popularity observed for the oldest songs (i.e., songs older than 19 weeks). Such behavior occurs as a result of the existence of a very restricted set of songs that are continuously listened by the users. Some of these songs are from artists such as *Beatles*, *U2* or *Michael Jackson*, which, interestingly, present a more stable popularity over time.

Our next analysis refers to the *Mean Age of Items*, considering both user (**PV-4**) and system (**PV-5**) perspectives. According to the users perspective, Figure 3 (d), we found that consumed items age much more slowly than their actual aging rate in the system. That is, in general, the users consumption is focused on new items, rather than on items already known by him. Thus, the average age of consumed items in each week has becoming increasingly distant from the actual aging rate of users and items in the system (i.e., solid line in the plot). It is also important to emphasize that, although the consumed items age slowly, they do get old since the mean age curve is not a straight line parallel to the  $X$  axis, which shows that, even focusing their consumption on new items, users in *Last.fm* consume some old items. Thus, we may conclude that old items are part of the consumption behavior of users. Similar conclusions are obtained considering the system perspective, as shown in Figure 3 (e). Over time the mean age of consumed songs is lower than the actual users and items age, defining a curve below the continuous one plotted in the graphic. This shows that, besides focusing their consumption on items that users themselves know for a short time, some of those items are also recent in the system as a whole. In this case, however, the difference between the two aging rates is not significant. Thus, in general, old songs are slowly being abandoned, or "forgotten", and as new songs are consumed as soon as they are released.

Finally, we assess the correlation between the items' age for each user and the actual items age in the system. The plot in Figure 3 (f) represents this analysis. We observe that 35% of the differences between the actual items age and the items age recognized by each user, as defined in section 4, are smaller than 4 weeks. That is, more than one third of the items consumed by the users in our sample were in the system for less than 1 month. On the other hand, only 14% of the consumed items are older than 18 weeks (i.e., 4 months and a half) when a user has first consumed it. Therefore, based on the Verification Problem analysis, we describe *Last.fm* as a scenario in which there are clear evidences of the existence of the Oblivion Problem. Its users consume more often new items in the system and, over time, consume a given item less and less frequently. In addition, the items, in general, become quickly less popular as they age. We also found that the users consumption, in its majority, consists of new items in the system that were first consumed recently.

### 5.2.2 Utility Analysis

We start our Utility Analysis by the *Percentage of Successful Items* (**UA-1**), such as shown in Figure 4 (a). For this analysis, we consider as time granularity one month and the number  $X$  of standard deviations equal to 2. That is, for a song being successful in a given month  $M$  for a user  $U_i$ , it should exhibit a frequency of occurrence, at least, two standard deviations higher than mean frequency of consumption of  $U_i$  at  $M$ . As we can observe, the mean percentage of distinct successful songs for each user of our sample is about 8% of the song set listened by him or her each month. However, we also found higher standard deviation values. Although we can identify only very few successful items for some

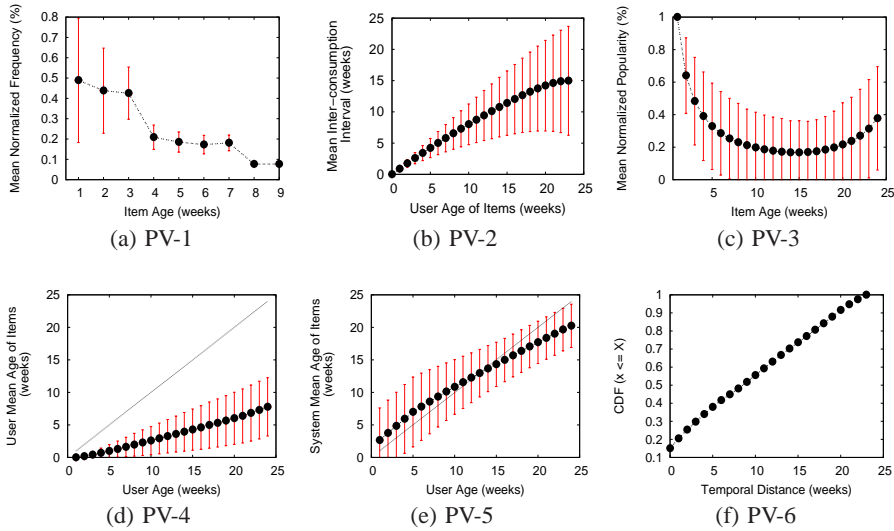


Figure 3: Results for Problem Verification

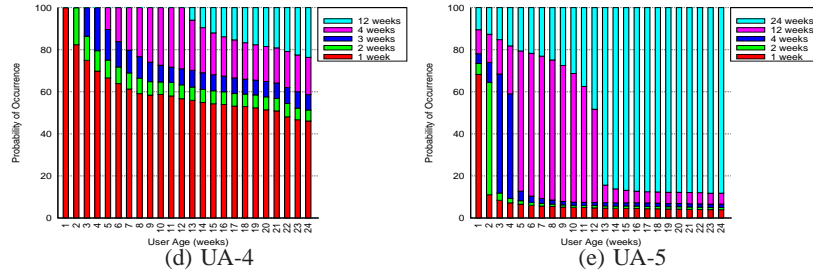
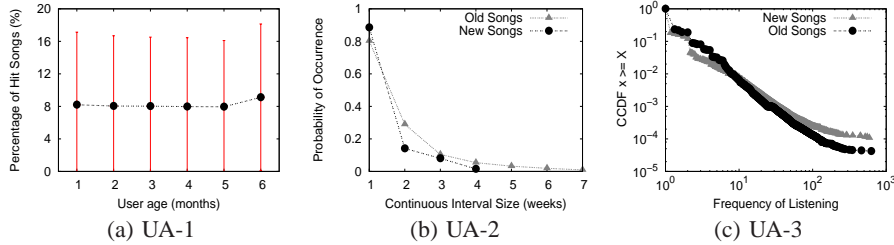


Figure 4: Results for Utility Analysis

users, in general, it is possible to identify a meaningful number of successful items, defining a rich set of relevant items over time for diversity strategies.

Considering the *Probability of Continuous Return (UA-2)*, we found a very distinct behavior, regarding old and new songs, as shown in Figure 4 (b). While new items tend to be listened for a short period of time, songs known for a long time are consumed continuously for larger periods. This fact demonstrates that the subset of older songs, even not being the majority, represents more consistently the users taste over time. In a complementary way, the plot in Figure 4 (c) shows the complementary cumulative distribution function (CCDF) for the *Continuous Frequency Analysis (UA-3)*. As expected, new songs present a higher probability of being listened more frequently, since the users are still “learning” them. Contrasting this findings to those presented in the plot of Figure 3 (a), we can conclude that, although forgotten songs are likely to be listened, they will not be listened as intensively as the moment when the users knew these songs.

Considering now the *Analysis of Age Distribution of Items* for both user (UA-4) and system (UA-5) perspectives, we also have interesting results, as shown in Figures 4 (d) and (e), respectively.

From the user perspective, we found that, despite the consumption behavior of *Last.fm* users is focused on recently discovered items, as the users age they tend to listen to old songs more often. That is, over time, users tend to keep listening to songs known for a long time. This finding is particularly relevant for us, since it demonstrates that a portion of songs being consumed by a user is composed of old songs. Consequently, forgotten items would be relevant even to diversify this specific subset of songs.

Finally, we focus on the system perspective, as shown in Figure 4 (e), which requires a careful analysis. First, it is important to remember that we do not know the actual age of the items in the system. As we consider as the “birth moment” of each item the first time it has been consumed by any user of our sample, it is expected that at age 1 most of the users consumes 1 week old songs. However, we can see that over time users fail to bring new items to our sample and start to listen to items that already belong to it. That is, although users look for new items in the system, they increasingly consider novel items already present in the system. Further, as the number of existing items grows over time, the probability of a user to find a recent addition to the system that are also relevant to her or him decreases.

In summary, we can conclude that, despite the continuous and fast changes in terms of user taste, expressed through the listened songs, the few remaining old songs that he or she still listens regularly represent exactly the subset of songs listened to by a longer period of time. In addition, over time, *Last.fm* users tend to anchor their consumption more and more on the older songs in the system. Consequently, these songs represent a relevant set of information about users, demonstrating the significance of old items in the music recommendation domains. Thus, we believe that rescuing relevant forgotten songs would be particularly promising for diversifying not only the overall list of songs listened by each user but also this subset of old songs.

## 6. DISCUSSION

Obviously, once we have shown the existence and relevance of the Oblivion Problem, our next step is to identify forgotten items, and then select a subset of most relevant ones for RSs. A straightforward way of identifying the items forgotten by a user, or by the system as a whole, can be derived from the problem definition. After identifying the Ascension and Lapse moments of each item for each user, we need only to select the items that have had at some moment, at least, one *Ascension Moment* and are no longer being consumed by users (i.e., are beyond the *Lapse Moment*). The selection process of a subset of relevant items, since not all forgotten items are likely to be recommended, is more challenging. As the probability function  $f_{i,j}(t)$ , of a user  $U_i$  consumes an item  $I_j$ , itself is, in practice, impossible to be modeled, we need to define and evaluate heuristic strategies, defining such probabilities over time and then select the items with highest probabilities at each moment  $t$ . Further, considering the trade-off of recommending old and new items, in order to achieve high accuracy and diversity rates simultaneously, a final step is related to incorporate the selected forgotten items into traditional RSs.

It is also interesting to discuss novelty issues related to the Oblivion Problem. As the primary goal is to recommend items already known by users, our first aim is to consider that there are no novelty gains in such recommendations. But, actually do forgotten items represent any sort of novelty to users? In the literature, novelty of a piece of information refers to how different it is regarding “what has been previously seen”, by a specific user, or by a community as a whole [4]. However, we argue that it is an incomplete statement, missing a relevant issue: for how long an information is known. We can redefine novelty of a piece of information as how different it is with respect to “what has been previously seen **in a recent moment properly identified**”. That is, relevant items that used to be consumed long ago could represent, in some sense, a degree of novelty, since we are assuming that hardly the users would remember by themselves most of those “lost” items. Therefore, a promising direction consists in investigating in deep the novelty issues related to the Oblivion Problem.

### 6.1 The Cold Start Duality

Going one step further in the discussion about solving the Oblivion Problem, we emphasize an interesting view that could help us in this task: its duality with the Cold Start problem [14]. Considering the Oblivion Problem, the role of recommendations extrapolates the task of indicating to users items that he or she may eventually like. Now it includes rescuing relevant items that, due to the RSs inadequacy for it, the natural growth of the tail and the competition process among the items by the user attention, get lost amid a wide range of options. Such rescuing aims to bring up old relevant items and to provide, besides the diversity improvement, surprise and good memories to the users. However, the challenge of recommending forgotten items relies on the fact that both domain

and users evolve over time. Furthermore, such items have usually a very long past, but no recent information. As current RSs take into account, in general, only most recent information for providing the recommendations, forgotten items, controversially, suffer from lack of information. Therefore, the Oblivion Problem may be seen as the dual of the *Cold Start* problem, in which the difficulty of recommending new items also occur due to lack of information, but now due to lack of past.

Cold Start is a classic problem in traditional collaborative filtering recommendation [14]. It is hard to generate recommendations for new items because there is not enough experience data about the new items to make reliable correlations with other items. Pure user-oriented collaborative filtering cannot help in a cold-start setting, since no user preference information is available to form any basis for recommendations. A usual solution for this problem consists in using a content based recommendation approach, or even a hybrid approach, combining collaborative filtering with content based techniques [1].

Given this duality between the Cold Start and the Oblivion Problem, a first attempt to address the later would be to apply the same solutions currently used for Cold Start. For instance, using content-based techniques we can derive weights for the forgotten items according to their similarity with items currently consumed by the users. Also, hybrid methods may also be useful to retrieve forgotten items. However, the application of such methods to this scenario is even more challenging, since the attributes that would be selected to describe the content of the items must have a time-invariant character. This is an important requirement since the users taste evolves. Therefore, using information such as artist name and music genre among others may not be a good choice given that users might change their interests, in particular w.r.t. artists or genres over time. Thus, temporal correlations between items is an aspect to be incorporated into hybrid methods, in order to address the Oblivion Problem. Further, we believe that most of the evaluation metrics and strategies proposed for Cold Start problem could, with some changes, be applied to evaluate RSs designed to deal with the Oblivion Problem.

## 7. RELATED WORK

The growing relevance of Recommender Systems (RSs) in various domains have boosted research on this topic recently. Despite the advances achieved, there are still several challenges associated with this task, such as a proper user taste modeling, huge volume of items, and sparsity or lack of information about users [1]. Among these challenges, we highlight two that have received prominent attention: diversity on recommendation lists and temporal dynamics inherent to this kind of domains.

Recent studies found that diversity, combined with a high accuracy rate, is a relevant requirement associated with the usual taste similarity issue [16]. In [20], for instance, the diversification of recommendation lists is extensively addressed. The authors propose a similarity metric using a taxonomy-based classification and use it to compute an intra-list similarity metric that determines the overall diversity of the recommended list. Another study has examined the conditions in which diversity can be increased without loss of similarity, and presented an approach to determine such similarity, preserving increases in diversity when possible [12]. In [19], the diversification goal is seen as a binary optimization problem and a solution strategy to this problem consists in relaxing it to a trust-region problem. Further, the role of diversity in traditional recommender systems is clarified in [11], highlighting the pitfalls of naively incorporating current diversity enhancing techniques into existing recommender systems.

Considering efforts on temporal dynamics, it is almost consen-

sual that accurately capturing user preferences over time is a major challenge for RSs. Therefore, numerous studies have tried to characterize, to model and to propose new strategies to deal with this problem without penalizing accuracy [10]. Ding [6] presented a novel algorithm to compute the temporal weights for items so that older items get smaller values. This approach has a disadvantage due to latest data are not always important while old data are not trivial all the time. Recently, Koren [9] predicted movie ratings for Netflix by modeling the temporal dynamics via a factorization model. Analogously, many time-evolving models [17] introduced time as a universal dimension shared by all users. It is remarkable that simple correlations over time are typically not meaningful, since users change their preferences due to different external events. Some studies argue that the time dimension is a local effect and should not be used for comparison among users [18].

Unlike other work, we address both aforementioned challenges, diversification and temporal dynamics, simultaneously in the context of the Oblivion Problem. That is, we address temporal dynamics using a new strategy that enhances the diversity in RSs by exploiting the subset of items consumed by users in the past (i.e., forgotten items), thus providing a differentiated and promising source of recommendations compared to traditional techniques. In fact, most techniques that focus on temporal dynamics aim to define a set of items that best match the most recent behavior or desires of the users [5]. We argue, however, that some of these users' wishes might be met by old items. Further, given the large amount of existing information about those forgotten items, we believe that such items may enhance the usual trade-off between diversity and accuracy in RSs. A study that deserves a deeper analysis, given its similarity to our work w.r.t. goals, is presented in [10]. However, its authors propose techniques that, by looking at recent recommendations provided to users, avoid that the same items are recommended to users over and over again through time.

## 8. CONCLUSIONS AND ONGOING WORK

In this paper, we present a differentiated and promising strategy to increase diversity in recommendation lists, based on the temporal dynamics inherent to recommendation domains. Such strategy defines a new problem for recommender systems, the Oblivion Problem, that aims to identify which items have been successful in the past for a given user (i.e., forgotten items) and exhibit a high probability of being consumed by this user again in the present. Besides formalizing this problem, we propose a methodology for its identification in real scenarios. The application of our methodology on a sample of *Last.fm* demonstrates the existence of the Oblivion Problem in this scenario, as well as the potential usefulness of recommending forgotten items in this case.

An immediate step of our work consists in developing techniques that are able to automatically identify the forgotten items relevant to each user, individually, or to the system as a whole. Later, we aim to build a recommender system that incorporates such items to recommendation lists, without penalizing the accuracy, in order to increase the diversity in the RSs. Finally, we highlight as another relevant direction the problem verification in distinct scenarios, such as recommendation of contacts in social networks, which present characteristics different from those observed in music recommendation.

## 9. ACKNOWLEDGMENTS

This work is partially supported by CNPq, Finep, Fapemig and InWeb – the Brazilian National Institute of Science and Technology for the Web.

