Proceedings of the

RecSys 2011

Workshop on

Human Decision Making in Recommender Systems (Decisions@RecSys'11)

and

User-Centric Evaluation of Recommender Systems and Their Interfaces - 2

(UCERSTI 2)

affiliated with the

5th ACM Conference on Recommender Systems

October 23-27, 2011

Chicago, IL, USA

Alexander Felfernig, Li Chen, Monika Mandl,

Martijn Willemsen, Dirk Bollen and Michael Ekstrand (eds.)

Preface (Decisions@RecSys'11)

Interacting with a recommender system means to take different decisions such as selecting a song/movie from a recommendation list, selecting specific feature values (e.g., camera's size, zoom) as criteria, selecting feedback features to be critiqued in a critiquing based recommendation session, or selecting a repair proposal for inconsistent user preferences when interacting with a knowledge-based recommender. In all these scenarios, users have to solve a decision task. The major focus of this workshop (Decisions@RecSys'11) are approaches for efficient human decision making in different recommendation scenarios.

The complexity of decision tasks, limited cognitive resources of users, and the tendency to keep the overall decision effort as low as possible leads to the phenomenon of bounded rationality, i.e., users are exploiting decision heuristics rather than trying to take an optimal decision. Furthermore, preferences of users will likely change throughout a recommendation session, i.e., preferences are constructed in a specific decision environment and users do not know their preferences beforehand.

Decision making under bounded rationality is a door opener for different types of non-conscious influences on the decision behavior of a user. Theories from decision psychology and cognitive psychology are trying to explain these influences, for example, decoy effects and defaults can trigger significant shifts in item selection probabilities; in group decision scenarios, the visibility of the preferences of other group members can have a significant impact on the final group decision.

The major goal of this workshop is (was) to establish a platform for industry and academia to present and discuss new ideas and research results that are related to human decision making in recommender systems. The workshop consists (consisted) of technical sessions in which results of ongoing research are (were) presented, informal group discussions on focused topics, and a keynote talk given by Anthony Jameson from DFKI, Germany.

The topics of papers submitted to the workshop can be summarized as follows:

- Decision heuristics: the role of decision heuristics/phenomena (e.g., decoys and anchoring) in the construction of recommender applications.
- Recommender user interfaces: impact of recommender interfaces on human decision-making behavior.
- Group decision making: group recommendation algorithms and group decision strategies.
- Emotion-based recommendation: emotion detection and emotion-aware recommender applications.
- New application domains: smart homes and intelligent data management.

The workshop material (list of accepted papers, invited talk, and the workshop schedule) can be found at the Decisions@RecSys 2011 workshop webpage: recex.ist.tugraz.at/RecSysWorkshop.

Alexander Felfernig, Li Chen, and Monika Mandl
October 2011

Preface (UCERSTI 2)

Research on ”Human-Recommender Interaction” is scarce. Algorithm optimization and off-line testing using measures like RMSE are dominant topics in the RecSys community, but theorizing about consumer decision processes and measuring user satisfaction in online tests is less common. Researchers in Marketing and Decision-Making have been investigating consumer choice processes in great detail, but only sparingly put this knowledge to use in technological applications. Human-Computer Interaction has been focusing on the usability of interfaces for ages, but does not seem to link research on consumer choice and recommender system interfaces.

During RecSys 2010, we organized the first UCERSTI workshop to bridge these gaps. Two keynote speeches, 7 accepted papers and a lively panel discussion introduced the visitors of RecSys 2010 to the field of Human-Recommender Interaction. By means of UCERSTI 2 we hope to further strengthen the bonds between these researchers, to exchange new experiences, and meet other new researchers working on user-centric research in Recommender Systems.

The papers cover the following topics:

- Preference elicitation methods and Decision Making research
- Applications of psychological theory and models in Recommender systems
- User-adaptive recommender interfaces
- Quantitative evaluation of recommender systems such as controlled experiments and field trials
- User-recommender interaction measurement techniques such as questionnaires and process data analysis
- User acceptance of recommender systems

UCERSTI 2 also includes a panel discussion, introduced by Joseph A. Konstan and Bart Knijnenburg, on "Recommender system evaluation: creating a unified, cumulative science”.

Panel description:

The evaluation of recommender systems is typified by a proliferation of claims, metrics and procedures. A review of research papers in Recommender Systems shows a number of typical claims:

- This is an innovative way of recommending
- This algorithm is more accurate than others
- This algorithm is faster for large data sets than others

- This algorithm is better than others along a particular dimension (e.g., diversity, novelty)
- This way of eliciting ratings leads to greater accuracy of recommendations
- This recommender system (algorithm, interface, etc.) is preferred by users
- This recommender system (algorithm, interface, etc.) leads to greater long-term user retention than other systems

For each of these claims recommender systems researchers and practitioners have developed several distinct metrics to evaluate them, as well as a diverse set of procedures to conduct the evaluation. This apparent heterogeneity stands in the way of scientific progress. Researchers face the impossible challenge of selecting a subset of claims/metrics/procedures that allows for comparability of their work with previous studies. To create a rigorous, cumulative science of recommender systems, we need to take a step back and reflect on our current practices.

This reflection is partly philosophical: Which of the possible investigative claims are worthy of our consideration? The answer to this question depends on the purpose or goal we ascribe to a recommender system, whom we feel should benefit from it, and where we believe the field of recommender systems blends into other fields. In other words, we need to decide on what a ”good recommender system” really is.

It is also partly practical: As scientists, we need to understand best practices for providing the evidence to back up these claims, and for providing such evidence in a way that allows our field to move forward. Some claims (e.g., novelty) can simply be supported by a review of related work. Others (e.g., user satisfaction) require careful experimental designs that isolate and make salient as much as possible the factor being studied so that differences in results can be attributed to that factor. Still others (e.g., algorithmic performance) require standardization of metrics and evaluation procedures to ensure apples-to-apples comparisons against the best prior work.

This panel will address the general challenge of building a rigorous, cumulative science out of recommender systems with a specific focus on experiment design and standardization in support of better user-centered evaluation.

More information on UCERSTI2 at: http://ucersti.ieis.tue.nl/

Martijn Willemsen, Dirk Bollen and Michael Ekstrand

October 2011

Workshop Committee (Decisions@RecSys'11)

Chairs

Alexander Felfernig, Graz University of Technology
Li Chen, Hong Kong Baptist University

Organization

Monika Mandl, Graz University of Technology

Program Committee

Mathias Bauer, Mineway, Germany
Shlomo Berkovsky, CSIRO, Australia
Robin Burke, De Paul University, USA
Li Chen, Hong Kong Baptist University, China
Hendrik Drachsler, Open University of the Netherlands
Alexander Felfernig, Graz University of Technology, Austria
Gerhard Friedrich, University of Klagenfurt, Austria
Sergiu Gordea, Austrian Institute for Technology, Austria
Mehmet Goker, Salesforce, USA
Andreas Holzinger, Medical University Graz, Austria
Dietmar Jannach, University of Dortmund, Germany
Alfred Kobsa, University of California, USA
Gerhard Leitner, University of Klagenfurt, Austria
Walid Maalej, Technische Universität München, Germany
Monika Mandl, Graz University of Technology, Austria
Francisco Martin, BigML, USA
Alexandros Nanopoulos, University of Hildesheim, Germany
Francesco Ricci, University of Bolzano, Italy
Olga Santos, UNED, Spain
Monika Schubert, Graz University of Technology, Austria
Markus Strohmaier, Graz University of Technology, Austria
Erich Teppan, University of Klagenfurt, Austria
Nava Tintarev, University of Aberdeen, UK
Marc Torrens, Strands, Spain
Alex Tuzhilin, New York University, USA
Markus Zanker, University of Klagenfurt, Austria
Christoph Zehentner, Graz University of Technology, Austria

Additional Reviewers

Ge Mouzhi, University of Dortmund, Germany

Workshop Committee (UCERSTI)

Organization

Martijn Willemsen, Human-Technology Interaction, Eindhoven University of Technology, Netherlands
Dirk Bollen, Human-Technology Interaction, Eindhoven University of Technology, Netherlands
Michael Ekstrand, GroupLens Research, Department of Computer Science and Engineering University of Minnesota, USA

Program Committee

Benedict G. C. Dellaert, Department of Business Economics, Erasmus University Rotterdam, The Netherlands
Maciej Dabrowski, Digital Enterprise Research Institute, National University of Ireland, Galway, Ireland
Alexander Felfernig, Software Technology Institute, Graz University of Technology, Germany
David Geerts, Centre for User Experience Research, University of Leuven, Belgium
Kristiina Karvonen, Helsinki Institute for Information Technology HIIT, Aalto, Finland
Alfred Kobsa, Donald Bren School of Information and Computer Sciences, University of California, Irvine, USA
Bart Knijnenburg, Donald Bren School of Information and Computer Sciences, University of California, Irvine, USA
Artus Krohn-Grimberghe, Information Systems and Machine Learning Lab, University of Hildesheim, Germany
Sean M. McNee, FTI Technology, USA
Steffen Rendle, Steffen Rendle, Social Network Analysis, University of Konstanz, Germany
Sylvain Senecal, Department of Marketing, HEC Montreal, Canada

Table of Contents (Accepted Full Papers)

Decisions @ RecSys 2011

UCERSTI 2

Decoy Effects in Financial Service E-Sales Systems

Erich Christian Teppan
Dept. of Applied Informatics
Alpen-Adria Universitaet Klagenfurt
9020 Klagenfurt, Austria
Erich.Teppan@uni.klu.ac.at

Alexander Felfernig, Klaus Isak
Dept. of Software Technology
Graz Instititute of Technology
8010 Graz, Austria
Alexander.Felfernig@ist.tu-graz.at
Klaus.Isak@ist.tu-graz.at

ABSTRACT

Users of E-Sales platforms typically face the problem of choosing the most suitable product or service from large and potentially complex assortments. Whereas the problem of finding and presenting suitable items fulfilling the user's requirements can be tackled by providing additional support in the form of recommender- and configuration systems, the control of psychological side effects resulting from irrationalities of human decision making has been widely ignored so far. Decoy effects are one family of biases which have been shown to be relevant in this context. The asymmetric dominance effect and the compromise effect have been shown to be among the most stable decoy effects and therefore also carry big potential for biasing online decision taking. This paper presents two user studies investigating the impacts of the asymmetric dominance and compromise effect in the financial services domain. While the first study uses synthesized items for triggering a decoy effect, the second study uses real products found on konsument.at, which is an Austrian consumer advisory site. Whereas the results of the first study prove the potential influence of decoy effects on online decision making in the financial services domain, the results of the second study provide clear evidence of the practical relevance for real online decision support- and E-sales systems.

Categories and Subject Descriptors

H.5.2 [**INFORMATION INTERFACES AND PRESENTATION**]: User Interfaces—*Graphical user interfaces (GUI)*

General Terms

Human Factors, Experimentation, Theory

Keywords

Decision Phenomena, Decoy Effects, Consumer Decision Making, E-Sales Platforms.

Decisions@RecSys 2011, Chicago, USA

1. INTRODUCTION

It is often hard for customers of E-sales platforms to find suitable products or services (denoted as items for the remainder of this paper) which match their requirements. This challenge is triggered by the size and complexity of the underlying item assortment. Recommender applications facilitate the item identification process by proactively supporting the customer/user in different types of decision scenarios [7]. These systems have been a very active research field for many years which resulted in different solutions for many item domains [10][13][15][19]. What has been widely ignored by the E-sales and recommender community is that once sets of items are presented on some sort of result page, decision phenomena occur which can have significant impacts on customer decision making [30].

One family of effects which have been shown to be relevant in this context are decoy effects [30]. The decoy effect induces an increased attraction of target items with respect to competitor items due to the existence of so-called decoy items. In other words, the target items are those which (should) profit more from the existence of the decoys than the competitors. Two prominent types of decoy effects are the *asymmetric dominance effect* (ADE) [11] and the *compromise effect* (CE) [31]. These two decoy effects differ in terms of the relative positions (described by the corresponding attribute dimensions – in our example: *optical zoom* and *resolution*) of the decoy items in the item landscape (see Figure 1). Compared to the target item, an asymmetrically dominated decoy item (see {d1, d2, d3} in Figure 1) is worse in every dimension (d1) or worse in at least one dimension and equal in the other dimensions (d2 and d3). Compared to the competitor, the asymmetrically dominated decoy item is - though worse in some dimensions - also better in some dimensions. In other words, there are dimensions where the decoy item defeats the competitor, but the decoy defeats the target in none of the given dimensions.

Table 1 shows a simplified example of the ADE (d1) with two attribute dimensions and two items in the domain of digital cameras: The target item is better than the competitor in the dimension resolution (8 mpix) whereas the competitor is better in the dimension optical zoom (6x). In theory, the addition of an asymmetrically dominated decoy the attractiveness of the target increases.

The asymmetry induced by the decoy is most easily shown by the corresponding domination graph which outlines the superiority/inferiority relations between all items in every dimension. Figure 2 is showing the corresponding domination graphs for the example in Table 1.

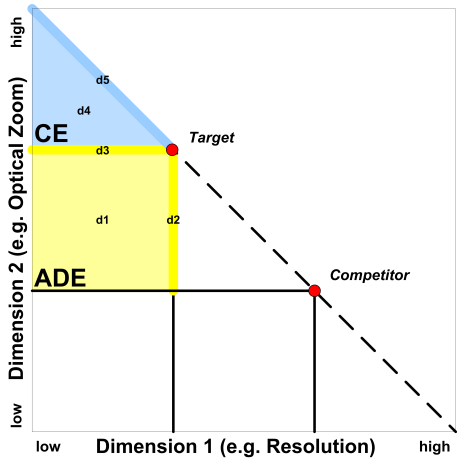


Figure 1: Asymmetrically dominated decoy items (area ADE) and decoy items triggering a compromise effect (area CE) for the benefit of the target.

	Competitor	Target	Decoy
Resolution	8 mpix	5 mpix	6 mpix
Optical Zoom	3x	6x	2x

Table 1: Example of the ADE in the domain of digital cameras. The additional presentation of the decoy shifts the attraction towards the target.

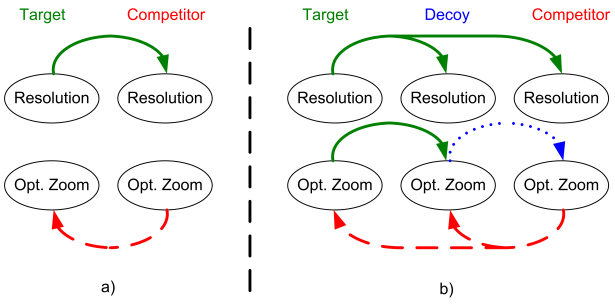


Figure 2: Domination graph: Without decoy (a) the target as well as the competitor dominate each other in one dimension. After inclusion of the decoy (b) the equality seems to change as the target dominates the decoy in both dimensions whereas the competitor dominates the decoy in only one dimension and is even dominated by the decoy in one dimension (blue/spotted arrow).

	Competitor	Target	Decoy
Resolution	10 mpix	7 mpix	3 mpix
Optical Zoom	3x	6x	7x

Table 2: Example of the CE in the domain of digital camera. The additional presentation of the decoy makes the target a good compromise.

In a set without decoy (a), the target and the competitor items are dominating each other in the same number of attributes (i.e. in our case in on attribute dimension each). Due to the inclusion of a decoy item the situation changes (b). Now the target dominates the rest of the set more than the competitor (i.e. three arrows vs. two arrows). As a direct consequence of asymmetrical dominance, d1, d2, and d3 are inferior items such that the overall utility calculated with some objective utility function (e.g. multi attribute utility theory [32]) is lower compared to the target.

Another important decoy effect is the compromise effect [24][31] (see d4 and d5 in Figure 1). The key reason for the existence of this effect is the fact that consumers rather prefer items with medium values in all dimensions than items with extreme values ("good" compromise items). This aspect of human choice behavior is denoted extremeness aversion [28]. Table 2 shows a very simple example.

Again, by the addition of the compromise decoy (d4) the attractivity of the target item is increased compared to the attractivity of the competitor item. The distinction between d4 and d5 is based on an objective utility function [23]. Having such a utility function, all items positioned on the diagonal in Figure 1 are pareto optimal. As a consequence, a d5-decoy has the same overall utility as the target, i.e. does not constitute an inferior item. In this case the only mechanism causing the compromise effect is extremeness aversion. As a d4-decoy also constitutes an extreme item, it triggers extremeness aversion. Additionally it constitutes an inferior item such that the occurring tradeoff contrasts support positive influences for the target. Tradeoff contrasts exist when the advantages of one item outweigh the advantages of another item. In the example of Table 2, the target is much better in the dimension *resolution* than it is defeated by the decoy in the dimension of *optical zoom*. As discussed above, the extreme case of a tradeoff contrast leads to dominance.

The major contributions of this paper are the following: We provide an in-depth analysis of the existence of decoy effects in the financial services domain. In this context we show the existence of decoy effects for result sets with more than three items and also show the effects on the basis of commercial product assortments. The investigations concentrate on the two most important effects, namely the asymmetric dominance effect and the compromise effect. All presented studies have been carried out online and unsupervised and thus preserved a maximum of real world conditions. The results of the presented empirical studies clearly show the impact of decoy effects on item selection behavior of users. Consequently, although not taken into account up to now, these effects play a major role for the construction of recommender and esales applications.

The remainder of this paper is organized as follows. In Section 2 we provide an overview of related work. In Section 3 we discuss the results of a user study based on a synthesized set of financial services. In the following (Sec-

tion 4) we present the results of the second decoy study which is based on a real-world dataset (bankbooks from konsument.at). The impact of decoy effects on the construction of recommender applications is summarized in Section 5. With Section 6 we conclude the paper and provide an outlook of future work.

2. RELATED WORK

The main reason why decoy effects occur in human decision making is that humans often do not act fully rational. Fully rational agents apply some sort of value maximization model like the multi attribute utility theory, multiple regression, or Bayesian statistics in order to find an optimal solution [23][25][32]. All these approaches are computationally very expensive, but human decision taking is normally bounded by time restrictions, limited cognitive capacities, and limited willingness to accept cognitive effort. This is the reason why humans apply in many circumstances heuristic approaches (i.e. rules of thumb).

In contrast to rationality, this concept is called *bounded rationality* or *procedural rationality* [12][22][26][27]. Gerd Gigarenzer has shown in multiple experiments, that heuristic, bounded rational approaches can be as accurate as some fully rational concept like multiple regression [8][9]. Unfortunately, there are cases where bounded rationality acts as a door opener for systematic misjudgements which builds the grounding for decision phenomena/effects. Based on misjudgements due to bounded rationality, these decision effects bear the danger of suboptimal decision making [30].

Decoy effects [1][11][17][20][21][31] are one family of such effects which have the potential of severely impacting on the perceived value of goods and services. Basically, there exist three types of decoy effects: the attraction effect [21], the asymmetric dominance effect [11], and the compromise effect [24][31]. In existing literature, the expressions decoy effect, asymmetric dominance effect and attraction effect are often used synonymously, as the asymmetric dominance effect is the most prominent and stable decoy effect, and the attraction effect could be seen as the more general effect sharing the principle of tradeoff contrasts [28]. A clear distinction between the different effects, the corresponding decoy items, and the different mechanisms working behind the different decoy effects can be found in [29]. Since the 1980's a lot of research has been done in order to investigate decoy effects.

While the existence of such biases has been shown in quite a number of publications there has not been done much research in investigating the impacts of such decision biases in real world sales platforms with realistic environments and on the basis of real market data. This is out of two reasons: First, the investigation of some decision effect under clean room conditions makes it possible to eliminate a maximum of disturbing influences and therefore also maximizes and purifies the measured effect. Second, it is not easy to get good market data as companies are usually very reserved concerning the proliferation of business intelligence. Although the investigation of cognitive biases without real market conditions are indeed relevant from the basic research point of view, the practical relevance for real world applications cannot be assessed because a particular bias can be too small in relation to other overlaying (uncontrolled) effects such that the practical relevance for real world applications is possibly not given.

Closing this gap, this paper is in the line of research investigating decoy effects in realistic settings, as all studies are carried out unsupervised using a recommender like online system. The second study presented in this paper uses real market data (i.e. real capital savings books) taken from an independent consumer information site (www.konsument.at). Moreover, financial services constitutes a high-involvement decision domain, such that decoy effects should be less likely than in low-involvement domains where the user does not put too much energy into the decision process.

3. EXPERIMENT WITH SYNTHESIZED SETS OF FINANCIAL SERVICES

In order to investigate the influence of the Asymmetric Dominance- and Compromise Effects (ADE and CE) on product selection tasks in the financial services domain, a corresponding online user study was carried out. The experiment was two folded: Subjects (Students of the Alpen-Adria Universitaet Klagenfurt) had to accomplish one decision task for each effect (Asymmetric Dominance- and Compromise Effect). Altogether there were 535 valid sessions whereby 358 were from female persons. The subject's age ranged from 18 to 76 years (mean = 25.8, std = 7.2).

3.1 Compromise Effect

Design

The first decision task the subjects had to accomplish was to decide which type of financial service they would choose if they had 5000 Euros. Depending on the products the subjects had to choose from, three groups were differentiated: The *control* group with the product types *public_bonds*, *gold*, *mixed_funds*, group *Decoy_A* with the product types *bankbook* (=decoy), *public_bonds*, *gold*, *mixed_funds*, and group *Decoy_B* containing the product types *public_bonds*, *gold*, *mixed_funds*, *shares* (=decoy) (see Figure 3).

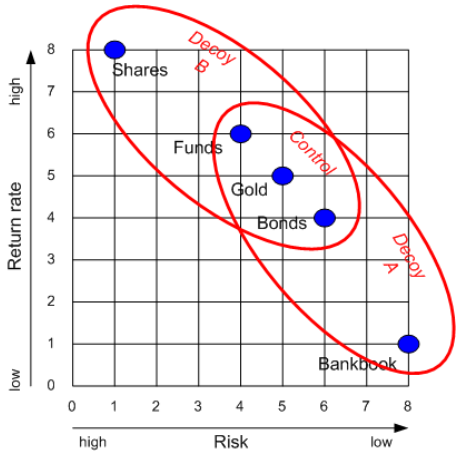


Figure 3: Product landscape of CE-task: three alternative items (funds, gold, bonds) with the corresponding decoy items (shares and bankbook, respectively).

The utility of each product was described in terms of risk and return rate (see Figure 3 and Table 3), whereby low risk and a high return rate was interpreted as good (i.e. high utility value).

As exact preference models were not given equal weighted

Product type	Return Rate	Risk
Bankbook	1	8
Public bonds	4	6
Gold	5	5
Mixed funds	6	4
Shares	8	1

Table 3: Product utilities in CE-task.

Multi attribute utility Theory (MAUT [32]) was used for designing suitable options. Although exact knowledge about user preferences (e.g. attribute weights) would be preferred also a linear equal weight model does the job as all hypotheses are tested on behalf of corresponding control groups revealing the actual preferences. The product types bonds, gold, and funds have the same overall utility (= 10) and therefore no tradeoff contrasts (TC) occur (see Table 3). The extreme options bankbook and shares have a little lower overall utility (= 9). Adding such options leads to TCs and therefore can cause compromise effects.

There were two hypotheses postulated:

- H1: Choice of Bonds is increased by the presence of Bankbook.
- H2: Choice of Funds is increased by the presence of Shares.

Results

Generally, users preferred low risk items over high return items. Comparing the choice distribution of the control group with group Decoy_A [H1], it can be said that more people chose bonds in the decoy group than in the control group (see Figure 4). In fact, the presence of bankbook made bonds the strongest option whereas in the control group gold was the most often chosen product type. The corresponding statistical analysis of bonds choices in the two groups showed a strong tendency (Fisher's Exact Test, one-sided: p <.079). Comparing the choice distribution of the control group and group Decoy_B [H2] the effect is even clearer. The increase of funds choices in presence of shares was highly significant (Fisher's Exact Test, one-sided: p <.001). It is notable that in all three groups the compromise options (i.e. the product groups in the middle) scored better than the extreme options.

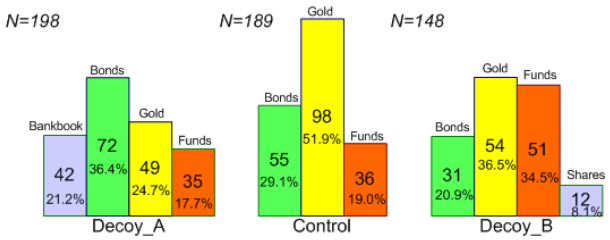


Figure 4: Choice distribution for the CE-task.

3.2 Asymmetric Dominance Effect

Design

In the second decision task the subjects also had to imagine they had 5000 Euros for investment. In this case they only could choose among various bankbooks. Depending on the products the subjects had to choose from, four groups were differentiated: The *control* group contained the products *bankbook1*, *bankbook2*, *bankbook3*. Group *Decoy_1* contained *bankbook1*, *bankbook2*, *bankbook3*, *decoy1*. Group *Decoy_2* contained *bankbook1*, *bankbook2*, *bankbook3*, *decoy2*, and group *Decoy_3* contained *bankbook1*, *bankbook2*, *bankbook3*, *decoy3* (see Figure 5). *decoyX* denotes an asymmetrically dominated decoy for *bankbookX*. When presenting the items to the user, the decoy items (decoy1, decoy2, decoy3) were called *bankbook4* (in order to avoid experimental side effects triggered by the item name).

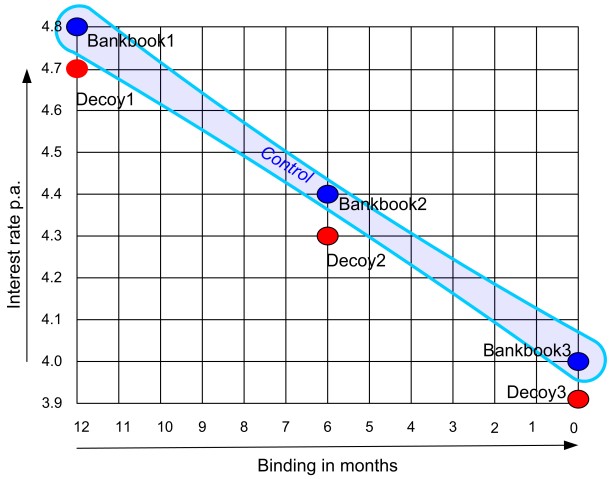


Figure 5: Product landscape of ADE-task: three alternative bankbooks with the corresponding decoy items.

The utility of each product was described in terms of interest rate per year (p.a.) and binding in months (i.e. the time within it is not possible to withdraw the money), whereby low binding and high interest rate was interpreted as good (i.e. high utility value). Figure 5 and Table 4 summarize the settings.

Product	Interest rate p.a.	Binding in months
Bankbook1	4.8	12
Bankbook2	4.4	6
Bankbook3	4.0	0
Decoy1	4.7	12
Decoy2	4.3	6
Decoy3	3.9	0

Table 4: Product attributes in ADE-task.

In the control group no tradeoff contrasts (TCs) where existent as the product with the highest interest rate had also the longest binding and vice versa. The decoy products (decoy1, decoy2, decoy3) constitute asymmetrically dominated alternatives (e.g. decoy1 is only dominated by bankbook1, etc). Additionally to the ADE-constellation there can also be found further TCs between the decoy and the non dominating bankbooks (i.e. compromise effects).

There were three hypotheses postulated:

- H3: Choice of Bankbook1 is increased by the presence of Decoy1.
- H4: Choice of Bankbook2 is increased by the presence of Decoy2.

- H5: Choice of Bankbook3 is increased by the presence of Decoy3.

Results

In this case users preferred high return rates over binding in years. Comparing the number of subjects choosing bankbook1 in the control group and in the group Decoy_1 one can remark a non-significant increase by 1.5% (Fisher's Exact Test, one-sided: p <.448, see Figure 6) [H3]. Comparing the choice distribution of the control group and group Decoy_2 the effect was significant. The increase of bankbook2 choices in presence of decoy2 made up 12.1% (Fisher's Exact Test, one-sided: p <.001) [H4]. The decoy3 in the group Decoy_3 increased the bankbook3 choices by 6.8% compared to the control group (Fisher's Exact Test, one-sided: p <.127) [H5].

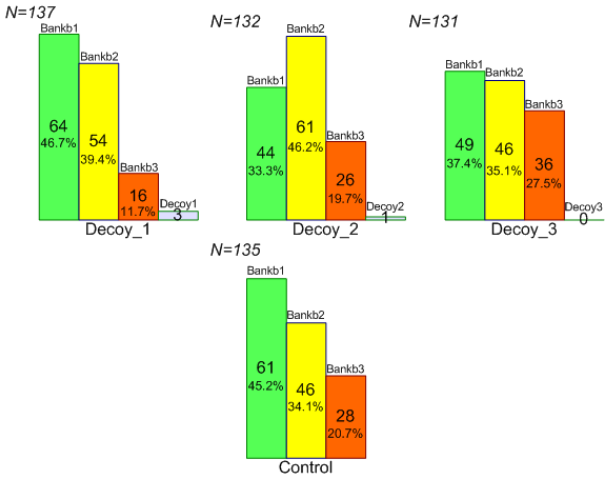


Figure 6: Choice distribution for the ADE-task.

4. EXPERIMENT WITH A REAL-WORLD SET OF FINANCIAL SERVICES

Design

The first step in order to come up with a realistic set of items was to find a suitable product domain. The domain of capital savings books was found to be perfect for our purposes because of the following reasons:

- Savings books can be well described by two dimensions, which is binding (i.e. the period in which it is not possible to withdraw the money) and interest rate (p.a.). This offers the possibility to stick to the simple two-dimensional item landscape.
- There is lots of comparable market data available.

The experimental items for the different choice sets were chosen on the basis of a products list given by Konsument.at, a well-known independent consumer information site.[1]. Konsument.at listed capital savings books having a binding period between one and five years. The products of the two- and four year categories having the highest interest rates of that category were chosen as competing items A and B.

[1]Please note that the experiment was already carried out in 2009, such that the market data was up to date at this time.

Additionally, two asymmetrical dominated decoy items dA (decoy for A) and dB (decoy for B) were defined by choosing the items with the second best interest rates of the two and four year categories. The extreme products with a binding period of one and five years showing the highest interest rates in the respective categories were constituting the corresponding compromise decoys cA (decoy for A) and cB (decoy for B). Table 5 and Figure 7 are showing the resulting product landscape of the experimental items. It has to be noted that the design is not completely symmetric as the dominated items (dA and dB) are always inferior in the binding dimension, such that dA constitutes a d3-decoy (see Figure 1) whereas dB constitutes a d2-decoy.

Item	A	B	dA	dB	cA	cB
Interest rate p.a. (%)	3,00	3,77	2,75	3,60	2,25	4,00
Binding (years)	2	4	2	4	1	5
Bank	Deniz	Auto	Erste	Direkt	Direkt	Direkt

Table 5: Attribute values of the experimental items used in the different experimental groups.

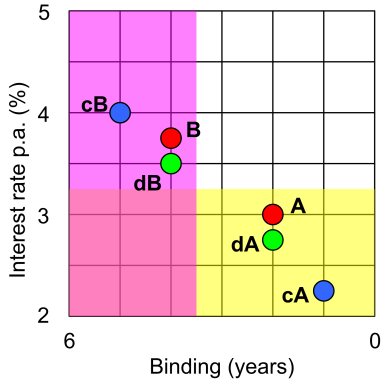


Figure 7: Product landscape: two alternative items with the corresponding decoy items.

Grounding on the experimental super set in Table 5 and Figure 7, experimental sets were defined and categorised according to the decoy added to the core setting (i.e. only the competing items A and B). The control (Control) set is consisting of only the competing items A and B. In the decoy sets one out of four possible decoys (dA, dB, cA, cB) was added, which should evoke the asymmetric dominance- or compromise effect (ADE or CE) for the benefit of A or B, respectively.

SetId	Item 1	Item 2	Item 3	Decoy Type
0	A	B		Control
1	A	B	dA	ADE - pro A
2	A	B	dB	ADE - pro B
3	A	B	cA	CE - pro A
4	A	B	cB	CE - pro B

Table 6: Experimental item sets and type of decoy.

The experiment was designed to be carried out online and unsupervised. Subjects (students of University of Klagenfurt) were invited by email containing a link to the online

experiment to take part in the experiment. Figure 8 is showing a screenshot of one experimental situation. Subjects were asked to imagine to have 10000 Euros for investment and to choose their favorite option out of a set of proposed items (i.e. the capital savings books). The subjects were assigned randomly to one of the defined settings. Furthermore, the position of the presented items was random.

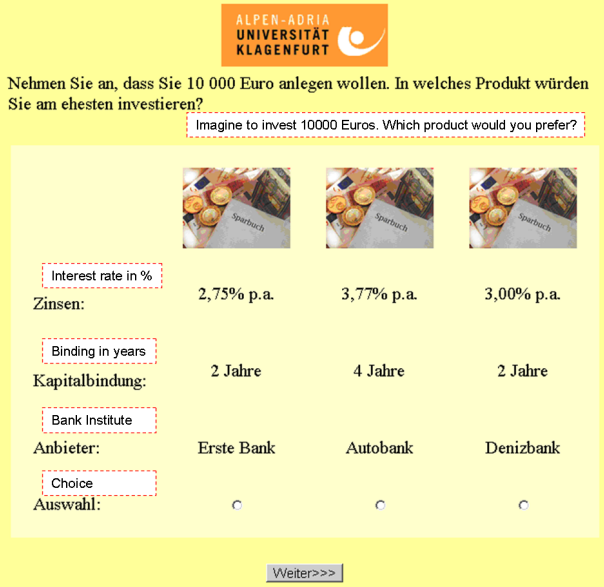


Figure 8: Screenshot of the online experiment. The translations have been added post hoc.

Following the current theory, the control set should offer the most objective view on the competing items A and B as there are no decoy effects, and thus should build the baseline. With respect to this baseline, the following hypotheses were formulated:

- H1: In setting 1, the asymmetric dominated decoy shifts attraction for the benefit of A (damages B).
- H2: In setting 2, the asymmetric dominated decoy shifts attraction for the benefit of B (damages A).
- H3: In setting 3, the compromise decoy shifts attraction for the benefit of A (damages B).
- H4: In setting 4, the compromise decoy shifts attraction for the benefit of B (damages A).

Results

Table 7 shows the experimental outcome for all five settings. It becomes obvious that only in group 2 the decoy was able to lift the number of target choices. In the other groups it seems that the choices of decoys were too many such that the absolute number of choices of both, A and B, were decreased. For the groups 3 and 4, this is not surprising, as non-dominated decoys (like a CE decoy) do not represent inferior options. The reason why the decoy in group 1 was chosen unexpectedly often must be the bank name. Whereas all other decoys were products from 'Denizbank', the decoy in group 1 was a product of 'Erste Bank', which obviously is a bank with better reputation.

In order to carve out the asymmetric influence of the decoy on A and B, Table 8 lists only the choices of A or B, neglecting the decoy choices. Now it is revealed that except in group 3, where the relation between A and B kept almost the same (i.e. H3 is not supported), the decoy pulled away more choices from the competitor than from the target, i.e. damaged the attraction of the target less than the attraction of the competitor (i.e. H1, H2, H4 are supported). Hence, the decoys rather caused an asymmetric detraction rather than an asymmetric attraction. The reason, why there could not be revealed a compromise effect in group 3, is most probably the distance between the decoy and the target (see Figure 7). The distance (i.e. cumulated attribute differences) plays a significant role for the strength of the decoy, such that the bigger difference is the less is the asymmetric influence of an intended decoy.

Although the absolute choices of a target product are not imperatively raised by a decoy item, there are nevertheless two possibilities how bank institutes could benefit from decoy effects. First, it is possible to shift the attraction within a bank's product assortments, as the bank's reputation (i.e. name) cannot have any influence (i.e. it is the same for all products). For example, it would be possible to decrease the attraction of products which show low marginal return (i.e. competitor) or to increase the attraction of products which show high marginal return. In this case, because of the possibly many decoy choices, it would be crucial that the decoy also shows a high marginal return rate in order to improve the overall result.

The second possibility for exploitation addresses the possibility for a bank itself being the target. In this case it is more convenient to think about products as parts of the bank's product portfolio. When considering portfolios, the introduction of a decoy product could significantly take away choices of the competitor banks portfolio for the sake of the target bank's portfolio. Thereby it does not matter which of the products in the portfolio benefits.

SetId	Decoy Type	A	B	Decoy	Total
0	Control	31	16		47
		66.0%	34.0%		100.0%
1	ADE - pro A	31	9	9	49
		63.3%	18.4%	18.4%	100.0%
2	ADE - pro B	24	24	2	50
		48.0%	48.0%	4.0%	100.0%
3	CE - pro A	25	13	9	47
		53.2%	27.7%	19.1%	100.0%
4	CE - pro B	20	16	18	54
		37.0%	29.6%	33.3%	100.0%

Table 7: Results of the experiment.

5. RELEVANCE FOR E-SALES SYSTEMS

In principle, decoy effects occur in any system where competing choice options are presented concurrently. Obviously, this is the case for many e-sales systems like shop applications, recommender- and configurations systems, or many other online decision support systems. Although, depending on the application, there are various situations during the user sessions where cognitive biases like decoy effects can play an important role, the most important phase for decoy effects constitutes the product presentation phase. During this phase purchase offers (in shopping systems) or recom-

SetId	Decoy Type	A	B	Total
0	Control	31	16	47
		66.0%	**34.0**%	100.0%
1	ADE - pro A	31	9	40
		77.5%	**22.5**%	100.0%
2	ADE - pro B	24	24	48
		50.0%	**50.0**%	100.0%
3	CE - pro A	25	13	38
		65.8%	**34.2**%	100.0%
4	CE - pro B	20	16	36
		55.6%	**44.4**%	100.0%

Table 8: Results of the experiment, leaving out decoy choices.

mended items (in recommender systems) are typically presented concurrently and the user (consumer) finds himself in some sort of decision dilemma. Here, decoy effects can manifest in suboptimal decision making as decoy effects bias the perceived utility of the concurring options. This may further result in product purchases which are not optimal for the consumer, the vendor, or both.

In the case of dialog-based systems (i.e. systems which gather user information by posing questions and proposing possible answers) decoy effects can also influence the answers given by the users during the dialog. This can influence the accuracy and furthermore the time-efficiency of such systems. For case case-based systems like tweaking-critiquing recommenders with multiple items to be criticized concurrently [5] this is obvious as any cycle basically constitutes a new product presentation phase.

Another aspect which is somewhat orthogonal to the biasing of decisions is the fact, that decoy effects can have a positive effect on the decision confidence [30]. This means that decoys manage to seemingly alleviate a decision situation such that users feel more confident about their decisions. Altogether, the above mentioned aspects offer a big potential for e-sales systems for optimizing the decision making process and also the quality of the taken decisions.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have presented the results of a series of empirical studies that clearly show the impact of different types of decoy effects on the item selection behavior of a user in the context of financial services decision making. The existence of decoy effects has been shown for non-classical scenarios with more than three items in the result set in order to show the existence of decoy effects for real world scenarios. Therefore, we analyzed the existence of decoy effects on the basis of the bankbook dataset provided by the Austrian consumer advisory platform *konsument.at*. The results of our studies have a significant impact on the design of future e-sales systems since it is obvious that item selection behavior is not based on a complete analysis of the set of offered or recommended items. Item selection is often subject to the application of a set of simple heuristics which is the reason for the observed decoy effects. Taking into account these heuristics, and thus better understanding human decision taking, can have positive effects in terms of a higher confidence in the set of presented items. Moreover, controlling such effects also offers the possibility of increasing the probability of selection of certain items.

Apart from the ongoing investigation of diverse decision biases in the context of e-sales systems, a main focus of our future work is the implementation of a framework which allows to identify and control decision biases. In particular, we are working on a decoy filter for recommender systems which is able to identify biased item sets and calculates a set of items to be removed or added in order to objectify the decisions. Specifically in the context of recommender systems this could lead to a big improvement in terms of recommendation accuracy and user trust.

Acknowledgement

The work presented in the paper has been conducted within the scope of the XPLAIN-IT project which is financed by the Privatstiftung Kaerntner Sparkasse.

7. REFERENCES

[1] D. Ariely, T. Wallsten, Seeking subjective dominance in multidimensional space: An exploration of the asymmetric dominance effect, Organizational Behaviour and Human Decision Processes, 63(3), 223-232, 1995.

[2] R. Burke, Knowledge-based recommender systems. Encyclopedia of Library and Information Systems, 69(32):180-200, 2000.

[3] R. Burke, Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12(4):331-370, 2002.

[4] S. Callander and C. H.Wilson, Context-dependent Voting, Quarterly Journal of Political Science, 1: 227-254, 2006.

[5] L. Chen, P. Pu, Interaction design guidelines on critiquing-based recommender systems, User Modeling and User-Adapted Interaction 19(3): 167-206, 2009.

[6] A. Colman, B. Pulford, and F. Bolger. Asymmetric dominance and phantom decoy effects in games, Journal of Organizational Behavior and Human Decision Processes 104(2007): 193-206.

[7] A. Felfernig, G. Friedrich, and L. Schmidt-Thieme. Introduction to the IEEE Intelligent Systems Special Issue: Recommender Systems, 22(3):18-21, 2007

[8] Gigerenzer, G., Bounded rationality: Models of fast and frugal inference, Swiss Journal of Economics and Statistics, 133(2/2), 201-218, 1997.

[9] Gigerenzer, G., Todd, P., ABC Research Group, Simple heuristics that make us smart, New York/Oxford: Oxford University Press, ISBN: 9780195121568, 1999.

[10] 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.

[11] Huber, J., Payne, J. W., Puto, C., Adding Asymmetrically Dominated Alternatives: Violations of Regularity and the Similarity Hypothesis, The Journal of Consumer Research, Vol. 9(1), 90-98, 1982.

[12] Kahneman, D., Maps of bounded rationality: psychology for behavioral economics, The American Economic Review. 93(5), 1449-1475, 2003.

[13] J. Konstan, B. Miller, J. Herlocker, L. Gordon, and J. Riedl. GroupLens: Applying Collaborative Filtering to Usenet News, Communications of the ACM, 40(3): 77-87, 1997.

[14] F.Y. Kuoa, T.H. Chub, M.H. Hsuc, H.S. Hsieha. An investigation of effort-accuracy trade-off and the impact

of self-efficacy on Web searching behaviors, Decision Support Systems, 37: 331-342, Elsevier, 2004.

[15] R. J. Mooney, L. Roy, Content-based book recommending using learning for text categorization, 5th ACM Conference on Digital Libraries, ACM Press, 2000.

[16] M. Ouyang, Does the Decoy Effect Exist in the Marketplace? An Examination of the Compromise Effect, Congress 2004 de l'Association des Sciences Administrative du Canada, 2004.

[17] R. Paramesh, Independence of Irrelevant Alternatives, Econometrica, 41(5):987-991, 1973.

[18] J.W. Payne, J.R. Bettman, and E.J. Johnson. The Adaptive Decision Maker, Cambridge University Press, Cambridge, England (1993)

[19] M. Pazzani and D. Billsus. 1997. Learning and Revising User Profiles: The Identification of Interesting Web Sites, Machine Learning. (27), 313-331, (1997).

[20] J. Quesada, N. Chater, P. Otto., and C. Gonzalez., An explanation of decoy effects without assuming numerical attributes. 27th Annual Meeting of the Cognitive Science Society. Chicago Lawrence Erlbaum Associates, 2005.

[21] S. Ratneshwar, A. D. Shocker, D. W. Stewart, Toward understanding the attraction effect: the implications of product stimulus meaningfulness and familiarity, Journal of Consumer Research, 13:520-533, 1987.

[22] Rubinstein, A., Modeling Bounded Rationality, The MIT Press, Cambridge(Massachusetts)/London(England), ISBN-10: 0262681005, 1998.

[23] C. Schmitt, D. Dengler, M. Bauer. The MAUT Machine - an Adaptive Recommender System, ABIS, 2001.

[24] M. Schweizer, Kontrast- und Kompromisseffekt im Recht am Beispiel der lebenslaenglichen Verwahrung, Schweizerische Zeitschrift fur Strafrecht, 123(4):438-457, 2005.

[25] Simon, H. A., A behavioral model of rational choice, The quarterly Journal of Economics, Vol. 69, 99-118, 1955.

[26] Simon, H. A., Theories of Bounded Rationality, Decision and Organisation, Radner and Radner (Eds.), 1972.

[27] Simon, H. A., From Substantive to Procedural Rationality, Method and Appraisal in Economics, Latsis (Eds.), 1976.

[28] I. Simonson,A. Tversky, Choice in context: Tradeoff contrast and extremeness aversion, in: Journal of Marketing Research (39), 281-292, 1992.

[29] Teppan, E. C., Recommendation beyond rationality, Dissertation, University of Klagenfurt, 2010.

[30] Teppan, E., Felfernig, A., Impacts of Decoy Elements on Result Set Evaluation in Knowledge-Based Recommendation, International Journal of Advanced Intelligence Paradigms (IJAIP), Vol. 1(3), 358-373, 2009.

[31] B. Wernerfelt, A Rational Reconstruction of the Compromise Effect: Using Market Data to infer Utilities, Journal of Consumer Research, 21:627-33, 1995.

[32] D. Winterfeldt, W. Edwards, Decision Analysis and Behavioral Research, Cambridge University Press,Cambridge, England, 1986.

Affective recommender systems: the role of emotions in recommender systems

Marko Tkalčič
University of Ljubljana Faculty of electrical engineering
Tržaška 25, Ljubljana, Slovenia
marko.tkalcic@fe.uni-lj.si

Andrej Košir
University of Ljubljana Faculty of electrical engineering
Tržaška 25, Ljubljana, Slovenia
andrej.kosir@fe.uni-lj.si

Jurij Tasič
University of Ljubljana Faculty of electrical engineering
Tržaška 25, Ljubljana, Slovenia
jurij.tasic@fe.uni-lj.si

ABSTRACT

Recommender systems have traditionally relied on data-centric descriptors for content and user modeling. In recent years we have witnessed an increasing number of attempts to use emotions in different ways to improve the quality of recommender systems. In this paper we introduce a unifying framework that positions the research work, that has been done so far in a scattered manner, in a three stage model. We provide examples of research that cover various aspects of the detection of emotions and the inclusion of emotions into recommender systems.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

recommender systems, emotions

1. INTRODUCTION

In the pursuit of increasing the accuracy of recommender systems, researchers started to turn to more user-centric content descriptors in recent years. The advances made in affective computing, especially in automatic emotion detection techniques, paved the way for the exploitation of emotions and personality as descriptors that account for a larger part of variance in user preferences than the generic descriptors (e.g. genre) used so far.

However, these research efforts have been conducted independently, stretched among the two major research areas, *recommender systems* and *affective computing*. In this paper we (i) survey the research work that helps improving recommender systems with affective information and (ii) we provide a unifying framework that will allow the members of the research community to identify the position of their activities and to benefit from each other's work.

Decisions@RecSys 2011, Chicago, USA

2. THE UNIFYING FRAMEWORK

When using applications with recommender systems the user is constantly receiving various stimuli (e.g. visual, auditory etc.) that induce emotive states. These emotions influence, at least partially (according to the bounded rationality model [16]) the user's decisions on which content to choose. Thus it is important for the recommender system application to detect and make good use of emotive information.

2.1 Describing emotions

There are two main approaches to describe the emotive state of a user: (i) the *universal emotions model* and (ii) the *dimensional model*. The universal emotions model assumes there is a limited set of distinct emotional categories. There is no unanimity as to which are the universal emotions, however, the categories proposed by Ekman [10] (i.e. happiness, anger, sadness, fear, disgust and surprise) appear to be very popular. The dimensional model, on the contrary, describes each emotion as a point in a continuous multidimensional space where each dimension represents a quality of the emotion. The dimensions that are used most frequently are valence, arousal and dominance (thus the VAD acronym) although some authors refer to these dimensions with different names (e.g. pleasure instead of valence in [20] or activation instead of arousal in [13]). The circumplex model, proposed by Posner et al. [24], maps the basic emotions into the VAD space (as depicted in Fig. 1)

2.2 The role of emotions in the consumption chain

During the user interaction with a recommender system and the content consumption that follows, emotions play different roles in different stages of the process. We divided the user interaction process in three stages, based on the role that emotions play (as shown in Fig. 2): (i) *the entry stage*, (ii) *the consumption stage* and (iii) *the exit stage*.

The work surveyed in this paper can be divided in two main categories: (i) generic emotion detection algorithms (that can be used in all three stages) and (ii) usage of emotion parameters in the various stages. This paper does not aim at providing an overall survey of related work but rather to point out good examples of how to address various aspects of recommender systems with the usage of techniques

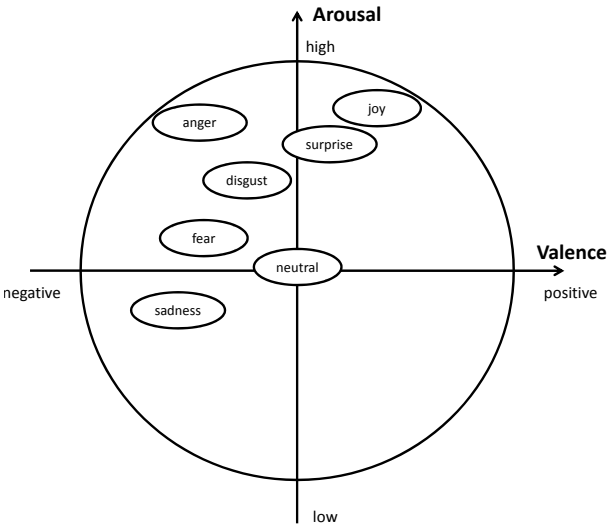


Figure 1: Basic emotions in the valence-arousal plane of the dimensional model

borrowed from affective computing.

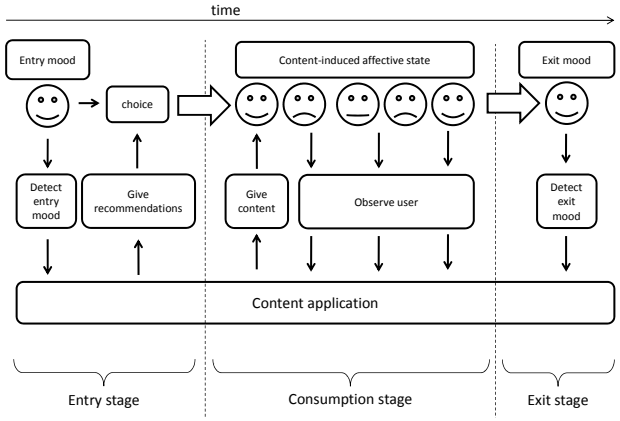


Figure 2: The unifying framework: the role of emotions in user interaction with a recommender system.

In the remainder of the paper we address each stage separately by surveying the existing research work and providing lists of open research areas. At the end we discuss the proposed framework and give the final conclusions.

3. DETECTING AFFECTIVE STATES

Affective states of end users (in any stage of the proposed interaction chain) can be detected in two ways: (i) explicitly or (ii) implicitly. The implicit detection of emotions is more accurate but it's an intrusive process that breaks the interaction. The implicit approach is less accurate but it's well suited for user interaction purposes since the user is not aware of it. Furthermore, Pantić et al. [22] argued that explicit acquisition of users' affect has further negative properties as users may have side-interests that drive their explicit affective labeling process (egoistic tagging, reputation-driven tagging or asocial tagging).

The most commonly used procedure for the explicit asessment of emotions is the Self Assessment Manikin (SAM) developed by [7]. It is a questionnaire where users assess their emotional state in the three dimensions: valence, arousal and dominance.

The implicit acquisition of emotions is usually done through a variety of modalities and sensors: video cameras, speech, EEG, ECG etc. These sensors measure various changes of the human body (e.g. facial changes, posture changes, changes in the skin conductance etc.) that are known to be related to specific emotions. For example, the Facial Action Coding System (FACS), proposed by Ekman [9], maps emotions to changes of facial characteristic poionts. There are excellent surveys on the topic of multimodal emotion detection: [31, 22, 14]. In general, raw data is acquired from one or more sensors during the user interaction. These signals are processed to extract some low level features (e.g. Gabor based features are popular in the processing of facial expression video signals). Then some kind of classification or regression technique is applied to yield distinct emotional classes or continuous values. The accuracy of emotion detection ranges from over 90% on posed datasets (like the Kanade-Cohn dataset [18]) to slightly better than coin tossing on spontaneous datasets (like the LDOS-PerAff-1 dataset [29]) [27, 6].

4. ENTRY STAGE

The first part of the proposed framework (see Fig. 2) is the entry stage. When a user starts to use a recommender system, she is in an affective state, the *entry mood*. The entry mood is caused by some previous user's activities, unknown to the system. When the recommender system suggests a limited amount of content items to the user, the entry mood influences the user's choice. In fact, the user's decision making process depends on two types of cognitive processes, the rational and the intuitive, the latter being strongly influenced by the emotive state of the user, as explained by the bounded rationality paradigm [16]. For example, a user might want to consume a different type of content when she is happy than when she is sad. In order to adapt the list of recommended items to the user's entry mood the system must be able to detect the mood and to use it in the content filtering algorithm as contextual information.

In the entry part of user-RS interaction one of the aspects where emotions can be exploited is to influence the user's choice. Creed [8] explored how the way we represent information influences the user's choices.

It has been observed by Porayska-Pomsta et al. [23] that in tutoring systems there is a strong relation between the entry mood and learning. They analysed the actions that a human tutor took when the student showed signs of specific affective states to improve the effectiveness of an interactive learning environment.

A user modeling approach that maps a touristic attraction with a piece of music that induces a related emotion has been developed by Kaminskas and Ricci [17]. Their goal was to find an appropriate musical score that would reinforce the affective state induced by the touristic attraction.

Using the entry mood as a contextual parameter (as described by Adomavicius and Tuzhilin in [2]) could improve the recommender's performance. Both Koren et al. [19] and Baltrunas et al. [5] suggest using the matrix factorization approach and enrich it with contextual parameters. At the context-aware recommender systems contest in 2010 [1] the goal was to select a number of movies that would fit the user's entry mood. The contest winners' contribution, Shi et al. [25] used several approaches amog which the best was the joint matrix factorization model with a mood-specific regularization.

As an extension to the usage of emotions as contextual information an interesting research area is to diversify the recommendations. For example, if a user is sad, would it be better to recommend happy content to cheer her up or to recommend sad content to be in line with the current mood? Research on information retrieval results diversification is getting increased attention, especially after the criticism of the recommendation bubble has started[2]. Although we are not aware of any work done on results diversification connected with emotions, a fair amount of work has been done on political news aggregators in order to stimulate political pluralism [21].

5. CONSUMPTION STAGE

The second part of the proposed framework is the consumption stage (see Fig. 2). After the user starts with the consumption of the content she experiences affective responses that are induced by the content. Depending on the type of content, these responses can be (i) single values (e.g. the emotive response to watching an image) or (ii) a vector of emotions that change over time (e.g. while watching a movie or a sequence of images). Figure 3 shows how emotions change over time in the consumption stage. The automatic detection of emotions can help building emotive profiles of users and content items that can be exploited for content-based recommender algorithms.

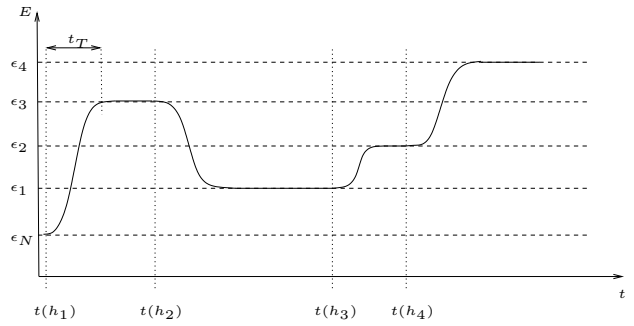


Figure 3: The user's emotional state ϵ is continuously changing as the time sequence of the visual stimuli $h_i \in H$ induce different emotions.

Using emotional responses for generating implicit affective tags for content is the main research area in the consumption section. Pantić et al. [22] argued why the usage of automatic emotion detection methods improves content tagging: the minimization of the drawbacks caused by egoistic tagging, reputation-driven tagging and asocial tagging. They also anticipate that implicit tagging can be used for user profiling in recommender systems.

Joho et al. [15] used emotion detection from facial expressions to provide an affective profile of video clips. They used an item profile structure that labels changes of users emotions through time relative to the video clip start. The authors used their approach for summarizing highlights of video clips.

Hanjalić et al. [11] approached the summarization of video highlights from the other side: they used the source's low level features (audio and video) to detect higlihjts without taking into account the responses of end users.

The research work described so far in this section is interesting because allows us to model the content items (images, movies, music etc.) with affective labels. These affective labels describe the emotions experienced by the users who consume the items. In our previous work [26] we have shown that the usage of such affective labels over generic labels (e.g. genre) significantly improves the performance of a content-based recommender system for images. We used explicitly acquired affective metadata to model the items and the users' preferences. However, in another experiment [28], where we used implicitly acquired affective metadata, the accuracy of the recommender system was significantly lower but still better than with generic metadata only.

In a similar experiment, Arapakis et al. [4] built a recommender system that uses real time emotion detection informaion.

6. EXIT STAGE

After the user has finished with the content consumption she is in what we call the exit mood. The main difference between the consumption stage and the exit stage is that the exit mood will influence the user's next actions, thus having an active part, while in the consumption stage the induced emotions did not influence any actions but were a passive response to the stimuli. In case that the user continues to use the recommender system the exit mood for the content just consumed is the entry mood for the next content to be consumed.

The automatic detection of the exit mood can be useful as an indicator of the user's satisfaction with the content. Thus the detection of the exit mood can be seen as an unobtrusive feedback collection technique.

Arapakis et al. [3] used the exit mood, detected through videos of users' facial expressions, as an implicit feedback in their recommender system for video sequences.

In an experiment with games, Yannakakis et al. [30], used heart rate activity to infer the "fun" that the subjects experience in physical interactive playgrounds.

7. OPEN RESEARCH AREAS

We identified four main areas where further research should be conducted in order to build true affective recommender systems: (i) using emotions as context in the entry stage, (ii) modeling affective content profiles, (iii) using affective profiles for recommending content and (iv) building a set of datasets.

Although some work has been carried out on exploiting the entry mood we believe that there is still the need to answer tha basic question of the entry stage: *which items to recommend when the user is in the emotive state A?*. We

[1] `http://www.dai-labor.de/camra2010/`
[2] `http://www.thefilterbubble.com/`

further believe that there are firm differences between what the user wants now and what is good for a user on a long run. Thus bringing the research on results diversification (see the work done in [21, 1]) into affective recommender systems is a highly important topic.

Affective content profiling is still an open question, especially profiling content items that last longer than a single emotive response. The time dependancy of content profiles has also o strong impact on the algorithms that exploit the profiles for recommending items.

With the except of the LDOS-PerAff-1 dataset [29] (which is limited in the amount of content items and users), the research community does not have a suitable dataset upon which to work. It is thus required that a large-scale dataset, compareable to the MovieLens or Netflix datasets, is built.

8. CONCLUSION

In this paper we have provided a framework that describes three ways in which emotions can be used to improve the quality of recommender systems. We also surveyed some work that deals with parts of the isuues that arise in te pursuit of affective recommender systems.

An important issue in recommender systems, especially when it comes to user-centric systems, is to move from data-centric assessment criteria to user-centred assessment criteria. We have not addressed this issue in this paper as it appears to larger dimensions. The recsys community has so far relied on metrics borrowed from information retrieval: confusion matrices, precision, recall etc. (see [12] for an overview). However recommender systems are used by end users and thus the assessment of the end users should be taken more into account. We suggest to move towards metrics that take into account the user experience as pointed out in `http://www.usabart.nl/portfolio/KnijnenburgWillemsen-UMUAI2011_UIRecSy.pdf`.

9. REFERENCES

[1] G. Adomavicius and Y. Kwon. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering*, (99):1–1, 2011.

[2] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS)*, 23(1):103–145, 2005.

[3] I. Arapakis, J. Jose, and P. Gray. Affective feedback: an investigation into the role of emotions in the information seeking process. *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, (January):395–402, 2008.

[4] I. Arapakis, Y. Moshfeghi, H. Joho, R. Ren, D. Hannah, and J. M. Jose. Enriching user profiling with affective features for the improvement of a multimodal recommender system. *Proceeding of the ACM International Conference on Image and Video Retrieval - CIVR '09*, (i):1, 2009.

[5] L. Baltrunas. Exploiting contextual information in recommender systems. *Proceedings of the 2008 ACM conference on Recommender systems - RecSys '08*, page 295, 2008.

[6] M. S. Bartlett, G. C. Littlewort, M. G. Frank, C. Lainscsek, I. R. Fasel, and J. R. Movellan. Automatic Recognition of Facial Actions in Spontaneous Expressions. *Journal of Multimedia*, 1(6):22–35, Sept. 2006.

[7] M. Bradley and P. Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.

[8] C. Creed and R. Beale. Using emotion simulation to influence user attitudes and behavior. *HCI workshop at BCS 2005*, pages 1–3, 2005.

[9] P. Ekman. Facial expression and emotion. *American Psychologist*, 48(4):384, 1993.

[10] P. Ekman. Basic Emotions. In *Handbook of Cognition and Emotion*, pages 45—-60. 1999.

[11] A. Hanjalic. Adaptive extraction of highlights from a sport video based on excitement modeling. *IEEE Transactions on Multimedia*, 7(6):1114–1122, Dec. 2005.

[12] J. L. Herlocker, J. A. Konstan, L. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst*, 22(1):5–53, 2004.

[13] S. V. Ioannou, A. T. Raouzaiou, V. a. Tzouvaras, T. P. Mailis, K. C. Karpouzis, and S. D. Kollias. Emotion recognition through facial expression analysis based on a neurofuzzy network. *Neural networks : the official journal of the International Neural Network Society*, 18(4):423–35, May 2005.

[14] A. Jaimes and N. Sebe. Multimodal human–computer interaction: A survey. *Computer Vision and Image Understanding*, 108(1-2):116–134, 2007.

[15] H. Joho, J. M. Jose, R. Valenti, and N. Sebe. Exploiting facial expressions for affective video summarisation. *Proceeding of the ACM International Conference on Image and Video Retrieval - CIVR '09*, page 1, 2009.

[16] D. Kahneman. A perspective on judgment and choice: mapping bounded rationality. *The American psychologist*, 58(9):697–720, Sept. 2003.

[17] M. Kaminskas and F. Ricci. Location-Adapted Music Recommendation Using Tags. *User Modeling, Adaption and Personalization*, pages 183–194, 2011.

[18] T. Kanade, J. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 46–53. IEEE, 2000.

[19] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89, Apr. 2010.

[20] A. Mehrabian. Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in Temperament. *Current Psychology*, 14(4):261–292, Dec. 1996.

[21] S. Munson, D. X. Zhou, and P. Resnick. Sidelines: An algorithm for increasing diversity in news and opinion aggregators. *Proceedings of ICWSM09 Conference on Weblogs and Social Media. San Jose, CA.*, 2009.

[22] M. Pantic and A. Vinciarelli. Implicit human-centered tagging [Social Sciences. *IEEE Signal Processing*

Magazine, 26(6):173–180, Nov. 2009.

[23] K. Porayska-Pomsta, M. Mavrikis, and H. Pain. Diagnosing and acting on student affect: the tutor's perspective. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, 18(1-2):125–173, 2007.

[24] J. Posner, J. a. Russell, and B. S. Peterson. The circumplex model of affect: an integrative approach to affective neuroscience, cognitive development, and psychopathology. *Development and psychopathology*, 17(3):715–34, Jan. 2005.

[25] Y. Shi, M. Larson, and A. Hanjalic. Mining mood-specific movie similarity with matrix factorization for context-aware recommendation. *Proceedings of the Workshop on Context-Aware Movie Recommendation*, pages 34–40, 2010.

[26] M. Tkalčič, U. Burnik, and A. Košir. Using affective parameters in a content-based recommender system for images. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, pages 1–33–33, Sept. 2010.

[27] M. Tkalčič, A. Odić, A. Košir, and J. Tasič. Comparison of an Emotion Detection Technique on Posed and Spontaneous Datasets. *Proceedings of the 19th ERK conference, Portorož, 2010*, 2010.

[28] M. Tkalčič, A. Odić, A. Košir, and J. Tasič. Impact of Implicit and Explicit Affective Labeling on a Recommender System's Performance. *Joint Proceedings of the Workshop on Decision Making and Recommendation Acceptance Issues in Recommender Systems (DEMRA 2011) and the 2nd Workshop on User Models for Motivational Systems: The affective and the rational routes to persuasion (UMMS 2011)*, page 112, 2011.

[29] M. Tkalčič, J. Tasič, and A. Košir. The LDOS-PerAff-1 Corpus of Face Video Clips with Affective and Personality Metadata. *Proceedings of Multimodal Corpora: Advances in Capturing, Coding and Analyzing Multimodality (Malta, 2010), LREC*, page 111, 2009.

[30] G. N. Yannakakis, J. Hallam, and H. H. Lund. Entertainment capture through heart rate activity in physical interactive playgrounds. *User Modeling and User-Adapted Interaction*, 18(1-2):207–243, Sept. 2008.

[31] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang. A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions. *IEEE Trans. Pattern Analysis & Machine Intelligence, Vol. 31, No.*, 1pp:39–58, 2009.