## 10. REFERENCES

- [1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE*, 17(6):734–749, 2005.
- [2] C. Anderson. *The long tail*. Gramedia Pustaka Utama, 2006.
- [3] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–470, 2002.
- [4] P. Castells, S. Vargas, and J. Wang. Novelty and diversity metrics for recommender systems: Choice, discovery and relevance. In *Proc. of International Workshop DDR*, pages 29–37, 2011.
- [5] T. Cebrián, M. Planagumà, P. Villegas, and X. Amatriain. Music recommendations with temporal context awareness. In *Proc. of the 4th RecSys, RecSys '10*, pages 349–352, New York, NY, USA, 2010. ACM.
- [6] Y. Ding and X. Li. Time weight collaborative filtering. In *Proc. of the 14th C*, pages 485–492. ACM, 2005.
- [7] K. Hoashi, K. Matsumoto, and N. Inoue. Personalization of user profiles for content-based music retrieval based on relevance feedback. In *Proc. of the 11th ICM*, pages 110–119. ACM New York, NY, USA, 2003.
- [8] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. of ICDM-08, 8th IEEE ICDM*, pages 263–272, 2008.
- [9] Y. Koren. Collaborative filtering with temporal dynamics. In *Proc. of the 15th ACM SIGKDD*, pages 447–456. ACM New York, NY, USA, 2009.
- [10] N. Lathia, S. Hailes, L. Capra, and X. Amatriain. Temporal diversity in recommender systems. *Proc. of SIGIR'10*, pages 210–217, 2010.
- [11] L. McGinty and B. Smyth. On the role of diversity in conversational recommender systems. *Case-Based Reasoning Research and Development*, 2003.
- [12] D. McSherry. Diversity-conscious retrieval. *Advances in Case-Based Reasoning*, pages 27–53, 2002.
- [13] Y. Park and A. Tuzhilin. The long tail of recommender systems and how to leverage it. In *Proc. of the ACM RecSys*, pages 11–18. ACM, 2008.
- [14] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proc. of the 25th SIGIR*, pages 253–260. ACM Press, 2002.
- [15] B. Schwartz. *The paradox of choice: Why more is less*. Harper Perennial, 2005.
- [16] B. Smyth and P. McClave. Similarity vs. diversity. *Case-Based Reasoning Research and Development*, pages 347–361, 2001.
- [17] J. Sun, C. Faloutsos, S. Papadimitriou, and P. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proc. of the 13th SIGKDD*, pages 687–696. ACM, 2007.
- [18] L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, and J. Sun. Temporal recommendation on graphs via long-and short-term preference fusion. In *Proc. of the 16th ACM SIGKDD*, pages 723–732. ACM, 2010.
- [19] M. Zhang and N. Hurley. Avoiding monotony: improving the diversity of recommendation lists. In *Proc. of the ACM RecSys*, pages 123–130. ACM, 2008.
- [20] C. Ziegler, S. McNee, J. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proc. of the 14th WWW*, pages 22–32. ACM, 2005.

# A Framework for Recommending Collections

Jennifer Golbeck  
College of Information Studies  
University of Maryland, College Park, MD  
jgolbeck@umd.edu

Derek L. Hansen  
College of Information Studies  
University of Maryland, College Park, MD  
dlhansen@umd.edu

## ABSTRACT

To date, the vast majority of recommender systems research has addressed the problem of recommending individual items that the user will like. Recommending collections of items rather than individual items is an important open space of research in the recommender systems community. In this paper, we present a comprehensive framework for describing and evaluating collections of items. This framework is designed to be domain independent and applicable to any collection recommendation problem. Our framework includes a categorization scheme for describing collections and a list of features upon which a collection can be evaluated. We present a number of examples that showed how these different attribute and evaluation techniques can be combined and applied in a given domain. We then discuss issues relevant to the building of these systems. This includes challenges in obtaining data about users' preferences for collections. We propose methods that include obtaining and analyzing existing collections from websites and developing multi-player online games to generate data about replacements and preferences. In addition, we look at how collection recommenders could be used to assist users either by creating collections from scratch or by assisting users in their own collection creation tasks. We believe this framing of an important problem will lead to new research in the development and evaluation of algorithms for recommending collections in interesting applications and with cross-domain applicability.

## Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous

## General Terms

Human Factors

## Keywords

recommender systems, collections, collection recommender systems

## 1. INTRODUCTION

To date, nearly all research in recommender systems has focused on recommending individual items that the user will like. This has been successful and is useful in a wide range of domains. Recommending collections of items as a distinct task from recommending individual items also has broad applicability, but has received very little attention in the literature.

There are many reasons to recommend collections of items. In many cases, items in a collection are complementary so that the value of each item is increased when it is combined with other items (e.g., ice-cream, banana, and hot fudge). Even when items are not complementary, recommending a bundle of items can help extract more consumer surplus by essentially sorting individuals into groups with different reservation prices [1]. Information goods such as music, digital books, and software are particularly well suited for combining into collections. Those selling such collections can benefit from the "predictive power of bundling" which under many conditions can lead to increases in sales, efficiency, and profits when compared to selling the items individually [3]. The benefits of bundling items are suggested by the use of Amazon's "Buy Together Today" offers that provide one price for a bundle of items (e.g., two related books).

Many popular online systems thrive on user-generated collections. Collections not only are valuable in themselves, they often provide a meaningful activity that keeps participants coming back. Many major music websites facilitate the creation of user-generated playlists (e.g., imeem, Rhapsody). Tools like iTunes Genius automatically create playlists from music in your iTunes library, as well as recommend related music. They have received mixed reviews. Sites like Playlist.com, Mixpod, and MixTape.me create communities around the creation and sharing of music collections.

Yet, even these sites provide few tools to augment the playlist creation process through recommendations that consider the collection as a whole. In other domains collections are also common: Amazon has Listmania!, Flickr has Galleries, clothing stores such as Marie Claire allow you to create collections of clothing and accessories on a virtual model, recipe sites like AllRecipies have recipe boxes, and Colourlovers.com recommends colors that go well together. Nearly all of these sites currently only support manual creation of collections, often missing out on the opportunity to recommend items

that fit well with partially completed collections.

There are countless varieties of collections that can be recommended. Stock portfolios, playlists, menus, and hobby collections (e.g. Hummel figurines) are just a few diverse examples. The domain of a collection is certainly important for judging its quality. The domain includes the type of item being recommended, the environment in which the collection will be created and used, and by whom the collection will be created and used. However, a general framework that is domain independent has many benefits. It leads to a higher-level understanding of collections and allows recommendation strategies to be shared more easily between domains by identifying strategies and algorithms that work for certain classes of similar collections. For example, a technique for recommending meal plans for diabetics may also be useful in recommending stock portfolios because both are unordered collections treated as a unit that must meet a set of constraints.

Some top- $n$  collection recommenders have focused on the quality of the set of recommended items as a whole collection, particularly with respect to diversity of items [2, 4, 20, 12, 19]. We discuss this work further below. However, there are many aspects of collection quality to be considered beyond what has been treated in top- $n$  recommenders so far. In our previous work [7], we introduced some preliminary notions of collection recommendation. We focused on a certain subset of collections, of which mix tapes served as the canonical example. In this paper, we present a more comprehensive framework for describing collections by type and feature. We also present a full set of attributes by which the quality of a collection can be measured independent of its domain. Finally, we discuss challenges to collecting data needed to support the creation and evaluation of collection recommender systems and describe how the framework can be used in different ways to assist the user in building collections.

## 2. A FRAMEWORK FOR COLLECTIONS

A framework for collections must describe all the features of a collection that allow different types of collections to be compared and contrasted independent of their domain. These features do not address the quality of a collection; they simply describe it. We introduce four features in the framework: collection type - unit or selection, ordered or unordered, finite or infinite, and constrained or unconstrained. We also present six different attributes that can be measured to determine the quality of a collection: individual item ratings, order interaction effects, item co-occurrence effects, size, diversity, coverage, and balance.

### 2.1 Attributes of Collections

#### 2.1.1 Collection Type: Unit or Selection

Some collections are treated as a single unit to be used as and evaluated as a single item (e.g. an outfit for a person to wear). Other collections are designed to be drawn from (e.g. a library of books). Here, we elaborate more on this distinction.

*Unit Collections.* Many collections are used as a single item. When all items in a collection are used together, as though the collection is its own item, we call this a *unit collection*.

For example, a mixtape is made up of a series of songs, but the quality of the collection can be evaluated separately from the songs themselves. A mix tape that randomly pulls together songs from completely unrelated genres – some classical songs, some gangsta rap, some death metal, and easy listening – and presents them in no particular order may be considered to be of lower quality than a tape that has a theme to tie all of its songs together, with carefully selected ordering to provide smooth transitions between songs, and with a diverse yet compatible set of songs that are enjoyable both individually and in relation to one another. Stock portfolios, family meals, edited volumes, and the collection of readings in a syllabus are other examples.

*Selection Collections.* In contrast to a unit collection are collections that are not designed to be used all at once, rather they exist as a set from which the user can draw a subset of items when needed. We call these *selection collections*.

A library is an example of a selection collection. We can evaluate the quality of the collection as a whole by considering how well it meets the needs it was set up to address. A library of cookbooks, for example, is not designed to be used as a unit where every cookbook is used at once. Rather, individual books are selected out of the collection and used when needed. Music libraries, wardrobes, and menus are other examples.

#### 2.1.2 Ordered or Unordered

The order of items in a collection may be important or not. In the mix tape example above, ordering is very important to making a good tape. In other collections, order does not make sense, such as a stock portfolio or a collection of accessories for an outfit. Finally, depending on the domain, order may sometimes be important for a collection and not other times. A cookbook is a collection of recipes. One could argue that the order in which recipes appear in the cookbook is not important since the book is not read consecutively, but rather accessed at arbitrary points. On the other hand, if the book is treated as a unit, the ordering of items may tell a story or otherwise improve the experience of using the book, so it may matter.

#### 2.1.3 Constrained or Unconstrained

For certain collections, there are constraints that must be met in order for the collection to be useful. A stock portfolio must be within a certain range of risk. As a more complicated example, a medical diet must have a certain number of calories, nutrients, and a balance of protein, carbohydrates, and fat. Certain foods may also be excluded. When recommending a collection with constraints, each item must be evaluated with respect to these requirements before it is added to the collection.

Note that some combinations of constraints can lead to computationally intractable problems. If, for example, the user

is designing a daily meal plan and needs to select a set of foods that has at least 1,500 calories and cannot exceed 1,700 calories, where foods must come from a balance of categories (starches, vegetables, proteins) and also achieve the recommended daily allowances of a set of nutrients, the problem quickly begins to look like a variant of the Knapsack Problem which is NP-Complete [5]. Recommender systems are not designed to search for optimal solutions; they find preferences. Thus, when putting constraints on collections it is important to consider if a recommender system is the appropriate technique for selecting items. Recommenders should be used when constraints are simple and user preferences are important, not when finding items that meet the constraints is the difficult problem.

#### 2.1.4 *Finite or Infinite*

While no collection is truly infinite, we borrow the concept of finite and infinite horizons from game theory. In an iterated game, a finite horizon describes when the players know how many times the game will be played. An infinite horizon describes when the game is played such that the players do not know when it will end, or it is played so many times that the end seems so far away that the players treat it as though it will continue indefinitely. Translating this to collections, some collections have a fixed size that is small enough where users can consider the whole collection at once; we call these finite collections. Other collections are designed to be ever-increasing in size and these are what we call infinite collections. It is not that the set of items that make up the collection is infinite, it is that they are cycled through continuously without end.

To understand the distinction between a finite and infinite collection consider a music playlist of classical music. The playlist itself could be considered a finite collection that stands on its own. Now consider a system that continuously samples from the classical music playlist, which can be thought of as a "seed" collection. It may randomly select songs from the playlist or it may select songs to play next based on rules such as "don't play the same song twice in a row" or "play songs from different time periods." No matter the case, the entire music stream would be considered an infinite collection and it could be judged independently of the underlying collection of songs from which it pulls.

Most of the examples mentioned so far – cookbooks, mix tapes, stock portfolios – are finite. A radio station or an ongoing meal plan for an individual are examples of infinite collections.

## 2.2 Valuing Collections

To create systems that recommend collections of items, we must have a method of scoring a collection to determine if one collection is better than another. In this section we describe a set of measures that can be used to evaluate collections. Note that some of these evaluation methods will not apply to all type-feature combinations of collections.

### 2.2.1 *Individual Item Values*

In current research that recommends sets of items, the individual item value has been the primary – and often the only – concern. When creating a collection, including items that

the user will like will certainly make it better. Previous work has shown this for mix tapes, where collections with many highly rated songs outperformed those with many poorly rated songs [16]. Thus, collection recommenders should consider the user preferences for individual items. Evaluating the quality of a given item for a user is where the bulk of existing recommender systems research has focused [8]. Existing techniques can be used for this part of the evaluation.

Note that the tolerance for lower value items may be higher in a selection collection than in a unit collection because not every item in a selection collection need be used. Having a song that I don't particularly care for in my iTunes library (selection collection) doesn't lower the value of the entire collection as much as it would if it were included in a mix tape of 10 songs designed to be played straight through (item collection).

### 2.2.2 *Order Interaction*

In ordered collections, ordering can impact the quality of the collection in several ways. Absolute placement of an item can be important; some items may work better in a given position. For example, an overview article may fit best at the beginning of an edited collection of articles rather than in the middle or at the end. Similarly, some songs may work well as the first or last song in a mix tape. Or, songs with certain characteristics (e.g., "favorite" songs) may work best as first songs, a hypothesis our mix tape experiment discussed below. Note that this type of absolute placement order effect does not apply to infinite collections. It only applies to selection collections inasmuch as the absolute ordering helps with the selection process itself (e.g., items are listed alphabetically or sortable by other characteristics).

Relative placement of items to one another can also be important when ordering items in a collection. Two items may go very well together in a particular order, while other pairs may clash when placed in sequence. The relative ordering effects are typically independent of their absolute placement in the collection. For this reason they are applicable to infinite collections as well as finite collections. If two songs clash with one another, this is likely true if they are part of a mix tape (finite collection) or a radio station play sequence (infinite collection). Or, songs with certain characteristics (e.g., favorite songs) may work best as first songs, as demonstrated in earlier work [7].

### 2.2.3 *Item Co-Occurrence Effects*

Regardless of whether a collection is ordered, the interaction of items within it can affect its quality. Some items work well together and others do not. These co-occurrence effects are one of the most important factors in the success or failure of many collections.

It can be a complex task to evaluate co-occurrence effects. Even two items that both have high individual item ratings may not work well together. Someone might like chocolate and also like pickles, but not the two together. This is a rather intuitive effect when considering pairs, but gets more complicated when considering the quality of larger sets of items such as a triple.

For example, chocolate bars and graham crackers are a fine

combination; marshmallows and chocolate bars are also; and marshmallows and graham crackers are as well. None of these pairs are poor but neither are they exceptional. However, the combination of all three into a smore makes a much beloved snack for many people. The combination of all three items is better than would be indicated by looking at the three pairs. On the other hand, three items that are very good pairwise can make a bad triple. Consider building a research team of two professors and one graduate student. The professors may work well together, and each may work well with the student. However, all three may have trouble working together. The presence of a student may bring out some tension between the faculty members about who is in control, and the student may have trouble balancing work or contradictory instructions from them.

Similar scenarios can be made moving up from groups of three to four, and so on. While it is certainly useful to look at the compatibility of groups of two or even three items, this approach quickly becomes computationally difficult, requiring  $O(n^k)$  comparisons for groups of size  $k$ .

Co-occurrence effects are most relevant to unit collections, where each item is directly tied to other items in a whole. However they may apply to selection collections. For example, a music selection collection that includes many music genres (e.g., rap, country, and gospel) may lose credibility and be valued less by those who strongly dislike one of those genres, even if they would not select songs of that genre when using the collection.

#### 2.2.4 Size

For finite collections, the number of items in the collection may be important. Collections can be too big or too small, depending on the domain or purpose. Consider a collection of accessories for an outfit. Even if all of them work well together, there still may be too many for the collection to be considered good. On the other hand, a mix tape with only three songs would often be considered too short. Selection collections typically benefit from an increase in size since larger collections mean more options from which to choose. However, even selection collections can grow too large, making selections too challenging or time intensive.

#### 2.2.5 Item Distribution

For collections to be successful, they may need items to be distributed in certain ways. For example, having diversity among recommended items has been shown to be important. Similarly, in some domains it is important to have items that cover a set of sub-categories and/or have a proper balance across those sub-categories. In these latter cases, the value derives not from the items simply being different from another, but from the fact that there are different categories represented and the distribution of items over categories is appropriate to the domain. These three ideas are distinct, but obviously strongly interrelated. To emphasize these differences we discuss each of these in separate sections, recognizing that their relationship is a tight one.

**Item Diversity.** Diversity of recommended items is one feature of collections that has been addressed by a handful of researchers in recent years, although it deserves much more

attention. Although not discussed in the context of collections, researchers have recognized problems with many existing recommender systems which suggest the top-N items (e.g., Amazon’s list of the 5 most related books) [2, 4, 20, 12]. The problem they have identified is that the items often lack sufficient diversity, recommending items that the person already knows or recommending items that are too similar (e.g., songs all by the same artist). Even though the items each have a high probability of being liked, they fail to satisfy the user’s desire to be exposed to new material. Researchers have recognized that to overcome this problem they must consider the top-N recommended items as a “portfolio” rather than individual items [2].

Some authors have developed algorithms that recognize the need to balance and diversify recommendation lists in order to reflect a user’s complete array of interests. For example, Zeigler et al. have considered the entire top-N “portfolio” in the context of recommending books [20]. They develop a “topic diversification” algorithm that balances accuracy of suggestions with an individual’s full range of interests using existing hierarchical book classifications. They also develop a metric for measuring intra-list similarity that is generic enough to refer to different kinds of item features such as genre, author, timeframe. Their metric is designed for a case where order is not important since rearranging positions of recommendations in a top-N list does not affect the list’s intra-list similarity metric.

Their user study found that item-based algorithms benefited from a small boost in diversity, while user-based algorithms did not [20]. Zhang and Hurley develop a general approach to considering diverse subsets of items (e.g., top-N lists) by considering the problem as the optimization of an objective function under constraints of a certain type [19]. They also develop an objective measure of diversity which only requires that there is a measure of the dissimilarity between each pair of items. Finally, diversity of a set is addressed in [13] in the context of news aggregators where users vote on articles. The approaches that these papers have developed could be applied to collections in addition to top-N lists.