Using latent features diversification to reduce choice difficulty in recommendation lists

Martijn C. Willemsen
Eindhoven University of Technology
Human-Technology Interaction group
IPO 0.17, P.O. Box 513
5600 MB Eindhoven, Netherlands
M.C.Willemsen@tue.nl

Bart P. Knijnenburg
University of California, Irvine
Department of Informatics
DBH 5091
Irvine, CA 92697
bart.k@uci.edu

Mark P. Graus
Eindhoven University of Technology
Human-Technology Interaction group
IPO 0.20, P.O. Box 513
5600 MB Eindhoven, Netherlands
M.P.Graus@tue.nl

Linda C.M. Velter-Bremmers
Eindhoven University of Technology
Human-Technology Interaction group
L.C.M.Bremmers@student.tue.nl

Kai Fu
Eindhoven University of Technology
Human-Technology Interaction group
K.Fu@student.tue.nl

ABSTRACT
An important side effect of using recommender systems is a phenomenon called "choice overload"; the negative feeling incurred by the increased difficulty to choose from large sets of high quality recommendations. Choice overload has traditionally been related to the size of the item set, but recent work suggests that the diversity of the item set is an important moderator. Using the latent features of a matrix factorization algorithm, we were able to manipulate the diversity of the items, while controlling the overall attractiveness of the list of recommendations. In a user study, participants evaluated personalized item lists (varying in level of diversity) on perceived diversity and attractiveness, and their experienced choice difficulty and tradeoff difficulty. The results suggest that diversifying the recommendations might be an effective way to reduce choice overload, as perceived diversity and attractiveness increase with item set diversity, subsequently resulting in participants experiencing less tradeoff difficulty and choice difficulty.

Categories and Subject Descriptors
H.1.2 [Models and principles]: User/Machine Systems software psychology; H.4.2 [Information Systems Applications]: Types of Systems-decision support; H.5.2 [Information Interfaces and Presentation]: User Interfaces-evaluation / methodology, interaction styles, user centered design

General Terms
Algorithms, Experimentation, Human Factors, Theory

Keywords
Choice Overload, Diversification, User-centric evaluation

1. INTRODUCTION
Recommender systems support users in content discovery and exploration by providing items that match their personal preferences. Based on preference information (e.g. the user's ratings of known items, or past purchases) a recommender system predicts which items the user will like. The output of a typical recommender system is a ranked list of items with the highest predicted ratings. Provided that these predictions are accurate, the personalized recommendations are highly attractive. There is however a downside to attractive recommendations: psychological research on choice overload suggests that choosing an item from such a set might be a difficult task. Previous research [2] has shown that longer lists of attractive personalized recommendations can result in choice difficulty and subsequently in choice overload. In this paper we will investigate to what extend the difficulty of choosing from the list is related to the diversity of the items, while keeping the overall attractiveness of the set constant. Our results will show that users like item sets that are diversified and experience less choice difficulty in these sets, which suggests that diversification might be effective in reducing choice overload.

2. RELATED WORK
2.1 Choice overload
Choice overload [8, 16] refers to the difficulty decision makers experience when choosing from a large set of good alternatives. Choice overload has originally been related to the *size of the item set*, which creates two opposing effects. On one hand, larger sets are more attractive as they potentially provide more benefits for the decision maker, but at the same time larger sets have increased opportunity costs such as comparison costs, potential regret of not choosing the best option, and increased expectations which might not be met by the large set [16, 19].

As these opportunity costs tend to increase faster than the benefits associated with larger sets, decision makers usually find it easier to choose from a smaller set, and are often more satisfied with their actual choice when choosing from a smaller set. For example, in the original study by Iyengar and Lepper [8] visitors of a supermarket where more attracted towards a tasting booth that displayed 24 types of jam, rather than 6 types. However, only 3% of people who visited the booth with the large set of items bought jam, whereas 30% of the visitors of the small set booth bought jam (and reported to be more satisfied with their purchase). Other researchers have shown similar effects for other consumer products such as gift boxes [15] and coffee [13].

Choice overload effects can also occur in item lists generated by recommender systems. Bollen et al. [2] performed a user study with a matrix factorization movie recommender. They used three conditions: a small top-5 list, a large high quality top-20 list and a large lower quality 20 item list, composed of the top-5 plus 15 items with lower-ranked movies. Users experienced significantly more choice difficulty when presented with the high quality top-20 item list, compared to the other two lists. This increased

difficulty counteracted the increased attractiveness of the larger set, showing that in the end, choice satisfaction in all three list conditions was about the same. In other words, although users found the long high quality list more attractive, they engaged in increased effort when evaluating its items, compared to the other two lists. Behavioral data corroborated these findings.

2.2 Factors underlying choice overload

Within the psychological literature there is a strong debate as to how omnipresent the choice overload phenomenon is. A recent meta-analysis by Scheibehenne et al. [16] across 50 studies shows that the overall effect size is zero, showing that in some cases longer items lists are detrimental and in some cases beneficial. Several preconditions are identified as potential causes for choice overload. A larger choice set might only result in decreased satisfaction and increased choice deferral when there are no strong prior preferences or dominant options in the item set [3, 13]. Indeed, Scheibehenne et al. show in their meta-analysis that expertise and prior preferences reduce choice overload, showing that it is most likely to occur for sets in which there is little variability in the attractiveness of the individual items and in which no specific items stand out (i.e., the items all fit the preferences of the decision maker equally well). In the psychological and marketing literature, the choice overload effect is mostly studied using item sets that are not personalized and therefore contain a wide variety of different items that do not necessarily lie in a person's field of interest. Recommender systems, on the other hand, provide lists that are optimized for the decision-maker. For such sets the preconditions for choice overload are easily met, as recommendation lists contain many, highly attractive items, none of which are clearly dominating. This suggests (in line with Bollen et al. [2]) that choice overload might be an important (and undesirable) side effect of personalized recommender systems.

Scheibehenne et al. [17] suggest that there are several moderators not included in their meta-analysis that might influence the level of choice overload and that require more research. We will focus predominantly on moderators related to the composition and perception of the item set, as these are moderators that can be investigated effectively by manipulating the output of a recommender system.

Two important moderators related to the composition of the item set are recognized by Scheibehenne et al. [17]: the categorization of the list and the diversity of the list. The first moderator suggests that if items are categorized, cognitive effort is reduced, resulting in less choice overload and higher satisfaction [13]. Categorization of recommendations in a recommender system has been shown to increase user satisfaction, effectiveness and confidence [7]. Though it has not yet been established if categorization can also reduce choice overload in recommender systems, we will focus on the second moderator of choice overload that has been part of ongoing research in both recommender systems and psychology: item set diversity.

2.3 The role of item set diversity

Scheibehenne et al. [16] indicate that item set diversity is another important moderator of choice overload. They argue that until now diversity has not been precisely controlled in studies of choice overload, and that the lack of control over this variable might be one reason why studies on choice overload show such volatile results. The benefit of using recommender algorithms is that these do allow us to control item attractiveness and item set diversity at the same time, offering a precise control of the composition of the item set. But to understand better the relation between diversity and choice difficulty, we will first discuss some of the existing literature that has investigated the differences in choice difficulty between uniform and diverse item sets.

Reutskaja et al. [15] found that the more uniform an item set gets, the more difficult it becomes to make a choice. Fasolo et al. [4] studied real world assortments and showed that as the number of items in a set increase, the differences between the options decrease. Specifically, differences between the relevant attribute values become smaller and the density of the set increases. Density is the distance, measured one product attribute at a time, between one product and its closest neighbor (e.g., the inter product distance is larger for the attribute „duration" when the movies in the list are 60, 120 and 180 minutes long, than for a list of movies that are 60, 80 and 100 minutes long). As density increases the differences between products decrease (i.e. the items in a high density set are more uniform, whereas items in a low density set are more diverse).

Fasolo et al. [4] argue that in uniform sets, with attribute levels close to one another, it is hard to decide which option is better. Users of typical recommender systems, where item sets contain many items that are highly similar, might therefore experience a large amount of *choice difficulty*, as they may find it difficult to justify one decision over another. People prefer to make decisions they can easily justify [22], especially when item sets become larger [21]. Therefore, people may prefer diversified item sets over uniform item sets, as items from diversified lists provide clear reasons to be chosen.

However, there is another side of diversified sets, which has received little discussion in the literature concerning choice overload. As options become more diverse, they might encompass difficult *tradeoffs* that generate conflicts that require a lot of effort to resolve, as one always needs to sacrifice something when choosing one item over another. Scholten and Sherman [18] propose a double mediation model, which shows a U-shaped relation between the size of the tradeoffs in a set and the amount of conflict that is generated by the set. They suggest that both very uniform and very diverse sets might be difficult to choose from, compared to sets of average diversity. Choosing from a uniform choice set, compared to a diversified choice set, is harder because one lacks compelling reasons to pick any option. However, the tradeoffs one has to make are smaller, which makes it easier to choose because no great sacrifices need to be incurred. Conversely, making a choice from a diverse choice set, compared to a uniform choice set, is easier because better arguments can be made for a certain option. However, the tradeoffs are larger, which makes it more difficult to make the decision, because greater sacrifices need to be incurred.

Given the fact that recommender systems are prone to induce choice overload and that researchers in psychology do not yet agree whether increased diversity leads to more or less difficulty, there is a need to study the effect of item set diversity on choice difficulty and tradeoff difficulty in greater detail. In the present study, we employ a diversification algorithm that allows us to tightly control the diversity of the recommended item set, thereby allowing us to investigate how different levels of diversification influence the user"s perception and experiences (in terms of tradeoff difficulty and choice difficulty) of the item set. For this purpose, we chose a recommender using a matrix factorization algorithm, as the latent features used by these algorithms provide ideal means to control diversity and tradeoffs.

2.4 Matrix factorization and tradeoffs

Matrix factorization algorithms try to express movies and users in terms of vectors on a set of latent features. Features are extracted in such a way that the relative positions of an item-user pair can be used to predict the rating for the user on that item as accurately as possible. The recommendations provided are those items with highest predicted rating for the user.

Koren, Bell and Volinsky [11] claim that the vectors in matrix factorization model "measure the extent to which the item possesses those [features], positive or negative", and that these features relate to real-world concepts, for example „Geared towards males/females" and „Escapist/Serious". This is similar to how choice sets are described in Multi-attribute utility theory (MAUT) used in the psychology of judgment and decision making [1]. MAUT describes choice options on a set of common dimensions (attributes; e.g., hard disk space, processor speed or battery life for a notebook), and assumes that one's preferences can be described by a series of weights that denote the relative importance of these attributes to a decision maker.

Given the similarity between attributes and the latent features in matrix factorization, the latent feature space could be used to vary the diversity of recommendation lists. The latent feature space allows us to select subsets of recommendations with a specific density (i.e. a specific distribution across the latent features), while keeping its overall attractiveness (in terms of predicted ratings) constant. Arguably, this diversity manipulation directly affects underlying psychological concepts responsible for choice difficulty and tradeoff difficulty and therefore is more effective in helping us understand choice overload than other diversity manipulations that are often based on external information. For example, Ziegler et al. [23] investigated that diversification by using a subset of the Top-50 recommendations and diversifying this subset in terms of Intra-List Similarity based on a separate, external ontology.

3. EXPERIMENT

3.1 Goal and hypotheses

The present study aims to investigate how diversification of a list of recommended items affects the perceived diversity and perceived attractiveness of the list, and how these factors subsequently affect tradeoff difficulty and choice difficulty. Using a diversification algorithm on the latent features of a matrix factorization algorithm, we vary the density of the list of recommendations while keeping the overall attractiveness of the list (in terms of the predicted ratings) constant. By using three levels of diversification we investigate whether the relation between diversity and difficulty is linear (higher diversity always simplifies choosing as it is easier to find reasons to choose one over another, the predominant view in the literature on choice overload) or U shaped (diversity only helps to a certain level, but a high diversity might result in large tradeoffs between items that are effortful to resolve and that might for some people result in too high sacrifices, as suggested by Scholten and Sherman [18]). To accurately measure the role of diversity, we employ a within-subject design in which each participant is presented (sequentially) with a low, medium and high diversity list. To prevent possible order effects, the order of these lists is randomized over participants.

Between subjects we also vary the number of items in the list on 5 levels (5, 10, 15, 20 or 25), as the literature suggests that diversification might have a stronger impact for larger lists. However, given that recommenders output personalized and highly attractive sets of items, we might find that even for short lists, diversification has a strong impact on experienced difficulty.

To measure the subjective perceptions and experiences of these recommendations after the presentation of each list, we employ the user-centric framework for user experience of recommender systems as described in Knijnenburg et al [10]. Based on this framework we expect that the effect of diversification of the recommender output on subjective experience with the list (how difficult is it to make tradeoffs and choose from the list) is mediated by subjective perceptions of the diversity and attractiveness of the list. In particular, we expect that item lists that are more diverse (i.e., have a lower density on the attributes) are perceived as more varied and potentially also as more attractive, and that these two factors affect the experience of tradeoff difficulty and choice difficulty.

3.2 Manipulating diversity in a matrix factorization model

Our study uses a 10-dimensional prediction model. The goal of our diversification algorithm is to generate three lists of movies that are about equally attractive, but that differ in how much they vary on the latent features of the algorithm. Our diversification is performed on the personalized top-200 of the ranked output of the recommender algorithm. In a prior study this number was established to allow for a large enough range in potential diversification while at the same time not differing too much in attractiveness: the difference between the highest and lowest predicted ratings in a typical top-200 list for this prediction model is 0.76 on a five-point scale, which is not much higher than the mean absolute error in the predictions of the prediction model we used.

For every participant three different sets of N movies were extracted from these top 200 recommendations. The low diversification set consisted of the N movies closest to the centroid of the top 200 recommendations (see Figure 1). For the high diversification, a greedy algorithm was used to select the set of N movies with the highest inter-item distances (using city block distance). The algorithm started with the movie closest to the centroid. The distance to each of the remaining movies was calculated and the one with the highest distance was added to the recommendation list. For the next steps, the distances between all items in the recommendation list and all remaining items were

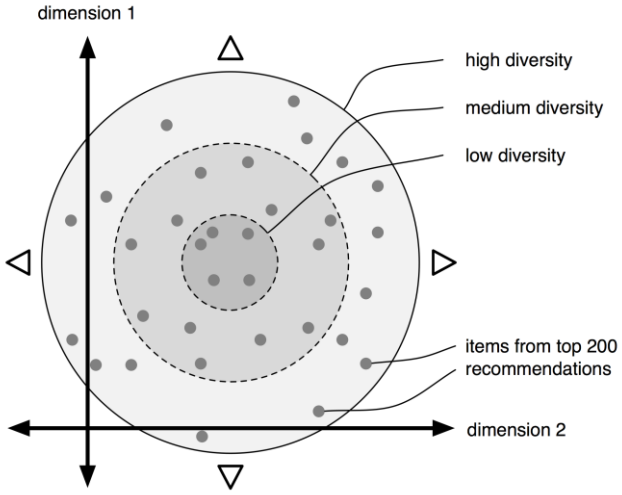


Figure 1: Schematic representation (simplified in two dimensions) of our item diversification in the top 200 set

calculated and the item with the maximum distance was added to the recommendation list, until the required number of recommendations was reached. Initial analyses on the used data set suggested that a boundary around 50% would result in movies with a medium density, so to derive a set with medium diversification level, the same greedy algorithm was used, but it was restricted to the 100 movies closest to the centroid instead of using the entire set.

3.3 System

For the study a movie recommender was developed based on a web-interface used previously in the MyMedia project[1], using a Matrix Factorization algorithm for the calculation of the recommendations. The dataset used for the experiment was the 10M MovieLens dataset[2]. In order to maximize the probability that users knew some of the movies presented in the initial rating task, movies from before 1994 and their corresponding ratings were removed from the dataset, resulting in a set of 5.6 million ratings by 69820 users on 5402 movies. We further enriched the MovieLens dataset with a short synopsis, cast, director and a thumbnail image of the movie cover taken from the Internet Movie Database. The Matrix Factorization algorithm used 10 latent features, a maximum iteration count of 100, a regularization constant of 0.0001 and a learning rate of 0.01. Using a 5-fold cross validation on the used dataset, this specific combination of data and algorithm resulted in an RMSE of 0.854 and an MAE of 0.656, which is up to standards. An overview of metrics is given by [5].

3.4 Design and procedure

The study consisted of three parts. In the first part, participants answered a set of questions to measure a number of individual characteristics. In their meta-analysis, Scheibehenne et al. [16] show that the characteristics expertise and prior preference are important moderators of choice overload. Therefore, we constructed a set of items to measure movie expertise and strength of preferences. We also measured maximizing tendency of our participants, using the short 6-item version [14] of the maximization questionnaire by Schwarz [20]. Schwarz defines people who always try to make the best possible choice as maximizers, and people who aim for "good enough" as satisficers. Maximizers consider more options whereas satisficers stop looking when they have found an item that meets their standards. Therefore the search costs of maximizers are higher and consequently it is suggested that they are more prone to choice overload.

After these questions, the second part of the study was used to gather rating information from the participant to be able to calculate and provide personalized recommendations. In this phase the participants were asked to rate a total of ten movies. They were presented with ten randomly selected movies at a time, with the instruction to rate only the movies they were familiar with (ratings were entered on a scale from 1 to 5 stars). After inspecting and (possibly) rating some of the ten movies shown, users could get a new list of movies by pressing a button. When the participant had entered ten or more ratings in total, they would be guided to the third part.

In the third part the participant sequentially received three times a list with recommendations, each time with a different level of diversification (the order of diversification levels was randomized). Depending on the condition, the participant was shown a rank-ordered list of between 5 and 25 movies (list length was manipulated between subjects) represented by a movie title. The predicted rating (in stars and one point decimal value) was shown next to the title. If the participant hovered over one of the titles, additional information appeared in a separate preview panel. This additional information consisted of the movie cover, the title of the movie, a synopsis, the name of the director(s) and part of the cast. Before moving to the next list, participants were presented with a short questionnaire of 16 items, measuring choice difficulty, tradeoff difficulty, perceived diversity and perceived attractiveness of the presented list. Participants thus answered these questions about each of the three lists.

3.5 Participants

Participants for this study were gathered using an online participant database. Participants were compensated with 3 euro (about 4 US dollars) for participating. 97 participants completed the study (mean age: 29.2 years, sd=10.3, 52 females and 45 males).

4. RESULTS

4.1 Measures

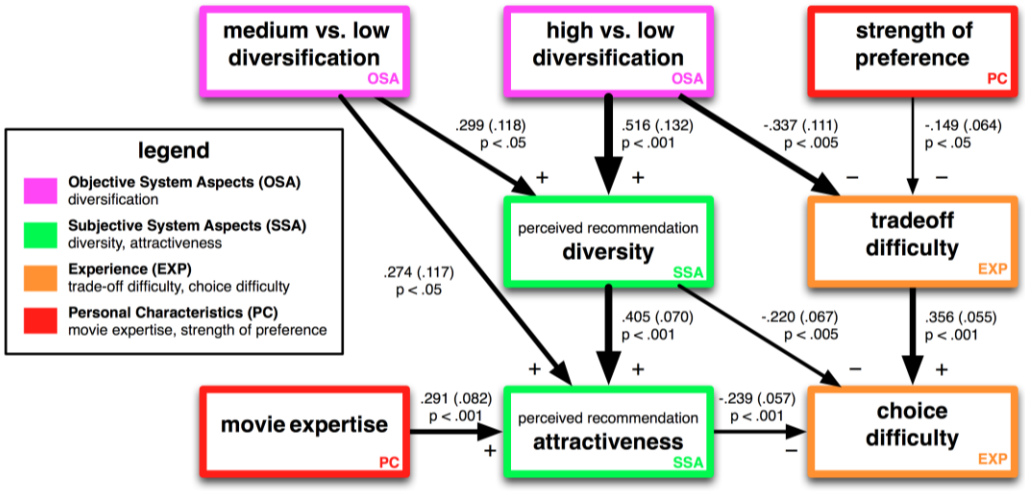
The items in the questionnaires were first submitted to an exploratory factor analysis (EFA) to determine whether their covariances naturally reproduced the predicted constructs. The EFA used repeated ordinal dependent variables, a weighted least squares estimator and Geomin rotation. After deleting items with low communalities or high cross-loadings, the analysis produced five correlated factors with a good model fit ($\chi^2(86) = 176.4$, $p < .001$, CFI = .988, TLI = .977, RMSEA = .060)[3], a good factor definition (lowest relevant loading: .576, highest irrelevant loading: .225) and a good discriminant validity (highest factor correlation: .396). These are the resulting five factors and their underlying items:

- Perceived recommendation diversity (5 items):
  - The list of movies was varied.
  - All the movies were similar to each other.
  - Most movies were from the same genre.
  - Many of the movies in the list differed from other movies in the list.
  - The movies differed a lot from each other on different aspects.
- Perceived recommendation attractiveness (5 items):
  - I would give the recommended movies a high rating.
  - The list of movies showed too many bad items.
  - The list of movies was attractive.
  - I didn't like any of the recommended items.
  - The list of recommendations matched my preferences.
- Strength of preference (3 items):
  - I have clearly defined preferences concerning movies.
  - I know what kind of movies I like.
  - Most of the time I let someone else pick a movie for me.

[1] See http://www.mymediaproject.org/

[2] The MovieLens dataset is available at http://grouplens.org.

[3] Based on extensive simulations, Hu and Bentler [6] propose cut-off values for these fit indices to be: CFI > .96, TLI > .95, and RMSEA < .05.



- Movie expertise (4 items):
  - I am a movie lover.
  - Compared to my peers I watch a lot of movies.
  - Compared to my peers I am an expert on movies.
  - I only know a few movies.
- Maximizing tendency (3 items):
  - No matter what I do, I have the highest standards for myself.
  - I never settle for the second best.

Two additional items were selected as indicators:

- Choice difficulty ("I would find it difficult to choose a movie from this list")
- Tradeoff difficulty ("I had to put a lot of effort into comparing the different aspects of the movies")

Nine additional items failed to contribute to a single factor, and were therefore deleted from the analysis.

4.2 Manipulation checks

We compared the resulting diversity, predicted ratings and variance of the predicted ratings in our data to check our diversification algorithm. As can be seen in Table 1, our diversification algorithm indeed increases the average range of the scores on the 10 matrix factorization features (calculated through Equation 1), a proxy of the level of attribute diversity.

$$AFSR = \sum_{1}^{10} \frac{\max(x_i) - \min(x_i)}{10} \quad (1)$$

Table 1: Diversity and predicted ratings of the presented items in our study

diversity	Average feature score range (AFSR) mean (SE)	Predicted rating mean (SE)	SD predicted rating mean (SE)
Low	0.959 (0.015)	4.486 (0.042)	0.163 (0.010)
Medium	1.273 (0.016)	4.486 (0.041)	0.184 (0.011)
High	1.744 (0.024)	4.527 (0.039)	0.206 (0.013)

At the same time the predicted ratings do not differ between the three levels of diversification showing that we manipulated diversity independent of (predicted) attractiveness. The standard deviation of the predicted ratings for the three set does increase slightly with increasing diversity.

4.3 SEM Model

The subjective constructs from the EFA were organized into a path model using a confirmatory structural equation modeling (SEM) approach with repeated ordinal dependent variables and a weighted least squares estimator. In the resulting model, the subjective constructs are structurally related to each other and to the conditions (list length and diversification level). The model was constructed based on the user-centric framework for user experience of recommender systems described in Knijnenburg et al. [10]. In the final model, the maximizer scale did not relate to any other variable, and was therefore removed from the analysis. The manipulation "list length" (whether participants were shown 5, 10, 15, 20 or 25 recommendations) also did not have a significant influence on the other variables. The results are therefore collapsed over these conditions. We also did not observe any effect of the order in which the three lists were presented.

The final model had a good model fit ($\chi^2(179) = 256.5$, $p < .001$, CFI = .989, TLI = .987, RMSEA = .041). Figure **2** displays the effects found in this model. Factor scores in the final model are standardized; the numbers on the arrows (A → B) denote the estimated mean difference in B, measured in standard deviations, between participants that differ one standard deviation in A. The number in parentheses denotes the standard error of this estimate, and the p-value below these two numbers denotes the statistical significance of the effect. As per convention, only effects with $p < .05$ are included in the model. The medium and high diversification conditions are compared to the low diversification baseline condition; numbers on the arrows originating in the conditions denote the mean differences between participants in medium or high diversification condition and participants in the low diversification condition.

To better understand the effects, we plotted the marginal means of the subjective constructs in the mid and high diversification condition relative to low diversification condition in Figure 3. Our diversification algorithm significantly affects the perceived diversity in linear fashion, with medium and high diversification

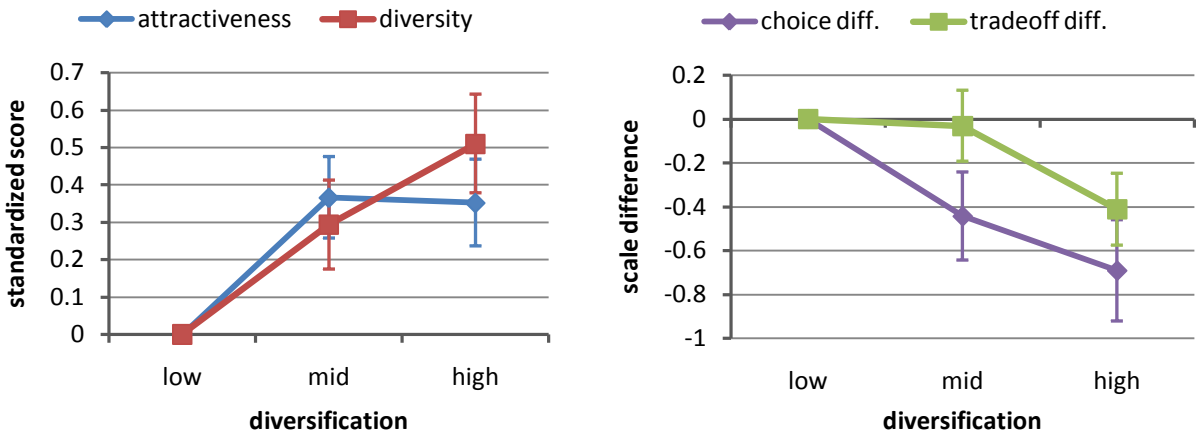


Figure 3: Left side: marginal means of perceived attractiveness diversity scores for mid and high diversification relative to low diversification: Right side: relative scale differences of mid and high diversification scores relative to the low diversification scores. Error bars are one std. err of the mean.

resulting in significantly higher perceived diversity than the low diversification condition. Higher perceived diversity subsequently increases the perceived attractiveness of the recommendations. The medium diversification condition also has a direct positive effect on attractiveness, making medium diversification as attractive as the high diversification (and both are significantly more attractive than low diversification, see Figure 3). There is also a direct effect of expertise, a personal characteristic, on attractiveness, showing that expert participants report higher attractiveness ratings.

In terms of tradeoff difficulty, we observe that this is significantly (and negatively) influenced by the high diversification condition, as well as a main effect of strength of preferences. So people experience less tradeoffs with very high diversification and if their self-reported strength of preference is higher, they also experience less tradeoff difficulty. The negative effect of diversification goes against the expectation that higher diversification leads to options that encompass larger tradeoffs between the attributes. Potentially the high diversification does not generate items that encompass apparent tradeoffs in the specific domain of movies.

All these constructs together influence the choice difficulty experienced by the user, which goes up with increased tradeoff difficulty, but goes down with increased diversity and attractiveness. The net result of our diversification manipulation on choice difficulty is negative: the higher the diversity of the set, the more attractive and diverse, and the less difficult it is to choose from the set (the marginal effects in Figure 3 suggest that choice difficulty decreases almost linearly with diversification level).

Our two experience constructs, tradeoff and choice difficulty, suggest that difficulty within this domain and for this recommender system is caused predominantly by lack of diversity in the item set, probably because the items are too similar to the user be able to make up her mind and find an option that is easy to justify. We do not find any evidence for the tradeoff difficulty related to conflicting preferences, which according to Scholten and Sherman [18] may occur when items become too diverse and tradeoffs become difficult to resolve.

5. Conclusion and Discussion

This paper shows how choice difficulty and tradeoff difficulty associated with a list of recommendations are influenced by the diversity of the list on the underlying latent features. By increasing the diversity of the items on these underlying dimensions, we increase the perceived diversity and attractiveness of the set, and subsequently reduce choice difficulty. Our net result thus is not a U-shaped relation between diversity and choice difficulty, but rather a simple linear downward trend.

We also do not observe an effect of item size on the perceived diversity or the experienced difficulty. Though intuitively, one would expect such an effect, we did not observe large variations across the different lengths. Given that our diversification algorithm finds items that are maximally spaced out from each other on the latent features within the set of equally attractive options, this might be not very surprising: when all items are good, diversification helps for both small and large sets. We would thus expect the effect of diversification to be roughly equal for different list lengths. Moreover, as the 5 different list lengths were manipulated between subjects, we have limited statistically power to detect such small differences.

We may have found a lack of an effect of item size on choice difficulty (which deviates from previous studies of choice overload) because we did not ask our participants to actually make a choice from the item sets. In the current study we explicitly did not investigate choice overload in the classical sense (showing that people defer choice or are less satisfied with their chosen option afterwards) but tried to investigate first whether diversifying on the underlying latent features of the recommender algorithm could influence the perceived diversity, attractiveness and difficulty. In a follow-up study, we plan to use this diversification to further investigate if manipulating the diversity (combined with item set size) will influence choice overload. In that study we will explicitly ask people to choose an item from the list of recommendations and also measure their satisfaction with the chosen item (as in Bollen et al. [2]).

Our results seem to corroborate the idea that latent features in matrix factorization algorithms have psychological meaning, as was suggested before [11]. Others [9, 12] already suggested that

many of the findings from decision-making research could be very relevant for the development of recommender systems. Our data suggests that manipulating the underlying latent feature spaces could be one key to further understand and explore the role of important effects such as context and reference points in recommender systems.

6. ACKNOWLEDGMENTS

We would like to thank Niels Reijmer and Dirk Bollen for helpful discussions on the study.

7. REFERENCES

[1] Bettman, J.R., Luce, M.F. and Payne, J.W. 1998. Constructive Consumer Choice Processes. *Journal of Consumer Research*. 25, 3 (Dec. 1998), 187-217.

[2] Bollen, D., Knijnenburg, B.P., Willemsen, M.C. and Graus, M. 2010. Understanding choice overload in recommender systems. *Proceedings of the fourth ACM conference on Recommender systems - RecSys '10* (Barcelona, Spain, 2010), 63-70 DOI=http://doi.acm.org/10.1145/1864708.1864724

[3] Chernev, A. 2003. Product assortment and individual decision processes. *Journal of Personality and Social Psychology*. 85, 1 (2003), 151-162.

[4] Fasolo, B., Hertwig, R., Huber, M. and Ludwig, M. 2009. Size, entropy, and density: What is the difference that makes the difference between small and large real-world assortments? *Psychology and Marketing*. 26, 3 (Mar. 2009), 254-279.

[5] 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 (Jan. 2004), 5–53.

[6] Hu, L.-tze and Bentler, P. 1999. Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling: A Multidisciplinary Journal*. 6, 1 (1999), 1-55.

[7] Hu, R. and Pu, P. 2011. Enhancing recommendation diversity with organization interfaces. *Proceedings of the 16th international conference on Intelligent user interfaces* (New York, NY, USA, 2011), 347–350.

[8] Iyengar, S.S. and Lepper, M.R. 2000. When choice is demotivating: Can one desire too much of a good thing? *Journal of Personality and Social Psychology*. 79, 6 (2000), 995-1006.

[9] Jameson, A., Gabrielli, S., Kristensson, P.O., Reinecke, K., Cena, F., Gena, C. and Vernero, F. 2011. How can we support users‟ preferential choice? *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems* (Vancouver, BC, Canada, 2011), 409–418.

[10] Knijnenburg, B.P., Willemsen, M.C., Gartner, Z., Soncu, H. and Newell, C. Explaining the User Experience of Recommender Systems. *accepted to User Modeling and User-Adapted Interaction. http://t.co/cC5qPr9*.

[11] Koren, Y., Bell, R. and Volinsky, C. 2009. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*. 42, 8 (2009), 30-37.

[12] Mandl, M., Felfernig, A., Teppan, E. and Schubert, M. 2011. Consumer decision making in knowledge-based recommendation. *Journal of Intelligent Information Systems*. 37, 1 (Aug. 2011), 1-22.

[13] Mogilner, C., Rudnick, T. and Iyengar, S.S. 2008. The Mere Categorization Effect: How the Presence of Categories Increases Choosers‟ Perceptions of Assortment Variety and Outcome Satisfaction. *Journal of Consumer Research*. 35, 2 (Aug. 2008), 202-215.

[14] Nenkov, G.Y., Morrin, M., Ward, A., Schwartz, B. and Hulland, J. 2008. A short form of the Maximization Scale: Factor structure, reliability and validity studies. *Judgment and Decision Making Journal*. 3, 5 (Jun. 2008), 371-388.

[15] Reutskaja, E. and Hogarth, R.M. 2009. Satisfaction in choice as a function of the number of alternatives: When "goods satiate." *Psychology and Marketing*. 26, 3 (Mar. 2009), 197-203.

[16] Scheibehenne, B., Greifeneder, R. and Todd, P.M. 2010. Can There Ever Be Too Many Options? A Meta- Analytic Review of Choice Overload. *Journal of Consumer Research*. 37, 3 (Oct. 2010), 409-425.

[17] Scheibehenne, B., Greifeneder, R. and Todd, P.M. 2009. What moderates the too-much-choice effect? *Psychology and Marketing*. 26, 3 (Mar. 2009), 229-253.

[18] Scholten, M. and Sherman, S. 2006. Tradeoffs and theory: The double-mediation model. *Journal of Experimental Psychology: General*. 135, 2 (May. 2006), 237-261.

[19] Schwartz, B. 2004. The tyranny of choice. *Scientific American*. 290, 4 (Apr. 2004), 70-75.

[20] Schwartz, B., Ward, A., Monterosso, J., Lyubomirsky, S., White, K. and Lehman, D.R. 2002. Maximizing versus satisficing: Happiness is a matter of choice. *Journal of Personality and Social Psychology*. 83, 5 (2002), 1178-1197.

[21] Sela, A., Berger, J. and Liu, W. 2009. Variety, Vice, and Virtue: How Assortment Size Influences Option Choice. *Journal of Consumer Research*. 35, 6 (Apr. 2009), 941-951.

[22] Shafir, E., Simonson, I. and Tversky, A. 1993. Reason-based choice. *Cognition*. 49, 1-2 (Nov. 1993), 11-36.

[23] Ziegler, C.-N., McNee, S.M., Konstan, J.A. and Lausen, G. 2005. Improving recommendation lists through topic diversification. *Proceedings of the 14th international conference on World Wide Web* (Chiba, Japan, 2005), 22–32.

Users' Decision Behavior in Recommender Interfaces: Impact of Layout Design

Li Chen and Ho Keung Tsoi
Department of Computer Science
Hong Kong Baptist University
Hong Kong, China
{lichen, hktsoi}@comp.hkbu.edu.hk

ABSTRACT
Recommender systems have been increasingly adopted in the current Web environment, to facilitate users in efficiently locating items in which they are interested. However, most studies so far have emphasized the algorithm's performance, rather than from the user's perspective to investigate her/his decision-making behavior in the recommender interfaces. In this paper, we have performed a user study, with the aim to evaluate the role of layout designs in influencing users' decision process. The compared layouts include three typical ones: list, grid and pie. The experiment revealed significant differences among them, with regard to users' clicking behavior and subjective perceptions. In particular, pie has been demonstrated to significantly increase users' decision confidence, enjoyability, perceived recommender competence, and usage intention.

Categories and Subject Descriptors
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms
Design, Experimentation, Human Factors.

Keywords
Users' decision behavior, recommender system, interface layout, user study.

1. INTRODUCTION

Although recommender systems have been popularly developed in recent years as personalized decision support in social media and e-commerce environments, more emphasis has been placed on improving algorithm accuracy [10], and less on studying users' actual decision behavior in the recommender interfaces. On the other hand, according to user studies conducted in other areas, users will likely adapt their behavior when being presented with different information presentations. For instance, in a recent study done by Kammerer and Gerjets, the presentation of Web search engine results by means of a grid interface seems to prompt users to view all results at an equivalent level and to support their selection of more trustworthy information sources [7]. Braganza et al. also investigated the difference between one-column and multi-column layouts for presenting large textual documents in web-browsers [1]. They indicated that users spent less time scrolling and performed fewer scrolling actions with the multi-column layout.

RecSys'11 Workshop on Human Decision Making in Recommender Systems, October 23-27, 2011, Chicago, IL, USA

Unfortunately, little is known about the impact of recommender interface's layout on users' decision-making behavior. There is also lack of studies that examined whether users would perceive differently, especially regarding their decision confidence and perceived system's competence, due to the change of layout. Thus, in this paper, we are particularly interested in exploring users' behavior in the recommender interface when it is presented with three layout designs: list, grid and pie. As a matter of fact, most of current recommender systems follow the list structure, where recommended items are listed one after another. The grid layout, a two-dimensional display with multiple rows and columns, has also been applied in some recommender sites to display the items. As the third alternative design, pie layout, though it has been rarely used in recommender systems, has been proven as an effective menu design for accelerating users' selection process [2]. The comparison among them via user evaluation could hence tell us which layout would be most desirable to optimize the recommender's benefits. That is, with the ideal layout design, users can be more active in clicking recommendations, be more confident in their choices, and be more likely to adopt the recommender system for repeated uses.

Concretely, we evaluated three layout designs from both objective and subjective aspects to measure users' decision performance. The objective measures include users' clicking behavior (e.g., the first clicked item's position, the amount of clicked items, etc.), and time consumption. Subjective measures include users' decision confidence, perceived interface competence, enjoyability, and usage intention. These measurements are mainly based on the user evaluation framework that we have established from prior series of user studies on recommenders [4,8,9]. We thus believe that they can be appropriately utilized as the standard to assess user behavior. Relative to our earlier work [5], this paper was for the first time to investigate the effect of basic layouts of recommender interfaces on users' decision process, which is also new in the general domain of recommender systems, to the best of our knowledge.

2. THREE LAYOUT DESIGNS

2.1 List Layout

As mentioned above, most existing recommender systems employ the standard one-dimensional ranked-order *list* style, where all items are displayed one after the other. For instance, MovieLens is a typical collaborative filtering (CF) based movie recommender system (www.movielens.org). In this system, items are ranked by their CF scores in the descending order and presented in the list format. The score represents the item's matching degree with the current user's interest.

Figure 1.a shows the sample layout (where every position is for placing one item). The number of shown items varies among

existing systems. Some systems (e.g., Criticker.com) limit the number to 10 or less, while some systems (like MovieLens) give a list of items as many as possible and divide them into pages (e.g., one page displays a fixed number of items). Each item is usually described with its basic info (e.g., thumbnail image, name, rating). When users click an item, more of its details will be displayed in a separate page.

2.2 Grid Layout

The *grid* layout design has also been applied in some existing websites (e.g., hunch.com). In this interface, recommendations are presented in multiple rows and columns, so several items are laid out next to each other in one line. The regular presentation is to align the items horizontally (line by line). For example, as shown in Figure 1.b, the positions 1, 2, 3, …, 12 are respectively allocated with items that are ranked 1st, 2nd, 3rd, …, 12st according to their relevance scores.

Because users likely shift eyes to nearby objects [6], we were interested in verifying whether the grid format would stimulate users to discover more items than in list.

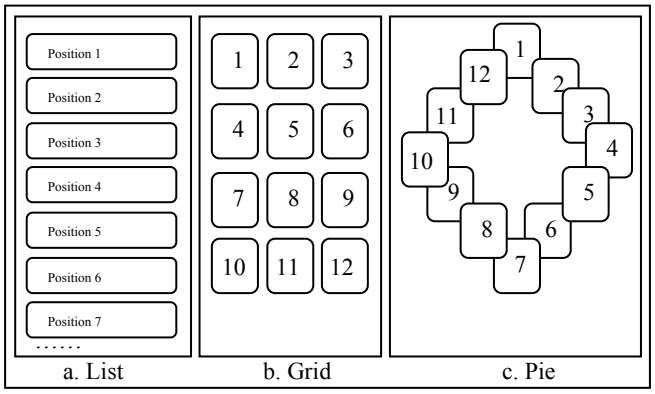


Figure 1. The three layout designs for recommender interface (the number refers to the position of a recommendation).

2.3 Pie Layout

Another two-dimensional layout design is to place the items in the compass format, i.e., *pie* layout. This idea originates from the comparison of linear menu (i.e., the alphabetic ranked-order of menu choices) and pie menu [2]. In the pie menu, items are placed along the circumference of a circle at equal radial distances from the center. The distance to and size of the target can be seen as an effect on positioning time according to Fitts' law [3]. Researchers previously found that due to the decreased distance (i.e., the minimum distance needed to highlight the item as selected) and increased target size, users selected items slightly faster. The drift distance after target selection and error rates were also minimized.

We thus believe that the pie layout could offer a novel alternative and potentially more effective design to be studied. The reason is that it would support users to have a quicker overview of all displayed items, as the interface consumes greater width but less height. In addition, it would allow users to click items faster, because the mean distance between items is reduced.

When we concretely implemented this interface, we adhere to the regular clockwise direction to display the items along the circle, with the most relevant item placed at the first position (see Figure 1.c).

3. PROTOTYPE IMPLEMENTATION

We implemented a movie recommender system with the three layout versions. The recommending mechanism is primarily based on the hybrid of tag suggestions and tag-aware item recommendation [11]. Specifically, based on the user's initial tag profile, the system will first recommend a set of tags from other users as suggestions to enrich the new user's profile. In the mean time, a set of movie items with higher matching degree with the user's current tag profile is returned as item recommendations. If the user modifies her/his profile, the set of recommendations will be updated accordingly. More concretely, the control flow of the system works in the following four steps:

Step 1. To begin, the new user is asked to specify a reference product (e.g., a favorite movie) as the starting point. The product and its associated tags (as annotated by other users) are then stored in the user's profile. Alternatively, s/he can directly input one or more tag(s) for building her/his initial profile.

Step 2. *Profile-based Item Recommendation*. Based on the profile, the system generates a set of item recommendations (i.e., movies in our prototype) to the user via the weighted combination of FolkRank and content-based filtering approaches. Specifically, FolkRank transforms the tripartite graph found in the folksonomic systems into the two-dimension hyper-graph. In parallel, the content-based filtering approach rank items based on the correlation between the content of the items (i.e., title, keywords, and user-annotated tags) and the user's current profile. A tuning parameter is dynamically set to adjust the two approaches' relative weights in producing the top *k* recommendations.

Step 3. *Tag recommendation*. In the recommender interface, the system not only returns item recommendations, but also a set of tags to help users further enrich their profile if they need. To generate the tag recommendation, we first deployed the Latent Dirichlet Allocation (LDA), which is a dimensionality reduction technique, to extract common topics among all user tags in the database. Each topic represents a cluster, and all the extracted clusters were then applied to match with the current user's tag profile. New tags from the best matching clutsers are then retrieved as recommended tags to the user. These tags' associated items are also integrated into the process of generating item recommendations in the next cycle if any of them were selected by the user. Moreover, the tag recommendations were grouped into three categories in the interface: *factual tags* (i.e., the tag describes a fact of the item, "rock"), *subjective tags* (the people's opinion, "cool") and *personal tags* (used to organize the user's own collection, e.g., "my favorites"). The grouping is automatically performed. For example, if the tag is a common keyword in the item's basic descriptions, it is treated as *factual tags*. General Inquirer[1] , a content analysis program, is employed to determine whether a tag is subjective. The rest of the tags that do not belong to the first two categories are then considered to be *personal tags*.

Step 4. If the user has done any modifications on her tag profile, it will be used to produce a finer-grained item recommendation in the next interaction cycle (returning to step 2).

The process from Step 2 to Step 4 continues till the user selects item(s) as her/his final choice(s), or quit from the system without

[1] http://www.webuse.umd.edu:9090/

selecting any recommendations. More details about the algorithm steps can be referred to [11].

To build the prototype, we crawled 998 movies and their info (including posters, names, overall ratings, number of reviewers, directors, actors/actresses, plots, etc.) from IMDB (Internet Movie Database) site. These movies' associated tags were extracted from MovieLens for building the tag base.

Concretely, the system returns 24 movie recommendations at a time. The 24 movies are sorted in the descending order by their relevance scores, and then divided into two pages (i.e., each page with 12 movies). The switching to the second page is through the "More Movies" button. Such design could enable us to evaluate user behavior not only in a single page, but also their switching behavior across pages (i.e., whether they click the button to view more items).

The recommended movies are presented differently in the three layout versions (see Figure 3). In the *list layout*, the 12 movies in one page are displayed in the list style, where the ranked 1st is positioned at the top, followed by the ranked 2nd one (the ranked 1st one means that the movie has the highest score among the 12 movies). In the *grid layout*, three movies are displayed along one row and four in one column. More specifically, the first row shows the ranked 1st, 2nd and 3rd movies from left to right, the second row is with the ranked 4th, 5th, 6th movies, and so on. In the pie layout, the 12 movies (each with the same target size as in grid) are presented in a clockwise direction, with the ranked 1st movie at the 12 clock's position, 2nd at the 1 clock's position, and so forth.

In all of the three interfaces, each movie has a poster image, name, rating, number of reviews and a brief plot. More of the movie's details can be accessed by clicking it. A separate detail page will then show the movie's director(s), actor/actress info, detailed plot, and give links to IMDB and trailer, etc. If users like this movie, they could click the button "My Choice" at the detail page.

There is also a profile area in the three interfaces, which allows users to modify their tag profile by selecting the system-suggested ones or inputting their own. In list and grid, it is placed on the left panel, and in pie, it is in the central part.

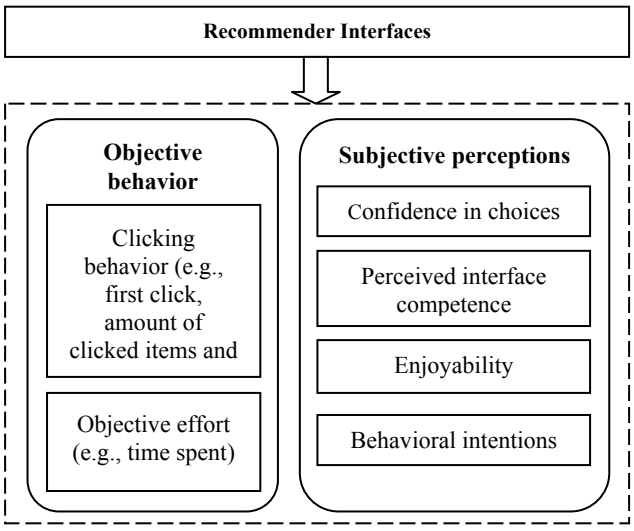


Figure 2. Objective and subjective measures in the user study.

4. EXPERIMENT SETUP

4.1 Measures

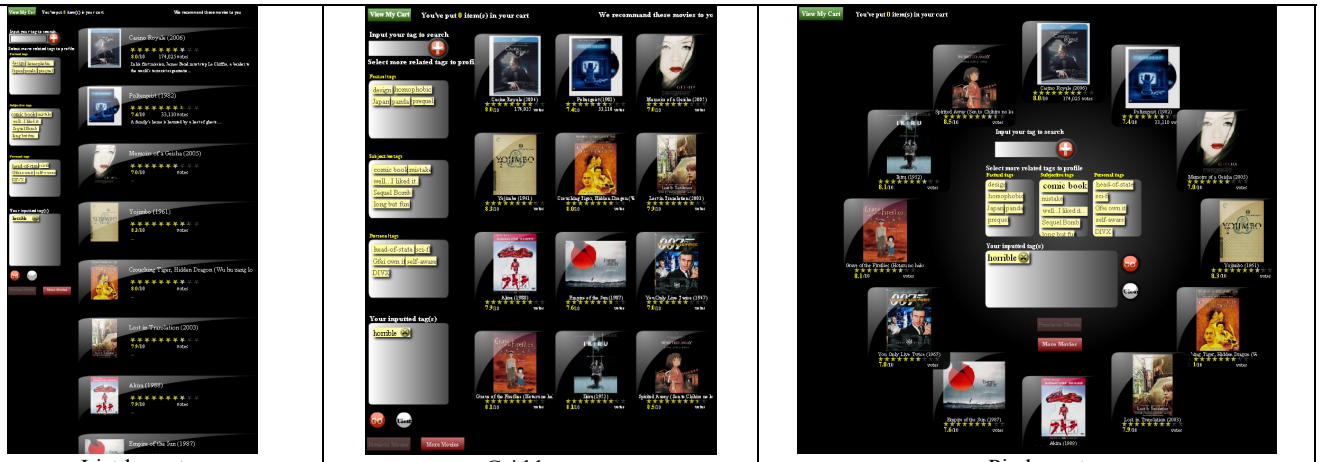
Identifying the appropriate criteria for assessing a recommender system from the user's perspective has always been a challenging issue. Accumulated from our previous experiences on this track [4,8,9], a set of measures have been established. The framework not only includes objective interaction effort that users have spent with the system (e.g., time consumption), but also users' perceived confidence in choices that they made in the recommender and their intention to repeatedly use the system. More specifically, in this experiment, in order to in depth identify the three layouts' respective effects on user behavior, we assessed the following aspects (see Figure 2).

4.1.1 Objective Measures

The objective measures mainly include quantitative results from analyzing users' actual behavior in using the interface. Concretely, they cover two major aspects.

Clicking behavior. It has been broadly recognized that users' clicking decisions on the recommender interface (i.e., clicking an item to view its detailed info) reflects their interest in the item. Therefore we recorded users' clicking behavior and clicked items' positions. The goal was to evaluate whether the clicking would be influenced by the layout, and which interface could support users to easily find interesting items. Specifically, the clicking behavior was analyzed via three variables: 1) the users' *first clicked item's position*, from which we could know whether users' first click falls on the most relevant item (as predicted by the system) or not. 2) *All clicks on distinct items that a user has made* throughout her/his session of using the interface. This variable can expose the distribution of clicks over different areas on the interface. The comparison among all users could further reveal their similar clicking pattern. In addition, the total amount of clicked items could tell us how many items interested the user when s/he was confronted with the whole set of recommendations in the respective layouts. 3) *Frequency of clicking "more movies"*. Such action indicates that users switched to the next page to view more recommended items. If the frequency is higher, one possible explanation is that users felt enjoyable while using the interface and were motivated to take the effort in viewing more items, or it is because users cannot find the interesting items at the first page. Thus, this number should be analyzed in combination with other variables, especially users' subjective opinions on the interface, so that we could more fairly attribute it to the pros or cons of the interface.

Objective effort consumption. Besides above mentioned analyses on users' clicking behavior, we also recorded the time a user spent in completing the task on the specific interface. This value can be used to represent the amount of objective effort that users exerted while using the interface. In fact, it has been frequently adopted in related literatures to be an indicator of the system's performance [10]. However, less time does not mean that users would perceive less effort taken or have better decision quality [8]. That is why we included various subjective constructs (see the next subsection) to better understand the interface's true merits.



List layout | Grid layout | Pie layout

Figure 3. A movie recommender interface with three layout versions.

4.1.2 Subjective Measures

Users' decision confidence and perception of the interface were mainly obtained through the post-task survey. Actually, the subjective measures can be quite useful to expose the competence of the interface in assisting users' decision-making and its ability in increasing users' intention to use the system again. The variables that we have used in this experiment cover four constructs: *decision confidence, perceived interface competence*, *enjoyability*, and *behavioral intentions*. The perceived interface competence was qualitatively measured through multiple dimensions: users' perception of item/tag recommendation quality, perceived ease of use of the interface in searching for info, and perceived ease of use in modifying their profile. The behavioral intention was assessed from users' intention to use the interface again.

Table 1 lists all of the questions we used to measure these subjective variables. In the form of questionnaire, each question was required to respond on a 5-point Likert scale from "strongly disagree" (1) to "strongly agree" (5).

Table 1. Questions to measure users' subjective perceptions

Measured variables	**Question responded on a 5-point Likert scale from "strongly disagree" to "strongly agree"**
Decision confidence	Q1: I am confident that I found the best choices through the interface.
Perceived recommender interface's competence	Q2: The interface helps me find some good movies; Q3: This interface provides some good "tag" suggestions to help me specify criteria; Q4: I found it easy to use the interface to search for movies; Q5: I found it easy to modify my profiles in the interface.
Enjoyability	Q6: I felt enjoyable while using this interface.
Behavioral Intention	Q7: I am inclined to use this interface again.

4.2 Experiment Procedure and Participants

The primary factor manipulated in the experiment is *layout* as we prepared with three versions in the prototype system: list, grid, pie. To compare the three layouts, we applied the within-subjects experiment design. That is, every participant was required to evaluate all of them one by one, but the interfaces' appearance order was randomized in order to avoid any carryover effects (so there are six possible sequences of displaying the three layouts). To evaluate each layout, a concrete task was assigned to the user. Concretely, each layout interface was randomly associated with one scenario for the user to play the role and perform the situational task. For example, one scenario is "*This is October, the festival Halloween is coming. John is a college student, and he would like to organize an event to watch movie with his friends at his home. After discussing with his friends, they would like to watch a horror movie in this festival. John is responsible for choosing some movies as candidates. Please imagine yourself as John and use the interface to find three candidates that you would like to recommend to your friends.*" The other two scenarios were respectively for Valentine's Day, and the military subject. In each scenario, the user was encouraged to freely use the interface to find three most suitable movies according to her/his own preferences.

The experiment was setup as an online procedure. It contains the instructions, recommender interfaces and questionnaires, so that users could easily follow and we could also automatically record all of their actions in a log file. The same administrator supervised the experiment for all participants.

A total of 24 volunteers (12 females) were recruited. 3 are with age less than 20, 1 with age above 30, and the others are between 20 and 30. Most of them are students in the university, pursuing Bachelor, Master or PhD degrees, but their studying majors are diverse. All participants had visited movie recommender sites (e.g., Yahoo movie) before the experiment, and 58.3% have even visited the indicated sites at least a few times every three months. The participants also specified the mean reasons that will motivate them to repeatedly use such a site. Among the various reasons, the ease of use of the site's user interface was indicated as the most important factor (with the importance rate 3.83 in the range of 1 to 5). The second important factor is the site's ability in helping them find movies that they like (3.79), followed by the site's reputation (3.5).

4.3 Results

4.3.1 Objective Behavior

For each layout version, we first counted the number of users' first clicks that fall on a particular position and then classified them into areas. Specifically, in one interface, each area contains three adjacent positions (e.g., 1-3 positions compose the first area, 4-6 form the second area, and so on). Areas 5 to 8 refers to the positions at the second recommendation page of the interface.

Figure 4 shows the actual distribution. In total, 8, 10, and 8 users have clicked item in the first area respectively in list, grid and pie interfaces. Then in the list and pie, there exists a linear drop from areas 1, to 2, then to 3. In area 4, the list's curve returns to the same level of area 2, but in pie it goes much higher even beyond the level of area 1. In grid, a sharp drop appears from areas 1 to 2. Then the curve rebounds and reaches to a level equivalent in areas 3 & 4. Another interesting finding is that there are 3, 2, and 1 of users' first clicks were at the second page respectively in list, grid and pie (i.e., in areas 5 to 8). To rank these areas by the amounts of first clicks, we can see that the hotter areas in list are 1, 2 & 4. In grid, they are 1, 3 & 4, and in pie, they are 1 and 4.

To further investigate the hot areas throughout a user's whole interaction session, we counted her/his total clicks made on each interface. The average numbers of items clicked by a single user are 3.96, 3.875, and 4.84 in list, grid and pie respectively. The difference between grid and pie is even marginally significant (p = 0.076, t = -1.86, by paired samples t-test). The exact distribution of the average user's clicks among the eight areas is shown in Figure 5, from which we can see that above 50% of a user's clicks on list were in areas 1 (28.42%) and 4 (27.37%), followed by areas 3 and 2. In grid and pie, the two hotter areas are also 1 and 4, but the comparison regarding areas 2 and 3 shows that the clicks on them are more evenly distributed in pie (respectively 17.24% and 18.10%), which in fact also has higher total amount of clicks than in grid.

Moreover, the clicking distribution across pages 1 and 2 is significantly different among the three interfaces. More clicks appeared in grid's second page (24.73% accumulated from areas 5 to 8), and pie's (19.83%), against 7.37% in list. This finding suggests that grid and pie might more likely stimulate users to click the "More Movies" button for viewing more recommended items. In this regard, we further found that 50% of users have actually gone to the second page while using grid, followed by 41.7% users who did so in pie, and 25% in list (p = .056 between grid and list, t = -2.01).

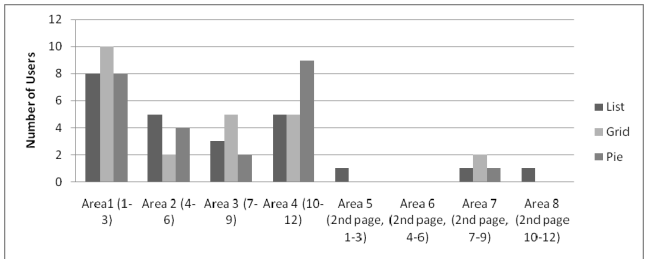


Figure 4. The distribution of users' first clicks.

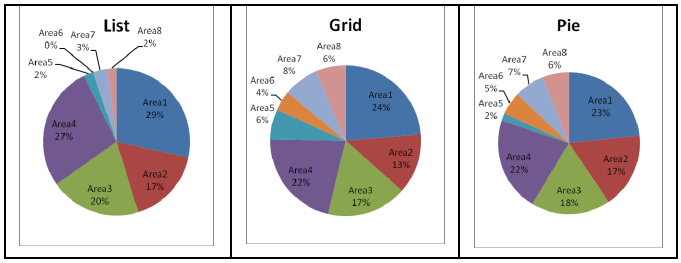


Figure 5. The distribution of an average user's whole clicks during her interaction session with an interface.

As for the total time spent on each interface, on average, it is 156.375 seconds in list, 109.875 seconds in grid, and 152.667 in pie. Though it took longer in list and pie, the differences are not significant (p > 0.1 by ANOVA and three pairs of t-test).

4.3.2 Subjective Perceptions

Besides measuring users' objective behavior, we were driven to further understand their subjective perceptions such as decision confidence, perceived ease of use of the recommender interface, and intention to use it again in the future, as described in Section 4.1.2.

Significant differences were found in respect of these subjective measures (see Table 2). First of all, most of users were confident that they found the best choices through pie. The mean score is 3.54 which is marginally significantly higher than the average in list (vs. 3.125, p = .076, t = -1.85). The grid's score is in between (3.33). Secondly, due to the change of layout, users perceived pie more competent in helping them find good movies (3.58 vs. 3.29 in grid, p = .09, t = -1.77; list: 3.33), easier to use (3.5 in pie against 3 in list, t = -2.77, p = .01; the difference between grid and list is also marginally significant: 3.375 vs. 3, p = .095), and easier to modify their profile (3.375 in pie vs. 3.04 in list, t = -1.88, p = .07). Moreover, users rated higher on pie's ability in providing good tag suggestions (3.46 in pie vs. 3 in list, t = .2.41, p = .02; vs. 2.9 in grid, t = -2.25, p = .03). They also felt more enjoyable while using pie than list (3.42 against 2.875 in list, t = -2.72, p = .01; grid: 3.12). The median and mode values are also reported in Table 2.

Table 2. Users' subjective perceptions with the three layouts (L: List; G: Grid; P: Pie)

	Mean (st.d)			Median			Mode		
	L	G	P	L	G	P	L	G	P
Q1	3.125 (.85)	3.33 (.92)	**3.54*^{L}** (.78)	3	3.5	4	3	4	4
Q2	3.33 (.82)	3.29 (1.04)	**3.58*^{G}** (.72)	3	3.5	4	4	4	4
Q3	3 (.88)	**3.375*^{L}** (.92)	**3.5*^{L}** (.88)	3	3	4	3	3	4
Q4	3 (.98)	2.92 (1.02)	**3.46*^{L,G}** (.88)	3	3	3.5	4	3	4
Q5	3.04 (.91)	3.17 (.92)	**3.375*^{L}** (.92)	3	3	4	3	3	4
Q6	2.875 (.74)	3.17 (.96)	**3.42*^{L}** (.93)	3	3	3.5	3	4	4
Q7	2.92 (.83)	3.17 (.92)	**3.29*^{L}** (.95)	3	3	3.5	3	3	4

Note: Asterisks denote highly or marginally significant differences to the respective abbreviated interfaces (by paired samples t-test).

4.3.3 User Comments

At the end of the study, we also asked each user to give some free comments on the interfaces. 9 users explicitly praised pie. As quoted from their words, "*it is easy for me to see all without scrolling the page*", "*easy, clear, more information*", "*easy to use*", "*no need to loop around as the movies are all in the middle*", etc. Similar preference was also given to grid: "*I can get a glimpse of all movies within a page*", "*the layout of displaying movie is good for browsing*", "*it lists more movies*", "*the item displayed clearly, and no need to scroll up or scroll down for watching the information*". Thus, the obvious advantage of pie and grid, as user perceived, is that they allow them to easily see many choices without scrolling and facilitate them to browse and seek info. On the other hand, the comments to list were mainly negative (as stated by 14 users): "*find the movie difficultly*", "*need to scroll down*", "*not easy to use*", "*I can't see all suggested movies at once*", "*too long inefficient take effort to scroll*", etc. Therefore, the frequent reason behind users'

disliking is that the list is not easy for them to see all suggested movies and demands more effort.

5. CONCLUSIONS AND FUTURE WORK

In conclusion, this paper reports our in-depth studying of users' decision behavior and attitudes in different recommender interface layouts. Specifically, we compared three typical layout designs: list, grid and pie. The results revealed that in list and grid, users' first clicks largely fall in the top three positions, but in pie they also came to other areas. The distribution of an average user's whole set of clicks in an interface further showed that though the top three positions (i.e., the area 1) and the last three positions (i.e., the area 4) are commonly popular in the three layouts, the clicks are more evenly distributed in pie among all areas at its first page. Grid and pie are even more active in stimulating users to click items in the next recommendation page. From subjective measures and user comments, we found that users did prefer using pie and grid to list. Moreover, pie has been demonstrated with significant benefits in increasing users' decision confidence, perceived interface competence, enjoyability, and usage intention.

For our future work, we will conduct more user studies, including eye-tracking experiments, to track users' eye-movement behavior in the recommender interfaces. Another interesting topic will be to investigate the interaction effect from items' relevance ordering with the layout. That is, when the ordering was changed (i.e., reversed ascending order instead of regular descending order), would users' behavior be influenced or not? In fact, with the varied ordering condition, we are able to identify whether users would spontaneously evaluate the item's relevance, or their selection behavior would be largely influenced by the layout. For example, in the list interface, would they still select items at the top though they are least relevant? The relative role of layout against the relevance ordering could be hence revealed.

6. REFERENCES

[1] Braganza, C., Marriott, K., Moulder, P., Wybrow, M. and Dwyer, T. Scrolling behaviour with single- and multi-column layout. In *Proc. WWW 2009*, 831-840.

[2] Callahan, J., Hopkins, D., Weiser, M. and Shneiderman, B. An empirical comparison of pie vs. linear menus. In *Proc. CHI 1988*, ACM, 95-100.

[3] Card, S. K., Newell, A. and Moran, T. P. *The Psychology of Human-Computer Interaction*. L. Erlbaum Assoc. Inc., Hillsdale, NJ, USA, 1983.

[4] Chen, L. and Pu, P. Evaluating critiquing-based recommender agents. In *Proc. AAAI 2006*, 157-162, 2006.

[5] Chen, L. and Pu, P. Eye-Tracking Study of User Behavior in Recommender Interfaces. In *Proceedings of 2010 International Conference on User Modeling, Adaptation and Personalization (UMAP'10)*, 375-380, Big Island, Hawaii, USA, June 20-24, 2010.

[6] Halverson, T. and Hornof, A. J. A minimal model for predicting visual search in human-computer interaction. In *Proc. CHI 2007*, 431-434.

[7] Kammerer, Y. and Gerjets, P. How the interface design influences users' spontaneous trustworthiness evaluations of web search results: comparing a list and a grid interface. In *Proc. ETRA 2010*, 299-306.

[8] Pu, P and Chen, L. Trust-Inspiring Interfaces for Recommender Systems. *Journal of Knowledge-Based Systems (KBS)*, vol. 20 (6), 542-556, 2007.

[9] Pu, P and Chen, L. A user-centric evaluation framework of recommender systems. In *Proc. RecSys'10 Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI'10)*, 14-21, 2010.

[10] Ricci, F., Rokach, L., Shapira, B., and Kantor, P.B. (Eds.) *Recommender System Handbook*. Springer, 2011.

[11] Tsoi, H. K. and Chen, L. Incremental tag-aware user profile building to augment item recommendations. In *2nd Workshop on Social Recommender Systems (SRS'11) in CSCW'11,* 2011.

Visualizable and explicable recommendations obtained from price estimation functions

Claudia Becerra and
Fabio Gonzalez
Intelligent Systems Research Laboratory (LISI)
Universidad Nacional de Colombia, Bogota
[cjbecerrac,fagonzalezo]@unal.edu.co

Alexander Gelbukh
Center for Computing Research (CIC)
National Polytechnic Institute (IPN)
Mexico DF, 07738, Mexico
gelbukh@gelbukh.com

ABSTRACT

Collaborative filtering is one of the most common approaches in many current recommender systems. However, historical data and customer profiles, necessary for this approach, are not always available. Similarly, new products are constantly launched to the market lacking historical information. We propose a new method to deal with these "cold start" scenarios, designing price-estimation functions used for making recommendations based on cost-benefit analysis. Experimental results, using a data set of 836 laptop descriptions, showed that such price-estimation functions can be learned from data. Besides, they can also be used to formulate interpretable recommendations that explain to users how product features determine its price. Finally a 2D visualization of the proposed recommender system was provided.

Categories and Subject Descriptors

H.1.2 [**User/Machine Systems**]: Human information processing; H.4.2 [**Types of Systems**]: Decision support

General Terms

Experimentation

Keywords

Apriori recommendation, Cold-start recommendation, Price estimation functions

1. INTRODUCTION

The internet and e-commerce grow exponentially. As a result, decision-making process about products and services is becoming increasingly complex. These processes involve hundreds and even thousands of choices and a growing number of heterogeneous features for each product. This is mainly due to the introduction and constant evolution of new markets, technologies and products.