**Coverage.** Diversity deals with having items that are different from one another. These differences may be based on the attributes of the items themselves or on the categories or genres into which the items fall. Increased diversity will have more items that are in different categories or have different attributes, or the magnitude of the difference between items will increase. Coverage, on the other hand, is interested only in the categories in which recommended items are found. Furthermore, coverage measures which categories are covered by the recommendation.

A collection may have high item diversity but poor coverage. For example, a cookbook may include a wide range of recipes of several types, suggesting high item diversity. However, it may not include any desserts or side dishes, suggesting poor coverage, particularly if it were a general purpose cookbook. Conversely, a cookbook with good coverage of all of the types of dishes may lack enough diversity of recipes and ingredients to make it valuable



The needed coverage will depend on the domain and intended use. While diversity and coverage are closely related, they are certainly distinct concepts measure different interactions between items in a set.

**Balance.** Balance is closely related to coverage. While coverage addresses if there are or are not items recommended in a specific category (essentially a binary measure), balance describes the distribution of items in categories. Balance can be applied with or without any category coverage requirements. Simply having a “good” proportion of recommended items among categories may be sufficient to make a good collection.

In the cookbook example above, all categories may be covered, but the balance may be poor. If, for example, the book was marketed as a general cookbook but 90% of the recipes were for main dishes featuring chicken, it would not be well balanced for its type.

### 3. EXAMPLE COLLECTIONS

There are countless types of collections with different features, requirements, and domains. We present several example collections to illustrate some of the different possibilities, see how they relate to our framework, and review related research that has been done in the space of collection recommendation. Table 3 shows even more examples and how they fit into the framework.

#### 3.1 Family Dinner

As opposed to a multi-course meal, a family dinner is one where all dishes are served at once. The meal usually includes a main course and several side dishes, often including a vegetable and a starch. Depending on the number of people and the occasion, there may be a large number of options (e.g. American Thanksgiving dinner which often includes 6 or more side dishes) or only one choice for each category (one main course, one starch, and one vegetable dish).

This menu is not ordered. Thus, there are no ordering effects in the menu, but other interaction effects are present. Ideally, each menu item would be enjoyable by itself, and the combinations of items work well pairwise and overall. A menu with tacos as a main course would probably not serve cranberry sauce as a side dish. The size of the meal also matters. For two people, a dinner with 12 different dishes is likely to be considered to have too many items, whereas a meal with only two dishes may be completely appropriate.

The items in the meal are usually expected to provide some coverage of different categories (e.g. a main course and side dish). Among these dishes, there must be proper balance. Many people would consider a dinner for four with four loaves of garlic bread and one small piece of lasagna to share among all four people improperly balanced, even though it covers the main course and side dish categories. However, diversity of items beyond coverage and balances is sometimes but not always a requirement.

As just one example, a meal of fried chicken, french fries, and a biscuit has very little variety diversity relative to what is possible in a meal; two of the three items are fried, two are

starches, and everything is similarly flavored and textured. While not the healthiest option, many people would consider this a tasty dinner and a good combination of items. Thus, while variety has its place, it is not always an important component of a single meal. Generally, these meals are not constrained, but if the domain is shifted to one of dieting or where there are medical conditions to be considered, constraints on many aspects of the meal could arise.

#### 3.2 Collectible Card Games

Collectible Card Games, like Magic the Gathering, are games where players build decks of cards from their collections, and play a game with at least one other player using those cards. Thus, the overall collection of cards is a selection collection, since individual items are chosen from it to be used in a particular game. The quality of a collection is generally judged by its size, diversity, coverage, and balance.

With more cards, the player has more options in creating a deck. Thus, larger collections are almost always better. Games have different categories of cards, and having a proper balance among those categories, covering all the categories in some way, and having a wide range of cards from common to rare and across categories is important. Interestingly, though, the user’s preference for individual cards does not generally impact the quality of a collection.

For example, in Magic the Gathering a large proportion of the cards - roughly 1/3rd - are common cards called “lands”. These are necessary in this proportion for game play, but the value of an individual land card is extremely low. Common, low-valued cards of other types are also necessary to have well represented in the collection because they are needed in most decks for the player to be effective. Generally, individual cards that are rare and highly valued cannot be used extensively in a game deck because of the way the game is played, and this means they are also a small part of the overall collection. Thus, in this example, individual item values are not important to the value of the collection. Diversity, coverage, and balance, on the other hand, are critical.

#### 3.3 Music Libraries and Playlists

One space of collection recommendation that has received significant attention in the literature is playlist generation. These systems build lists of songs for users based on their known preferences. However, much of this research focuses on building a list of songs where each song is evaluated individually; little attention is paid to the quality of the collection as a whole with focus on interaction effects, co-occurrence relationships, order effects, etc.

Consider an individual who has an iTunes Music Library of a few hundred songs. The library itself can be considered a collection, one that is typically a finite, selection collection where order is not particularly important (except perhaps to help locate a song). Constraints on the collection may include hard drive space and cost. Note that we could use the music library as the seed for an infinite collection that continuously played music from the library (e.g., in random order).

Although talking about music libraries can be useful in some contexts, users typically consider individual playlists - collec-

**Table 1: A table of collection types with indications of the value measures that may apply to them. Note that these are intended as examples but there may be cases for a given type of collection where a different mix of measures would be used.**

Collection	Features			Value Measures							
	Type	Finite	Ordered	Constrained	Individual Items	Order Interaction	Co-Occurrence	Size	Diversity	Coverage	Balance
Stock Portfolio	Unit	X		X	X			X	X	X	X
Mix Tape	Unit	X	X		X	X	X	X	X		
Playlist	Unit		X		X	X	X		X	X	X
Family Dinner	Unit	X			X		X	X		X	X
Fashion Runway Collection	Unit	X	X	X	X	X	X	X	X	X	X
Collectible Card Games	Selection	X						X	X	X	X
Medical Meal Plan	Selection			X	X				X	X	X
Cookbook	Selection	X			X				X	X	X
Radio Station	Selection		X		X	X	X		X	X	X
Board of Directors	Unit	X	X	X	X		X	X	X		

tions of songs that are pulled from a personal music library (or larger music database) into some coherent collection. Playlists can be hand-crafted or automatically generated. Indeed, automatic playlist generation via systems such as iTunes Genius, The Filter, and MusicIP are already popular. In these tools, users typically provide a seed song and the system automatically creates a list of related songs from the user’s library, often using content-based approaches that measure the similarity of songs based on various dimensions (e.g., rhythm, artist, genre) (e.g., [15]).

These automatic playlist generators don’t typically pay attention to order, simply showing the top-N similar songs, perhaps with a few dissimilar songs thrown in at the end of the list to enhance diversity (e.g., [11]). A few novel systems such as PATS try to balance a desire for coherence (i.e., similarity of songs) and variation (diversity of songs) by assuring that the same song is not recommended multiple times [16]. Their approach was successful in that PATS-generated playlists outperformed randomly assembled playlists [16]. A user study of an automatic playlist generator running on a mobile device showed that there was significant interest in such tools and that there is a need to group or spread out songs that are overly similar (e.g., from the same artist) [11], suggesting that relative order effects are important.

As with the iTunes Music Library example, playlists can be used as seeds for infinite collections that cycle through the songs in the playlist, as for example occurs when songs from a playlist are selected and played as background music at a party. The way in which songs from the playlist are cycled through may take into consideration order effects, diversity, coverage, and balance, or it could be completely random.

Playlists also highlight how it is possible to conflate several

types of collections. Note that the quality of the music library and the quality of a playlist created from that library are related, but different. The music library should be evaluated as its own collection. Since the music collection serves at least in part as a selection collection from which playlists can be created, the music library should be fairly large, diverse, and have good balance and coverage. If the library is only used for a specific genre (e.g. classical music) it should still have all those attributes within the given genre. The playlists created from this library obviously depend on the collection of items available, but are judged on other criteria. This will include the diversity of songs selected, the order interaction as one song flows to the next, its coverage and balance, and the quality of the individual items.

### 3.4 Mixtapes

Unlike playlists, which often serve as seeds for infinite collections that can continue forever, mixtapes are always finite collections, usually with fewer than 20 songs. This difference allows for consideration of absolute placement in the ordering (e.g. which song goes first or last), and farther reaching interaction effects as we judge the flow of songs over the whole collection rather than within a sliding window.

In previous work [7], we ran experiments with users, asking them to create mix tapes of 10 songs from a set of 15 possible songs. Subjects were also asked what factors they thought were important in making a good mixtape. Our results showed that subjects included songs they liked more often (individual item values), that the first song on the mixes was rated significantly higher than songs in other positions (order interaction effects), and certain songs appeared together much more often than expected while others were never used together (co-occurrence effects). In the open responses, 70% of subjects said that there should be a theme

to a mixtape (co-occurrence effects on a larger scale than pairwise interactions) and 2/3rds of subjects said that the order of songs is important. These quantitative and qualitative results show that apart from the individual songs that make them up, mixtapes have value as collections and that certain features can make one mix better than another.

#### 4. RECOMMENDER SYSTEMS FOR COLLECTIONS

Once this background data is available and the type and attributes of the collection have been identified, there are many ways a collection recommender system can be used. While item recommender systems generally suggest one item or a set of items from which the user can choose, collection recommenders have more possibilities. They can suggest whole collections, assist users in their creation of collections, and help improve existing collections by offering additions, removals, and replacements according to constraints or the user's preferences.

Certainly, recommending entire collections from scratch is important and useful. There are many domains where fully automated collection generation is desired. For example, if a user is at the gym with her MP3 player, she may not have time to create a playlist from scratch. In this case, a system that automatically chooses and orders songs with little to no user intervention is desirable. Users with little to no knowledge of the stock market may have no preferences about individual stocks, and so after specifying constraints for the portfolio, a system that automatically selects investments would be useful. In fact, this latter example is similar to the way people invest when choosing a fund; they do not focus on the individual items but rather select an existing collection with attributes that best meet their desires for risk, return, etc.

On the other hand, there are also many cases where users do not want fully automated recommendations of collections. Rather, they would prefer a system that helps them in their own collection creation. One domain where this has been studied is in playlist generation. Users have complained that automatic playlist generators remove the fun of creating playlists and do not provide enough possibilities for customizing playlists [11]. One approach to overcome this problem is to create a semi-automatic playlist generator such as SatisFly that augments the creation of playlists by recommending songs that fit various specified constraints [17]. This general approach leads to questions not just in collection recommendation but also in designing appropriate user interfaces and social practices around the use of these system. These recommender systems that augment collection creation will need to walk a fine line between suggesting content while still facilitating exploration and autonomy.

With proper background knowledge, these recommenders can also be built into existing systems. For example, a person with a Hummel figurine collection may search eBay for new items. A collection recommender could work on top of eBay, searching available items and ranking those which would add the most value to the existing collection. Similarly, a recipe website that allowed users to input the dishes they planned to serve could suggest other recipes to fill out the meal with compatible items. Making changes to existing

collections could also be a useful application of these algorithms. Someone may have a recipe and want a substitution for an item.

For example, someone who does not like asparagus may ask the system to recommend a replacement for a stir fry, and the system could look at its underlying data and suggest snow peas as a substitute. More generally, systems could allow users to increase the level of diversity in a collection along a sliding scale or highlight items that may be problematic when placed together.

Indeed, optimizing any feature of collections - diversity, individual item preference, etc. - by adding, removing, or changing items are all valid and useful techniques for recommender systems in this area.

Although many collections are used by an individual, other collections are used by many people. These shared collections are a particularly interesting area of future research. Indeed, group recommender systems that balance the preferences of multiple individuals to recommend items are an active area of research [14, 9]. Issues such as diversity, item co-occurrence interaction effects, coverage, and balance within collections seem particularly important within a group context.

Finally, it is worth noting that it is possible that for some types of collections it will simply not be possible to produce a recommender algorithm that takes into account all the value measures that apply to the collection. The data space may simply be too sparse, even in the most well used systems. The interaction of items in a collection and the connection between those interaction effects and personal taste may also be too complex for a recommender system to address. As algorithms for these systems and data collection mechanisms are developed, the limitations will become clearer.

#### 5. CONCLUSIONS

Recommending collections of items rather than individual items is an important open space of research in the recommender systems community. In this paper, we presented a comprehensive framework for describing and evaluating collections independent of their domain. Collection types include unit or selection collections, ordered and unordered, finite and infinite, and constrained or unconstrained.

The quality of these collections is judged based on the value of the individual items, order interaction (on ordered collections), co-occurrence effects, size, diversity, coverage, and balance. We presented a number of examples that showed how these different attribute and evaluation techniques could be combined and applied in a given domain.

Work that looks at more diverse types of collections will provide many valuable insights into collection recommendation generally as well as to the specific domain. There is also independent research to be done in the data collection techniques. The games research described in [10, 6, 18] has been successful in gathering data for individual item collection, and projects that extend this research to collections would be interesting and relatively straightforward to conduct. Our framework helps in this area particularly because

once a data collection technique is developed for a particular problem, it should be immediately and directly applicable to problems with the same framework attributes and valuation methods.

In addition, collection recommender systems can support a variety of different applications: automatic collection creation, augmented collection development, and item selection. These techniques will all require usability research in addition to development of the algorithms themselves.

## 6. REFERENCES

- [1] W. Adams and J. Yellen. Commodity bundling and the burden of monopoly. *The Quarterly Journal of Economics*, pages 475–498, 1976.
- [2] K. Ali and W. van Stam. Tivo: making show recommendations using a distributed collaborative filtering architecture. In *Proceedings of the 2004 ACM Conference on Knowledge Discovery and Data Mining*, pages 394–401, New York, NY, USA, 2004. ACM.
- [3] Y. Bakos and E. Brynjolfsson. Bundling information goods: Pricing, profits, and efficiency. *Management Science*, pages 1613–1630, 1999.
- [4] K. Bradley and B. Smyth. Improving recommendation diversity. In *Proceedings of AAAI'01: The Sixteenth International Conference on Artificial Intelligence*, 2001.
- [5] M. Garey, D. Johnson, et al. *Computers and Intractability: A Guide to the Theory of NP-completeness*. wh freeman San Francisco, 1979.
- [6] S. Hacker and L. von Ahn. Matchin: eliciting user preferences with an online game. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 1207–1216, New York, NY, USA, 2009. ACM.
- [7] D. Hansen and J. Golbeck. Mixing it up: Recommending collections of items. In *CHI '09: Proceedings of the SIGCHI conference on Human factors in computing systems*, 2009.
- [8] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, 2004.
- [9] A. Jameson. More than the sum of its members: Challenges for group recommender systems. In *Proceedings of the working conference on Advanced visual interfaces*, pages 48–54. ACM New York, NY, USA, 2004.
- [10] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 1197–1206, New York, NY, USA, 2009. ACM.
- [11] A. Lehtiniemi and J. Seppänen. Evaluation of automatic mobile playlist generator. In *Mobility '07: Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology*, pages 452–459, New York, NY, USA, 2007. ACM.
- [12] S. M. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*, pages 1097–1101, New York, NY, USA, 2006. ACM.
- [13] S. Munson, D. X. Zhou, and P. Resnick. Sidelines: An algorithm for increasing diversity in news and opinion aggregators. In *Proceedings of the 3rd International AAAI Conference on Weblogs and Social Media*, 2009.
- [14] M. OConnor, D. Cosley, J. Konstan, and J. Riedl. PolyLens: A recommender system for groups of users. In *Proceedings of the European Conference on Computer-Supported Cooperative Work*, pages 199–218, 2001.
- [15] E. Pampalk, A. Flexer, and G. Widmer. Hierarchical organization and description of music collections at the artist level. In *Proceedings of the 2005 European Conference on Digital Libraries*, pages 37–48, 2005.
- [16] S. Pauws and B. Eggen. PATS: Realization and user evaluation of an automatic playlist generator. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 222–230, 2002.
- [17] S. Pauws and S. van de Wijdeven. Evaluation of a new interactive playlist generation concept. In *ISMIR International Conference on Music Information Retrieval*, pages 638–643, 2005.
- [18] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326, New York, NY, USA, 2004. ACM.
- [19] M. Zhang and N. Hurley. Avoiding monotony: improving the diversity of recommendation lists. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 123–130, New York, NY, USA, 2008. ACM.
- [20] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the Fifteenth World Wide Web Conference*, pages 22–32, New York, NY, USA, 2005. ACM.

# Helping Users Perceive Recommendation Diversity

Rong Hu

Human Computer Interaction Group  
Swiss Federal Institute of Technology (EPFL)  
CH-1015, Lausanne, Switzerland

rong.hu@epfl.ch

Pearl Pu

Human Computer Interaction Group  
Swiss Federal Institute of Technology (EPFL)  
CH-1015, Lausanne, Switzerland

pearl.pu@epfl.ch

## ABSTRACT

The recommendation diversity is increasingly being recognized as an important issue in satisfying users' needs for recommender systems. Various diversity-enhancing methods have been developed to increase diversity while making personalized recommendations to users. However, one crucial issue remains. Could the diversity, as system designers have carefully incorporated, be perceived by users and influence their interaction behaviors? In this paper, we try to investigate whether this issue can be addressed at the interface level. Our goal is to understand design issues that enhance users' perception of recommendation diversity and more importantly their satisfaction. A within-subject user study was conducted to compare an organization interface, which groups recommendations into categories, with a standard list interface. Our user study results show that the organization interface indeed effectively increased users' perceived diversity of recommendations, especially perceived categorical diversity. Correlation results reveal that the perceived categorical diversity in recommendation lists has a significant correlation with users' perceived ease of use of a system, perceived usefulness of the system and attitudes towards the system, thereby resulting in a positive effect on their intention to use the system. We conclude by proposing design guidelines based on our study observations.

## Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – Evaluation/methodology.

## General Terms

Design.

## Keywords

Diversity, Recommender System, User Study, Interface Design, User Satisfaction.

## 1. INTRODUCTION

During a long period, prediction accuracy was considered as the sole criterion when evaluating recommender systems' quality. However, recent studies have increasingly indicated that accuracy is not enough for a satisfying recommender system, in particular from a user's point of view [8, 13]. Other criteria, such as diversity and serendipity, are emerging as important characteristics for consideration to generate *useful* recommendations [5, 8]. In this paper, we focus on recommendation diversity issues.

Permission Copyright is held by the author/owner(s). Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011), held in conjunction with ACM RecSys 2011. October 23, 2011, Chicago, Illinois, USA.

Diversity is an intrinsically desirable property for a recommender system. Firstly, users' needs are commonly uncertain beforehand [15, 16]. Varied options could broaden users' domain knowledge about the recommended items and help them clarify their requirements. Secondly, recommender systems are expected to help users explore and discover new items of interest [8]. For users, it is more valuable to obtain the recommendations that they would love, but are different from those which they have already purchased or used [9]. For e-commerce websites, recommending varied items has the potential to make more profits by increasing the sales diversity [7]. Thirdly, it is important for recommender systems to convince users that the recommended item is the best one for them. The existence of diversity in recommendations has the capability of decreasing the difficulty of making a choice and enhancing users' confidence in their choices by providing comparison among recommendations [2, 10].

Even though many diversity-enhancing algorithms have been proposed in the literature [1, 12, 14, 21, 22, 23], few studies have investigated users' perception of recommendation diversity and how such a perception could influence their satisfaction and acceptance of a system. In [23], Ziegler et al. did a large scale online study, and their online experimental results show that users' overall satisfaction with recommendation lists not only depends on accuracy, but also on the range of reading interests covered. They also found that human perception can only capture a certain level of diversification inherent to a list. Beyond that point, it is difficult for users to notice the increasing diversity degree. Therefore, it is worth investigating how to help users overcome the cognitive limitation and be aware of the existence of diversity in recommendation lists, aimed at achieving a high level of satisfaction to a system.

Currently, the conventional ranked list interface is still a popular way of displaying search/recommendation results. However, this method is highly inefficient in some cases [3]. For example, the number of retrieved search results can be easily beyond the extent of human cognitive capability. Users tend to focus on the top of a list and items that are located farther down in the list would attract little attention. By nature, the ranked list interface is likely to impede users' perception of the diversification of recommendations. Therefore, we are considering whether alternative approaches, such as a proper interface layout design, could augment users' diversity perception.

In this paper, we conducted a within-subject user study, comparing an organization-based interface, which groups recommendations and displays them in a category style [3, 16], with a conventional list interface, while keeping the recommendations in the two systems identical. We utilized Amazon.com as our experimental platform due to its well-known reputation in the field of recommender systems. Its standard list interface for recommendations was replaced by an organization-

based interface with the help of a proxy program. In this study, we attempt to answer the following two research questions:

- 1) How can interface designs influence users' perceived diversity?
- 2) How does diversity perception affect users' satisfaction of a system?

The contributions of this paper include three aspects. Our results suggest that the organization-based interface indeed effectively increased users' perceived diversity of recommendations, especially perceived categorical diversity (i.e., users perceive that various kinds of items were recommended to them). In addition, we empirically explored the influence of perceived diversity on users' acceptance of a recommender system. Correlation results show that categorical diversity more significantly influences users' perceived usefulness of the recommender, their attitudes toward the system and their intentions to use the system. Finally, based on the findings in this study, we proposed specific design guidelines.

The remainder of the paper is organized as follows. We first provide an overview of related research work on diversity enhancing technologies and diversity-related user studies in recommender systems. In Section 3, we describe the organization-based interface design methods. In Section 4, we present a detailed description of our experiment, including experiment design, evaluation metrics, and dataset, followed by the experimental results, discussion and the derived design guidelines. Finally, we present the conclusions and future work.

## 2. RELATED WORK

Traditional diversity-enhancing methods are operated as a heuristic search. The bounded greedy algorithm proposed in [1, 21] is the first attempt to explicitly enhance the diversity of a recommendation list without significantly compromising their query similarity characteristics in case-based recommender systems. It first ranks all recommendable items according to their similarity to the current query. Then, it sequentially transfers items from this ranked list to a final recommendation list such that each selected item maximizes the product of its similarity to the target query and its diversity relative to the cases that have already been selected. Most diversity-enhancing methods follow this fundamental re-ordering strategy [14, 19].

The concept of diversity was also considered in the design of critiquing-based recommender systems. Pu and Chen [4, 16] proposed a dynamic compound critiques generation method, which takes diversity among critiques into account. McCarthy et al. [11] also proposed an idea of generating diverse compound critiques in the context of conversational recommender systems.

Zhang and Hurley [22] suggested presenting the competing concerns of similarity and diversity as constrained binary optimization problems. They applied their optimization strategy to the top- $N$  prediction problem and achieved improvements on both diversity and accuracy compared to a standard item-based collaborative filtering algorithm.

McGinty and Smyth [12] highlighted the pitfalls of naively incorporating diversity-enhancing techniques into existing recommender systems and proposed an adaptive diversity-enhancing algorithm. They pointed out that diversity should be provided adaptively. When a recommender system appears to be close to the target case, diversity should be limited to avoid

missing it. But when the recommender system is not correctly focused, diversity can be used to help refocus more effectively.

In [23], the authors proposed a topic diversification approach based on taxonomy-based similarity. They compared not only the accuracy measures in different levels of diversification for both user-based and item-based CF, but also subjective satisfaction results from a large scale user survey. Their results show that users' overall satisfaction of recommendation lists goes beyond accuracy and involves other factors, e.g., the users' perceived list diversity. Their work first shed light on the critical value of diversity from the perspective of users.

Castagnos et al. [2] investigated the impact of recommenders on users' product search patterns by observing their interaction behaviors with an online product retail website with an eye tracking system. They demonstrated that users' need for diversity led them to use the recommender systems, compared to the traditional information filtering tools. Furthermore, they found that the diversified recommendations could enhance users' confidence by providing the capability of comparison. To conclude their findings, they proposed a time-dependent satisfaction model which demonstrates the dynamic compromise between accuracy and diversity in recommender system. Our work is similar to theirs. Differently, we investigate the relations between perceived diversity and users' acceptance of the system in a within-subject user study by comparing the influence of two interface designs.

## 3. ORGANIZATION-BASED INTERFACE

The idea of organization-based interfaces was first proposed as an explanation interface, with the aim of inspiring users' trust in recommender systems [16]. Pu and Chen implemented more than 13 paper prototypes of organization-based interfaces to explore the design dimensions. Based on the results of testing these prototypes with real users in the form of pilot studies and interviews, they derived five design principles: 1) categorize remaining recommendations according to their similar tradeoff properties relative to the top candidate; 2) propose improvements and compromises in the category title using conversational language; keep the number of tradeoff attributes under five to avoid information overload, e.g., "these products are cheaper and lighter, but have slower processor speed"; 3) eliminate dominated categories, and diversify the categories in terms of their titles and contained recommendations; 4) include actual products in a recommended category; 5) rank recommendations within each category by exchange rate (i.e., the preference-based utility value relative to the top candidate) rather than similarity measure. Consequently, the organization-based interface design essentially considers the diversity issue both among categories and within each category.

Previous studies have indicated that organization-based interface designs are highly effective in building users' trust of a recommender system, with the benefit of increasing users' intention to return to the agent and saving users' cognitive effort [16]. More recently, Chen and Pu [3] performed a user study with an eye-tracker to compare the efficacy of two recommender interface designs, list-based and organization-based interfaces, in affecting users' decision making strategies through the observation of users' eye movements and product selection behavior. Their results showed that organization-based interfaces can significantly attract users' attentions to more items with the resulting benefit of enhancing their objective decision quality. Based on their findings, we assume that the organization interface

designs have the capability of assisting users in perceiving the diversity of recommendation lists. In our experiment, we utilized a variation of the conventional organization-based interface approach, Editorial Picked Critiques (EPC) technique, to generate categories for our organization-based interface. We will introduce EPC technique in detail in the following section.

### 3.1 Editorial Picked Critiques (EPC)

EPC was originally developed in the context of applying critiquing-based recommendation technology to public taste products such as music, films, perfumes, fashion goods and wine [18]. In contrast to high-involvement products such as PCs, digital cameras, users tend to spend less time choosing public taste goods and are more likely to rely on public opinions or experts’ advice to make decisions [20]. EPC was designed to take into account the public opinions, popularity information and editorial suggestions, as well as the needs for personalization and diversity.

EPC first identifies five important unit critique categories that match users’ attention and needs for public taste goods: price-driven critiques, popularity-based critiques, diversity-driven critiques, similarity-driven critiques, and special recommendation (similar to editorials special picks). Items in the similarity-driven critiques are those which are similar to the selected product and could be generated by recent similarity-based recommendation approaches, such as content-based or collaborative filtering methods. This category is titled as “people who like this may also like”.

Compound critiquing categories are generated on the basis of these unit critiques. In [18], a set of five compound categories were proposed for perfume products: “more popular and cheaper”, or “more popular but more expensive” in the case that the former category does not contain any products, “same brand and cheaper” or “same brand but more expensive”, “just as popular and cheaper”, “same price range and just as popular”, and finally “people who like this also like”. When generating recommendations for each category, users’ preferences are taken into account.

In our experiment, we adopted these compound categories proposed in [18] as our classification categories for the organization interface. We remapped the recommendations from Amazon into these five categories, and we used Amazon’s bestselling order and customers’ ratings as a popularity measure. The items which cannot be categorized into any of the first four categories are put into the category “people who like this also like”.

## 4. EXPERIMENT

### 4.1 Materials

A well-known commercial website, Amazon.com, was used as our experimental platform due to its high reputation in the field of recommender systems. Its standard list interface was used as the baseline. The organization version was achieved with the help of an open-source filtering HTTP proxy program, PAW<sup>1</sup>. The recommendation list we used was “Customer Who Viewed This Item Also Viewed” in the detailed information page for each product (perfume in our experiment). Unlike the organization-based interface designs in [2, 3, 16], the categories in this study were organized in a tab-based structure to better conform to the horizontal list style in that website. By clicking on each tab, users could see the recommendations in the corresponding category.

<sup>1</sup> <http://paw-project.sourceforge.net>

**Table 1. Demographic characteristics of participants.**

Gender	Male	Female
	10	10
Nationality	Chinese (10), Swiss(2), Indian(3), Romanian(1), Croatian(1), Portuguese(1), Iranian(1), Georgian(1)	
Education	Bachelor, Master, Doctor	
Profession	student, research assistant, engineer, interface designer	
Age	21-30	31-40
	19	1

The categories which had no products were not presented. A screenshot of the organization (ORG) interface is shown in Figure 1. The original list-view (LIST) interface used in the website was adapted to only show five products each time to remain consistent with the organization-based interface. A screenshot of the list interface is shown in Figure 2. In either interface, the number of displayed recommendations was restricted to be the same (five in the current study) and the “next” and “previous” buttons were used to explore more items in a list. In order to avoid confusion, we removed the recommendation list of “Customers Who Bought This Also Bought” from the page. In addition, we placed the section “Customers Who Viewed This Item Also Viewed” just beneath the selected product so that users could easily notice it.


### 4.2 Dataset and Participants

The dataset of perfumes used in this experiment was crawled from Amazon and updated just before launching the study to ensure that we had a dataset containing the most recent and popular fragrance products available on the market. In our experiment, 21,071 items were accessible, covering 13,246 items for women (6,281 Eau de Toilette, 689 Cologne and 6,276 Eau de Parfum) and 7,825 items for men (6,066 Eau de Toilette, 1,474 Cologne and 285 Eau de Parfum).

A total of 20 participants (10 females) were recruited in our user study. The incentive for the participants was a lottery: one out of the 20 users could win a 100 CHF gift voucher to purchase one of the perfumes the winner put in the basket during the study. These participants were from 8 different countries with various professions (student, research assistant, engineer, interface designer); their age ranged from 20 to 40, and they represented various educational backgrounds (from bachelor, master or Ph.D.). The details of their demographic characteristics are shown in Table 1. In addition, four background questions were asked in terms of users’ previous computer knowledge, internet usage, perfume knowledge and experience with Amazon. All participants said that they were regular computer users and used the Internet frequently. 11 participants indicated “agree” to the statement “I have knowledge about perfume”, 8 participants marked “neutral” and just one said “disagree”. 17 of the participants had used Amazon before.

### 4.3 Evaluation Criteria

In order to evaluate users’ perceived qualities of a recommender, we used a simplified version of a user-centric recommender evaluation model (*ResQue*) [17]. More specifically, two questions were designed to measure users’ diversity perception of the recommendation lists. One referred to the difference among categories, querying whether “the items recommended to me are of various kinds” (called *categorical diversity*). The other



**Nautica White Sail by Nautica for Men 3.4 oz Eau De Toilette Spray**  
by [Nautica](#)  
No customer reviews yet. [Be the first.](#)

List Price: ~~\$68.00~~  
Price: **\$20.64** & eligible for **FREE Super Saver Shipping** on orders over \$25. [Details](#)  
You Save: **\$37.36 (64%)**

Size: 3.4 oz

**In stock but may require an extra 1-2 days to process.**  
Ships from and sold by [Amazon.com](#).

**14 new** from \$14.99

**Special Shipping Information:** This item **cannot be returned** and has additional shipping restrictions. [See details.](#)


[See larger image and other views](#)  
[Share your own customer images](#)

**Customers Who Viewed This Item Also Viewed**

More popular and cheaper	Same brand but more expensive	Just as popular but cheaper	Same price range and just as popular	People who like this also like
 <a href="#">Nautica Blue By Nautica For Men Edt Spray 3.4 Oz</a> ★★★★★ (9) \$15.28	 <a href="#">Diesel Plus Plus By Diesel For Men, Eau De Toilette Spray 2.5 Oz.</a> ★★★★★ (12) \$10.45	 <a href="#">Nautica Classic 3.3 oz. Eau De Toilette Spray Men</a> ★★★★★ (2) \$14.99	 <a href="#">Lomani By Lomani For Men, Eau De Toilette Spray 3.4-Ounce Bottle</a> ★★★★★ (6) \$8.19	 <a href="#">Diesel Zero Plus By Diesel For Men, Eau De Toilette Spray 2.5 Oun...</a> ★★★★★ (10) \$13.20

page 1 of 2

**Figure 1. The simulated organization interface (content is identical to the recommendation results below).**



**Nautica White Sail by Nautica for Men 3.4 oz Eau De Toilette Spray**  
by [Nautica](#)  
No customer reviews yet. [Be the first.](#)

List Price: ~~\$68.00~~  
Price: **\$20.64** & eligible for **FREE Super Saver Shipping** on orders over \$25. [Details](#)  
You Save: **\$37.36 (64%)**

Size: 3.4 oz






**In stock but may require an extra 1-2 days to process.**  
Ships from and sold by [Amazon.com](#).

**14 new** from \$14.99

**Special Shipping Information:** This item **cannot be returned** and has additional shipping restrictions. [See details.](#)

[See larger image and other views](#)  
[Share your own customer images](#)

**Customers Who Viewed This Item Also Viewed**

 <a href="#">B United By Benetton For Men, Eau De Toilette Spray 3.3 Ounces</a> \$14.96	 <a href="#">Adidas Game Spirit By Adidas For Men, Eau De Toilette Spray, 3.4...</a> \$7.50	 <a href="#">Diamonds &amp; Emeralds Perfume by Elizabeth Taylor Eau De Parfums</a> ★★★★★ (1) \$8.06 - \$42.99	 <a href="#">Nautica Blue By Nautica For Men Edt Spray 3.4 Oz</a> ★★★★★ (9) \$15.28	 <a href="#">Perry Ellis 360 Red By Perry Ellis For Men, Eau De Toilette Spr...</a> ★★★★★ (15) \$21.34
--	--	--	--	--

page 1 of 6

**Figure 2. The standard list interface.**

considers the difference among each item, asking whether “the items recommended to me are similar to each other” (also called *item-to-item diversity*). We also tried to investigate the influence of perceived diversity on users’ acceptance of a recommender system. In our evaluation, we took into account perceived ease of use and usefulness of a system (facilitation, effectiveness, and supportiveness), users’ attitudes towards the system (satisfaction, conviction, and confidence), and behavioral intentions to use it (intention to reuse, intention to tell friends, and intention to purchase). Besides, we measured users’ perception on recommendation quality. Table 2 lists all of the questions as measures of these subjective variables. Each question was

required to respond on a 5-point Likert scale from “strongly disagree” (1) to “strongly agree” (5).

#### 4.4 Experiment Design and Procedure

Our user study was conducted in a within-subjects design. All participants used both interfaces, and then filled in a post-stage assessment questionnaire for the respective interface (see Table 2). In the end, they were asked to answer about their preferences on these two interfaces. All participants were randomly assigned to two experimental conditions, with a differing order in using the two interfaces. That is, 10 users in one condition evaluated the list view interface first and then the organization view interface; the



**Table 2. Post-stage assessment questionnaire.**

ID	Questions
Q1	I am interested in the items recommended to me.
Q2	The items recommended to me are of various kinds.
Q3	The items recommended to me are similar to each other. (reversal question)
Q4	Finding an item to buy with the help of the recommender is easy.
Q5	The recommended items effectively helped me find the ideal product.
Q6	I feel supported in selecting the items to buy with the help of the recommender.
Q7	Overall, I am satisfied with the recommender provided by this system.
Q8	I am convinced of the products recommended to me.
Q9	I am confident I will like the items recommended to me.
Q10	I will use this recommender again.
Q11	I will tell my friends about this recommender.
Q12	I would buy the items recommended, given the opportunity.

other condition had a reverse order. Counterbalance measures were taken to eliminate fatigue and learning effects as much as possible.