Unfortunately, human capacity for decision-making is too limited to address the complexity of this scenario. Studies in psychology field have shown that human cognitive capacities are limited from five to nine alternatives for simultaneous comparison [17, 14]. Consequently, making a purchasing decision at an e-commerce store that does not provide tools to assist decision-making, is a task that largely overwhelms human capacities. Moreover, several studies have shown that this problem generates adverse effects on people such as: regret due to the selected option, dissatisfaction due to poor justification for the decision, uncertainty about the idea of "best option", and overload of time, attention and memory (see [20]).

Many recommender systems approaches have addressed this problem through collaborative filtering [6] based on product content (i.e. descriptions) and on customer information [2, 9]. This approach recommends products similar to those chosen by similar users. On the other hand, latent semantics approaches [10] have been successfully used to build affinity measures between products and users. Most of the aforementioned approaches have been applied in domains with products such as books and movies that remain available long enough to collect enough historical data to build a model [4, 12].

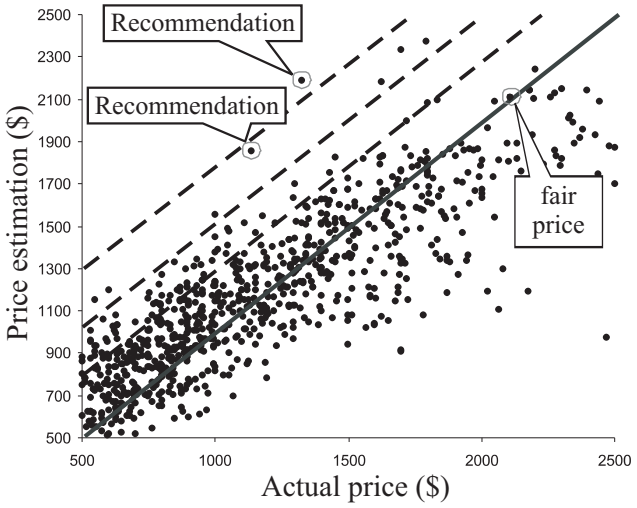
While impressive progresses have been made in the field using collaborative filtering, the relevance of current approaches in domains with frequent changes in products is still an open question [8]. For example, customer-electronics domain is characterized by products with a very short life cycle in the market and a constant renewal of technologies and paradigms. Collaborative approaches face two major problems in this scenario [13]. First, product features are constantly redefined, making difficult for users to identify relevant attributes. Second, historical sales product data become obsolete very quickly due to the frequent product substitution. This problem of making automatic recommendations without historical data is known as *cold-start recommendation* [18].

In this paper, we propose a new cold-start method based on an estimate of the benefit to the user when purchasing a product. This function is formulated as the difference between estimated and real prices. Therefore, our approach recommends products with high benefit-cost ratio to find "best-deals" on a data set of products. Figure 1 shows an example of such recommendations based on utility functions displaying 900 laptop computers. In this figure, the features of laptops below the line in bold, indicating fair prices, do not justify prices of laptops.

The rest of the paper is organized as follows. In Section 2, the necesary background and proposed method are presented. In Section 3, an evaluation methodology and some data refinements are proposed and applied to the model. Finally, in Section 4, some concluding remarks are briefly discussed.

RecSys'11, October 23–27, 2011, Chicago, Illinois, USA.

Figure 1: Graphic recommender based on price estimates



2. APRIORI RECOMMENDATIONS USING UTILITY FUNCTIONS

The general intuition of method is led by the lexicographical criterion [22]. That is, users prefer products that offer more value for their money. Clearly, this approach is not applicable to all circumstances, but it is general enough when customer profiles are not available in cold-start scenarios.

When a user purchases a product x_i, a utility function $utility(x_i)$ provides an estimation of the difference between the estimated price $f(x_i)$ and the market price y_i, that is $utility(x_i) = f(x_i) - y_i$. Thus, the products in the market are represented as a set $X = \{x_1, x_2, ..., x_n, ..., x_N\}$, where each product x_i is a vector characterized in a feature space $\mathbb{R}^M$. With these data, a regression model, learned from X and the vector of prices y, generates price estimations $f(x_i)$ required for calculation of the utility. Finally, the utility function is computed on all products thus providing an ordered list with the top-n apriori recommendations.

Estimates of price $f(x_i)$ can be obtained by a linear-regression model as:

$$f(x_i) = \beta_o + \sum_{m \in \{1,...,M\}} \beta_m x_{im}. \quad (1)$$

This model is equivalent to an additive value function used in the decision-making model *SAW* (simple additive weighting) [5], but with coefficients β_m learned automatically. Clearly, the recommendations obtained from these estimates can be explained to users, since each term $\beta_m x_{im}$ represents the money contribution to the final price estimate provided by the m-th feature of the i-th product.

The quality of the apriori recommendations obtained with the proposed method depends primarily on three factors: the amount of training data, the accuracy of price estimates $f(x_i)$, and the ability to extract user-understandable explanations from the regression model. Certainly, linear models, such as that of eq. 1, offer good interpretability, but in many cases, these models generate high rates of error in their predictions when the interactions among features are complex. These models are known as *weak regression models*. On the other hand, discriminative approaches, such as *support vector regression* [19], provide better models with lower error rates but also with lower interpretability.

This trade-off can be overcome with a hybrid regression model as *3FML* (three-function meta-learner) [3]. This meta-regressor combines two different regression methods in a new improved combined model in a way similar to other meta-algorithms such as *voting*, *bagging* and *AdaBoost* (see [1]). Unlike these methods, 3FML uses one regression method to make price predictions and another to predict the error. As long as the former regression method is weak, stable and interpretable, the latter can be any other regression method regardless its interpretability. As a result, the combined regression preserves the same interpretability level of the first regressor but with lower error rate.

A linear regression model can be trained to learn parameters β_m by minimizing the least squared error from data [15]. This first model can be used by 3FML to build a base regression model $f_0(x)$ with the full dataset. Then, this model is used to divide the data into two additional groups depending on whether the obtained price predictions were below or above the training price, given a difference threshold θ. Next, using the same base-regression method, two additional models $f_{+1}(x)$ and $f_{-1}(x)$ are trained with the pair of subsets called respectively, *upper model* and *lower model*. Figure 3 illustrates *upper*, *base* and *lower* models compared to the target function, which is the price in a data set of laptop computers. The three resulting models are combined using an aggregation mechanism – called *mixture of experts* [11] – with the following expression:

$$\hat{f}(x_i) = \sum_{l \epsilon \mathbb{H}} w_l(x_i) f_l(x_i), \quad (2)$$

having

$$\sum_{l \epsilon \mathbb{H}} w_l(x_i) = 1, i \in \{1 \dots n\} \quad (3)$$

$\mathbb{H}$ is a dictionary of experts consisting on the base model and two additional specialized models, $\mathbb{H} = \{f_{-1}(x), f_0(x), f_{+1}(x)\}$. The gating coefficient w_{li} establishes the level of relevance of the l model into the final price prediction for the i-th product.

In 3FML model, coefficients $w_{li} = w_l(x_i)$ are obtained by chaining a membership function w_l for each regression model to a function α that depends on the errors of the three models, $w_l(x_i) = w_l(\alpha(f_{-1}(x_i), f_0(x_i), f_{+1}(x_i), y_i))$. These membership functions $w_l(\alpha)$ are similar to those used in fuzzy sets [23] but these satisfy the constraint given by eq. 3. Three examples of those functions are shown in Figure 2; one triangular and two Gaussian. Clearly, the range of the error function α must agree with the domain of the membership functions. For instance, if the domain of the membership functions is $[0, 1]$, an appropriate function α_i must return a value close to 0.5 when y_i is better modeled by $f_0(x_i)$. Similarly, reasonable values for α_i, if y_i is better modeled by $f_{-1}(x_i)$ or $f_{+1}(x_i)$, are respectively 0.0 and 1.0.

Such function α can be arithmetically constructed (see [3] for triangular and Gaussian cases) and α_i can be obtained for every x_i. 3FML makes use of a second regression method to learn a function for α_i. This function is called α-learner, which seeks to predict the same target y_i but indirectly through the errors obtained by f_{-1}, f_0 and f_{+1}. The estimates obtained with α-learner are used in com-

Figure 2: Triangular and Gaussian membership functions

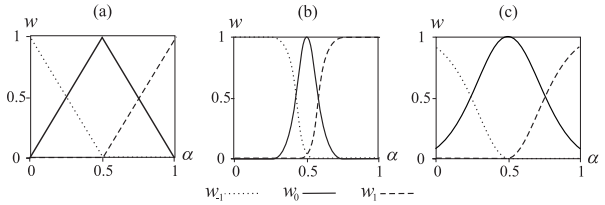
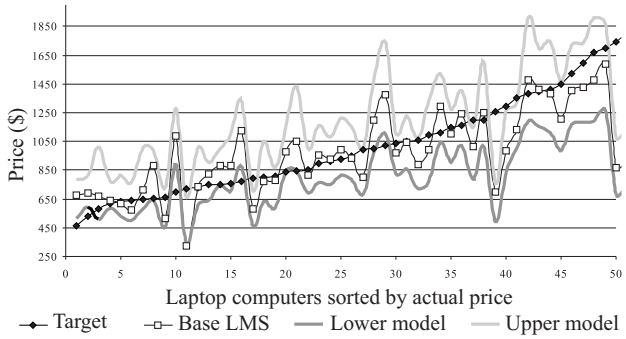


Figure 3: 3FML's three regression models graph



bination with the membership functions to get coefficients $w_l(x_i)$. Therefore, final predictions are obtained with a different linear model for each target price y_i. The resulting model is also linear, but different for each product instance in function to x_i:

$$\hat{f}(x_i) = \hat{\beta}_0(x_i) + \sum_{m \epsilon \{1,...,M\}} \hat{\beta}_m(x_i) x_{im}, \qquad (4)$$

where

$$\hat{\beta}_o(x_i) = \sum_{l \in \mathbb{H}} \beta_{lo} w_l(x_i) \; ; \; \hat{\beta}_m(x_i) = \sum_{l \in \mathbb{H}} \beta_{lm} w_l(x_i).$$

Clearly, the model in eq. 4 is as user-explainable as that of eq. 1.

The effect of α-learner in eq. 4 is that the entire data set is clustered into three latent classes. These classes can be considered as market segments namely: high-end, mid-range and low-end products. Many commercial markets exhibit this segmentation, e.g. computers, mobile phones, cars, etc.

3. EXPERIMENTAL VALIDATION

The aim of experiments is to build a model that provides a cost-benefit ranking of a set of products where each product is represented as a vector of features. To assess the quality of this ranking, two factors are observed. First, the error of the price-estimation regression should be low to make sure that this function provides a reasonable explanation of the data set. Second, the model must be interpretable and discovered knowledge must be consistent with market data. For example, if a proposed model discovers a ranking of how much money each operating system contributes to laptop prices, this ranking should be in agreement the prices of retail versions of the same operating systems.

In addition, the full features set of the top-10 recommended products is provided along with a 2D visualization of the entire data set. These resources allow the reader – guided by a brief discussion – to qualitatively evaluate the recommendations obtained with the proposed method.

Table 1: Attributes in *Laptops_17_836* data set

Feature name	Type	% missing
Manufacturer	Nominal	0.00%
Processor Speed	Numeric	0.40%
Installed Memory	Numeric	1.90%
Operating System	Nominal	0.00%
Processor	Nominal	0.20%
Memory Technology	Nominal	7.20%
Max Horizontal Resolution	Numeric	7.90%
Warranty-Days	Numeric	15.50%
Infrared	Nominal	0.00%
Bluetooth	Nominal	0.00%
Docking Station	Nominal	0.00%
Port Replicator	Nominal	0.00%
Fingerprint	Nominal	0.00%
Subwoofer	Nominal	0.00%
External Battery	Nominal	0.00%
CDMA	Nominal	0.00%
Price	Numeric	0.00%

3.1 Data

The data is a set of 836 laptop computers each represented by a vector of 69 attributes including price, which is the attribute to be estimated. Data were collected by Becerra[1] from several U.S. e-commerce sites (e.g. Pricegrabber, Cnet, Yahoo, etc.), during the second half of 2007 within a month. A subset of 17 features was selected using the correlation-based selection method proposed by Hall [7]. We call this dataset *Laptops_17_836*; all its features and percentage of missing values are shown in Table 1.

3.2 Price estimation results

For the construction of the price-estimation function, several regression methods were used, namely: least mean squares linear regression (LMS) [15], M5P regression tree [21, 16], support vector regression (SVR) [19] and three-function meta-learner (3FML, described in previous section). 3FML provides three interpretable linear models: *upper*, *base* and *lower* models, which can be associated with product classes. Finally, estimated price for each laptop was obtained with the combination of these three models using eq. 4 with the weights obtained from α-Learner and Gaussian membership functions.

The performance of each method was measured using root-mean-square error (RMSE) defined as:

$$RMSE = \sqrt{\frac{\sum_i \left(\hat{f}(x_i) - y_i\right)^2}{|X|}}.$$

The data set was randomly divided into 75% for training and 25% for testing. Ten different runs of this partition ratio were used for each method. These ten RMSE results were averaged and reported. Table 2 shows the results, their standard deviation (in parentheses) and some model parameters.

[1]http://unal.academia.edu/claudiabecerra/teaching

The method with lowest RMSE was SVR with a complexity parameter $C = 100$ using radial basis functions (RBF) as kernel. However, interpretability of this model is quite limited, given the embedded feature space induced by the kernel. On the other hand, LMS and 3FML provide straightforward interpretation of β coefficients, which represent the amount of the contribution of each feature to the product estimated price. Clearly, 3FML was the method that better coped with this interpretability-accuracy trade-off.

Table 2: 10 runs average RMSE results for price estimates obtained with several regression methods

Regression model	Avg. RMSE
M5P regression tree	239.70(21.57)
Least Mean Squares (LMS)	259.87(17.90)
ε-SVR, $C = 100$, linear kernel	258.93(16.93)
3FML (LMS as base model)	233.48(14.76)
3FML (ε-SVR, $C = 100$, linear kernel)	**223.76(8.57)**
ε-SVR, $C = 100$, RBF kernel $\sigma = 7.07$	230.23(12.27)

3.3 Evaluation and feedback

In this section the price estimation function obtained using 3FML is manually analyzed checking coherence of β coefficients with real facts of the market. Particularly, coefficients for attributes *operating system*, *processor* and numerical features are reviewed, and – when necessary – some refinements are proposed to the data sets to deal with discussed issues.

3.3.1 Operating System *attribute analysis*

Table 3 shows the distribution of the different operating systems into the entire data set of laptops and the abbreviations that we use to refer them at Table 5 and Table 4.

In order to evaluate the portion of the price estimation model related to *operating system* (OS) attribute, coefficients of this feature are compared with related Microsoft's retail prices. Table 4 shows public retail prices for *Windows Vista*™ published at 2007-3Q. In spite that at that date, *Windows Vista*™ operating system had already six months of launched, many brand new laptops still had pre-installed previous *Windows XP*™. Thus, we consider for analysis *Windows XP Pro*™ equivalent to *Windows Vista Business*™, as well as, *Windows XP*™ equivalent to *Windows Vista Home Premium*™. This assumption is also coherent with the observed behavior in Microsoft's price policy that keeps prices of previous product releases invariable during version transition periods.

It is interesting to highlight the behavior of 3FML model with *Windows Vista Ultimate*™. Although this OS version occurs only at 1.32% of instances (see Table 3), it is correctly recognized as the most expensive OS (see Table 4) by the upper model. This fact corrects an erroneous tendency recognized by base and lower models. In general terms, for other OS versions, 3FML managed to predict similar ordering as that of retail prices.

3.3.2 Processor *attribute coefficients*

As shown in Table 1, *Laptops_17_836* data set has two features to describe the main processors of laptops , they are: *Processor Speed* (numeric) and *Processor* (nominal). The former is the processor clock rate and the latter is a text string that contains — in most of cases – the manufacturer, the product family and the model (e.g. "Intel Core 2 Duo Mobile T7200"). Unlike OS attribute, which has only seven possible alternatives, *Processor* attribute has 133 possible processor models. Moreover, the frequencies of occurrence of each processor model exhibit a Zipf-type distribution (see Figure 4). Thus, approximately half of the 836 laptops have only 8 different processors and more than 80 processors occur only in one laptop. Part of this sparseness is due to missing information, abbreviations and formatting.

Table 3: Proportions of operating systems ocurrences in *Laptops_17_836* data set

Operating System	#	%
Vista Home Premium™ (WinVHP)	251	30.02%
WinXP Pro™ (WinXPP)	208	24.88%
WinXP™ (WinXP)	151	18.06%
Win. Vista Business™ (WinVB)	137	16.39%
Win. Vista Home Basic™ (WinVHB)	44	5.26%
Mac OS™ (MacOS)	34	4.07%
Win. Vista Ultimate™ (WinVU)	11	1.32%
Total	836	100%

Table 4: Retail prices for different editions of *Windows Vista*™

O.S.→	WinVHB	WinVHP	WinVB	WinVU
Retail price*	$199.95	$259.95	$299.95	$319.95

*http://www.microsoft.com/windows/windows-vista/compare-editions (site consulted in September 2007)

The *Processor* attribute, as found in the data set, can generate a detrimental effect on the price-estimation function. Besides, β coefficients could hardly be explained and their evaluation against market facts could lead to misleading results. Thus, the model was withdrawn from *Processor* attribute and it was renamed as *Proc. Family*. In addition, the data set was enriched manually adding the following four processor related attributes:

- *L2-Cache*: processor cache in Kibibytes (2^{10} bytes).
- *Hyper Transport*: frontal bus clock rate in Mhz.
- *Thermal Design*: maximum dissipated power in watt.
- *Process Technology*: CMOS technology in nanometre.

This new data set is referred as *Laptops_21_836* data set. Performance results of new price-estimation functions are shown in Table 6. Clearly, SVR and 3FML obtained substantial improvements using this new data set.

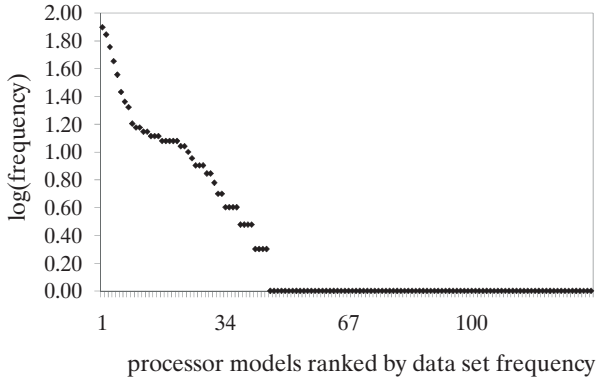
Similarly to the analysis made for OS attribute, processors families also have a consumer-value ranking given by their technology, which can be compared to a ranking taken from an interpretable price-estimation function. The technology ranking of Intel processors is: (1st) *Core 2 Duo*™, *Core Duo*™, *Core Solo*™, *Pentium Dual Core*™ and *Celeron*™. Same for AMD's processors: (1st) *Turion*™, *Athlon*™ and *Sempron*™ [2]. We extracted a ordering for processor fami-

[2]see http://www.notebookcheck.net/Notebook-Processors.129.0.html for a short description of mobile processor families (site consulted in June 2011)

Table 5: 3FML base, upper and lower model coefficients $\beta_{s.o}$ for *operating system* attribute

Base model		Upper model		Lower model	
S.O.	$\beta_{s.o}$	S.O.	$\beta_{s.o}$	S.O.	$\beta_{s.o}$
WinVU	323.3	WinVB	185.6	WinVB	127.3
WinVB	260.3	WinXPP	184.5	WinXPP	127
WinXPP	249.8	MacOS	169.2	MacOS	95.2
MacOS	245.9	WinVU	96.4	WinVHP	24.7
WinVHP	116.7	WinVHP	57	WinVU	0.0
WinXP	94.3	WinXP	27.8	WinVHB	0.0
WinVHB	0.0	WinVHB	0.0	WinXP	-7.1

Figure 4: Distribution of *Processor* attribute



lies by their corresponding β coefficients from 3FML models. Results for this ranking – means and standard deviation – making 10 runs with different samples of 75% training and 25% test are shown in Table 7.

Results in Table 7 show how *upper model* better ordered processor families with high technological ranking. Similarly, *lower model* does a similar work recognizing *Sempron*™ family at the lowest rank.

3.3.3 Numerical attributes coefficients

This subsection present a brief discussion on the interpretation of β coefficients extracted from the price-estimation function for some numeric attributes (shown in Table 8). Although this interpretation is clearly subjective, it reveals some laptop-market facts, which were extracted in an unsupervised way from the data.

For instance, consider *Thermal Design* attribute. Negative values in the β coefficients reveal a fact: the lesser power the CPU dissipates, the higher the laptop's price.

Table 6: RMSE for regression price estimates in *Laptops_21_836* data set

Regression model	RMSE
ε-SVR ($C = 1$, lineal kernel)	254.56(11.75)
ε-SVR ($C = 100$, RBF kernel,)	219.16(9.88)
3FML*	220.91(10.97)

* Base model: ε-SVR, $C = 1$, lineal kernel. α-Learner: ε-SVR, $C = 100$, RBF kernel.

Table 7: Processor families rankings obtained from 3FML price-estimation function

Upper model		Lower model	
Intel Core2 Duo	7.4(0.8)	Intel Core2 Duo	7.6(0.5)
Intel Core Duo	7.2(1.2)	Intel Core Solo	6.2(1.7)
Intel Core Solo	5.3(2.1)	AMD Athlon	5.7(3.8)
Intel Celeron	5.1(1.7)	Intel Core Duo	4.8(2.1)
PowerPC	4.6(2.6)	AMD Turion	4.8(1.8)
Pent DualCore	3.7(1.4)	Pent DualCore	4.3(2.1)
AMD Sempron	3.4(2.4)	PowerPC	4.3(3.1)
AMD Turion	3.3(1.8)	Intel Celeron	3.3(1.3)
AMD Athlon	1.8(1.4)	AMD Sempron	2.8(2.3)

Base model	
Intel Core Solo	8.5(0.7)
Intel Core2 Duo	8.3(0.7)
Intel Core Duo	6.8(0.9)
Pent DualCore	5.1(1.4)
Intel Celeron	5.0(0.7)
AMD Turion	3.7(1.1)
PowerPC	2.9(2.1)
AMD Sempron	2.6(1.8)
AMD Athlon	2.1(1.3)

Table 8: β coefficients for numerical attributes from 3FML model with *Laptops_21_836* data set

Feature name	Upper	Base	Lower
β_0	0.23	0.12	0.06
Warranty Days	0.04	0.01	-0.01
Installed Memory	-0.11	0.17	0.10
Max. Horizontal Resolution	0.12	0.37	0.15
Processor Tech.	0.30	0.08	0.05
Thermal Desing	-0.01	-0.37	-0.27
Hyper Transport	-0.02	0.25	0.05
L2-Cache	0.08	0.16	0.11
Processor Speed	-0.03	0.25	0.17

Besides, these coefficients also shows that this effect affects prices more at mid-range and low-end laptop-market segments. Similarly, *Max. Horizontal Resolution* attribute reveals that this feature has greater impact on the mid-range laptop market prices.

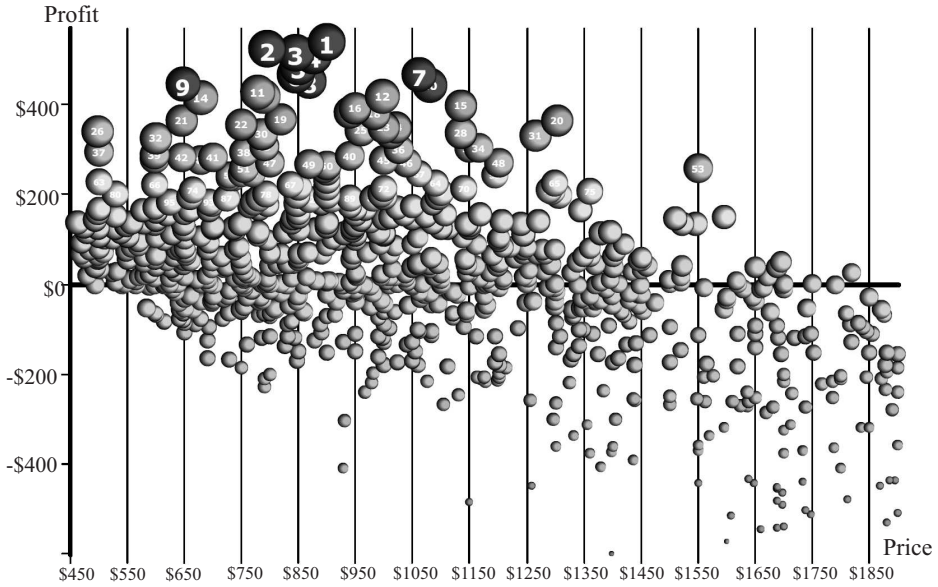
Interestingly, there is a phenomenon revealed by the features that are easy perceived by users, such as *Installed Memory*, *Max. Horizontal Res.* (number of horizontal pixels on screen), *L2-Cache* and *Processor Speed*. That is: those features have considerably less effect on prices in high-end than in mid-range and low-end market segments. This phenomenon can be explained by the fact that "luxury" goods justify their price more by attributes such as brand-label, exclusive features and physical appearance rather than for their configuration.

3.4 Recommendations for users

3.4.1 Top-10 recommendations

After the quantitative evaluation (i.e. regression error) and qualitative assessment (i.e. agreement with market facts) using 3FML model, the resulting functions provided reasonable estimates of price and support elements to explainable

Figure 5: Visualization of 836 laptops recommendation ranking



recommendations such as rankings and weights of attributes. After obtaining the estimates of prices, the profit for each laptop is calculated from the difference between this estimate and real price. Table 9 shows the top-10 recommendations with the highest profit among all 836 laptops.

The first and second top-ranked laptops have similar configurations, but even small differences make comparison difficult at first sight. The second laptop has better price, more memory, docking station and ports replicator slots. Unlike, the former has higher screen resolution and a fingerprint sensor. These differences can be compared quantitatively with the help of β coefficients provided by the model. However, a better explanation of the #1 recommended choice is a market fact extracted from the obtained manufacturer ranking showed in Table 10. The three regression models identify the *Lenovo*™ brand better ranked than *HP*™. Therefore, the first recommended laptop becomes a "best deal" given the standard prices of Lenovo at the time. Similarly, recommendations #7, #9 and #10 seem to get their high user profit not because of their configuration features, but because of their label *Sony*™, which do better positions on the ranking of manufacturers than its counterparts.

Second and third recommendations only differ in *Processor Speed* attribute. Clearly, the estimated cost of that differentia is the numerical difference between their estimated prices, which is $42. Nevertheless, their real price difference is $50. This explains the order of position in the ranking assigned by the recommender system to the #2 and #3 recommendations. More pair-wise comparisons and evaluations could be made but are omitted due to space limitations.

These paired comparisons become cognitively more difficult when the number of features, differences and instances increases. However, the proposed recommender method provides reasonable explanations no matter how much data is involved, and these can be provided by user request. This is important because cold-start recommender systems need to establish trust in users due of the lack of collaborative support.

3.4.2 2D Visualization

Ordered lists are the most common format to present recommendations to users. However, despite having such an ordination, establish the most appropriated choice for a particular user is a difficult task. Therefore, we propose a novel visualization method for our recommender system. The purpose of this is to enable users to build a mental model of the market. When users do not have a clear aim or a defined budget, this tool provides a rough idea of the number of options and prices. In addition, visualization can help the short-term memory decreasing cognitive load and highlight the recommended options.

The proposed 2D visualization is shown in Figure 5. The horizontal axe represents actual price and the vertical axe represents the profit, which is the difference between the estimated and actual price. Each laptop is represented as a bubble, where larger radius and warmer colors (darker gray in the grayscale version) means higher profit-price differences. Besides, the number of ranking was included in the top-99 recommendations.

This visualization highlights other "best deals" that are hidden in the ranking list. For instance, consider recommendation #53 (see Figure 5 in the coordinates $1550 price and $260 profit). Perhaps this is an interesting option to consider if user's budget is over $1500. Similarly, recommendation #26 can be quickly identified as the best option for buyer on a low budget.

The proposed visualization also allows a qualitative assessment of the price-estimation function. For instance, consider the laptops above $1300, this function has difficulties to predict prices using the current set of features, which in turn appears to be very effective for mid-range prices. This problem could be solved indentifying and adding to the set of attributes those distinctive features of high-end laptops, namely: shockproof devices, special designs, colors, housing materials, exclusive options, etc.

Table 9: Detailed top-10 ranked recomendations

Recommendation rank →	#1	#2	#3	#4	#5
Horizontal Resol.	900 pixels	800 pixels	800 pixels	1536 pixels	768 pixels
Memory Tech.	DDR2	DDR2	DDR2	DDR2	DDR2
Inst. Memory	512 MB	1024 MB	1024 MB	1024 MB	1024 MB
Family	Core Duo	Core Duo	Core Duo	Core2 Duo	Core2 Duo
Processor Speed	1830 GHz	1830 GHz	2000 GHz	1500 GHz	2000 GHz
L2 Cache	?*	?	?	2048 kB	4096 kB
Hyper Transp	?	?	?	667 Mhz	667 Mhz
Thermal Design	?	?	?	35	34
Process Tech.	?	?	?	65nm	65nm
Manufacturer	Lenovo	HP	HP	Lenovo	HP
Op. System	WinXPP	WinXPP	WinXPP	WinVB	WinXPP
Warranty Days	1095	1095	1095	365 W	1095 W
IBDPFWC**	YNYYYNN	YYYYYNN	YYYYYNN	NNYYNNN	YYYYYNN
Actual Price	$ 899	$ 795	$ 845	$ 875	$ 849
Estimated Price	$ 1,438	$ 1,319	$ 1,361	$ 1,383	$ 1,332
Profit	$ 539	$ 524	$ 516	$ 508	$ 483

Recommendation rank →	#6	#7	#8	#9	#10
Horizontal Resol.	1050 pixels	800 pixels	800 pixels	800 pixels	800
Memory Tech.	DDR2	DDR2	DDR2	DDR2	DDR2
Inst. Memory	1024 MB	1024 MB	1024 MB	512 MB	1024 MB
Family	Core Duo	Core2 Duo	Core Duo	Core Duo	Core2 Duo
Processor Speed	2000 GHz	2160 GHz	1830 GHz	1660 GHz	2000 GHz
L2 Cache	2048 kB	4096 kB	?*	2048 kB	4096 kB
Hyper Transp	667 Mhz	667 Mhz	?	667 Mhz	800 Mhz
Thermal Design	31 W	34 W	?	31 W	35 W
Process Tech.	65nm	65nm	?	65nm	65nm
Manufacturer	Lenovo	Sony	HP	Sony	Sony
Op. System	WinXP_Pro	WinXP_Pro	WinXP_Pro	WinXP_Pro	V_Business
Warranty Days	365	365	1095	365	365
I BDPFWC**	NYNNYNN	NYYYNNN	YYYYYNN	NNYYNNNN	NYYYYNN
Actual Price	$ 845	$ 1,060	$ 868	$ 649	$ 1,080
Estimated Price	$ 1,312	$ 1,526	$ 1,319	$ 1,093	$ 1,522
Profit	$ 467	$ 466	$ 451	$ 444	$ 442

* Question mark stands for missing values.
** Initials I B D P F W C stand for *Infrared*, *Bluetooth*, *Docking Station*, *Port Replicator*, *Fingerprint*, *Subwoofer* and *CDMA*.

Table 10: Average ranking of *Manufacturer* attribute using 3FML at *Laptops_21_836* data set

Base model		Upper model		Lower model	
Asus	10.6(0.7)	Dell	10.0(0.9)	Asus	10.2(0.8)
Sony	9.6(1.2)	Fujitsu	9.2(1.0)	Fujitsu	9.6(1.5)
Fujitsu	8.4(1.4)	Sony	7.4(1.8)	Sony	8.1(1.2)
Dell	7.9(0.9)	Asus	6.8(1.8)	Dell	7.3(1.4)
Apple	7.0(2.4)	Apple	6.1(1.3)	Apple	6.7(2.0)
Lenovo	6.5(1.6)	Lenovo	4.1(2.4)	Lenovo	5.2(1.5)
Toshiba	5.0(1.2)	Gateway	4.0(2.9)	Toshiba	4.7(1.5)
Acer	4.7(1.2)	Averatec	3.8(1.3)	Acer	3.9(1.3)
HP	2.3(0.7)	Acer	3.6(1.3)	HP	3.4(1.7)
Averatec	2.1(1.3)	HP	3.3(1.9)	Gateway	1.9(0.8)
Gateway	1.9(1.0)	Toshiba	2.4(1.5)	Averatec	1.9(2.1)

4. CONCLUSIONS

We presented a novel product recommender system based on an interpretable price-estimation function, which estimates the economic benefit for the customer to buy a product in a particular market. Accurate and interpretable price estimations were obtained using the 3FML (three-function meta-learner) method. This regression method allows the combination of an interpretable regressor (e.g. LMS) to estimate prices and an uninterpretable regressor (e.g. SVR) to identify the latent class of each product. The combined model obtained better price estimates than LMS, SVR and M5P regression tree, while it kept a high level of interpretation.