The user study was run at the office of an administrator who supervised the experiment and assisted participants to successfully complete all tasks, with the help of a desktop computer. Users' click behaviors were automatically recorded into log files. At the beginning, participants were asked to read a printed introduction and debriefed on the upcoming tasks. They then answered a series of background and demographic questions. In order to clarify the evaluated interfaces to the participants, two printed screenshots were shown and a brief description was given by the administrator. Then, they started using these two interfaces.

Participants were given specific tasks when using each interface. In the first interface, we asked a user to find up to three perfumes that he/she has never heard of or used before and would be willing to purchase for himself/herself given the opportunity and put them into the shopping cart. When using the second interface, the user was asked to search for three perfumes which he/she would be willing to purchase for someone of the opposite gender as a gift, in order to reduce the potential influence of users' familiarity with the product domain after using the first interface. After using each interface, the user was asked to fill in a post-stage assessment questionnaire to evaluate the interface he/she just tested. The questions are listed in Table 2.

Finally, all participants were asked to answer a questionnaire about their preferences on these two interfaces in terms of five aspects: general preference, informative, useful, good at recommending, and good at helping perceived diversity. These questions are listed in Table 3.

## 5. RESULTS ANALYSIS

### 5.1 Users' Subjective Evaluation

All responses for the post-stage questions were analyzed using paired sample t-tests. The results are shown in Figure 3. The questions marked with (\*\*) denote that a significant difference

**Table 3. Preference questionnaire.**

ID	Questions
P1	Which recommendation interface did you prefer?
P2	Which recommendation interface did you find more informative?
P3	Which recommendation interface did you find more useful?
P4	Which recommendation interface was better at recommending perfumes you like?
P5	Which recommendation interface was better at helping perceive the diversity of recommendations?

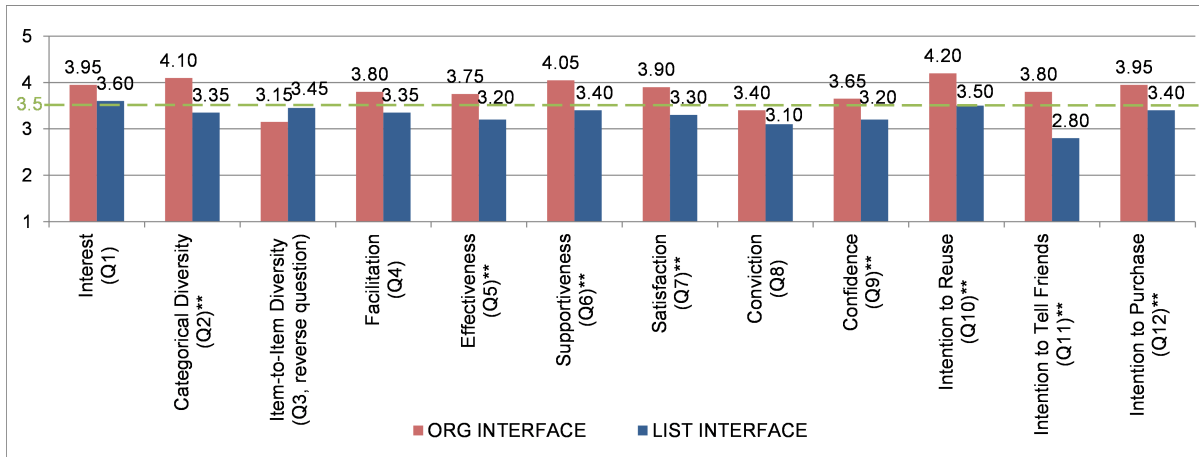
were observed among users' responses. The detailed analysis is as follows. Users found the recommended items from both interfaces to be interesting (Q1) with a slight advantage for ORG ( $p = 0.07$ ). It means that the subjective accuracy of the two interfaces is not significantly different.

With respect to users' perceived diversity in both interfaces, we asked two questions. One emphasizes the categorical difference (Q2). The other simply considers the general differences between each item (Q3). Interestingly, we could see from the results that the difference between the two interfaces was only significant with respect to the question Q2. That is, the level of perceived categorical diversity in the organization interface was significantly higher than that of the list interface (mean = 4.1, SD = 0.788 for ORG, vs. mean = 3.35, SD = 0.988 for LIST,  $p < 0.05$ ,  $t = 3.68$ ). However, no significant difference was measured on item-to-item diversity ( $p = 0.186$ ). Users seemed to disagree that items were similar to each other in both interfaces (reverse scale of item-to-item diversity). Therefore, we conclude that the organization-based interface helped users' awareness of the diversity present by variety differences.

Perceived ease of use and usefulness of the system were evaluated in terms of three aspects: facilitation (Q4), effectiveness (Q5), and supportiveness (Q6). While users found ORG is more easy to use (Q4), the difference between ORG and LIST was slightly significant ( $p = 0.09$ ). On the other hand, users thought that the recommended items were significantly more effective in helping them find the ideal product (Q5) in ORG (mean = 3.75, SD = 0.851, vs. mean = 3.2, SD = 1.005 for LIST,  $p < 0.05$ ,  $t = 2.773$ ). They also felt more supported in selecting the items to buy with the help of ORG (Q6, mean = 4.05, SD = 0.686, vs. mean = 3.4, SD = 1.095 for LIST,  $p < 0.05$ ,  $t = 2.371$ ).

In order to evaluate users' attitude towards the tested interfaces, three evaluation measures were considered: satisfaction (Q7), conviction (Q8), and confidence (Q9). Users expressed significantly higher satisfaction for ORG (Q7, mean = 3.9, SD = 0.912, vs. mean = 3.3, SD = 0.923 for LIST,  $p < 0.05$ ,  $p < 0.05$ ,  $t = 3.559$ ). In addition, they seemed to be more confident that they would like the recommended items in ORG (Q9, mean = 3.65, SD = 0.875, vs. mean = 3.2, SD = 0.894 for LIST,  $p < 0.05$ ,  $t = 2.269$ ). Therefore, users had more positive attitudes towards the ORG interface.

Significant differences were also revealed on the measures of users' behavioral intentions to use a system. More specifically, users scored significantly higher for ORG on reusing the system (Q10, mean = 4.2, SD = 0.834, vs. mean = 3.5, SD = 0.827 for LIST,  $p < 0.001$ ,  $t = 4.273$ ), telling friends about it (Q11, mean =

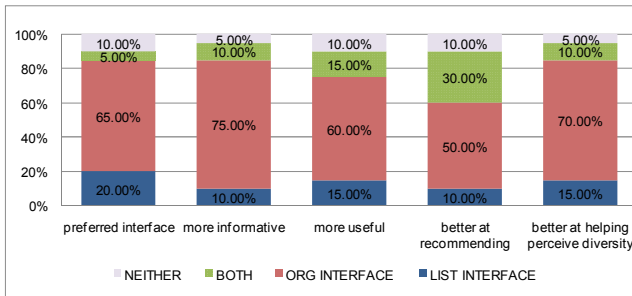


**Figure 3. Usability and user satisfaction assessment results. A cut off value at 3.5 represents agreement on the 5-point Likert scale. \*\* is marked for significant differences at the 5% level (p-value < 0.05).**

3.8, SD = 0.894, vs. mean = 2.8, SD = 0.894 for LIST,  $p < 0.001$ ,  $t = 4.359$ ) and purchasing the recommended items given the opportunity (Q12, mean = 3.95, SD = 0.686, vs. mean = 3.4, SD = 0.940 for LIST,  $p < 0.05$ ,  $t = 2.463$ ).

## 5.2 Final Preference

After evaluating two interfaces, users were asked to answer five questions regarding their preferences for these two interfaces. The results are shown in Figure 4. ORG got dominant preferences with more than 50% votes on all of the five questions. Particularly, 65% of users preferred the organization interface versus only 20% for the list interface, while 5% of them prefer both interfaces. More users thought that the organization-based interface was more informative (70% vs. only 10%), more useful (60% vs. 15%) and better at recommending items (50% vs. 10%). More importantly, 70% (vs. 15%) of users thought that the organization-based interface is better at helping them perceive the diversity of recommendations in contrast to the list interface.



**Figure 4. Preference Results.**

## 5.3 Correlation Analysis

We did a correlation analysis between the perceived diversity (both categorical diversity and item-to-item diversity) and other subjective measures, aimed at understanding how perceived diversity influences users' acceptance of a recommender system. The results are shown in Table 4. All correlations presented in boldface and with the symbol (\*\*) are statistical significant at the 0.05 level with two-tailed Pearson correlation coefficients.

More specifically, Table 4 shows the correlations between perceived categorical diversity and the other subjective measurements (perceived ease of use, perceived usefulness, attitudes and behaviors intentions to use). Perceived categorical diversity is highly positively related to the perceived ease of use (facilitation:  $r = 0.405$ ,  $p < 0.05$ ), and the perceived usefulness of the system (effectiveness:  $r = 0.451$ ,  $p < 0.01$ , supportiveness:  $r = 0.500$ ,  $p < 0.01$ ). In addition, perceived categorical diversity is significantly positively correlated with satisfaction ( $r = 0.576$ ,  $p < 0.001$ ), conviction ( $r = 0.456$ ,  $p < 0.01$ ) and confidence ( $r = 0.493$ ,  $p < 0.01$ ). The same correlation is found with respect to behavioral intentions to use (intention to reuse:  $r = 0.519$ ,  $p < 0.01$ , intention to tell friends:  $r = 0.428$ ,  $p < 0.006$ , intention to purchase:  $r = 0.386$ ,  $p < 0.05$ ).

On the contrary, the item-to-item diversity has a weaker correlation to the three subjective measure factors. It only has a significant correlation with facilitation ( $r = -0.322$ ,  $p < 0.05$ ) in the aspect of perceived usefulness. In the aspect of attitudes to the system, it is strongly related to conviction ( $r = -0.390$ ,  $p < 0.05$ ) and confidence ( $r = -0.426$ ,  $p < 0.01$ ). Furthermore, the item-to-item diversity is significantly correlated to intention to reuse ( $r = -0.434$ ,  $p < 0.01$ ).

## 5.4 Discussion and Design Guidelines

According to users' responses to the subjective questionnaires, we saw that users perceived more categorical diversity of recommendations in the organization interface compared to in the list interface. This suggests that the organization interface could indeed help users become aware of the diversity in recommendation lists, particularly the difference among categories which is difficult to perceive in the list view interface; there is a 22.4% increase. However, there is no significant statistical difference between the organization-based interface and the list-based interface with respect to the item-to-item diversity. The organization-based interface does not appear to be particularly advantageous in this case. After using two interfaces, users were asked to answer five questions regarding their preferences for these two interfaces. 70% (vs. 15%) of users thought that the organization-based interface is better at helping them perceive the diversity of recommendations in contrast to the list interface.

**Table 4. Correlation results on categorical and item-to-item diversity (\*\* denotes statistical significance at the 0.05 level, i.e., p-value<0.05).**

Factors		Correlation (Sig.)	
		Categorical Diversity (Q2)	Item-to-item Diversity (Q3)
Ease of Use	Facilitation (Q4)	<b>0.405(0.01**)</b>	<b>-0.322(0.043**)</b>
Perceived Usefulness	Effectiveness (Q5)	<b>0.451(0.003**)</b>	-0.247(0.124)
	Supportiveness (Q6)	<b>0.500(0.001**)</b>	-0.247(0.124)
Attitudes	Satisfaction (Q7)	<b>0.576(0.000**)</b>	-0.263(0.101)
	Conviction (Q8)	<b>0.456(0.003**)</b>	<b>-0.390(0.013**)</b>
	Confidence (Q9)	<b>0.493(0.001**)</b>	<b>-0.426(0.006**)</b>
Behavioral Intentions	Intention to reuse (Q10)	<b>0.519(0.001**)</b>	<b>-0.434(0.005**)</b>
	Intention to tell friends (Q11)	<b>0.428(0.006**)</b>	-0.097(0.553)
	Intention to purchase (Q12)	<b>0.386(0.014**)</b>	-0.226(0.161)

Previous studies have shown that the diversity of recommendation lists influences users satisfaction [23]. However, it is still not well understood why and how such an impact occurs. Our correlation results reveal that categorical diversity in recommendation lists influences users' perceived ease of use of a system, perceived usefulness of the system and attitudes towards the system, thereby resulting in a positive effect on their intention to use the system. While the item-to-item diversity has an impact on users' acceptance to the system as well, the effect is not as strong as with categorical diversity. On the other hand, our results empirically demonstrate that perceived diversity is indeed one critical factor influencing users' adoption of a recommender system due to its strong correlation with the factors (perceived ease of use, perceived usefulness, attitudes, and behavioral intentions) which are considered in users' acceptance models, like TAM [6].

Furthermore, the correlation results show that perceived diversity plays a role in providing supporting information, which leads to increased user confidence in a system. In previous research about diversity-enhancing techniques, diversity has only been demonstrated to help users reduce interaction cycles and more efficiently find the target item [12, 22]. Our empirical results indicate that users obtained more supportive and convincing information when they perceive diversity, and thereafter they felt more confident about their decisions. In other words, diversity can not only make recommendations covering a wide range of users' interests, but can also provide supportive information to aid users make decisions.

The current study confirmed the critical role of diversity in a recommender's success. It further shows promising results that contribute to the field:

1) Even though a number of diversity-enhancing techniques have been proposed in the literature, interface design issues relative to diversity have been overlooked. Our study demonstrates that a simple reorganization of the results into a category layout could have a strong positive effect on users' perceived qualities of the system, especially their satisfaction and intention to use and purchase. This suggests a novel research direction on the issue of diversity-enhancing technology.

2) Our results show that perceived *categorical* diversity has an even stronger influence on users' positive perception and

acceptance of a recommender system than item-to-item diversity. This highlights the critical role of categorical diversity on user experience of a recommender system. However, it doesn't mean the item-to-item diversity is trivial. According to users' responses, it is difficult for them to be aware of the item-to-item diversity in recommendations.

To conclude the findings of our study, we propose the following design guidelines.

**Guideline 1:** Take recommendation diversity into account when designing recommender systems.

**Guideline 2:** Make users aware of the diversity (both categorical diversity and item-to-item diversity) existed in recommendation lists by explaining the similarities and differences among the displayed items.

**Guideline 3:** Display recommendations in a category layout by adopting organization interface designs to enhance users' perception of the categorical diversity of the recommendations.

## 6. CONCLUSION AND FUTURE WORK

We conducted an in-depth user study to compare an organization-based interface with the standard list-based interface. Experimental results reveal that the ORG interface indeed influence the users' perception of the recommendation diversity. Users in the ORG interface had more strong perception of categorical diversity. Even though users found the recommended items to be interesting in both interfaces, ORG users were more satisfied with the recommender. While both interfaces were easy to use, ORG users indicated that the interface was more helpful for them in terms of locating the items they wanted to buy (decision support). Most importantly, ORG users are more likely to use the system again, tell their friends about it and buy the recommended items. Strong correlation has been found between perceived diversity and users' satisfaction.

Our future work includes validating our findings in other product domains, comprehensively investigating the influence of diversity on the success of a recommender system, exploring other formats of interface designs which can more effectively enhance users' experience with a recommender system.

## 7. ACKNOWLEDGMENTS

We thank EPFL, the Swiss National Science foundation, and the ministry of education of the People's Republic of China for supporting the reported research work. We are grateful to the participants of our user studies for their patience and time.

## 8. REFERENCES

- [1] Bradley, K. and Barry, S. 2001. Improving Recommendation Diversity. In *Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science* (Maynooth, Ireland, 2001), 85-94.
- [2] Castagnos, S., Jones, N. and Pu, P. 2010. Eye-tracking product recommenders' usage. In *Proceedings of the fourth ACM conference on Recommender systems* (Barcelona, Spain, 2010). ACM, 1864717, 29-36.
- [3] Chen, L. and Pu, P. 2010. Eye-Tracking Study of User Behavior in Recommender Interfaces. *User Modeling, Adaptation, and Personalization*, De Bra, P., Kobsa, A. and Chin, D., eds. LNCS 6075, Springer Berlin / Heidelberg, 375-380.
- [4] Chen, L. and Pu, P. 2007. Preference-Based Organization Interfaces: Aiding User Critiques in Recommender Systems. *User Modeling 2007*, Conati, C., McCoy, K. and Paliouras, G., eds. Lecture Notes in Computer Science 4511, Springer Berlin / Heidelberg, 77-86.
- [5] Cosley, D., Lawrence, S. and Pennock, D.M. 2002. REFEREE: an open framework for practical testing of recommender systems using ResearchIndex. In *Proceedings of the 28th international conference on Very Large Data Bases* (Hong Kong, China, 2002). VLDB Endowment, 1287374, 35-46.
- [6] Davis, F.D., Bagozzi, R.P. and Warshaw, P.R. 1989. User acceptance of computer technology: a comparison of two theoretical models. *Manage. Sci.* 35, 8, 982-1003.
- [7] Fleder, D.M. and Hosanagar, K. 2007. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM conference on Electronic commerce* (San Diego, California, USA, 2007). ACM, 1250939, 192-199.
- [8] Herlocker, J.L., Konstan, J.A., Terveen, L.G. and Riedl, J.T. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* 22, 1, 5-53.
- [9] Hijikata, Y., Shimizu, T. and Nishida, S. 2009. Discovery-oriented collaborative filtering for improving user satisfaction. In *Proceedings of the 14th international conference on Intelligent user interfaces* (Sanibel Island, Florida, USA, 2009). ACM, 1502663, 67-76.
- [10] Knijnenburg, B.P., Willemsen, M.C., Gantner, Z., Soncu, H. and Newell, C. 2011. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction Journal (UMUAI), Special Issue on User Interfaces for Recommender Systems* (Upcoming).
- [11] McCarthy, K., Reilly, J., Smyth, B. and McGinty, L. 2005. Generating Diverse Compound Critiques. *Artif. Intell. Rev.* 24, 3-4, 339-357.
- [12] McGinty, L. and Smyth, B. 2003. On the role of diversity in conversational recommender systems. In *Proceedings of the 5th international conference on Case-based reasoning: Research and Development* (Trondheim, Norway, 2003). Springer-Verlag, Berlin, 276-290.
- [13] McNee, S.M., Riedl, J. and Konstan, J.A. 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI '06 extended abstracts on Human factors in computing systems* (Montreal, Quebec, Canada, 2006). ACM, 1125659, 1097-1101.
- [14] McSherry, D. 2002. Diversity-Conscious Retrieval. In *Proceedings of the 6th European Conference on Advances in Case-Based Reasoning* (2002). Springer-Verlag, 219-233.
- [15] Mislevy, R.J. and Gitomer, D.H. 1996. The Role of Probability-Based Inference in an Intelligent Tutoring System. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research.* 5, 3-4, 253-282.
- [16] Pu, P. and Chen, L. 2006. Trust building with explanation interfaces. In *Proceedings of the 11th international conference on Intelligent user interfaces* (Sydney, Australia, 2006). ACM, 1111475, 93-100.
- [17] Pu, P. and Chen, L. 2010. A User-Centric Evaluation Framework of Recommender Systems. In *Proceedings of the ACM RecSys 2010 Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI)* (Barcelona, Spain, 2010). CEUR-WS.org, 14-21.
- [18] Pu, P., Zhou, M. and Castagnos, S. 2009. Critiquing recommenders for public taste products. In *Proceedings of the third ACM conference on Recommender systems* (New York, New York, USA, 2009). ACM, 1639760, 249-252.
- [19] Shimazu, H. 2001. ExpertClerk: navigating shoppers' buying process with the combination of asking and proposing. In *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 2* (Seattle, WA, USA, 2001). Morgan Kaufmann Publishers Inc., 1642287, 1443-1448.
- [20] Smith, D., Menon, S. and Sivakumar, K. 2005. Online peer and editorial recommendations, trust, and choice in virtual markets. *Journal of Interactive Marketing.* 19, 3, 15-37.
- [21] Smyth, B. and McClave, P. 2001. Similarity vs. Diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development* (2001). Springer-Verlag, 758890, 347-361.
- [22] Zhang, M. and Hurley, N. 2008. Avoiding monotony: improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM conference on Recommender systems* (Lausanne, Switzerland, 2008). ACM, 1454030, 123-130.
- [23] Ziegler, C.-N., McNee, S.M., Konstan, J.A. and Lausen, G. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web* (Chiba, Japan, 2005). ACM, 22-32.

# An evaluation of novelty and diversity based on fuzzy logic

Simone Santini\*  
Escuela Politécnica Superior  
Universidad Autónoma de Madrid  
simone.santini@uam.es

Pablo Castells  
Escuela Politécnica Superior  
Universidad Autónoma de Madrid  
pablo.castells@uam.es

## ABSTRACT

Information retrieval systems are based on an estimation or prediction of the *relevance* of documents for certain topics associated to a query or, in the case of recommendation systems, for a certain user profile.

Most systems use a graded relevance estimation (a.k.a. relevance status value), that is, a real value  $r(d, \tau) \in [0, 1]$  for the relevance of document  $d$  with respect to topic  $\tau$ . In retrieval systems based on the Probability Ranking Principle [9], this value has a probabilistic interpretation, that is,  $r(d, \tau)$  is equivalent (in rank) to the probability that a user will consider the document relevant. We contend in this paper for an alternative interpretation, where the value  $r(d, \tau)$  is considered as the fuzzy truth value of the statement “ $d$  is relevant for  $\tau$ ”. We develop and evaluate two measures that determine the quality of a result set in terms of *diversity* and *novelty* based on this fuzzy interpretation.

## 1. INTRODUCTION

Information Retrieval (IR) theory and systems revolve around the core –and ill-defined– notion of *relevance*. IR models, methods, evaluation and –if we may use the term– philosophy are concerned with the estimation, prediction, assessment, evaluation, formalization, and understanding of relevance. In a simple and generic formulation of a retrieval system (the one we shall use in this paper), we have a set  $\mathcal{T}$  of *topics* of interest, a data base  $\mathcal{D}$  of documents, and a function  $r$  over  $\mathcal{T} \times \mathcal{D}$ , where  $r(d, \tau)$  represents the *relevance status* of the document  $d$  for topic  $\tau$ . In the Boolean IR model, relevance takes values in  $\{0, 1\}$  or, more in general, in a set isomorphic to the boolean data type **2**;  $r(d, \tau) = \text{true}$  if document  $d$  is relevant for topic  $\tau$ , while  $r(d, \tau) = \text{false}$  if document  $d$  is not. This crude characterization has often proved insufficient: many algorithms and methods require a finer notion of the relevance of documents than simply declaring them relevant or not relevant. For this reason, IR systems usually work with a *graded* relevance  $r(d, \tau) \in [0, 1]$ .

How are we to interpret graded relevance? What is the precise meaning of a statement such as  $r(d, \tau) = 0.8$ ? This

\*This work was supported by the *Ministerio de Educación y Ciencia* under the grant *N. MEC TIN2008-06566-C04-02*, Information Retrieval on different media based on multidimensional models: relevance, novelty, personalization and context.

important semantic question is generally overlooked, mostly because in standard systems the way we interpret relevance does not make all that difference. IR systems return documents sorted by their relevance status value, and under any reasonable interpretation of  $r$ , it is always the case that a document with  $r(d, \tau) = 0.8$  is more “desirable” than a document with  $r(d, \tau) = 0.2$ , and should be returned in a higher position. This being the case, who cares what  $r(d, \tau) = 0.8$  really means? The issue, however, is quite important in more recent systems that deal with *diversity* and *novelty* [10, 1, 3]. In these cases, relevance status values are used in objective functions for retrieval result diversification, and ground truth relevance values are used as arguments in diversity-oriented IR quality metrics. Here, it is not just a matter of which documents are more relevant than others, but of which are the appropriate tools to manipulate relevance values. These tools depend on the way such relevance values are interpreted.

One common interpretation of relevance is *probabilistic* [9, 11, 1, 12]. In this interpretation, the value  $r(d, \tau)$  represents –or is rank-equivalent to– the probability that a user will consider  $d$  relevant for topic  $\tau$ . This identification has important consequences, as it entails that the appropriate machinery for manipulating relevance is Bayesian (e.g. multiplication for independent events, the Bayes theorem for conditional probabilities, etc.). As an alternative to the probabilistic interpretation, we explore a *fuzzy* (graded truth) interpretation of relevance, lifting the binary relevance assumption. Our motivation rests on the difference between uncertainty (caused by incomplete information) and fuzziness (which is a characteristic of linguistic descriptions such as *relevant*).

The endorsement of fuzziness over uncertainty entails a different choice of manipulation instruments. We shall use a version of fuzzy logic to express formally the statement that a set of result  $\mathcal{R}$  is *novel* (has no redundancies) and *diverse* (covers all the topics of interest). The fuzzy interpretation of the relevance will transform these statements too into fuzzy formulas, so that for each set of results  $\mathcal{R}$  we shall be able to give the degree of truth of the statement  $\mathcal{R}$  is *novel and diverse* and, consequently, to pick the set for which the statement is most true.

## 2. THE SEMANTIC OF RELEVANCE

As we have mentioned in the introduction, relevance is often given in the form of a real number, generally as  $r(d, \tau) \in [0, 1]$ . The obvious question to ask (one, as we shall see, that bears quite strongly on the form that the systems should

take) is: what is the interpretation that we should give to this value?

The most common interpretation of this value that is given in information retrieval is probabilistic, that is: *the value  $r(d, \tau)$  represents the probability that a user will consider document  $d$  relevant for topic  $\tau$* . The probabilistic framework entails that we are dealing with a situation in classical logic subject to uncertainty due to limited information. That is, the underlying model is still that of documents that either completely relevant or completely irrelevant (that is, relevance can be described within the framework of Boolean propositional logic), but we do not have enough information to make a determination [5].

We explore here an alternative logical framework for the question of relevance to be posed. In reality, the documents are given and known completely, so (within the limits of the modeling techniques used) instead of modeling the uncertainty in the determination of relevance, one may consider the relevance of a document for a certain topic as a *fuzzy truth value*. This corresponds to the most natural linguistic description that one might give of a document. One doesn't just describe a document as relevant or not relevant: one would rather say that a document is *not very* relevant, *somewhat* relevant, *very* relevant, and so on. These linguistic qualifiers are appropriately modeled with graded truth values rather than with formalisms that deal with uncertainty.

A good example of the difference between the two is given in [2]. Imagine a bottle of water locked in a pantry, so that we can't see it. We know that the bottle is either full or empty, but we have no information about which is which. We can model this situation of uncertainty by saying that with probability 0.5 the bottle is full. Even if we don't know which is which, the bottle is still either completely full or completely empty. The situation is the opposite if *we can see* the bottle and the bottle is half full. In this case, we have complete information: there is no uncertainty involved, and all observers will agree that the bottle is half full. We say in this case that the statement "the bottle is full" has a *truth value* of 0.5; we have fuzzyness, but no uncertainty.

Relevance assessment can be dealt with analogously: the values  $r(d, \tau)$  do not model uncertainty (since, as we have said, we have complete information about the documents), but the fuzzyness of the statement *document  $d$  is relevant for topic  $\tau$* . They are not probabilities, but degrees of truth. The assumption of graded truth entails that the right formalism to use is that of fuzzy logic, to which we shall give a brief introduction in the next section.

### 3. FUZZY LOGIC AND BL-ALGEBRA

There are several approaches to develop a fuzzy logic. One can start with the basic connective and an involutive negation [4], or define the operations based on a suitable t-norm. The latter approach, which we shall follow here, is based mainly on [7, 6].

DEFINITION 3.1. *A (continuous) t-norm is a continuous*

*function  $*$ :  $[0, 1]^2 \rightarrow [0, 1]$  such that, for all  $x, y, x \in [0, 1]$*

- i)  $x * y = y * x$  (commutativity)
- ii)  $(x * y) * z = x * (y * z)$  (associativity)
- iii)  $x \leq y \Rightarrow x * z \leq y * z$  (left monotony)
- iv)  $x \leq y \Rightarrow z * x \leq z * y$  (right monotony)
- v)  $1 * x = x$
- vi)  $0 * x = 0$

(1)

(Note that property iv is redundant, as it is a consequence of commutativity and left monotony.)

DEFINITION 3.2. *A BL-algebra is an algebra*

$$\mathbf{L} = ([0, 1], \cap, \cup, *, \Rightarrow, 0, 1) \quad (2)$$

where

- i)  $([0, 1], \cap, \cup, 0, 1)$  is a lattice with least element 0 and largest element 1;
- ii)  $(L, *, 1)$  is a commutative semigroup, where  $*$  is a t-norm;
- iii) for all  $x, y, z$ :
  - a)  $z \leq (x \Rightarrow y)$  iff  $x * z \leq y$ ;
  - b)  $x \cap y = x * (x \Rightarrow y)$ ;
  - c)  $x \cup y = ((x \Rightarrow y) \Rightarrow y) \cap ((y \Rightarrow x) \Rightarrow x)$ ;
  - d)  $(x \Rightarrow y) \cup (y \Rightarrow x) = 1$ .

Property a and the continuity of  $*$  imply that  $\Rightarrow$  is the residual of  $*$  [7]:

$$x \Rightarrow y = \sup\{z \mid z * x \leq y\} \quad (3)$$

that is, that  $\mathbf{L}$  is a residuated lattice. Property b and continuity imply that  $x \cap y = \min\{x, y\}$ , while property c and continuity imply that  $x \cup y = \max\{x, y\}$ .

The syntax of the fuzzy logic is based on two operators: the *strong conjunction*  $\sqcap$  and the implication  $\rightarrow$ , as well as the constant  $\bar{0}$ . Formulas are composed of propositional variables, the constant, and these operators. Well formed formulas are defined recursively: propositional variables and  $\bar{0}$  are well formed formulas; if  $\phi$  and  $\psi$  are well formed formulas then

$$\phi \sqcap \psi \quad \phi \rightarrow \psi \quad (\phi) \quad (4)$$

are as well. Nothing else is a well formed formula. Let  $W$  be the set of well formed formulas. An *evaluation function* assigns a value  $e(x)$  to each propositional variable  $x$  and extends to a function  $e : W \rightarrow [0, 1]$  through the definition

$$\begin{aligned} e(\bar{0}) &= 0 \\ e(x \sqcap y) &= e(x) * e(y) \\ e(x \rightarrow y) &= e(x) \Rightarrow e(y) \end{aligned} \quad (5)$$

Further connectives are defined as:

$$\begin{aligned} \phi \wedge \psi &\text{ is } \phi \sqcap (\phi \rightarrow \psi) && \text{(conjunction)} \\ \phi \vee \psi &\text{ is } ((\phi \rightarrow \psi) \rightarrow \psi) \wedge ((\psi \rightarrow \phi) \rightarrow \phi) && \text{(disjunction)} \\ \neg \phi &\text{ is } \phi \rightarrow \bar{0} && \text{(negation)} \\ \phi \equiv \psi &\text{ is } (\phi \rightarrow \psi) \sqcap (\psi \rightarrow \phi) && \end{aligned} \quad (6)$$

A formula  $\phi$  is a tautology if  $e(\phi) = 1$  for each evaluation function  $e$ . Based on this syntax and the algebraic semantics, different logic systems can be obtained by selecting different axioms. Here we shall use the standard axioms of [7]. The deduction rule is modus ponens. One consequence of the use of certain t-norms, which constitutes a problem in our case, is that the negation might degenerate into a two-values function, that is, with the residual of many t-norms we have

$$(x \Rightarrow 0) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

to avoid this, we requires that the Lukasiewicz axiom be true, namely that  $\neg\neg\phi = \phi$ . This constraints us to make use of the Lukasiewicz norm

$$x * y = \max\{0, x + y - 1\} \quad (8)$$

a choice that gives us

$$x \Rightarrow y = \begin{cases} 1 & \text{if } x < y \\ 1 + y - x & \text{otherwise} \end{cases} \quad (9)$$

$$\neg x = 1 - x$$

Finally, we efine the *true* constant  $\bar{1} = \neg\bar{0} = \bar{0} \rightarrow \bar{0}$ .

**THEOREM 3.1.** *The following are tautologies in the BL-algebra*

$$\begin{aligned} &\bar{1} \\ &\phi \rightarrow \bar{1} \end{aligned} \quad (10)$$

### 3.1 Quantifiers

In this paper, we shall define the semantics of formulas only on finite models. In this context, a quantifier can be seen as a mapping from the power set of the set of truth values to truth values. For example, the (classical) quantifier  $\forall$ , used in an expression like  $\forall x.p$ , where the model of  $x$  is the finite set  $X = \{x_1, \dots, x_n\}$  can be seen as a mapping  $\forall : 2^X \rightarrow 2$  such that  $\forall : \{p_1, \dots, p_n\} \mapsto \text{true}$  only if all the  $p_n$  are *true* [13]. While in classical logic there are two quantifiers ( $\forall, \exists$ ), in fuzzy logic there is an infinite family of quantifiers, which are used to model linguistic expressions such as *many, few, about ten*, etc. Here we shall only need quantifiers from the simplest of such family, the so-called *type <1> quantifiers* [8].

We shall define and analyze two families of quantifiers, which we shall call the *strong* and the *weak* family. In the weak family we give independent definitions of the universal and the existential quantifiers, that is, we will not use the classical logic equivalence  $\forall x.p \equiv \neg\exists x.\neg p$ . This will give us some more freedom to choose the t-norm on which we will base our system, since we will not have to worry too much if the negation degenerates into a binary-valued operation.

Given a finite model  $X = \{x_1, \dots, x_n\}$  and a unary logic function  $p$ , the expression  $\forall x.p$  is true to the extent that  $p$  is true for all the values  $x \in X$ . This entails the definition:

$$\forall x.p \text{ is } \bigwedge_{i=1}^n p(x_i) \quad (11)$$

and

$$e(\forall x.p) = \bigcap_{i=1}^n e(p(x_i)) = \min_X e(p(x_i)) \quad (12)$$

The existential quantifier we interpret independently as the quantifier that is true to the extent that *at least one* of the propositions  $p(x_i)$  is true, that is

$$\exists x.p \text{ is } \bigvee_{i=1}^n p(x_i) \quad (13)$$

and

$$e(\exists x.p) = \bigcup_{i=1}^n e(p(x_i)) = \max_X e(p(x_i)) \quad (14)$$

In the case of Lukasiewicz logic, in which  $\neg\neg\phi = \phi$ , the weak quantifiers still have the property that  $\exists x.p = \neg\forall x.\neg p$ , however this is not true in general.