The proposed method was tested with real-market data from a data set of laptops. The obtained price-estimation model was interpretable, allowing evaluation and refinement by domain experts and ensuring that price estimates are a coherent consequence of the product features. In addition, the obtained recommendations are easy to understand by users. For instance, feature rankings (e.g. ranking of CPU) and feature price contributions (e.g. cost per GB of main memory) are provided. Importantly, while the price estimates are obtained in a supervised way, other domain knowledge is extracted in a non-supervised way. Although the proposed method was tested in a particular domain (i.e. laptops), this same process can be applied to other domains that exhibit similar number of options and features.

Moreover, a user-friendly visualization method for recommendations was proposed using a 2D Cartesian metaphor and concrete variables such as cost and profit. This visualization allows users to make a quick mental map of a large market to explore and identify recommendations in different price ranges.

In conclusion, the proposed method is flexible and can be useful in e-commerce scenarios with products that allow the construction of price-estimation functions, such as customer-electronics products and others. Finally, our method fills a gap where recommender systems based on historical information fail because of the lack of such information.

5. ACKNOWLEDGEMENTS

This research is funded in part by the Bogota Research Division (DIB) at the National University of Colombia, and throught a grant from Colciencias, project 110152128465.

6. REFERENCES

[1] E. Alpaydin. *Introduction to Machine Learning*. The MIT Press, October 2004.

[2] M. Balabanovic and Y. Shoham. Fab: Content-based collaborative recommendation. *Communications of the Association for Computing Machinery*, 40(3):66–72, 1997.

[3] C. Becerra and F. Gonzalez. 3-functions meta-learner algorithm: a mixture of experts technique to improve regression models. In *DMIN08: Proceedings of the 4th international conference on data mining*, Las Vegas, NV, USA., 2008.

[4] J. Bennet and S. Lanning. The netflix prize. In *KDD Cup and Workshop*, 2007.

[5] R. T. Eckenrode. Weighting multiple criteria. *Management Sciences*, 12:180–192, 1965.

[6] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992.

[7] M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *ICML '00: Proceedings of the 17th International Conference on Machine Learning*, pages 359–366, San Francisco, CA, USA, 2000.

[8] E. Han and G. Karypis. Feature-based recommendation system. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 446–452, 2005.

[9] Borchers A. Riedl J. Herlocker J. L., Konstan J. A. An algorithmic framework for performing collaborative filtering. In *Proc. 22nd ACM SIGIR Conference on Information Retrieval*, pages 230–237, 1999.

[10] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. on Information Systems*, pages 89–115, 2004.

[11] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Comput.*, 3(1):79–87, 1991.

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

[13] M. Mandl, A. Felfernig, E. Teppan, and M. Schubert. Consumer decision making in knowledge-based recommendation. *Journal of Intelligent Information Systems*, 2010.

[14] G. A. Miller. The magical number seven: Plus or minus two: Some limits on our capability for processing information. *Psychological Review*, 63(2):81–97, 1956.

[15] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to linear regression analysis*. Wiley Interscience, 2006.

[16] R. J. Quinlan. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348, 1992.

[17] T. L. Saaty and M. S. Ozdemir. Why the magic number seven plus or minus two. *Mathematical and Computer Modelling*, 38(3):233–244, 2003.

[18] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR*, 2002.

[19] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[20] B. Schwartz. The tyranny of choice. *Scientific American*, pages 71–75, April 2004.

[21] Y. Wang and I. H. Witten. Induction of model trees for predicting continuous classes. In *Poster papers of the 9th European Conference on Machine Learning*. Springer, 1997.

[22] K. Yoon and C. Hwang. Multiple attribute decision making: An introduction. *Sage University papers*, 7(104), 1995.

[23] L. Zadeh. *Fuzzy Sets, Fuzzy Logic and Fuzzy Systems*. World Scientific, 1996.

Recommender Systems, Consumer Preferences, and Anchoring Effects

Gediminas Adomavicius
University of Minnesota
Minneapolis, MN
gedas@umn.edu

Jesse Bockstedt
George Mason University
Fairfax, VA
jbockste@gmu.edu

Shawn Curley
University of Minnesota
Minneapolis, MN
curley@umn.edu

Jingjing Zhang
University of Minnesota
Minneapolis, MN
jingjing@umn.edu

ABSTRACT

Recommender systems are becoming a salient part of many e-commerce websites. Much research has focused on advancing recommendation technologies to improve the accuracy of predictions, while behavioral aspects of using recommender systems are often overlooked. In this study, we explore how consumer preferences at the time of consumption are impacted by predictions generated by recommender systems. We conducted three controlled laboratory experiments to explore the effects of system recommendations on preferences. Studies 1 and 2 investigated user preferences for television programs, which were surveyed immediately following program viewing. Study 3 broadened to an additional context—preferences for jokes. Results provide strong evidence viewers' preferences are malleable and can be significantly influenced by ratings provided by recommender systems. Additionally, the effects of pure number-based anchoring can be separated from the effects of the perceived reliability of a recommender system. Finally, the effect of anchoring is roughly continuous, operating over a range of perturbations of the system.

1. INTRODUCTION

Recommender systems have become important decision aids in the electronic marketplace and an integral part of the business models of many firms. Such systems provide suggestions to consumers of products in which they may be interested and allow firms to leverage the power of collaborative filtering and feature-based recommendations to better serve their customers and increase sales. In practice, recommendations significantly impact the decision-making process of many online consumers; for example, it has been reported that a recommender system could account for 10-30% of an online retailer's sales [25] and that roughly two-thirds of the movies rented on Netflix were ones that users may never have considered if they had not been recommended to users by the recommender system [10]. Research in the area of recommender systems has focused almost exclusively on the development and improvement of the algorithms that allow these systems to make accurate recommendations and predictions. Less well-studied are the behavioral aspects of using recommender systems in the electronic marketplace.

Many recommender systems ask consumers to rate an item that they have previously experienced or consumed. These ratings are then used as inputs by recommender systems, which employ various computational techniques (based on methodologies from statistics, data mining, or machine learning) to estimate consumer preferences for other items (i.e., items that have not yet been consumed by a particular individual). These estimated preferences are often presented to the consumers in the form of "system ratings," which indicate an expectation of how much the consumer will like the item based on the recommender system algorithm and, essentially, serve as recommendations. The subsequent consumer ratings serve as additional inputs to the system, completing a feedback loop that is central to a recommender system's use and value, as illustrated in Figure 1. The figure also illustrates how consumer ratings are commonly used to evaluate the recommender system's performance in terms of accuracy by comparing how closely the system-predicted ratings match the later submitted actual ratings by the users. In our studies, we focus on the *feed-forward* influence of the recommender system upon the consumer ratings that, in turn, serve as inputs to these same systems. We believe that providing consumers with a prior rating generated by the recommender system can introduce anchoring biases and significantly influence consumer preferences and, thus, their subsequent rating of an item. As noted by [7], biases in the ratings provided by users can lead to three potential problems: (i) biases can contaminate the inputs of the recommender system, reducing its effectiveness; (ii) biases can artificially improve the resulting accuracy, providing a distorted view of the system's performance; (iii) biases might allow agents to manipulate the system so that it operates in their favor.

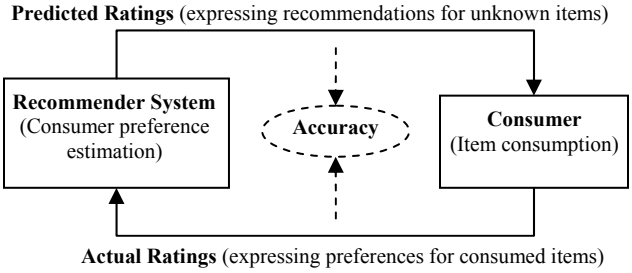


Figure 1. Ratings as part of a feedback loop in consumer-recommender interactions.

For algorithm developers, the issue of biased ratings has been largely ignored. A common underlying assumption in the vast majority of recommender systems literature is that consumers have preferences for products and services that are developed independently of the recommendation system. However, researchers in behavioral decision making, behavioral economics, and applied psychology have found that people's preferences are often influenced by elements in the environment in which preferences are constructed [5,6,18,20,30]. This suggests that the common assumption that consumers have true, non-malleable preferences for items is questionable, which raises the following question: *Whether and to what extent is the performance of recommender systems reflective of the process by which preferences are elicited?* In this study, our main objective is to answer the above question and understand the influence of a recommender system's predicted ratings on consumers' preferences. In particular, we explore four issues related to the impact of recommender systems: (1) The *anchoring* issue—understanding any potential anchoring effect, particularly at the point of consumption, is the principal goal of this study: Are people's preference ratings for items they just consumed drawn toward predictions that are given to them? (2) The *timing* issue—

Decisions@RecSys'11, October 27, 2011, Chicago, IL, USA.

does it matter whether the system's prediction is presented before or after user's consumption of the item? This issue relates to one possible explanation for an anchoring effect. Showing the prediction prior to consumption could provide a prime that influences the user's consumption experience and his/her subsequent rating of the consumed item. If this explanation is operative, an anchoring effect would be expected to be lessened when the recommendation is provided *after* consumption. (3) The *system reliability* issue—does it matter whether the system is characterized as more or less reliable? Like the timing issue, this issue is directed at illuminating the nature of the anchoring effect, if obtained. If the system's reliability impacts anchoring, then this would provide evidence against the thesis that anchoring in recommender systems is a purely numeric effect of users applying numbers to their experience. (4) The *generalizability* issue—does the anchoring effect extend beyond a single context? We investigate two different contexts in the paper. Studies 1 and 2 observe ratings of TV shows in a between-subjects design. Study 3 addresses anchoring for ratings of jokes using a within-subjects-design. Consistency of our findings supports a more general phenomenon that affects preference ratings immediately following consumption, when recommendations are provided.

2. BACKGROUND AND HYPOTHESES

Behavioral research has indicated that judgments can be constructed upon request and, consequently, are often influenced by elements of the environment in which this construction occurs. One such influence arises from the use of an anchoring-and-adjustment heuristic [6,30], the focus of the current study. Using this heuristic, the decision maker begins with an initial value and adjusts it as needed to arrive at the final judgment. A systematic bias has been observed with this process in that decision makers tend to arrive at a judgment that is skewed toward the initial anchor. Prior research on anchoring effects spans three decades and represents a very important aspect of decision making, behavioral economics, and marketing literatures. Epley and Gilovich [9] identified three waves of research on anchoring: (1) establishes anchoring and adjustment as leading to biases in judgment [5,9,21,29,30], (2) develops psychological explanations for anchoring effects [5,9,13,21,23], and (3) unbinds anchoring from its typical experimental setting and "considers anchoring in all of its everyday variety and examines its various moderators in these diverse contexts" ([9], p.21) [14,17]. Our study is primarily located within the latter wave while informing the second wave—testing explanations—as well; specifically, our paper provides a contribution both (a) to the study of anchoring in a preference situation at the time of consumption and (b) to the context of recommender systems.

Regarding the former of these contextual features, the effect of anchoring on preference construction is an important open issue. Past studies have largely been performed using tasks for which a verifiable outcome is being judged, leading to a bias measured against an objective performance standard (also see review by [6]. In the recommendation setting, the judgment is a *subjective* preference and is not verifiable against an objective standard. The application of previous studies to the preference context is not a straightforward generalization.

Regarding our studies' second contextual feature, very little research has explored how the cues provided by recommender systems influence online consumer behavior. The work that comes closest to ours is [7], which explored the effects of system-generated recommendations on user re-ratings of movies. It found that users showed high test-retest consistency when being asked to re-rate a movie with no prediction provided. However, when users were asked to re-rate a movie while being shown a "predicted" rating that was altered upward/downward from their original rating for the movie by a single fixed amount (1 rating point), they tended to give higher/lower ratings, respectively.

Although [7] did involve recommender systems and preferences, our study differs from theirs in important ways. First, we address a fuller range of possible perturbations of the predicted ratings. This allows us to more fully explore the anchoring issue as to whether any effect is obtained in a discrete fashion or more continuously over the range of possible perturbations. More fundamentally, the focus of [7] was on the effects of anchors on a *recall* task, i.e., users had already "consumed" (or experienced) the movies they were asked to re-rate in the study, had done so prior to entering the study, and were asked to remember how well they liked these movies from their past experiences. Thus, anchoring effects were moderated by potential recall-related phenomenon, and preferences were being remembered instead of constructed. In contrast, our work focuses on anchoring effects that occur in the construction of preferences at the time of actual consumption. In our study, no recall is involved in the task impacted by anchors, participants consume the good for the first time in our controlled environment, and we measure the immediate effects of anchoring.

Still, [7] provide a useful model for the design of our studies, with two motivations in mind. First, their design provides an excellent methodology for exploring the effects of recommender systems on preferences. Second, we build upon their findings to determine if anchoring effects of recommender systems extend beyond recall-related tasks and impact actual preference construction at the time of consumption. Grounded in the explanations for anchoring, as discussed above, our research goes beyond their findings to see if recommender system anchoring effects are strong enough to manipulate a consumer's perceptions of a consumption experience as it is happening.

Since anchoring has been observed in other settings, though different than the current preference setting, we begin with the conjecture that the rating provided by a recommender system serves as an anchor. Insufficient adjustment away from the anchor is expected to lead to a subsequent consumer preference rating that is shifted toward the system's predicted rating. This is captured in the following primary hypothesis of the studies:

> *Anchoring Hypothesis*: Users receiving a recommendation biased to be higher will provide higher ratings than users receiving a recommendation biased to be lower.

One mechanism that may underlie an anchoring effect with recommendations is that of *priming*, whereby the anchor can serve as a prime or prompt that activates information similar to the anchor, particularly when uncertainty is present [6]. If this dynamic operates in the current setting, then receiving the recommendation prior to consumption, when uncertainty is higher and priming can more easily operate, should lead to greater anchoring effects than receiving the recommendation after consumption. Manipulating the timing of the recommendation provides evidence for tying any effects to priming as an underlying mechanism.

> *Timing Hypothesis*: Users receiving a recommendation prior to consumption will provide ratings that are closer to the recommendation (i.e., will be more affected by the anchor) than users receiving a recommendation after viewing.

Another explanation proposed for the anchoring effect is a content-based explanation, in which the user perceives the anchor as providing evidence as to a correct answer in situations where an objective standard exists. When applied to the use of recommender systems and preferences, the explanation might surface as an issue of the consumer's trust in the system. Prior study found that increasing cognitive trust and emotional trust improved consumer's intentions to accept the recommendations [15]. Research also has highlighted the potential role of human-computer interaction and system interface design in achieving high consumer trust and acceptance of recommendations [7,19,22,28]. However, the focus of these studies differs from that underlying our research questions. In particular, the aforementioned prior studies focused on interface design (including presentation of items, explanation facilities, and rating scale definitions) rather than the anchoring effect of recommendations on the construction of consumer preferences. Our work was motivated in part by these studies to specifically highlight the role of *anchoring* on users' preference ratings.

In their initial studies, Tversky and Kahneman [30] used anchors that were, explicitly to the subjects, determined by spinning a wheel of fortune. They still observed an effect of the magnitude of the value from this random spin upon the judgments made (for various almanac-type quantities, e.g., the number of African countries in the United Nations). [27] also demonstrated anchoring effects even with extreme values (e.g., anchors of 1215 or 1992 in estimating the year that Einstein first visited the United States). These studies suggest that the anchoring effect may be purely a numerical priming phenomenon, and that the quality of the anchor may be less important.

In contrast, [20] found that the anchoring effect was mediated by the plausibility of the anchor. The research cited earlier connecting cognitive trust in recommendation agents to users' intentions to adopt them [15] also suggests a connection between reliability and use. To the extent that the phenomenon is purely numerically driven, weakening of the recommendation should have little or no effect. To the extent that issues of trust and quality are of concern, a weakening of the anchoring should be observed with a weakening of the perceived quality of the recommending system.

> *Perceived System Reliability Hypothesis*: Users receiving a recommendation from a system that is perceived as more reliable will provide ratings closer to the recommendation (i.e., will be more affected by the anchor) than users receiving a recommendation from a less reliable system.

To explore our hypotheses, we conducted three controlled laboratory experiments, in which system predictions presented to participants are biased upward and downward so our hypotheses can be tested in realistic settings. The first study explores our hypotheses by presenting participants with randomly assigned artificial system recommendations. The second study extends the first and uses a live, real-time recommender system to produce predicted recommendations for our participants, which are then biased upward or downward. The final study generalizes to preferences among jokes, studied using a within-subjects design and varying levels of rating bias. The next three sections provide details about our experiments and findings.

3. STUDY 1: IMPACT OF ARTIFICIAL RECOMMENDATIONS

The goals of Study 1 were fivefold: (1) to perform a test of the primary conjecture of anchoring effects (i.e., Anchoring Hypothesis) using artificial anchors; (2) to perform the exploratory analyses of whether participants behave differently with high vs. low anchors; (3) to test the Timing Hypothesis for anchoring effects with system recommendations (i.e., concerning differential effects of receiving the recommendation either before or after consuming the item to be subsequently rated) ; (4) to test the Perceived System Reliability Hypothesis for anchoring effects with system recommendations (i.e., concerning the relationship between the perceived reliability of the recommender system and anchoring effects of its recommendations); and (5) to build a database of user preferences for television shows, which would be used in computing personalized recommendations for Study 2.

3.1. Methods

216 people completed the study. Ten respondents indicated having seen some portion of the show that was used in the study (all subjects saw the same TV show episode in Study 1). Excluding these, to obtain a more homogeneous sample of subjects all seeing the show for the first time, left 206 subjects for analysis. Participants were solicited from a paid subject pool and paid a fixed fee at the end of the study.

In Study 1 subjects received *artificial* anchors, i.e., system ratings were not produced by a recommender system. All subjects were shown the same TV show episode during the study and were asked to provide their rating of the show after viewing. Participants were randomly assigned to one of seven experimental groups. Before providing their rating, those in the treatment groups received an artificial system rating for the TV show used in this study. Three factors were manipulated in the rating provision. First, the system rating was set to have either a *low* (1.5, on a scale of 1 through 5) or *high* value (4.5). Since [29] found an asymmetry of the anchoring effect such that high anchors produced a larger effect than did low anchors in their study of job performance ratings, we used anchors at both ends of the scale.

The second factor in Study 1 was the timing of the recommendation. The artificial system rating was given either *before* or *after* the show was watched (but always before the viewer was asked to rate the show). This factor provides a test of the Timing Hypothesis. Together, the first two factors form a 2 x 2 (High/Low anchor x Before/After viewing) between-subjects design (the top four cells of the design in Table 1).

Intersecting with this design is the use of a third factor: the perceived reliability of the system (*strong* or *weak*) making the recommendation. In the Strong conditions for this factor, subjects were told (wording is for the Before viewing/Low anchor condition): "Our recommender system thinks that you would rate the show you are about to see as 1.5 out of 5." Participants in the corresponding Weak conditions for the perceived reliability factor saw: "We are testing a recommender system that is in its early stages of development. Tentatively, this system thinks that you would rate the show you are about to see as 1.5 out of 5." This factor provides a test of the Perceived System Reliability Hypothesis. At issue is whether any effect of anchoring upon a recommendation is merely a numerical phenomenon or is tied to the perceived reliability and quality of the recommendation.

Since there was no basis for hypothesizing an interaction between timing of the recommendation and strength of the system, the complete factorial design of the three factors was not employed. For parsimony of design, the third factor was manipulated only within the Before conditions, for which the system recommendation preceded the viewing of the TV show. Thus,

within the Before conditions of the Timing factor, the factors of Anchoring (High/Low) and Reliability of the anchor (Strong/Weak) form a 2x2 between-subjects design (the bottom four cells of the design in Table 1).

In addition to the six treatment groups, a control condition, in which no system recommendation was provided, was also included. The resulting seven experimental groups, and the sample sizes for each group, are shown in Table 1.

Subjects participated in the study using a web-based interface in a behavioral lab, which provided privacy for individuals participating together. Following a welcome screen, subjects were shown a list of 105 popular, recent TV shows. TV shows were listed alphabetically within five genre categories: Comedy, Drama, Mystery/Suspense, Reality, and Sci Fi/Fantasy. For each show they indicated if they had ever seen the show (multiple episodes, one episode, just a part of an episode, or never), and then rated their familiarity with the show on a 7-point Likert scale ranging from "Not at all familiar" to "Very familiar." Based on these responses, the next screen first listed all those shows that the subject indicated having seen and, below that, shows they had not seen but for which there was some familiarity (rating of 2 or above). Subjects rated each of these shows using a 5-star scale that used verbal labels parallel to those in use by Netflix.com. Half-star ratings were also allowed, so that subjects had a 9-point scale for expressing preference. In addition, for each show, an option of "Not able to rate" was provided. Note that these ratings were not used to produce the artificial system recommendations in Study 1; instead, they were collected to create a database for the recommender system used in Study 2 (to be described later).

Table 1 Experimental Design and Sample Sizes in Study 1.

Control: 29			
Reliability condition	Timing condition	Low (anchor)	High (anchor)
Strong (reliability)	After (timing)	29	28
Strong (reliability)	Before (timing)	29	31
Weak (reliability)	Before (timing)	29	31

Following the rating task, subjects watched a TV episode. All subjects saw the same episode of a situation comedy. A less well-known TV show was chosen to maximize the likelihood that the majority of subjects were not familiar with it. The episode was streamed from Hulu.com and was 23 minutes 36 seconds in duration. The display screen containing the episode player had a visible time counter moving down from 20 minutes, forcing the respondents to watch the video for at least this time before the button to proceed to the next screen was enabled.

Either immediately preceding (in the Before conditions) or immediately following (in the After conditions) the viewing display, subjects saw a screen providing the system recommendation with the wording appropriate to their condition (Strong/Weak, Low/High anchor). This screen was omitted in the Control condition. Following, subjects rated the episode just viewed. The same 5-star (9-point) rating scale used earlier was provided for the preference rating, except that the "Not able to rate" option was omitted. Finally, subjects completed a short survey that included questions on demographic information and TV viewing patterns.

3.2. Results

All statistical analyses were performed using SPSS 17.0. Table 2 shows the mean ratings for the viewed episode for the seven experimental groups. Our preliminary analyses included data collected by survey, including both demographic data (e.g., gender, age, occupation) and questionnaire responses (e.g., hours watching TV per week, general attitude towards recommender systems), as covariates and random factors. However, none of these variables or their interaction terms turned out to be significant, and hence we focus on the three fixed factors.

We begin with analysis of the 2x2 between-subjects design involving the factors of direction of anchor (High/Low) and its timing (Before/After viewing). As is apparent from Table 2 (rows marked as Design 1), and applying a general linear model, there is no effect of Timing ($F(1,113) = 0.021$, $p = .885$). The interaction of Timing and High/Low anchor was also not significant ($F(1, 113) = 0.228$, $p = .634$). There is a significant observed anchoring effect of the provided artificial recommendation ($F(1, 113) = 14.30$, $p = .0003$). The difference between the High and Low conditions was in the expected direction, showing a substantial effect between groups (one-tailed $t(58) = 2.788$, $p = .0035$, assuming equal variances). Using Cohen's (1988) *d,* which is an effect size measure used to indicate the standardized difference between two means (as computed by dividing the difference between two means by a standard deviation for the data), the effect size is 0.71, in the medium-to-large range.

Table 2. Mean (SD) Ratings of the Viewed TV Show by Experimental Condition in Study 1.

Design 1	Design 2	Group (timing-anchor-reliability)	N	Mean (SD)
*	*	Before-High-Strong	31	3.48 (1.04)
*		After-High-Strong	28	3.43b (0.81)
		Control	29	3.22 (0.98)
	*	Before-High-Weak	31	3.08 (1.07)
	*	Before-Low-Weak	29	2.83 (0.75)
*		After-Low-Strong	29	2.88 (0.79)
*	*	Before-Low-Strong	29	2.78 (0.92)

Using only data within the Before conditions, we continue by analyzing the second 2 x 2 between-subjects design in the study (Table 2, rows marked as Design 2), involving the factors of direction of anchor (High/Low) and perceived system reliability (Strong/Weak). The anticipated effect of weakening the recommender system is opposite for the two recommendation directions. A High-Weak recommendation is expected to be less pulled in the positive direction compared to a High-Strong recommendation; and, a Low-Weak recommendation is expected to be less pulled in the negative direction as compared to Low-Strong. So, we explore these conjectures by turning to the direct tests of the contrasts of interest. There is no significant difference between the High and Low conditions with Weak recommendations ($t(58) = 1.053$, $p = .15$), unlike with Strong recommendations (as noted above, p = .0035). Also, the overall effect was reduced for the Weak setting, compared to the Strong recommendation setting, and was measured as a Cohen's $d = 0.16$, less than even the small effect size range. Thus, the subjects were sensitive to the perceived reliability of the recommender system. Weak recommendations did not operate as a significant anchor when the perceived reliability of the system was lowered.

Finally, we check for asymmetry of the anchoring effect using the control group in comparison to the Before-High and Before-Low groups. (Similar results were obtained using the After-High and After-Low conditions as comparison, or using the combined High and Low groups.) In other words, we already showed that the High and Low groups were significantly different from each other, but we also want to determine if each group differs from the Control (i.e., when no recommendation was provided to the users)

in the same manner. When an artificial High recommendation was provided (4.5), ratings were greater than those of the Control group, but not significantly so ($t(58) = 0.997$, $p = .162$). But when an artificial Low recommendation was provided (1.5), ratings were significantly lower than those of the Control group ($t(56) = 1.796$, $p = .039$). There was an asymmetry of the effect; however, the direction was opposite to that found by Thorsteinson et al. (2008). To study the effect further, Study 2 was designed to provide further evidence. So, we will return to the discussion of the effect later in the paper.

In summary, analyses indicate a moderate-to-strong effect, supporting the Anchoring Hypothesis. When the recommender system was presented as less reliable, being described as in test phase and providing only tentative recommendations, the effect size was reduced to a minimal or no effect, in support of the Perceived System Reliability Hypothesis. Finally, the Timing Hypothesis was not supported – the magnitude of the anchoring effect was not different whether the system recommendation was received before or after the viewing experience. This suggests that the effect is not attributable to a priming of one's attitude prior to viewing. Instead, anchoring is likely to be operating at the time the subject is formulating a response.

Overall, viewers, without a system recommendation, liked the episode (mean = 3.22, where 3 = "Like it"), as is generally found with product ratings. However, asymmetry of the anchoring effect was observed at the low end: Providing an artificial low recommendation reduced this preference more so than providing a high recommendation increased the preference. This effect is explored further in Study 2.

4. STUDY 2: IMPACT OF ACTUAL RECOMMENDATIONS

Study 2 follows up Study 1 by replacing the artificially fixed anchors with actual personalized recommendations provided by a well-known and commonly used recommendation algorithm. Using the user preferences for TV shows collected in Study 1, a recommender system was designed to estimate preferences of subjects in Study 2 for unrated shows. Because participants provide input ratings before being shown any recommendations or other potential anchors, the ratings were unbiased inputs for our own recommendation system. Using a parallel design to Study 1, we examine the Anchoring Hypothesis with a recommender system comparable to the ones employed in practice online.

4.1. Methods

197 people completed the study. They were solicited from the same paid subject pool as used for Study 1 with no overlap between the subjects in the two studies. Participants received a fixed fee upon completion of the study.

In Study 2, the anchors received by subjects were based on the recommendations of a true recommender system (discussed below). Each subject watched a show that he/she had indicated not having seen before – that was recommended by an actual real-time system based on the subject's individual ratings. Since there was no significant difference observed between subjects receiving system recommendations before or after viewing a show in Study 1, all subjects in the treatment groups for Study 2 saw the system-provided rating before viewing.

Three levels were used for the recommender system's rating provided to subjects in Study 2: Low (i.e., adjusted to be 1.5 points below the system's predicted rating), Accurate (the system's actual predicted rating), and High (1.5 points above the system's predicted rating). High and Low conditions were included to learn more about the asymmetry effect observed in Study 1. In addition to the three treatment groups, a control group was included for which no system recommendation was provided. The numbers of participants in the four conditions of the study are shown in Table 4 (Section 4.2).

Based on the TV show rating data collected in Study 1, an online system was built for making TV show recommendations in real time. We compared seven popular recommendation techniques to find the best-performing technique for our dataset. The techniques included simple user- and item-based rating average methods, user- and item-based collaborative filtering approaches and their extensions [2,4,24], as well as a model-based matrix factorization algorithm [11,16] popularized by the recent Netflix prize competition [3]. Each technique was evaluated using 10-fold cross validation based on the standard mean absolute error (MAE) and coverage metrics. Although the performances are comparable, the item-based CF performed slightly better than other techniques (measured in predictive accuracy and coverage). Also because the similarities between items could be pre-computed, the item-based technique performed much faster than other techniques. Therefore the standard item-based collaborative filtering approach was selected for our recommender system.

During the experiments, the system took as input subject's ratings of shows that had been seen before or for which the participant had indicated familiarity. In real time, the system predicted ratings for all unseen shows and recommended one of the unseen shows for viewing. To avoid possible show effects (e.g., to avoid selecting shows that receive universally bad or good predictions) as well as to assure that the manipulated ratings (1.5 points above/below the predicted rating) could still fit into the 5-point rating scale, only shows with predicted rating scores between 2.5 and 3.5 were recommended. When making recommendations, the system examined each genre in alphabetical order (i.e., comedy first, followed by drama, mystery, reality, and sci-fi) and went through all unseen shows within each genre alphabetically until one show with a predicted rating between 2.5 and 3.5 was found. This show was then recommended to the subject. When no show was eligible for recommendation, subjects were automatically re-assigned to one of the treatment groups in Study 1.

Our TV show recommender system made suggestions from a list of the 105 most popular TV shows that have aired in the recent decade according to a ranking posted on TV.com. Among the 105 shows, 31 were available for online streaming on Hulu.com at the time of the study and were used as the pool of shows recommended to subjects for viewing. Since our respondents rated shows, but viewed only a single episode of a show, we needed a procedure to select the specific episode of a show for viewing. For each available show, we manually compared all available episodes and selected the episode that received a median aggregated rating by Hulu.com users to include in the study. This procedure maximized the representativeness of the episode for each show, avoiding the selection of outlying best or worst episodes that might bias the participant's rating. Table 3 shows the distributions of rated and viewing-available shows by genre.

The procedure was largely identical to the Before and Control conditions used for Study 1. However, in Study 2, as indicated earlier, subjects did not all view the same show. TV episodes were again streamed from Hulu.com. The episode watched was either approximately 22 or 45 minutes in duration. For all subjects, the viewing timer was set at 20 minutes, as in Study 1. Subjects were instructed that they would not be able to proceed

until the timer reached zero; at which time they could choose to stop and proceed to the next part of the study or to watch the remainder of the episode before proceeding.

Table 3. Distribution of Shows.

Genre	Number of Shows	Available for Viewing
Comedy	22	7
Drama	26	8
Mystery/Suspense	25	4
Reality	15	4
Sci Fi and Fantasy	17	8
Total	105	31

4.2. Results

Since the subjects did not all see the same show, the preference ratings for the viewed show were adjusted for the predicted ratings of the system, in order to obtain a response variable on a comparable scale across subjects. Thus, the main response variable is the *rating drift*, which we define as:

Rating Drift = Actual Rating – Predicted Rating.

Predicted Rating represents the rating of the TV show watched by the user during the study as predicted by the recommendation algorithm (before any perturbations to the rating are applied), and Actual Rating is the user's rating value for this TV show after watching the episode. Therefore, positive/negative Rating Drift values represent situations where the user's submitted rating was higher/lower than the system's rating, as possibly affected by positive/ negative perturbations (i.e., high/low anchors).

Similarly to Study 1, our preliminary analyses using general linear models indicated that none of the variables collected in the survey (such as demographics, etc.) demonstrated significance in explaining the response variable. The mean (standard deviation) values across the four conditions of the study for this variable are shown in Table 4. Using a one-way ANOVA, overall the three experimental groups (i.e., High, Low, and Accurate) significantly differed ($F(2, 147) = 3.43$, $p = .035$).

Table 4. Mean (SD) Rating Drift of the Viewed TV Show by Experimental Condition, Study 2.

	Study 2	
Group	N	Mean (SD)
High	51	0.40 (1.00)
Control	48	0.14 (0.94)
Accurate	51	0.13 (0.96)
Low	47	-0.12 (0.94)

Providing an accurate recommendation did not significantly affect preferences for the show, as compared to the Control condition (two-tailed $t(97) = 0.023$, $p = .982$). Consistent with Study 1, the High recommendation condition led to inflated ratings compared to the Low condition (one-tailed $t(96) = 2.629$, $p = .005$). The effect size was of slightly less magnitude with Cohen's $d = 0.53$, a medium effect size. However, unlike in Study 1, the anchoring effect in Study 2 is symmetric at the High and Low end. There was a marginally significant effect of the recommendation being lowered compared to being accurate ($t(96) = 1.305$, $p = .098$, Cohen's $d = .30$), and a marginally significant effect at the High end compared to receiving Accurate recommendations ($t(100) = 1.366$, $p = .088$, Cohen's $d = .23$). Similar effects are observed when comparing High/Low to Control condition. In summary, the Anchoring Hypothesis is supported in Study 2, consistently with Study 1. However, the anchoring effects were symmetric in the overall analysis of Study 2 at the High and Low ends.