The second possibility is to define a *strong (universal) quantifier* using the strong conjunction. In this case we have

$$\forall x.p \text{ is } \prod_{i=1}^n p(x_i) \quad (15)$$

and

$$e(\forall x.p) = e(p(x_1)) * e(p(x_2)) * \dots * e(p(x_n)) \quad (16)$$

In this case we can't define the existential quantifier as we have done for the weak case, since we don't have a corresponding *strong* disjunction. Rather, we will resort to the standard idea from classical logic: *there is an  $x_i$  such that  $p(x_i)$  is true to the extent that "not for all  $x_i$  is  $p(x_i)$  false"*, that is:

$$\exists x.p \equiv \neg\forall x.\neg p \text{ is } \neg \bigwedge_{i=1}^n \neg p(x_i) \equiv (\bigwedge_{i=1}^n (p(x_i) \rightarrow \bar{0})) \rightarrow \bar{0} \quad (17)$$

The fuzzy logic that we have introduced is sound with respect to the BL-algebra (every theorem of fuzzy logic is a tautology in the BL-algebra) and the Lukasiewicz logic is complete with respect to the class of MV-algebras, that is, of the algebras such that, for all  $x$ ,  $((x \Rightarrow 0) \Rightarrow 0) = x$ . So, we have two ways to prove that a formula is true. We can either derive it from the axioms of fuzzy logic using modus ponens, or we can prove that it is a tautology in the BL-algebra (or in the MV-algebra, in the case of Lukasiewicz logic) based only on the general properties of the evaluation function, and independently of the evaluation of the predicate variables that appear in the formula. The first way is formally more correct, but much more labor-intensive. Since in this paper we shall not need too many properties, we shall in general resort to the second method.

**THEOREM 3.2.** *For both the weak and the strong quantifiers it is*

$$\forall x.\forall y.p \equiv \forall y.\forall x.p \quad (18)$$

(The proof is a simple application of the definition and associativity.)

The strong conjunction and the strong quantifier have a problem, which is particularly pernicious for our application. In most of the logic systems, the formula

$$\phi \rightarrow (\phi \sqcap \phi) \quad (19)$$

is not a theorem of Fuzzy logic<sup>1</sup>. The reason is that, for any

<sup>1</sup>An exception is Gödel logic, in which this is taken as an axiom. Gödel logic, however, entails that

$$e(\phi \sqcap \psi) = \min\{e(\phi), e(\psi)\}$$

that is, the t-norm  $*$  is "min".

t-norm that is not *min*, we have  $x * x < x$  so

$$e(\phi \rightarrow (\phi \sqcap \phi)) = e(\phi) \Rightarrow (e(\phi) * e(\phi)) \quad (20)$$

and, setting  $x = e(\phi)$  and  $y = e(\phi) * e(\phi) < x$ , we have

$$e(\phi \rightarrow (\phi \sqcap \phi)) = \sup\{z | x * z < y\} < 1 \quad (21)$$

The fact that  $e(\phi \sqcap \phi) < e(\phi)$  means that, if we take a series of predicates  $p(x_i)$  such that  $e(p(x_i)) = x$ , the value

$$e(\forall x.p) = \overbrace{x * x * \dots * x}^n \quad (22)$$

will become, for  $n$  large enough, equal to zero: the quantification of a large enough number of predicates that are not entirely true will yield false. For example, consider the case of the Lukasiewicz norm. Here  $x * x = \max\{0, 2x - 1\}$  and

$$\overbrace{x * x * \dots * x}^n = \max\{0, (n+1)x - n\} \quad (23)$$

so that for  $n > \frac{x}{1-x}$  the quantifier will be false. As we shall see in the following, if we look for a set  $\mathcal{R}$  with  $n$  results, we shall have to do several universal quantifications on universes with  $n$  members and, unless  $n$  is very small or the relevance of the documents is very close to 1, we shall get a score of 0 for all sets.

#### 4. DIVERSITY AND NOVELTY

We now have the tools to express the diversity and novelty of a set of result under the fuzzy interpretation of relevance. For the sake of clarity, we shall derive two separate predicates, one for diversity and one for novelty that we shall then join in a conjunction to derive the statement *set  $\mathcal{R}$  is novel (non-redundant) and diverse*. Here we assume that in all quantifications, the variables  $d$  and  $d'$  will range over  $\mathcal{R}$ , while the variable  $\tau$  will range over the set  $\mathcal{T}$  of topics. That is, we shall use the following short forms:

$$\begin{aligned} \forall d.p &\equiv \forall d.(d \in \mathcal{R} \rightarrow p) \\ \exists d.p &\equiv \exists d.(d \in \mathcal{R} \wedge p) \\ \forall \tau.p &\equiv \forall \tau.(\tau \in \mathcal{T} \rightarrow p) \\ \exists \tau.p &\equiv \exists \tau.(\tau \in \mathcal{T} \wedge p) \end{aligned} \quad (24)$$

A result set  $\mathcal{R}$  is *diverse* if for every topic there is a document in the set that is relevant for it. That is, the statement  $\mathfrak{D}(\mathcal{R})$  can be expressed simply as

$$\mathfrak{D}(\mathcal{R}) \equiv \forall \tau. \exists d.r(d, \tau) \quad (25)$$

A document is *novel* (or *non-redundant*) if there is at least one topic for which only that document is relevant, and a set is *novel* if all its documents are novel. That is:

$$\mathfrak{N}(\mathcal{R}) \equiv \forall d. \exists \tau.(r(d, \tau) \wedge \forall d'.(r(d', \tau) \rightarrow d = d')) \quad (26)$$

We shall call this the *weak* novelty. There is another possibility of defining novelty, which we shall call *strong*. We can require that there be no overlapping between the topics covered by the documents, that is, whenever a document  $d$  is relevant for a topic, no other document is relevant for that topic. That is:

$$\mathfrak{N}'(\mathcal{R}) \equiv \forall d. \forall \tau.(r(d, \tau) \rightarrow \forall d'.(r(d', \tau) \rightarrow d = d')) \quad (27)$$

We leave as an exercise to the reader to prove, using the definition of the quantifiers, the axioms and modus ponens, that, for an arbitrary  $\mathcal{R}$ ,

$$\mathfrak{N}'(\mathcal{R}) \rightarrow \mathfrak{N}(\mathcal{R}) \quad (28)$$

A set  $\mathcal{R}$  is *qualified* if it is diverse and novel. Since we have two versions of novelty, we have correspondingly two definitions of qualification. The strong qualification is defined as

$$\begin{aligned} \mathfrak{S}(\mathcal{R}) &= \mathfrak{D}(\mathcal{R}) \wedge \mathfrak{N}'(\mathcal{R}) \\ &= \forall \tau. \exists d.(r(d, \tau)) \wedge \forall d. \forall \tau.(r(d, \tau) \rightarrow \forall d'.(r(d', \tau) \rightarrow d = d')) \\ &= \forall \tau. (\exists d.r(d, \tau) \wedge \forall d.(r(d, \tau) \rightarrow \forall d'.(r(d', \tau) \rightarrow d = d'))) \end{aligned} \quad (29)$$

while the weak qualification is defined as

$$\begin{aligned} \mathfrak{S}(\mathcal{R}) &= \mathfrak{D}(\mathcal{R}) \wedge \mathfrak{N}(\mathcal{R}) \\ &= \forall \tau. \exists d.(r(d, \tau)) \wedge \forall d. \exists \tau.(r(d, \tau) \wedge \forall d'.(r(d', \tau) \rightarrow d = d')) \end{aligned} \quad (30)$$

Before we write down the evaluation functions for these formulas, we consider the translation of the logical function (of  $d$  and  $\tau$ )

$$r(d, \tau) \rightarrow \forall d'.(r(d', \tau) \rightarrow d = d') \quad (31)$$

The statement  $d' = d$  is crisp, so it evaluates to 0 or to 1. If  $d' \neq d$ , then

$$e(r(d', \tau) \rightarrow d = d') = e(r(d', \tau) \rightarrow \bar{0}) = e(\neg r(d', \tau)) \quad (32)$$

while if  $d = d'$

$$e(r(d', \tau) \rightarrow d = d') = e(r(d', \tau) \rightarrow \bar{1}) = 1 \quad (33)$$

The quantification, whichever form it takes, is a conjunction (either strong or weak), and 1 is its unit, so, in the case of the quantification we have

$$\forall d'.(r(d', \tau) \rightarrow d = d') = \bigwedge_{d' \neq d} \neg r(d', \tau) \quad (34)$$

and, in the case of strong quantification we have

$$\forall d'.(r(d', \tau) \rightarrow d = d') = \prod_{d' \neq d} \neg r(d', \tau). \quad (35)$$

As we have seen, in addition to the difference in the formula, we have different ways of implementing the quantifiers. Using the strong quantifiers on the strong formula leads to the  ${}^S\mathbf{S}$  (strong-strong) evaluation function

$$\begin{aligned} {}^S\mathbf{S}(\mathcal{R}) &= \prod_{\tau=1}^T \left[ \neg \prod_{d=1}^D \neg r(d, \tau) \prod \right. \\ &\quad \left. \prod_{i=1}^D (r(d, \tau) \rightarrow \prod_{d' \neq d} \neg r(d', \tau)) \right] \end{aligned} \quad (36)$$

while if we use the weak quantifiers, we get the  ${}^S\mathbf{W}$  evaluation function

$${}^S\mathbf{W}(\mathcal{R}) = \bigwedge_{\tau=1}^T \left[ \bigvee_{d=1}^D r(d, \tau) \wedge \bigwedge_{i=1}^D (r(d, \tau) \rightarrow \bigwedge_{d' \neq d} \neg r(d', \tau)) \right] \quad (37)$$

Similarly, the two versions of the weak formula are

$$\begin{aligned} {}^S\mathbf{W}(\mathcal{R}) &= \prod_{\tau=1}^T (\neg \prod_{d=1}^D \neg r(d, \tau)) \prod \\ &\quad \prod_{d=1}^D \left[ \neg \prod_{\tau=1}^T \neg r(d, \tau) \prod \prod_{d' \neq d} \neg r(d', \tau) \right] \end{aligned} \quad (38)$$

and

$${}^W\mathbf{W}(\mathcal{R}) = \bigwedge_{\tau=1}^T \bigvee_{d=1}^D r(d, \tau) \wedge \bigwedge_{d=1}^D \bigvee_{\tau=1}^T \left[ r(d, \tau) \wedge \bigwedge_{d' \neq d} \neg r(d', \tau) \right] \quad (39)$$



The observations of the previous section, in particular eq. (23) advise against the use of the strong quantifiers in large scale problems, so in the following we shall in general limit our considerations to the evaluation functions (37) and (39).

With these functions, we can formulate our two versions of the diversity and novelty optimization problem.

**STRONG FUZZY DIVERSITY(n)**: Given a data base of documents  $\mathcal{D}$ , a set of  $T$  categories  $\mathcal{T}$ , and the relevance measures  $r(d, \tau)$  with  $d \in \mathcal{D}$  and  $\tau \in \mathcal{T}$ , find the subset  $\mathcal{R} \subseteq \mathcal{D}$  with  $|\mathcal{R}| = n$  such that  ${}^w\mathbf{S}(\mathcal{R})$  is maximum.

The problem **WEAK FUZZY DIVERSITY(n)** is analogous but, in this case, the function that is maximized is  ${}^w\mathbf{W}(\mathcal{R})$ .

## 5. COMPLEXITY

Information retrieval with novelty and diversity often generates intractable problems [10] and our formulation is not, unfortunately, an exception, as we following theorems show. In order to show NP-completeness we have to transform the optimization problems into equivalent decision problems. The decision problem corresponding to **STRONG FUZZY DIVERSITY(n)** is the following:

**STRONG FUZZY DECISION(n)**: Given a data base of documents  $\mathcal{D}$ , a set of  $T$  categories  $\mathcal{T}$ , the relevance measures  $r(d, \tau)$  (with  $d \in \mathcal{D}$  and  $\tau \in \mathcal{T}$ ), and a number  $\rho \in [0, 1]$  does there exist a subset  $\mathcal{R} \subseteq \mathcal{D}$  with  $|\mathcal{R}| = n$  such that  ${}^w\mathbf{S}(\mathcal{R}) \geq \rho$ ?

The problem **WEAK FUZZY DECISION(n)** is defined analogously.

**THEOREM 5.1.** **WEAK FUZZY DECISION(n)** is NP-complete.

**PROOF.** We shall prove the theorem with a reduction from **X3C** (Exact cover by 3-sets). The statement of the problem is as follows: given a set  $X$  with  $|X| = 3q$  and a collection  $C$  of 3-element subsets of  $X$ , does  $C$  contain a subset  $C' \subseteq C$  such that every element of  $X$  occurs exactly in an element of  $C'$ ?

Note that, although the number of sets in  $C'$  is not explicitly stated in the theorem, the constraints of the problem entail that  $C'$  contains  $q$  sets.

We reduce the problem to **WEAK FUZZY DECISION** as follows. The set  $\mathcal{T}$  of categories will have one category for each element of  $X$ . There will be a document for each subset  $c \in C$ , and we shall set  $r(d, \tau) = 1$  if  $c$  contains the element of  $X$  represented by  $\tau$ , and 0 otherwise.

We claim that **WEAK FUZZY DECISION(q)** has a solution with  $\rho = 1$  if and only if **X3C** has a solution.

Remember that we can write

$${}^w\mathbf{W}(\mathcal{R}) = \mathfrak{D}(\mathcal{R}) \wedge \mathfrak{N}(\mathcal{R}) \quad (40)$$

where the logic quantifiers in  $\mathfrak{D}$  and  $\mathfrak{N}$  are interpreted in the weak sense. Consider the term  $\mathfrak{D}$ . We have

$$\mathfrak{D}(\mathcal{R}) = \min_{\tau \in \mathcal{T}} \max_{d \in \mathcal{D}} d(d, \tau) \quad (41)$$

$\mathfrak{D}(\mathcal{R}) = 1$  if and only if all the “max” that appear in the equation have a value of 1, that is, if and only if for each

category (viz. element of  $X$ ) there is a document (viz. subset in  $\mathcal{R}$ ) that contains it. In other words,  $\mathfrak{D}(\mathcal{R}) = 1$  iff

$$X \subseteq \bigcup_{c \in C'} c \quad (42)$$

Note however, that  $X$  is the universe of discourse, and that no subset  $c$  can contain any element not in  $X$ . So  $\mathfrak{D}(\mathcal{R}) = 1$  iff

$$X = \bigcup_{c \in C'} c \quad (43)$$

Suppose now that there is a solution to **X3C**. In this case, (43) holds, so  $\mathfrak{D}(\mathcal{R}) = 1$ . What about  $\mathfrak{N}(\mathcal{R})$ ? Suppose, by contradiction, that  $\mathfrak{N}(\mathcal{R}) < 1$ . Then there has to be at least one pair  $(d, \tau)$  such that

$$e(r(d, \tau) \wedge \bigwedge_{d' \neq d} \neg r(d', \tau)) < 1 \quad (44)$$

(The actual condition is stronger: there must be one such  $d$  for every  $\tau$ , but the weaker condition will do here.) So, there has to be  $d'$  such that  $e(r(d, \tau) \wedge \neg r(d', \tau)) < 1$ , that is,  $e(r(d, \tau) \wedge r(d', \tau)) > 0$ . Since the values of relevance are always 0 or 1, this means  $e(r(d, \tau) \wedge r(d', \tau)) = 1$ , so the element of  $X$  represented by  $\tau$  belongs to both  $d$  and  $d'$ , i.e. the set represented by  $d$  and  $d'$  are not disjoint, contradicting the fact that a solution was found.

Suppose now that there is a  $\mathcal{R}$  such that  ${}^w\mathbf{W}(\mathcal{R}) = 1$ . In this case, by (43),  $X = \bigcup c$ , that is, the documents in  $\mathcal{R}$  cover all categories. Since there are  $q$  documents,  $3q$  categories, and each document covers only 3 categories, if there were a category represented by more than a document there would also be a category not represented by any document. Since this is not the case, there are no overlaps between the documents, that is, the sets of  $C'$  are disjoint.

Note that in this case we didn't need the condition  $\mathfrak{N}$ : the constraints on the problem guarantee that even without this condition we would have solved **X3C**.  $\square$

**THEOREM 5.2.** **STRONG FUZZY DECISION(n)** is NP-complete.

The proof is based on the same reduction as that of the previous theorem.

## 6. THE BEHAVIOR OF THE FUNCTIONS

In this section we shall carry out a preliminary study of the two fuzzy evaluation functions that we are considering:  ${}^w\mathbf{S}$  and  ${}^w\mathbf{W}$ . Before this, we should make a few methodological considerations. There are, roughly speaking, three categories of methods that we can use to study these functions. We can study them analytically, expressing them in closed form; we can generate data using a known statistical distribution and determine the functions' behavior *vis à vis* certain controlled variables; or we can resort to user data collected from an existing system.

It should be evident that the latter solution, despite its widespread use, is inadequate in this case, since it doesn't allow a fine control over the independent variables and the controlled parameters of the evaluation. Tests on “real” are good for obtaining a qualitative impression of how a whole system works, but would make little sense in our predicament.