To pursue the results further, we recognize that one source of variation in Study 2 as compared to Study 1 is that different shows were observed by the subjects. As it turns out, 102 of the 198 subjects in Study 2 (52%) ended up watching the same Comedy show. As a result, we are able to perform post-hoc analyses, paralleling the main analyses, limited to this subset of viewers. The mean (standard deviation) values across the four conditions of these subjects for the main response variable are shown in Table 5. Using the same response variable of rating drift, the overall effect across the experimental conditions was marginally maintained ($F(2, 77) = 2.70$, $p = .07$. Providing an accurate recommendation still did not significantly affect preferences for the show, as compared to the Control condition (two-tailed $t(47) = 0.671$, $p = .506$). Consistent with Study 1 and the overall analyses, the High recommendation condition led to inflated ratings compared to the Low condition (one-tailed $t(51) = 2.213$, $p = .016$). The effect size was also comparable to the overall effect magnitude with Cohen's $d = 0.61$, a medium effect size.

However, for the limited sample of subjects who watched the same episode, the effects at the High and Low end were not symmetric. Compared to receiving an Accurate recommendation, there was a significant effect of the recommendation being raised ($t(52) = 1.847$, $p = .035$, Cohen's $d = .50$), but not of being lowered ($t(51) = 0.286$, $p = .388$).

Table 5. Mean(SD) Rating Drift for Subjects Who Watched the Same Comedy Show in Study 2.

Group	N	Mean (SD)
High	27	0.81 (0.82)
Control	22	0.53 (0.76)
Accurate	27	0.37 (0.93)
Low	26	0.30 (0.86)

Thus, the indicated asymmetry of the anchoring effect is different from the asymmetry present in Study 1, being at the High end rather than the Low end. Also, the asymmetry is not robust across the overall data. Indicated is that the underlying cause of asymmetries is situational, in this case depending upon specific TV show effects. When looking at effects across different TV shows (Table 4), the show effects average out and symmetry is observed overall. When looking at effects for a particular show (Tables 2 and 5), idiosyncratic asymmetries can arise.

5. STUDY 3: ACTUAL RECOMMENDATIONS WITH JOKES

Study 3 provides a generalization of Study 2 within a different content domain, applying a recommender system to joke preferences rather than TV show preferences. As in Study 2, the procedure uses actual recommendations provided by a commonly used recommendation algorithm. A within-subjects design also allows us to investigate behavior at an individual level of analysis, rather than in the aggregate. We apply a wider variety of perturbations to the actual recommendations for each subject, ranging from -1.5 to 1.5, the values used in Study 2, rather than just using a single perturbation per subject.

5.1. Methods

61 people received a fixed fee for completing the study. They were solicited from the same paid subject pool used for Studies 1 and 2 with no overlap across the three studies.

As with Study 2, the anchors received by subjects were based on the recommendations of a true recommender system. The item-

based collaborative filtering technique was used to maintain consistency with Study 2. The same list of 100 jokes was used during the study, though the order of the jokes was randomized between subjects. The jokes and the rating data for training the recommendation algorithm were taken from the Jester Online Joke Recommender System repository [12]. Specifically, we used their Dataset 2, which contains 150 jokes. To get to our list of 100, we removed those jokes that were suggested for removal at the Jester website (because they were either included in the "gauge set" in the original Jester joke recommender system or because they were never displayed or rated), jokes that more than one of the coauthors of our study identified as having overly objectionable content, and finally those jokes that were greatest in length (based on word count).

The procedure paralleled that used for Study 2 with changes adapted to the new context. Subjects first evaluated 50 jokes, randomly selected and ordered from the list of 100, as a basis for providing recommendations. The same 5-star rating scale with half-star ratings from Studies 1 and 2 was used, affording a 9-point scale for responses. Next, the subjects received 40 jokes with a predicted rating displayed. Thirty of these predicted ratings were perturbed, 5 each using perturbations of -1.5, -1.0, -0.5, +0.5, +1.0, and +1.5. The 30 jokes that were perturbed were determined pseudo-randomly to assure that the manipulated ratings would fit into the 5-point rating scale. First, 10 jokes with predicted rating scores between 2.5 and 3.5 were selected randomly to receive perturbations of -1.5 and +1.5. From the remaining, 10 jokes with predicted rating scores between 2.0 and 4.0 were selected randomly to receive perturbations of -1.0 and +1.0. Then, 10 jokes with predicted rating scores between 1.5 and 4.5 were selected randomly to receive perturbations of -0.5 and +0.5. Ten predicted ratings were not perturbed, and were displayed exactly as predicted. These 40 jokes were randomly intermixed. Following the first experimental session (3 sessions were used in total), the final 10 jokes were added as a control. A display was added on which subjects provided preference ratings for the 10 jokes with no predicted rating provided, again in random order. Finally in all sessions, subjects completed a short demographic survey.

5.2. Results

As with Study 2, the main response variable for Study 3 was Rating Drift (i.e., Actual Rating – Predicted Rating). As an illustration of the overall picture, Figure 2 shows the mean Rating Drift, aggregated across items and subjects, for each perturbation used in the study. In the aggregate, there is a linear relationship both for negative and positive perturbations. For comparison purposes, Table 6 shows the mean (standard deviation) values across the four perturbation conditions of Study 3 that were comparable to those used in Study 2 (aggregating across all relevant Study 3 responses). The general pattern for Study 3—using jokes and within-subjects design—parallels that for Study 2—using TV shows and a between-subjects design.

The within-subjects design also allows for analyses of the Anchoring Hypothesis at the individual level. We began by testing the slopes across subjects between negative and positive perturbations, and no significant difference was observed ($t(60) = 1.39$, two-tailed $p = .17$). We also checked for curvilinearity for each individual subject for both positive and negative perturbations. No significant departures from linearity were observed, so all reported analyses use only first-order effects. As an indicator of the magnitude of the effect, we examined the distribution of the correlation coefficients for the individual analyses. The mean magnitude of the relationship is 0.37, with values ranging from -0.27 to 0.87.

Overall, the analyses strongly suggest that the effect of perturbations on rating drift is not discrete. Perturbations have a continuous effect upon ratings with, on average, a drift of 0.35 rating points occurring for every rating point of perturbation (e.g., mean rating drift is 0.53 for a perturbation of +1.5).

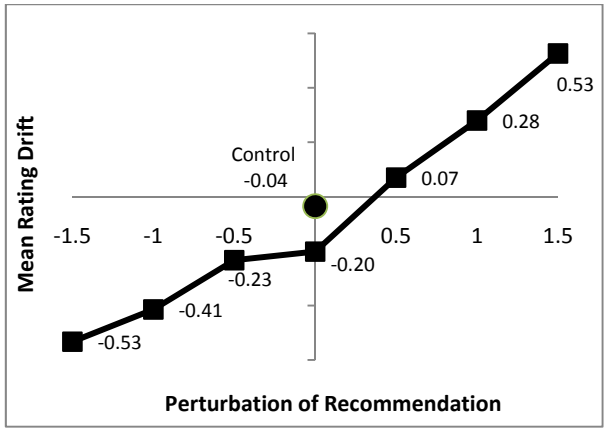


Figure 2. Mean Rating Drift as a Function of the Amount of Rating Perturbation and for Control Condition in Study 3.

Table 6. Mean (SD) Rating Drift, in the Comparable Conditions Used in Study 2 (±1.5, 0, Control), for Study 3.

Group	N	Mean (SD)
High	305	0.53 (0.94)
Control	320	-0.04 (1.07)
Accurate	610	-0.20 (0.97)
Low	305	-0.53 (0.95)

6. DISCUSSION AND CONCLUSIONS

We conducted three laboratory experiments and systematically examined the impact of recommendations on consumer preferences. The research integrates ideas from behavioral decision theory and recommender systems, both from practical and theoretical standpoints. The results provide strong evidence that biased output from recommender systems can significantly influence the preference ratings of consumers.

From a practical perspective, the findings have several important implications. First, they suggest that standard performance metrics for recommender systems may need to be rethought to account for these phenomena. If recommendations can influence consumer-reported ratings, then how should recommender systems be objectively evaluated? Second, how does this influence impact the inputs to recommender systems? If two consumers provide the same rating, but based on different initial recommendations, do their preferences really match in identifying future recommendations? Consideration of issues like these arises as a needed area of study. Third, our findings bring to light the potential impact of recommender systems on strategic practices. If consumer choices are significantly influenced by recommendations, regardless of accuracy, then the potential arises for unscrupulous business practices. For example, it is well-known that Netflix uses its recommender system as a means of inventory management, filtering recommendations based on the availability of items [26]. Taking this one step further, online retailers could potentially use preference bias based on recommendations to increase sales.

Further research is clearly needed to understand the effects of recommender systems on consumer preferences and behavior. Issues of trust, decision bias, and preference realization appear to be intricately linked in the context of recommendations in online marketplaces. Additionally, the situation-dependent asymmetry of these effects must be explored to understand what situational characteristics have the largest influence. Moreover, future research is needed to investigate the error compounding issue of anchoring: How far can people be pulled in their preferences if a recommender system keeps providing biased recommendations? Finally, this study has brought to light a potentially significant issue in the design and implementation of recommender systems. Since recommender systems rely on preference inputs from users, bias in these inputs may have a cascading error effect on the performance of recommender system algorithms. Further research on the full impact of these biases is clearly warranted.

7. ACKNOWLEDGMENT

This work is supported in part by the National Science Foundation grant IIS-0546443.

REFERENCES

[1] Adomavicius, G., and Tuzhilin, A. 2005. "Toward the Next Generation of Recommendation System: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Transactions on Knowledge and Data Engineering*, 17, (6), 734-749.

[2] Bell, R.M., and Koren, Y. 2007. "Improved Neighborhood-Based Collaborative Filtering," *KDD Cup'07*, San Jose, California, USA, 7-14.

[3] Bennett, J., and Lanning, S. 2007. "The Netflix Prize," *KDD-Cup and Workshop*, San Jose, CA, www.netflixprize.com.

[4] Breese, J.S., Heckerman, D., and Kadie, C. 1998. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *14th Conf. on Uncertainty in Artificial Intelligence*, Madison, WI.

[5] Chapman, G., and Bornstein, B. 1996. "The More You Ask for, the More You Get: Anchoring in Personal Injury Verdicts," *Applied Cognitive Psychology*, 10, 519-540.

[6] Chapman, G., and Johnson, E. 2002. "Incorporating the Irrelevant: Anchors in Judgments of Belief and Value.," in *Heuristics and Biases: The Psychology of Intuitive Judgment,* T. Gilovich, D. Griffin and D. Kahneman (eds.). Cambridge: Cambridge University Press, 120-138.

[7] Cosley, D., Lam, S., Albert, I., Konstan, J.A., and Riedl, J. 2003. "Is Seeing Believing? How Recommender Interfaces Affect Users' Opinions," *CHI 2003 Conference*, Fort Lauderdale FL.

[8] Deshpande, M., and Karypis, G. 2004. "Item-Based Top-N Recommendation Algorithms," *ACM Trans. Information Systems*, 22, (1), 143-177.

[9] Epley, N., and Gilovich, T. 2010. "Anchoring Unbound," *J. of Consumer Psych*, 20, 20-24.

[10] Flynn, L.J. January 23, 2006. "Like This? You'll Hate That. (Not All Web Recommendations Are Welcome.)." *New York Times*, from http://www.nytimes.com/2006/01/23/technology/23recommend.html.

[11] Funk, S. 2006. "Netflix Update: Try This at Home." 2010, from http://sifter.org/~simon/journal/20061211.html.

[12] Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. 2001. "Eigentaste: A Constant Time Collaborative Filtering Algorithm," *Information Retrieval*, 4, (2), 133-151.

[13] Jacowitz, K.E., and Kahneman, D. 1995. "Measures of Anchoring in Estimation Tasks," *Personality and Social Psychology Bulletin*, 21, 1161-1166.

[14] Johnson, J.E.V., Schnytzer, A., and Liu, S. 2009. "To What Extent Do Investors in a Financial Market Anchor Their Judgments Excessively?" Evidence from the Hong Kong Horserace Betting Market," *Journal of Behavioral Decision Making*, 22, 410-434.

[15] Komiak, S., and Benbasat, I. 2006. "The Effects of Personalization and Familiarity on Trust and Adoption of Recommendation Agents," *MIS Quarterly*, 30, (4), 941-960.

[16] Koren, Y., Bell, R., and Volinsky, C. 2009. "Matrix Factorization Techniques for Recommender Systems," *IEEE Computer Society*, 42, 30-37.

[17] Ku, G., Galinsky, A.D., and Murnighan, J.K. 2006. "Starting Low but Ending High: A Reversal of the Anchoring Effect in Auctions," *J. of Personality and Social Psych*, 90, 975-986.

[18] Lichtenstein, S., and Slovic, P. (eds.). 2006. *The Construction of Preference*. Cambridge: Cambridge University Press.

[19] Mcnee, S.M., Lam, S.K., Konstan, J.A., and Riedl, J. 2003. "Interfaces for Eliciting New User Preferences in Recommender Systems," in *User Modeling 2003, Proceedings*. Berlin: Springer-Verlag Berlin, 178-187.

[20] Mussweiler, T., and Strack, F. 2000. "Numeric Judgments under Uncertainty: The Role of Knowledge in Anchoring," *Journal of Experimental Social Psychology*, 36, 495-518.

[21] Northcraft, G., and Neale, M. 1987. "Experts, Amateurs, and Real Estate: An Anchoring-and-Adjustment Perspective on Property Pricing Decisions," *Organizational Behavior and Human Decision Processes*, 39, 84-97.

[22] Pu, P., and Chen, L. 2007. "Trust-Inspiring Explanation Interfaces for Recommender Systems," *Knowledge-Based Systems*, 20, (6), Aug, 542-556.

[23] Russo, J.E. 2010. "Understanding the Effect of a Numerical Anchor," *Journal of Consumer Psychology*, 20, 25-27.

[24] Sarwar, B., Karypis, G., Konstan, J.A., and Riedl, J. 2001. "Item-Based Collaborative Filtering Recommendation Algorithms," *10th International WWW Conference*, Hong Kong, 285 - 295.

[25] Schonfeld, E. July 2007. "Click Here for the Upsell." *CNNMoney.com*, from http://money.cnn.com/magazines/business2/business2_archive/2007/07/01/100117056/index.htm.

[26] Shih, W., S., K., and Spinola, D. 2007. "Netflix," *Harvard Business School Publishing*, (case number 9-607-138).

[27] Strack, F., and Mussweiler, T. 1997. "Explaining the Enigmatic Anchoring Effect: Mechanisms of Selective Accessibility," *Journal of Personality and Social Psychology*, 73, 437-446.

[28] Swearingen, K., and Sinha, R. 2001. "Beyond Algorithms: An Hci Perspective on Recommender Systems," *ACM SIGIR 2001 Workshop on Recommender Systems*, New Orleans, Louisiana.

[29] Thorsteinson, T., Breier, J., Atwell, A., Hamilton, C., and Privette, M. 2008. "Anchoring Effects on Performance Judgments," *Organizational Behavior and Human Decision Processes*, 107, 29-40.

[30] Tversky, A., and Kahneman, D. 1974. "Judgment under Uncertainty: Heuristics and Biases," *Science*, 185, 1124-1131.

Evaluating Group Recommendation Strategies in Memory-Based Collaborative Filtering

Nadia A. Najjar
University of North Carolina at Charlotte
9201 University City Blvd.
Charlotte, NC, USA
nanajjar@uncc.edu

David C. Wilson
University of North Carolina at Charlotte
9201 University City Blvd.
Charlotte, NC, USA
davils@uncc.edu

ABSTRACT

Group recommendation presents significant challenges in evolving best practice approaches to group modeling, but even moreso in dataset collection for testing and in developing principled evaluation approaches across groups of users. Early research provided more limited, illustrative evaluations for group recommender approaches, but recent work has been exploring more comprehensive evaluative techniques. This paper describes our approach to evaluate group-based recommenders using data sets from traditional single-user collaborative filtering systems. The approach focuses on classic memory-based approaches to collaborative filtering, addressing constraints imposed by sparsity in the user-item matrix. In generating synthetic groups, we model 'actual' group preferences for evaluation by precise rating agreement among members. We evaluate representative group aggregation strategies in this context, providing a novel comparison point for earlier illustrative memory-based results and for more recent model-based work, as well as for models of actual group preference in evaluation.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Algorithms, Experimentation, Standardization

Keywords

Evaluation, Group recommendation, Collaborative filtering, Memory-based

1. INTRODUCTION

Recommender systems have traditionally focused on the individual user as a target for personalized information filtering. As the field has grown, increasing attention is being given to the issue of group recommendation [14, 4]. Group recommender systems must manage and balance preferences from individuals across a group of users with a common purpose, in order to tailor choices, options, or information to the group as a whole. Group recommendations can help to support a variety of tasks and activities across domains that have a social aspect with shared-consumption needs. Common examples arise in social entertainment: finding a movie or a television show for family night, date night, or the like [22, 11, 23]; finding a restaurant for dinner with work colleagues, family, or friends [17]; finding a dish to cook that will satisfy the whole group [5], the book the book club should read next, the travel destination for the next family vacation [19, 2, 13], or the songs to play at any social event or at any shared public space [24, 3, 7, 9, 18].

Group recommenders have been distinguished from single user recommenders primarily by their need for an aggregation mechanism to represent the group. A considerable amount of research in group-based recommenders concentrates on the techniques used for a recommendation strategy, and two main group recommendation strategies have been proposed [14]. The first strategy merges the individual profiles of the group members into one group representative profile, while the second strategy merges the recommendation lists or predictions computed for each group member into one recommendation list presented to the group. Both strategies utilize recommendation approaches validated for individual users, leaving the aggregation strategy as a distinguishing area of study applicable for group-based recommenders.

Group recommendation presents significant challenges in evolving best practice approaches to group modeling, but even moreso in dataset collection for testing and in developing principled evaluation approaches across groups of users. Early research provided more limited, illustrative evaluations for group recommender approaches (e.g., [18, 20, 17]), but recent work has been exploring more comprehensive evaluative techniques (e.g., [4, 8, 1]). Broadly, evaluations have been conducted either via live user studies or via synthetic dataset analysis. In both types of evaluation, determining an overall group preference to use as ground truth in measuring recommender accuracy presents a complementary aggregation problem to group modeling for generating recommendations. Based on group interaction and group choice outcomes, either a gestalt decision is rendered for the group as a whole, or individual preferences are elicited and combined to represent the overall group preference. The former lends itself to user studies in which the decision emerges from

group discussion and interaction, while the latter lends itself to synthetic group analysis. Currently, the limited deployment of group recommender systems coupled with the additional overhead of bringing groups together for user studies has constrained the availability of data sets that can be used to evaluate group based recommenders. Thus as with other group evaluation efforts [4], we adopt the approach of generating synthetic groups for larger scale evaluation.

It is important to note that there are two distinct group modeling issues at play. The first is how to model a group for the purpose of making recommendations (i.e., what a group's preference outcome *will be*). We refer to this as the *recommendation group preference model* (RGPM). The second is how to determine an "actual" group preference based on outcomes in user data, in order to represent ground truth for evaluation purposes (i.e., what a group's preference outcome *was*). We refer to this as the *actual group preference model* (AGPM). For example, it might be considered a trivial recommendation if each group member had previously given a movie the same strong rating across the board. However, such an agreement point is ideal for evaluating whether that movie should have been recommended for the group.

In evaluating group-based recommenders, the primary context includes choices made about:

- the underlying recommendation strategy (e.g., content-based, collaborative memory-based or model-based)
- group modeling for making recommednations — RGPM (e.g., least misery)
- determining actual group preferences for evaluative comparison to system recommendations — AGPM (e.g., choice aggregation)
- choices about metrics for assessment (e.g., ranking, rating value).

Exploring the group recommendation space involves evaluation across a variety of such contexts.

To date, we are not aware of a larger-scale group recommender evaluation using synthetic data sets that (1) focuses on traditional memory-based collaborative filtering or (2) employs precise overlap across individual user ratings for evaluating actual group preference. Given the foundational role of classic user-based [22] collaborative filtering in recommender systems, we are interested in understanding the behavior of group recommendation in this context as a comparative baseline for evaluation. Given that additional inference to determine "ground truth" preference for synthetic groups can potentially decrease precision in evaluation, we are interested in comparing results when group members agree precisely in original ratings data.

In this paper, we focus on traditional memory-based approaches to collaborative filtering, addressing constraints imposed by sparsity in the user-item matrix. In generating valid synthetic groups, we model actual group preferences by direct rating agreement among members. Prediction accuracy is measured using root mean squared error and mean average error. We evaluate the performance of three representative group aggregation strategies (average, least misery, most happiness) [15] in this context, providing a novel comparison point for earlier illustrative memory-based results, for more recent model-based work, and for models of actual group preference in evaluation. This paper is organized as follows: section 2 overviews related researches. Section 3 outlines our group testing framework. Section 4 provides the evaluation using the proposed framework. Finally section 5 outlines our results and discussion.

2. RELATED WORK

Previous research that involves evaluation of group recommendation approaches falls into two primary categories. The first category employs synthetic datasets, generated from existing single-user datasets (typically MovieLens[1]). The second category focuses on user studies.

2.1 Group Aggregation Strategies

Various group modeling strategies for making recommendations have been proposed and tested to aggregate the individual group user's preferences into a recommendation for the group. Masthoff [16] evaluated eleven strategies inspired from social choice theory. Three representative strategies are average strategy, least misery, and most happiness.

- Average Strategy: this is the basic group aggregation strategy that assumes equal influence among group members and calculates the average rating of the group members for any given item as the predicted rating. Let n be the number of users in a group and *rij* be the rating of user j for item i, then the group rating for item i is computed as follows:

$$Gr_i = \frac{\sum_{j=1}^{n} r_{ji}}{n} \tag{1}$$

- Least Misery Strategy: this aggregation strategy is applicable in situations where the recommender system needs to avoid presenting an item that was really disliked by any of the group members, i.e., that goal is to please the least happy member. The predicted rating is calculated as the lowest rating of for any given item among group members and computed as follows:

$$Gr_i = \min_j r_{ji} \tag{2}$$

- Most Happiness: this aggregation strategy is the opposite of the least misery strategy. It applies in situations where the group is as happy as their happiest member and computed as follows:

$$Gr_i = \max_j r_{ji} \tag{3}$$

2.2 Evaluation with Synthetic Groups

Recent work by Baltrunas [4] used simulated groups to compare aggregation strategies of ranked lists produced by a model based collaborative filtering methodology using matrix factorization with gradient descent (SVD). This approach addresses sparsity issues for user similarity. The MovieLens data set was used to simulate groups of different sizes (2, 3, 4, 8) and different degrees of similarity (high, random). They employed a ranking evaluation metric, measuring the effectiveness of the predicted rank list using Normalized Discounted Cumulative Gain (nDCG). To account for the sparsity in the rating matrix nDCG was computed only over the items that appeared in the target user test set. The effectiveness of the group recommendation was measured as the average effectiveness (nDCG) of the group members where a higher nDCG indicated better performance.

[1]www.movielens.org

Chen et al. [8] also used simulated groups and addressed the sparsity in user-rating matrix by predicting the missing ratings of items belonging in the union set of items rated by group members. They simulated 338 random groups from the MovieLens data set and used it for evaluating the use of Genetic Algorithms to exploit single user ratings as well as item ratings given by groups to model group interactions and find suitable items that can be considered neighbors in their implemented neighborhood-based CF.

Amer-Yahia et al. [1] also simulated groups from MovieLens. The simulated groups where used to measure the performance of different strategies centered around a top-k TA algorithm. To generate groups a similarity level was specified, groups were formed from users that had a similarity value within a 0.05 margin. They varied the group similarity between 0.3, 0.5, 0.7 and 0.9 and the size 3, 5 and 8. It was unclear how actual group ratings were established for the simulated groups or how many groups were created.

2.3 Evaluation with User Studies

Masthoff [15] employed user studies, not to evaluate specific techniques, but to determine which group aggregation strategies people actually use. Thirty-nine human subjects were given the same individual rating sets from three people on a collection of video clips. Subjects were asked to decide which clips the group should see given time limitations for viewing only 1, 2, 3, 4, 5, 6, or 7 clips, respectively. In addition, why they made that selection. Results indicated that people particularly use the following strategies: Average, Average Without Misery and Least Misery.

PolyLens [20] evaluated qualitative feedback and changes in user behavior for a basic Least Misery aggregation strategy. Results showed that while users liked and used group recommendation, they disliked the minimize misery strategy They attributed this to the fact that this social value function is more applicable to groups of smaller sizes.

Amer-Yahia et al. [1] also ran a user study using Amazon's Mechanical Turk users, they had a total of 45 users where various groups were formed of sizes 3 and 8 to represent small and large groups. They established an evaluation baseline by generating a recommendation list using four implemented strategies. The resulting lists are combined into a single group list of distinct items and were presented to the users for evaluation where a relevance score of 1 was given if the user considered the item suitable for the group and 0 otherwise. They employed an nDCG measure to evaluate their proposed prediction lists consensus function. The nDCG measure was computed for each group member and the average was considered the effectiveness of the group recommendation.

Other work considers social relationships and interactions among group members when aggregating the predictions [10, 8, 21]. They model member interactions, social relationships, domain expertise, and dissimilarity among the group members when choosing a group decision strategy. For example, Recio-Garcia et al. [21] described a group recommender system that takes into account the personality types for the group members.

Berkovsky and Freyne [5] reported better performance in the recipe recommendation domain when aggregating the user profiles rather than aggregating individual user predictions. They implemented a memory-based recommendation approach comparing the performance of four recommendation strategies, including aggregated models and aggregated predictions. Their aggregated predictions strategy combined the predictions produced for each of the group members into one prediction using a weighted, linear combination of these predictions. Evaluation consisted of 170 users where a 108 of them belonged to a family group with size ranges between 1 and 4.

2.4 Establishing Group Preference

A major question that must be addressed in evaluating group recommender systems is how to establish the actual group preference in order to compare accuracy with system predictions. Previous work by [4, 8, 1] simulated groups from single-user data sets. Their simulated group creation was limited to groups of different sizes (representing small, medium and large) with certain degrees of similarity (random, homogeneous and heterogeneous). Chen et al. [8] used a baseline aggregation as the ground truth while [4] compares the effectiveness of the group-based recommendation to the effectiveness of the individual recommendations made to each member in the group. This led to our work in investigating ways to create synthesized groups from the most commonly used CF single-user data sets taking into consideration the ability to identify and establish ground truth. We propose a novel Group Testing Framework that allows for the creation of synthesized groups that can be used for testing in memory-based CF recommenders. In the remainder of the paper we give an overview of our proposed Group Testing Framework and we report on the evaluations we conducted using this framework.

Overall, larger-scale synthetic evaluations for group recommendation have not focused on traditional memory-based approaches. This may be because it is cumbersome to address group generation, given sparsity constraints in the user-item matrix. Moreover, only limited attention has been given to evaluation based on predictions, rather than ranking. Our evalution approach addresses these issues.

3. GROUP TESTING FRAMEWORK

We have developed a group testing framework in order to support evaluation of group recommender approaches. The framework is used to generate synthetic groups that are parametrized to test different group contexts. This enables exploration of various parameters, such as group diversity. The testing framework consists of two main components. The first component is a group model that defines specific group characteristics, such as group coherence. The second component is a group formation mechanism that applies the model to identify compatible groups from an underlying single-user data set, according to outcome parameters such as the number of groups to generate.

3.1 Group Model

In simulating groups of users, a given group will be defined based on certain constraints and characteristics, or *group model*. For example, we might want to test recommendations based on different levels of intra-group similarity or diversity. For a given dataset, the group model defines the space of potential groups for evaluation. While beyond the scope of this paper, we note that the group model for evaluation could include inter-group constraints (diversity across groups) as well as intra-group constraints (similarity within groups).

3.1.1 Group Descriptors

Gartrell et al. [10] use the term "group descriptors" for specific individual group characteristics (social, expertise, dissimilarity) to be accounted for within a group model. We adopt the *group descriptor* convention to refer to any quantifiable group characteristic that can reflect group structure and formation. Some of these group descriptors that can reflect group structure are user-user correlation, number of co-rated items between users and demographics such as age difference. We use these group descriptors to identify relationships between user pairs within a single user data set.

3.1.2 Group Threshold Matrix

A significant set of typical group descriptors can be evaluated on a pairwise basis between group members. For example, group coherence can be defined as a minimum degree of similarity between group members, or a minimum number of commonly rated items. We employ such pairwise group descriptors as a foundational element in generating candidate groups for evaluation. We operationalize these descriptors in a binary matrix data structure, referred to as the *Group Threshold Matrix* (GTM). The GTM is a square $n \times n$ symmetric matrix, where n is the number of users in the system, and the full symmetric matrix is employed for group generation. A single row or column corresponds to a single user, and a binary cell value represents whether the full set of pairwise group descriptors holds between the respectively paired users.

To populate the GTM, pairwise group descriptors are evaluated across each user pair in a given single-user dataset. The GTM enables efficient storage and operations for testing candidate group composition. A simple lookup indicates whether two users can group. A bitwise-AND operation on those two user rows indicates which (and how many) other users they can group with together. A further bitwise-AND with a third user indicates which (and how many) other users the three can group with together, and so on. Composing such row- (or column-) wise operations provides an efficient foundation for a generate-and-test approach to creating candidate groups from pairwise group descriptors.

3.2 Group Formation

Once the group model is constructed it can be applied to generate groups from any common CF user-rating data models as the underlying data source. The group formation mechanism applies the set of group descriptors to generate synthetic groups that are valid for the group model. It conducts an exhaustive search through the space of potential groups, employing heuristic pruning to limit the number of groups considered. Initially, individual users are filtered based on group descriptors that can be applied to single users (e.g., minimum number of items rated). The GTM is generated for remaining users. Baseline pairwise group descriptors are then used to eliminate some individual users from further consideration (e.g., minimum group size). The GTM is used to generate-and-test candidate groups for a given group size.

To address the issue of modeling actual group preferences for evaluating system predictions, the framework is tuned to identify groups where all group members gave at least one co-rated item the exact same rating among all group members. Such identified "test items" become candidates for the testing set in the evaluation process in conjunction with the corresponding group. We note that there are many potential approaches to model agreement among group members. In this implementation we choose the most straightforward approach, where the average rating among group members is equal to the individual group member rating for that item as a baseline for evaluation. We do not currently eliminate "universally popular" items, but enough test items are identified that we do not expect such items to make a significant difference. A common practice in evaluation frameworks is to divide data sets into test and target data sets. In this framework the test data set for each group would consist of the identified common item or items for that group.

4. EVALUATION

4.1 Baseline Collaborative Filtering

We implement the most prevalent memory-based CF algorithm, neighborhood-based CF algorithm [12, 22]. The basis for this algorithm is to calculate the similarity, w_{ab}, which reflects the correlation between two users a and b. We measure this correlation by computing the Pearson correlation defined as:

$$w_{ab} = \frac{\sum_{i=1}^{n}[(r_{ai} - \overline{r}_a)(r_{bi} - \overline{r}_b)]}{\sqrt{\sum_{i=1}^{n}(r_{ai} - \overline{r}_a)^2 \sum_{i=1}^{n}(r_{bi} - \overline{r}_b)^2}} \quad (4)$$

To generate predictions a subset of the nearest neighbors of the active user are chosen based on their correlation.

We then calculate a weighted aggregate of their ratings to generate predictions for that user. We use the following formula to calculate the prediction of item i for user a:

$$p_{ai} = \overline{r}_a + \frac{\sum_{b=1}^{n}[(r_{bi} - \overline{r}_b) \cdot w_{ab}]}{\sum_{b=1}^{n} w_{ab}} \quad (5)$$

Herlocker et al. [12] noted that setting a maximum for the neighborhood size less than 20 negatively affects the accuracy of the recommender systems. They recommend setting a maximum neighborhood size in the range of 20 to 60. We set the neighborhood size to 50 we also set that as the minimum neighborhood size for each member of the groups we considered for evaluation. Breese et al. [6] reported that neighbors with higher similarity correlation with the target user can be exceptionally more valuable as predictors than those with the lower similarity values. We set this threshold to 0.5 and we only consider the ones based on 5 or more co-rated items.

4.2 Group Prediction Aggregation

Previous group recommender research has focused on several group aggregation strategies for combining individual predictions. We evaluate the three group aggregation strategies which are outlined in section 2.1 as representative RGPMs. We compare the performance of these three aggregation strategies with respect to group characteristics: group size and the degree of similarity within the group.