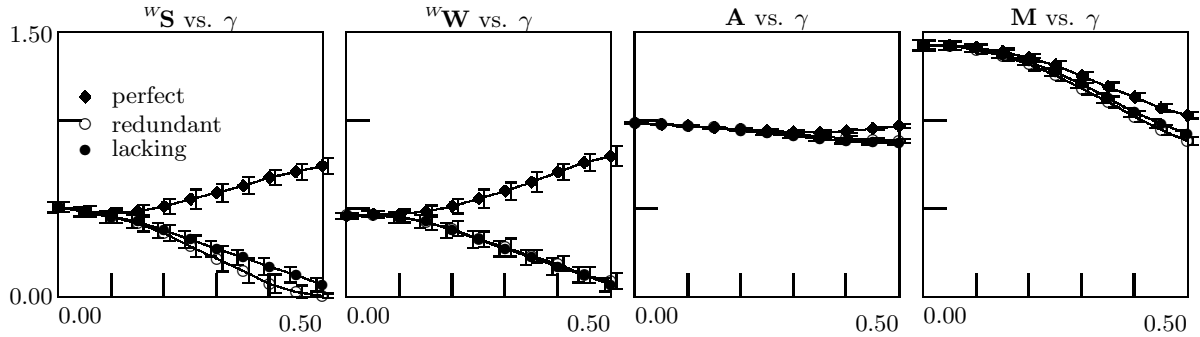


Figure 1: Redundant sets have  $r = 2$ , lacking sets have  $r = -2$ ; all the results have  $c = 24$  and  $r = 6$ .

Closed form solutions are clearly the best way to study a function, but they may be difficult to obtain under very general hypotheses. Here, we study analytically the behavior of our evaluation functions under a simple but telling special case: that of two topics. As we shall see later on, this setting is fairly representative of more general situations. For a more general setting, we recur to numerical calculations with controlled data sets. In this case, we not only calculate our two evaluation functions  ${}^w\mathbf{S}$  and  ${}^w\mathbf{W}$ , but compare them with two examples of the state of the art appeared in the literature: the probabilistic measure presented in [1] (and indicated in the following as **A**), and the *undirected compensatory* measure of [12] (indicated with **M**).

## 6.1 Closed-form model

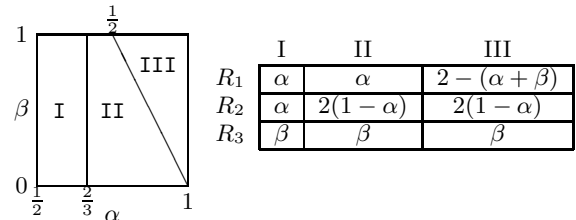
We consider a system with two categories, and result sets of two documents. We shall consider three sets,  $R_1$ ,  $R_2$ , and  $R_3$ . The documents of each set are represented as vectors, where the value  $\alpha > \frac{1}{2}$  represents relevance while the value  $\beta < \frac{1}{2}$  represents irrelevance for a particular topic. The three sets of two documents are as follows:

$$\begin{aligned}
 R_1 &: \begin{cases} d_1 = [\alpha, \beta] \\ d_2 = [\beta, \alpha] \end{cases} \\
 R_2 &: \begin{cases} d_1 = [\alpha, \alpha] \\ d_2 = [\beta, \alpha] \end{cases} \\
 R_3 &: \begin{cases} d_1 = [\alpha, \beta] \\ d_2 = [\beta, \beta] \end{cases}
 \end{aligned} \tag{45}$$

$R_1$  is the “perfect” set: document  $d_1$  is relevant for category  $\tau_1$ , and document  $d_2$  is relevant for  $\tau_2$ . The two documents cover the category range completely and without redundancy. In  $R_2$  the second document is redundant, as  $d_1$  already covers all categories, while in  $R_3$  no document covers category  $\tau_2$ . We shall say that  $R_2$  is *redundant* (viz. has positive redundancy) and that  $R_3$  is *lacking* (viz. has negative redundancy).

Consider first the function  ${}^w\mathbf{S}(\mathcal{R})$  which is, for each of the three result sets, a function of  $\alpha$  and  $\beta$  defined in the square  $\alpha \in [\frac{1}{2}, 1]$ ,  $\beta \in [0, \frac{1}{2}]$ .

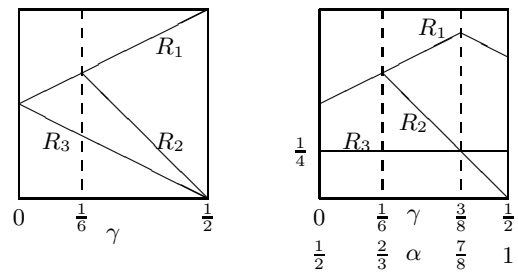
In order to determine the behavior of the function, we shall need to divide the square in three regions as illustrated below together with the values of the function in the three regions.



It must be noted that, for  $\alpha < \frac{2}{3}$ , this function doesn’t discriminate between the “perfect” set and the redundant one. The interpretation of this phenomenon hinges on the definition of redundancy. For low values of  $\alpha$ , it not so obvious that having two documents about the same topic constitutes a true redundancy, since the relevance of a document is low enough that a second document does indeed add relevance. To have a better idea of this phenomenon, consider two different parametrizations of  $\alpha$  and  $\beta$ . First, we consider a path in which  $\alpha$  and  $\beta$  start from a situation of complete confusion and diverge to a situation of crisp (binary) relevance. In particular, we shall consider the parametrization

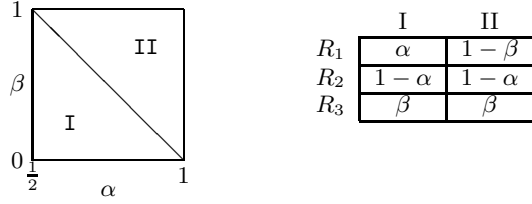
$$\alpha = \frac{1}{2} + \gamma \quad \beta = \frac{1}{2} - \gamma \tag{46}$$

with  $\gamma \in [0, \frac{1}{2}]$ . Then we shall consider the same parametrization of  $\alpha$ , but keeping  $\beta = \frac{1}{4}$ . The value of the function  ${}^w\mathbf{S}$  for the three result sets, as a function of  $\gamma$  with the two parametrizations is the following

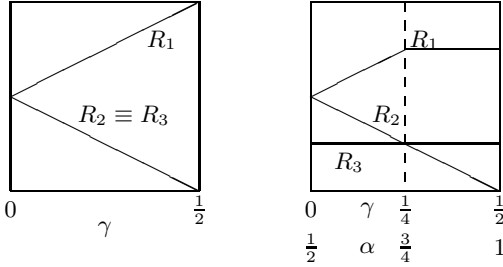


The behavior of the second curve for high values of  $\gamma$  (and therefore of  $\alpha$ ) is due to the presence of region III. In this case, each one of the two documents of the “perfect” set has a certain relevance not only for the category for which it is nominally relevant, but for the other as well ( $\beta = \frac{1}{4}$ ). When  $\alpha$  is high, the fact that, say,  $d_1$  is extremely relevant for  $\tau_1$  while  $d_2$  is also somewhat relevant for the same category creates some redundancy. It is therefore not surprising that in this region the value of the evaluation function begins to decrease, behaving exactly as it does in the case of the redundant set  $R_2$ .

In the case of the  ${}^w\mathbf{W}$  evaluation function, we only have to distinguish two regions, represented here with the corresponding function expressions.



Considering again the parametrization  $\gamma$  and the two previous examples ( $\alpha = \frac{1}{2} + \gamma$ ,  $\beta = \frac{1}{2} - \gamma$  and  $\alpha = \frac{1}{2} + \gamma$ ,  $\beta = \frac{1}{4}$ , respectively, we obtain the following behaviors (behaviors that, in this case, reserve no surprises).



## 6.2 Numerical tests

In order to extend the range of configurations in which we evaluate the functions, and in order to compare them with other functions appeared in the literature, we resorted to numerical evaluation in statistically controlled conditions. We consider a situation with  $c$  topics, in which we seek a result set of  $s$  documents. These values are always chosen in such a way that  $p = c/s$  is a natural number (this assumption doesn't restrict the scenario appreciably, and simplifies data generation). The "perfect" result set contains  $s$  documents, each one of which is relevant to  $p$  topics, without overlaps. This entails that this set is optimally diverse and novel. Imperfect sets are created using a redundancy parameter  $r$ , and having each one of the documents in the result set be relevant for  $p + r$  topics. If  $r < 0$  the set will be lacking (some topics will not be covered), while if  $r > 0$  the set will be redundant. Note that it must be  $1 - p \leq r \leq c - p$ . Relevance and irrelevance scores are modeled as two equally distributed random variables obtained starting with a normal distribution and clipping them to  $[0, 1]$ . That is, if

$$x'_r = N(\alpha, \sigma) \quad x'_{\bar{r}} = N(\beta, \sigma) \quad (47)$$

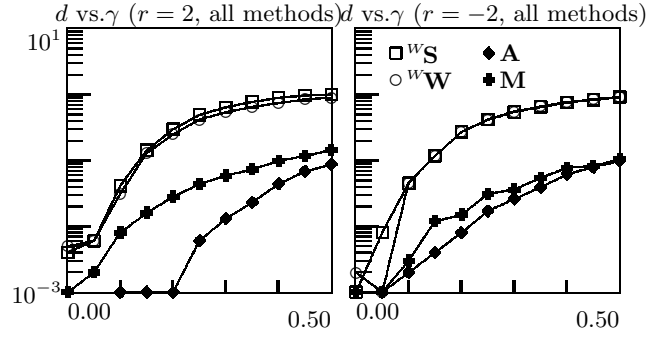
with  $\alpha \geq 1/2$  and  $\beta \leq 1/2$ , then the scores for relevance and non-relevance are

$$\begin{aligned} x_r &= \text{if}(x'_r < 0, 0, \text{if}(x'_r > 1, 1, x'_r)) \\ x_{\bar{r}} &= \text{if}(x'_{\bar{r}} < 0, 0, \text{if}(x'_{\bar{r}} > 1, 1, x'_{\bar{r}})) \end{aligned} \quad (48)$$

The distribution of the normal, for reasonable values of  $\alpha$  and  $\beta$ , if  $\sigma < 0.2$ ; for  $\sigma > 0.2$  the distortion due to clipping becomes preponderant and the results become hard to interpret. We chose to do all the measures with  $\sigma = 0.1$ .

The first diagram is a replica, in the new situation, of the analytical results, using the parametrization (46). The behavior, for the four functions under test, is shown in figure 5.

For  $\gamma = 0$  all documents are statistically the same, so none of the methods distinguish between them. As  $\gamma$  increases,



**Figure 2: Discrimination results for the four measures under test. Redundant sets have  $r = 2$  (graph on the left), lacking sets have  $r = -2$  (graph on the right). All the results have  $c = 24$  and  $r = 6$ .**

and the average difference between relevant and irrelevant documents becomes significant, all four methods separate the perfect set from the redundant and lacking ones (the  $t$ -test shows that with  $\gamma = 0.05$  the separation is already significant for all methods; this result applies to all other measurements so, from now on, in order to simplify the graphs, we will omit the indication of the variance). Qualitatively, we can observe that the two fuzzy measures appear to give a sharper separation between the perfect set and the other, as reflected by the separation of the curves.

In order to verify this effect, we have performed a series of *discrimination* measures. The idea is that, in order to separate the good results from the bad, we are often more interested in the relative difference between the scores than in the absolute values. For this reason, if  $u$  is the score given to a perfect set, and  $v$  is the score of a redundant or lacking set, we define the *discrimination coefficient* between the two as

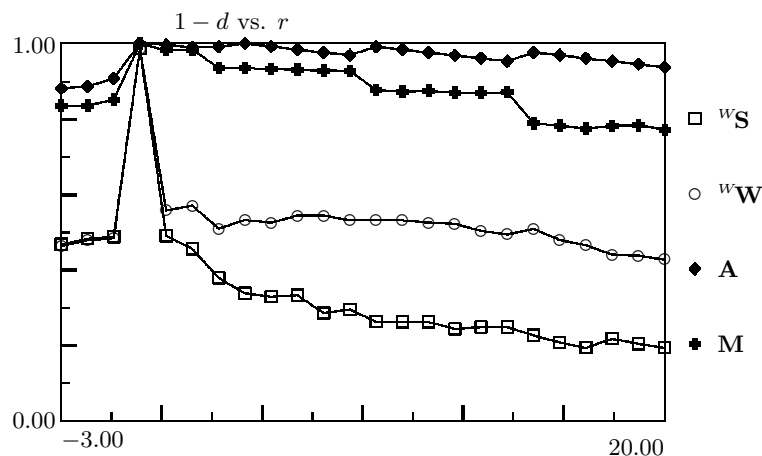
$$d = \frac{|u - v|}{u}. \quad (49)$$

This coefficient is independent of the scale of the measure, and it gives us the degree of separation between the perfect and redundant results as a fraction of the perfect score. The two graphs in figure 6.2 show the discrimination coefficients for the four measures under test<sup>2</sup>.

Here too we observe that the discrimination coefficient grows in a much sharper way in the case of the fuzzy measures than it does in the case of the other two.

As a final measure, we analyze the discrimination as a function of the redundancy (figure 6.2). We fix the averages of the relevance values to  $\alpha = 0.75$  and  $\beta = 0.75$ . We still have  $c = 24$  and  $r = 6$ , which leads to  $p = 4$ , so that the redundancy must be in the range  $-3 \leq r \leq 20$ . In order to make the graph clearer, we plot  $1 - d$  in lieu of  $d$ , so that the plot attains its maximum of 1 for  $r = 0$ , and decreases as  $r$  assumes positive or negative values. The graph confirms the main difference that we had already observed between the logic measure and the others that we are analyzing: in the case of the logic measures, the relative difference in score between the "perfect" score and the others is much more

<sup>2</sup>Note that the organization here is different from that of fig. 5: here each graph is relative to a single redundancy, and contains curves for all four measures. This solution would have been too confused for figure 5 due to the presence of the variance.



**Figure 3: Discrimination ( $1 - d$ ) results for versus redundancy for the four measures under test. All the results have  $c = 24$  and  $r = 6$ , which leads to  $p = 4$  and a range for the redundancy of  $[-3, 20]$ . Here we set  $\alpha = 0.75$ ,  $\beta = 0.25$ .**

pronounced; even relatively minor defects in the result will result in a considerable drop in the score.

## 7. CONCLUSIONS

We have presented a model of novelty and diversity consistent with the idea that relevance measures can be interpreted as fuzzy truth values, overcoming the binary relevance simplification. We have derived two different evaluation functions, depending on the specific form of the quantifier used, and we have compared them with two examples of the state of the art.

With respect to other functions, the main characteristics of the logic ones is the sharp decrease in the relative score difference between “perfect” sets and sets with even limited redundancy or lack. Whether this sharpness is an asset or a liability depends, of course, on the specifics of the system that one is designing. At the very least, however, the availability of the logic model provides additional tools to the designer of information retrieval and recommender systems.

A possible way to reduce this discrimination, that we shall study in the future, is to make use of other quantifiers. For example, instead of expressing logically the statement for each document  $d$  there is a category  $\tau$  that only  $d$  has, we could use a different type of fuzzy quantifier to express the statement for most documents  $d$  there is a category  $\tau$  that only  $d$  has.

## 8. REFERENCES

- [1] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Leong. Diversifying search results. In *Proceedings of WDSM '09*. ACM, 2009.
- [2] J. C. Bezdek and S. Pal. *Fuzzy models for pattern recognition*. New York:IEEE Press, 1996.
- [3] Charles Clarke, Maheedhar Kolla, Gordon Cormack, Olga Vechtomova, Azon Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the International ACM SIGIR Conference in Research and Developments in Information Retrieval*. ACM, 2008.
- [4] D. Dubois and H. Prade. A review of fuzzy set aggregation connectives. *Information Sciences*, 36:85–121, 1985.
- [5] Didier Dubois and Henri Prade. Possibility theory, probability theory and multiple-valued logics: A clarification. *Annals of Mathematics and Artificial Intelligence*, 32(1-4):35–66, 2001.
- [6] Francesc Esteva and LLuís Godo. Monoidal t-norm based logic: towards a logic for left-continuous t-norms. *Fuzzy sets and systems*, 124:271–88, 2001.
- [7] Petr Hájek. Basic fuzzy logic and BL-algebras. Technical report V736, Institute of Computer Science, Academy of Science of the Czech Republic, December 1996.
- [8] Michal Holčapek. L-fuzzy quantifiers of the type  $\langle 1^n, 1 \rangle$ . *Fuzzy sets and systems*, 159:1811–35, 2008.
- [9] S. Robertson. The probability ranking principle in IR. *Journal of documentation*, 33:294–304, 1977.
- [10] Simone Santini and Pablo Castells. Intractable problems in novelty and diversity. In *Actas de las XVI Jornadas de Ingeniería del Software y bases de datos*, 2011.
- [11] J. Wang and J. Zhu. Portfolio theory of information retrieval. In *Proceedings of the 32nd International ACM SIGIR Conference in Research and Developments in Information Retrieval*. ACM, 2009.
- [12] Yunjie Xu and Hainan Yin. Novelty and topicality in interactive information retrieval. *Journal of the American Society for Information Science and Technology*, 59(2):201–15, 2008.
- [13] Lofti A. Zadeh. A computational approach to fuzzy quantifiers in natural languages. *Computers and Mathematics with Applications*, 9:149–84, 1983.