4.3 Data Set

To evaluate the accuracy of an aggregated predicted rating for a group we use the MovieLens 100K ratings and 943 users data set. Simulated groups were created based on different thresholds defined for the group descriptors. The two

Table 1: Degrees of Group Similarity

Similarity Level	Definition $\forall i, j \in G$
High	$w_{ij} \geq 0.5$
Medium	$0.5 > w_{ij} \geq 0$
Low	$0 > w_{ij}$

Table 2: Similarity Statistics for Test Data Set

Degree of Similarity	Number of Valid Correlations	Average User-User Similarity
High	39,650	0.65
Medium	192,522	0.22
Low	95,739	-0.25

descriptors we varied were group size and degree of similarity among group members. We presume the same data set that is used to create the simulated groups is the same data set used to evaluate recommendation techniques.

By varying the thresholds of the group descriptors used to create the group threshold matrix we were able to represent groups of different characteristics, which we then used to find and generate groups for testing. One aspect we wanted to investigate is the affect of group homogeneity and size on the different aggregation methods used to predict a rating score for a group using the baseline CF algorithms defined in section 4.1. To answer this question we varied the threshold for the similarity descriptor and then varied the size of the group from 2 to 5. We defined three similarity levels: high, medium and low similarity groups as outlined in Table 1 where the inner similarity correlation between any two users i, j belonging to group G is calculated as defined in equation 1.

To ensure significance of the calculated similarity correlations we only consider user pairs that have at least 5 common rated items. For the MovieLens data set used we have a total of 444153 distinct correlations (943 taking two combinations at a time). For the three similarity levels defined previously the total correlation and average correlation are outlined in Table 2.

Table 3 reflects the GTM group generation statistics for the underlying data set used in our evaluation. Total combinations field indicate the number of possible group combinations that can be formed giving user pairs that satisfy our group size threshold descriptor. The valid groups field indicates the number of possible groups that satisfy both the size and similarity threshold whereas the testable groups are valid groups with at least one identified test item as described in section 3.2. As we increase the size of the groups to be created the number of combinations the implementation has to check increases significantly. We can also see that the number of testable groups is large in comparison to the number of groups used in actual user studies. As of this writing and due to system restrictions we were able to generate all testable groups for group size 2 and 3 across all similarity levels, group size 4 for low and high similarity level and group size 5 for the high similarity level.

4.4 The Testing Framework

The framework creates a Group Threshold Matrix based on the group descriptor conditions defined. In our implementation of this framework the group descriptors used to define inputs for the group threshold matrix are the user-user correlation and the number of co-rated items between any user pair. This forms the group model element of the testing framework. For the group formation element we varied the groups size and for each group the similarity category, 5000 testable groups were identified (with at least one common rating across group members). A predicted rating was computed for each group member and those values were aggregated to produce a final group predicted rating. Table 3 gives an overview of the number of different group combinations the framework needs to consider to identify valid, and testable groups. The framework exploits the possible combinations to identify groups where the group descriptors defined are valid between every user pair belonging to that group this is then depicted in the GTM.

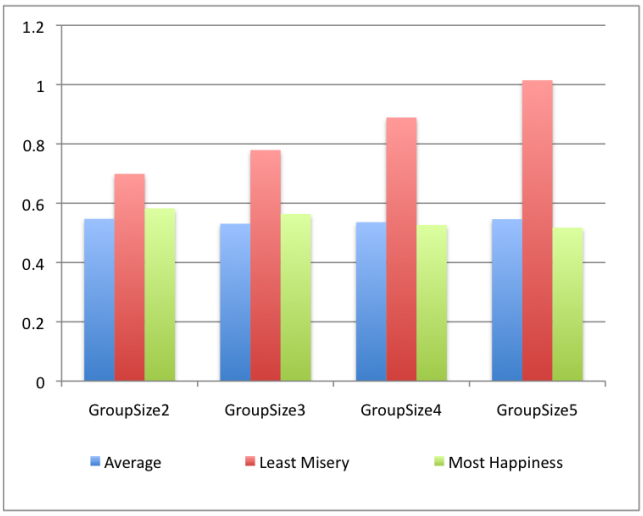


Figure 1: RMSE - High degree of similarity.

We then utilized the testing framework to assess the predicted rating computed for a group based on the three defined aggregation strategies in section 4.2. We compared the group predicted rating calculated for the test item to the actual rating using MAE and RMSE across the different aggregation methods.

It is worth noting here that just like any recommendation technique quality depends on the quality of the input data, the quality of the generated test set depends on the quality of the underlying individual ratings data set when it comes to the ability to generate predictions. For example, prediction accuracy and quality decrease due to sparsity in the original data set.

5. RESULTS AND DISCUSSION

Our evaluation goal is to test group recommendation based on traditional memory-based collaborative filtering techniques, in order to provide a basis of comparison that covers (1) synthetic group formation for this type of approach, and (2) group evaluation based on prediction rather than ranking. We hypothesize that aggregation results will support previous research for the aggregation strategies tested. In doing so, we investigate the relationship between the group's coherence, size and the aggregation strategy used. Figures 1-6 reflect the MAE and RMSE for these evaluated rela-

Table 3: Group Threshold Matrix Statistics

		2	3	4	5
High Similarity >= 0.5	Total Combinations	39,650	1,351,657	40,435,741	1,087,104,263
	Valid Groups	39,650	226,952	417,948	390,854
	Testable Groups	37,857	129,826	129,851	71,441
Medium >=0 < 0.5	Total combinations	192,522	30,379,236	3,942,207,750	434,621,369,457
	Valid groups	192,522	17,097,527		
	Testable groups	187,436	11,482,472		
Low similarity < 0.0	Total combinations	95,739	7,074,964	421,651,608	21,486,449,569
	Valid groups	95,739	1,641,946	6,184,151	
	Testable groups	87,642	470,257	283,676	

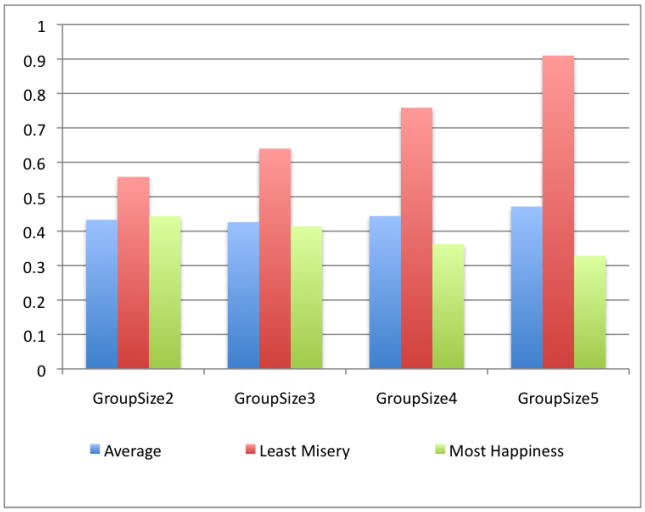


Figure 2: MAE - High degree of similarity.

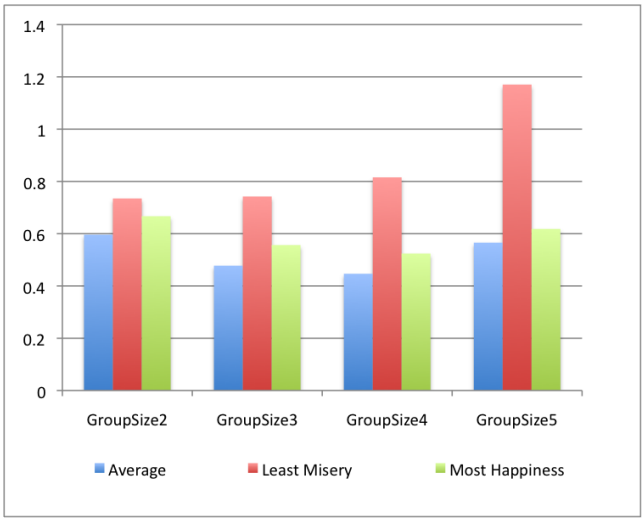


Figure 3: RMSE - Medium degree of similarity.

tionships. Examining the graphs for the groups with high similarity levels, Figures 1 and 2 show that average strategy and most happiness perform better than least misery. We conducted a *t-test* to evaluate the results significance and found that both MAE and RMSE for average and most happiness strategies, across all group sizes, significantly outperform the least misery strategy ($p<0.001$). For group sizes 2 and 3 there was no significant difference between the average and most happiness strategies ($p>0.01$). For group sizes 4 and 5 most happiness strategy performs better than the average strategy ($p<0.001$). Both least happiness and average strategies performance decreases as the group size grows. This indicates that a larger group of highly similar people are as happy as their happiest member.

Figures 3 and 4 show the RMSE and MAE for groups with medium similarity levels. The average strategy performs significantly better than most happiness and least misery across group sizes 2,3 and 4 ($p<0.001$). For the groups of size 5 there was no significant difference between average and most happiness strategies ($p>0.01$). For groups with medium similarity level the least misery strategy performance is similar to the groups with high coherency levels.

Figures 5 and 6 show the results for the groups with low similarity level. Examining the RMSE and MAE in these graphs the average strategy performs best across all group sizes compared to the other two strategies. MAE and RMSE for the average strategy for all group sizes with low

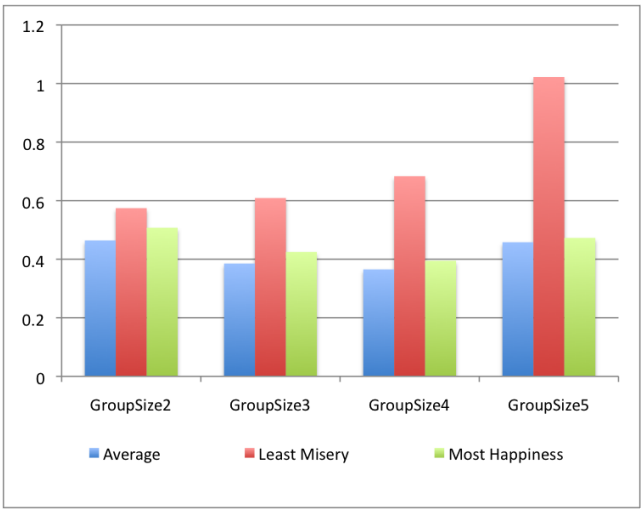


Figure 4: MAE - Medium degree of similarity.

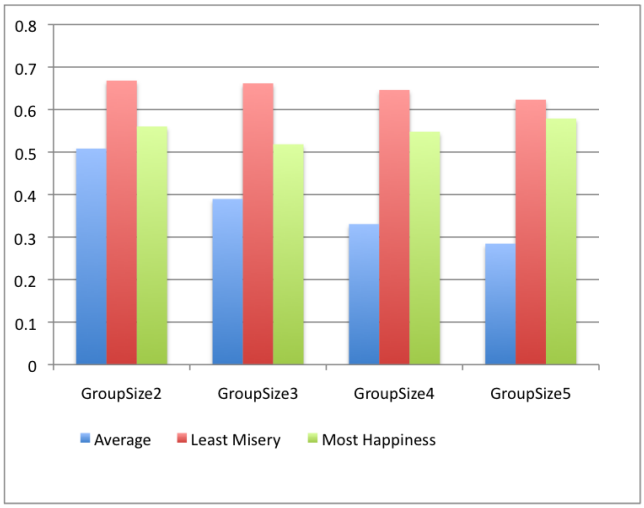


Figure 5: RMSE - Low degree of similarity.

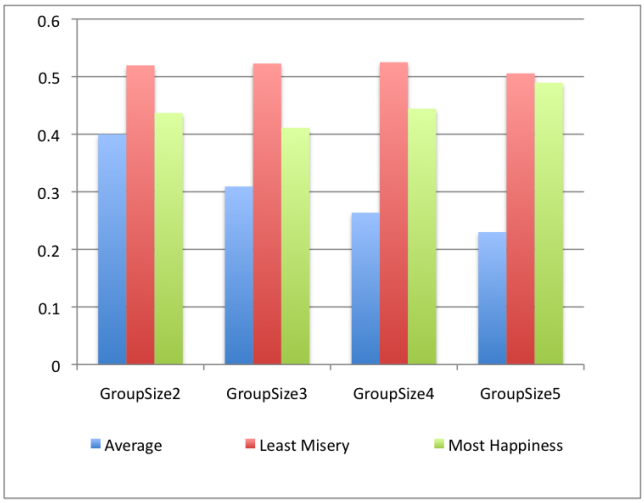


Figure 6: MAE - Low degree of similarity.

coherency had a statistically significant p value ($p<0.001$) compared to both least misery and most happiness strategies. Inconsistent with the groups with high coherency, for groups with low coherency the most happiness performance starts to decrease as the group size increases while the performance of the least misery strategy starts to increase.

These evaluation results indicate that in situations where groups are formed with highly similar members most happiness aggregation strategy would be best to model the RGPM while for groups with medium to low coherency average strategy would be best. These results using the 5000 synthesized groups for each category coincide with the results reported by Gartrell using real subjects. Gartrell defined groups based on the social relationships between the group members. They identified three levels of social relationships (couple, acquaintance and first-acquaintance) that might exist between group members. In their study to compare the performance of the three aggregation strategies across these social ties, they reported that for the groups of two members with a social tie defined as couple the most happiness strategy outperforms the other two. For the acquaintance groups, these groups had 3 members, the average strategy performs best, while for the first-acquaintance, they had one group with 12 members, the least misery strategy outperforms the best. It is apparent that their results for the couple groups performance is equivalent to our high-coherency groups, the acquaintance groups maps to the medium-coherency groups while the first-acquaintance groups follow the low-coherency groups. Masthoff studies reported that people usually used average strategy and least misery since they valued fairness and preventing misery. It is worth noting that her studies evaluated these strategies for groups of size 3 only without any reference to coherency levels.

6. CONCLUSION

As group-based recommender systems become more prevalent, there is an increasing need for evaluation approaches and data sets to enable more extensive analysis of such systems. In this paper we developed a group testing framework that can help address the problem by automating group formation resulting in generation of groups applicable for testing in this domain. Our work provides novel coverage in the group recommender evaluation space, considering (1) focus on traditional memory-based collaborative filtering, and (2) employs precise overlap across individual user ratings for evaluating actual group preference. We evaluated our framework with a foundational Collaborative Filtering neighborhood-based approach, prediction accuracy, and three representative group prediction aggregation strategies. Our results show that for small-sized groups with high-similarity among their members average and most happiness perform the best. For larger size groups with high-similarity performs most happiness performs better. For the low and medium similarity groups, average strategy has the best performance. Overall, this work has helped to extend the coverage of group recommender evaluation analysis, and we expect this will provide a novel point of comparison for further developments in this area. Going forward we plan to evaluate various parameterizations of our testing framework such as more flexible AGPM metrics (e.g. normalizing the ratings of the individual users).

7. REFERENCES

[1] S. Amer-yahia, S. B. Roy, A. Chawla, G. Das, and C. Yu. Group recommendation: Semantics and efficiency. *Proceedings of The Vldb Endowment*, 2:754–765, 2009.

[2] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso. Intrigue: Personalized recommendation of tourist attractions for desktop and hand held devices. *Applied Artificial Intelligence*, pages 687–714, 2003.

[3] C. Baccigalupo and E. Plaza. Poolcasting: A social web radio architecture for group customisation. In *Proceedings of the Third International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, pages 115–122, Washington, DC, USA, 2007. IEEE Computer Society.

[4] L. Baltrunas, T. Makcinskas, and F. Ricci. Group recommendations with rank aggregation and collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 119–126, New York, NY, USA, 2010. ACM.

[5] S. Berkovsky and J. Freyne. Group-based recipe recommendations: analysis of data aggregation strategies. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 111–118, New York, NY, USA, 2010. ACM.

[6] J. S. Breese, D. Heckerman, and C. M. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In G. F. Cooper and S. Moral, editors, *Proceedings of the 14^{th} Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[7] D. L. Chao, J. Balthrop, and S. Forrest. Adaptive radio: achieving consensus using negative preferences. In *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, GROUP '05, pages 120–123, New York, NY, USA, 2005. ACM.

[8] Y.-L. Chen, L.-C. Cheng, and C.-N. Chuang. A group recommendation system with consideration of interactions among group members. *Expert Syst. Appl.*, 34:2082–2090, April 2008.

[9] A. Crossen, J. Budzik, and K. J. Hammond. Flytrap: intelligent group music recommendation. In *Proceedings of the 7th international conference on Intelligent user interfaces*, IUI '02, pages 184–185, New York, NY, USA, 2002. ACM.

[10] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada. Enhancing group recommendation by incorporating social relationship interactions. In *Proceedings of the 16th ACM international conference on Supporting group work*, GROUP '10, pages 97–106, New York, NY, USA, 2010. ACM.

[11] D. Goren-Bar and O. Glinansky. Fit-recommend ing tv programs to family members. *Computers & Graphics*, 28(2):149 – 156, 2004.

[12] J. Herlocker, J. A. Konstan, and J. Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retr.*, 5:287–310, October 2002.

[13] A. Jameson. More than the sum of its members: challenges for group recommender systems. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '04, pages 48–54, New York, NY, USA, 2004. ACM.

[14] A. Jameson and B. Smyth. The adaptive web. chapter Recommendation to groups, pages 596–627. Springer-Verlag, Berlin, Heidelberg, 2007.

[15] J. Masthoff. Group modeling: Selecting a sequence of television items to suit a group of viewers. *User Modeling and User-Adapted Interaction*, 14:37–85, February 2004.

[16] J. Masthoff. Group recommender systems: Combining individual models. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 677–702. Springer US, 2011.

[17] J. F. McCarthy. Pocket restaurantfinder: A situated recommender system for groups. pages 1–10, 2002.

[18] J. F. McCarthy and T. D. Anagnost. Musicfx: an arbiter of group preferences for computer supported collaborative workouts. In *CSCW*, page 348, 2000.

[19] K. McCarthy, M. SalamÃş, L. Coyle, L. McGinty, B. Smyth, and P. Nixon. Cats: A synchronous approach to collaborative group recommendation. pages 86–91, Melbourne Beach, Florida, USA, 11/05/2006 2006. AAAI Press, AAAI Press.

[20] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl. Polylens: a recommender system for groups of users. In *Proceedings of the seventh conference on European Conference on Computer Supported Cooperative Work*, pages 199–218, Norwell, MA, USA, 2001. Kluwer Academic Publishers.

[21] J. A. Recio-Garcia, G. Jimenez-Diaz, A. A. Sanchez-Ruiz, and B. Diaz-Agudo. Personality aware recommendations to groups. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 325–328, New York, NY, USA, 2009. ACM.

[22] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *1994 ACM Conference on Computer Supported Collaborative Work Conference*, pages 175–186, Chapel Hill, NC, 10/1994 1994. Association of Computing Machinery, Association of Computing Machinery.

[23] C. Senot, D. Kostadinov, M. Bouzid, J. Picault, A. Aghasaryan, and C. Bernier. Analysis of strategies for building group profiles. In P. De Bra, A. Kobsa, and D. Chin, editors, *User Modeling, Adaptation, and Personalization*, volume 6075 of *Lecture Notes in Computer Science*, pages 40–51. Springer Berlin / Heidelberg, 2010.

[24] D. Sprague, F. Wu, and M. Tory. Music selection using the partyvote democratic jukebox. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, pages 433–436, New York, NY, USA, 2008. ACM.

Computing Recommendations for Long Term Data Accessibility basing on Open Knowledge and Linked Data

Sergiu Gordea
AIT - Austrian Institute of Technology GmbH
Donau-City-Strasse 1
Vienna, Austria
sergiu.gordea@ait.ac.at

Andrew Lindley
AIT - Austrian Institute of Technology GmbH
Donau-City-Strasse 1
Vienna, Austria
andrew.lindley@ait.ac.at

Roman Graf
AIT - Austrian Institute of Technology GmbH
Donau-City-Strasse 1
Vienna, Austria
roman.graf@ait.ac.at

ABSTRACT

Digital access to our cultural heritage assets was facilitated through the rapid development of the digitization process and online publishing initiatives as Europeana or the Google books project. As Galleries, Libraries, Archiving institutions and Museums (GLAM) created digital representations of their masterpieces new concerns arise regarding the long-term accessibility of digitized and digitally born content. Repository managers of institutions need to take well documented decisions with regard to which digital object representations to use for archiving or long term access to their valuable collections. The digital preservation recommender system presented within this paper aims at reducing the complexity in the process of decision making by providing support for classification and the preservation risk analysis of digital objects. Technical information which is available as linked data in open knowledge sources facilitates the construction of the DiPRec's recommender knowledge base. This paper presents the DiPRec recommender system, a community approach on how to achieve the generation of well founded and trusted recommendations through open linked data and inferred knowledge in the domain of long-term information preservation for GLAM institutions.

Categories and Subject Descriptors

H.3.7 [**Information Systems Applications**]: Digital Libraries; M.8 [**Knowledge Management**]: Knowledge Reuse

General Terms

Digital preservation, Recommender systems

Keywords

Knowledge based recommender, open recommendations, linked open data, preservation planning

1. INTRODUCTION

Knowledge based recommender systems (KBRs) as natural followers of expert systems are nowadays used for supporting the decision making process in multiple application areas as: e-commerce, financial services, tourism, etc. One of the most important challenges of KBRs is the construction of their underlying knowledge base. This is typically composed by sets of factual knowledge, i.e. information describing the application's domain and business rules. Both together enable the drawing of conclusions and support the decisions making process when analyzing the utility of a specific item in a given context as for example, analyzing the effectiveness of digitizing and publishing Mircea Eliade's book "History of Religious Ideas" within Google books.

Even though the world wide web has turned out to be the largest knowledge base, information published lacks an unified well-formed representation and mainly is intended for human readers. The Linked Open Data (LOD)[1] and Open Knowledge[2] initiatives address these weaknesses by describing a method on how to provide structured data in a well-defined and queriable format. By linking together and inferring properties of different independent and publically available information sources like FreeBase[3], DbPedia[4] and Pronom [5] within the specific context of a digital preservation scenario we shortcut the well known challenge of KBRs, the knowledge acquisition bottleneck.

In this paper we present our work carried out in the context of the Assets[6] project with the aim of preparing the ground for digital preservation within Europeana[7]. The Europeana portal serves as a central point for the large public to easily explore and research European cultural and scientific heritage online. It aggregates and collects data on digital resources from galleries, libraries, archives and museums accross Europe and by now manages about 19 million object descriptions collected from more than 15 hundred institutions. Within this very heterogeneous context it is easily understandable that digital objects are encoded in very heterogeneous file formats and versions throughout various different hardware and software content repository systems. Depending on the underlying use case it is likely that mul-

[1] http://linkeddata.org/
[2] http://www.okfn.org/
[3] http://www.freebase.com
[4] http://dbpedia.org/
[5] http://www.nationalarchives.gov.uk/PRONOM/
[6] http://www.assets4europeana.eu/
[7] http://www.europeana.eu/portal/

tiple representations of the same 'physical' object exist at a time. For example in most cases it is useful to provide access copies on demand which are easily distribuatable via the web while the master record needs to adhere to different requirements as for example the institution's long-term scenario and preservation policy.

A key topic in preservation planning is the file formats used for encoding the digital information. The Pronom Unique Identifiers (PUIDs) registry provides persistent, unique and unambiguous identifiers for file formats and therefore takes a fundamental role in the process of managing electronic records. Currently it lists information on about 820 different PUIDs. While some of the formats are properly documented, open-source and well supported, others may be outdated, redeemed by software vendors and no longer functional in modern operating systems. As always the the binary file's dependencies on the underlying platform, its configuration (codecs, plugins, etc.) as well as the rendering software are responsible on generating a concrete user performance, it is vital to have a solid understanding on all of them. This process is costly and requires a high degree of engineering expertise. Many of the GLAM institutions already outsource IT related activities and don't have the resources to keep track of the required level of complexity in house.

The Digital Preservation Recommender (DiPRec) system addresses the topics of 'preservation watch' and 'preservation policy recommendation'. It proposes a solution in the domain of digital long-term preservation for making documented recommendations based on risk scores, while the underlying knowledge base is built through a linked data approach. Information from FreeBase, DbPedia and Pronom in the areas of file formats, file conversions tools, hardware and software vendors is taken into account. The main contribution of this paper consists in the integration of open (general or domain specific) data when constructing knowledge based recommendations. The "knowledge acquisition bottleneck" and the high costs of setting up and maintaining KBRs are still an impediment for extensively adoption by the industry. Recommendations provided by DiPRec are meant to support GLAM institutions across Europe in the process of analyzing their digital assets. The technical foundation and the explanation of the DiPRec recommendations are computed on top of shared and collaboratively built data sources, trust in the area of LOD and digital preservation is a key issue which has been left out for this paper due to simplicity.

The novelty of our work consists in combining expert tools (as File, Droid or Fido) and automated object identification processes, with structured information (e.g. technical information on file formats) from open data repositories. This information is use for infering new knowledge, calculate preservation risks and finally for computing recommendations on preservation actions in the domain of digital long-term preservation. We present the rationale used for the construction of the DiPRec recommender by presenting concrete examples of a given content analysis which was provided for the Assets project. The rest of the paper is organized as follows; in Section 2 we present related work carried out on recommender systems and in the field of digital preservation. Section 3 highlights the architecture of DiPRec by comparing it against the construction of classical KBRs. The functionality provided by our system is explained in detail through a concrete example on the TIFF file format. The evaluation of our approach is presented in Section 4 by analyzing the digital collections of the Assets project. This is followed in the last Section of the paper (nr. 5) by the summarization of the concluding remarks for our work.

2. RELATED WORK

Knowledge Based Recommender systems gained broad popularity in e-commerce and e-tourism [7, 11, 24, 19] applications supporting customers in their decision making processes. The two most popular use cases are guidance through large and complex product offers (e.g. trip organization, feature selection of technical equipment) as well as accompanying the process of high cost decision making (e.g. financial investments). When designing the DiPRec recommender we took into consideration the Advisor Suite [12] and Planets Testbed infrastructure [16]. The main component of the Advisor Suite is a multipurpose workbench which offers support and advanced graphical user interfaces for constructing knowledge based recommenders. Advisor Suite features include the import of product catalogues, visual editing of a recommendation workflow and the generation of a runtime environment. The Planets[8] project focused on constructing practical services and tools for establishing empirical evidence in the process of informed decision making in the area of digital long-term preservation. A major achievement was the definition of basic nouns and verbs for core preservation operations. This allows to easily combine and swap tools within a preservation workflow and lead to a number of over fifty preservation services. Available services were deployed and tested within the Planets Testbed [22], a uniform environment for experimentation under well-defined and controlled surroundings. It provides automated quality assurance support for tools like DROID[9], JHOVE[10] and the eXtensible Characterisation Languages[11][5].

A key topic in preservation planning is the process of evaluating objectives under the limitation of well-known constraints. A state of the art report on technical requirements and standards as well as available tools to support the analysis and planning of preservation actions is given in [2]. Strodl et al. present the Planets preservation planning methodology Plato[12] by an empirical evaluation of image scenarios [21] and demonstrate specific cases of recommendations for image content in four major National Libraries in Europe[4]. After eliciting information regarding the preservation scenario (user requirements) the Plato tool is able to recommend specific preservation actions [3] for a given scenario. The tool was specifically designed to work on samples of the underlying data set and therefore is able to make use of XCL or similar tools for automated quality assurance and semi-automated evaluation of objectives. In contrast to these scenario evaluations, DiPRec aims at collecting information on a broader range from open linked data registries and dynamic knowledge sources. It can evaluate more general, even 'non-technical' objectives (e.g. what is the risk that no software vendor will support old formats like Word

[8] http://www.planets-project.eu/
[9] http://droid.sourceforge.net/
[10] http://hul.harvard.edu/jhove/
[11] http://planetarium.hki.uni-koeln.de/public/XCL/
[12] http://www.ifs.tuwien.ac.at/dp/plato/intro.html

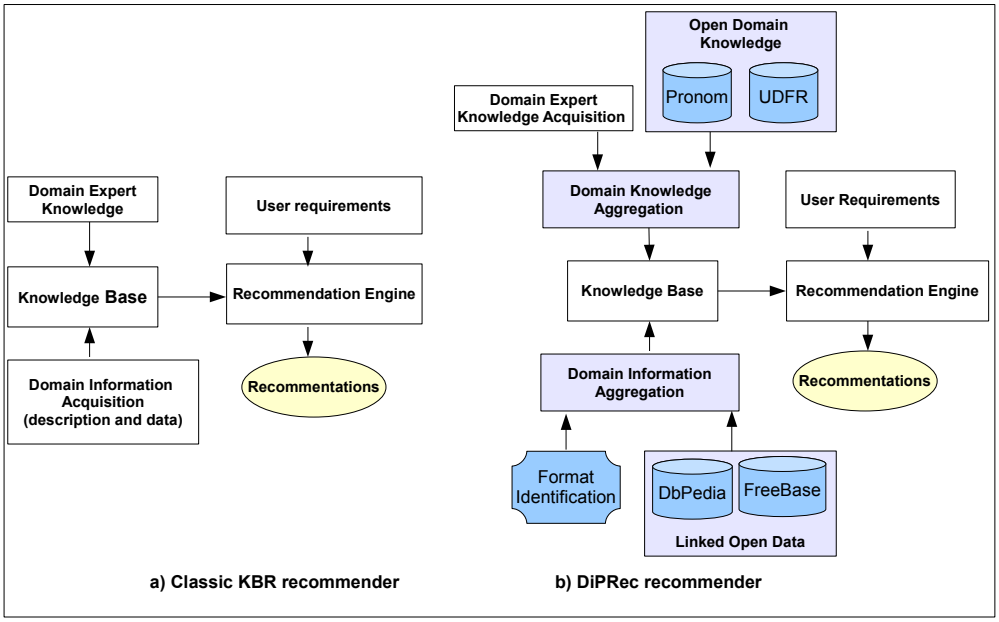


Figure 1: A comparison of regular KBRs and DiPRec recommender processes

3 documents?). This is a significant improvement over the Plato tool where all this information needs to provided by domain experts.

The Scape[13] project is one of the major current initiatives [18] which is partially funded by the European Union's FP7 on institutional preservation requirements. The project addresses besides the issues of scalable preservation and quality-assured preservation workflows also the topic of policy-based preservation planning and watch.

The paradigms of semantic Web and linked open data [6] transform the web from a pool of information into a valuable knowledge source of data according to the definitions of a knowledge management theory [17]. The exploitation of linked data as knowledge source for recommender system started as research topic in the last few years and was first applied to improve case-based and collaborative filtering recommenders [10, 9, 20]. In [20] the authors present the Talis Aspire system which is able to assists educational staff in picking educational web resources. The employment of linked data in collaborative filtering and case-based reasoning was explored by Heitmann and Hayes in [9] and [10].

3. SYSTEM OVERVIEW

Typically the creation of classic knowledge based recommender systems consists of three main tasks. Dealing with the collection of detailed descriptions of products offers is followed by the process of constructing a recommendation knowledge base (see section 3.2). At runtime user requirements elicitation takes place and recommendations are computed based on the underlying recommendation knowledge base and the items that match the given user requirements. DiPRec follows the same process but improves the way the knowledge base is built in order to reduce the efforts spent on domain knowledge acquisition. This is especially relevant for being exploited in GLAM preservation scenarios, where the underlying knowledge base contains broader information than the domain specific KBRs. Within the DiPRec recommender the Domain Information Aggregation module is responsible for collecting file format related information (e.g. formats, vendors, applications, etc.) from the open knowledge bases Pronom, DBPedia and Freebase. Furthermore the Domain Knowledge Aggregation module combines the outcome of a risk analysis process with the knowledge manually provided by domain experts. Figure 1 compares the process used by regular KBRs and the one presented by DiPRec which enhances the process of building the underlying knowledge base. In the following sections we present extended details on how the knowledge base of DiPRrec is built by using as example the Tagged Image File Format (TIFF).

The TIFF format is still very popular among the publishing industry, as it is a very adaptable file format although it did not have a major update since 1992. It was originally created by Aldus and since 2009 it is now under control of Adobe Systems. There are a number of extensions available (e.g. TIFF/IT, TIFF-FX) which have been based on the TIFF 6.0 specification, but not all of them are broadly used. A standard and broadly accepted approach in the archiving world is the migration of TIFF encoded content to the JPEG2000 format. In [4, 2] one can find the context in which several content providers took the decision to perform this kind of content migration. However within these scenarios, the context evaluation and the recommendation were computed by domain experts and by expert systems.

The DiPRec system, on the one hand applies to the approach of well-documented and trackable decision making, and at the same time it uses a semi-automatic approach on domain knowledge acquisition. This reduces the human effort invested by domain experts when providing reservation recommendations, reduces the financial efforts invested in the context evaluations, and in the same time is able to offer good quality recommendations.

[13] http://www.scape-project.eu/

FILE FORMAT DESCRIPTION	
Format Name	Tagged Image File Format (P), Tagged_Image_File_Format (D), Tagged_Image_File_Format(F)
Pronom Id	fmt/10 (P)
Mime Type	/media_type/image/tiff-fx, /media_type/image/tiff (F), image/tiff(P)
File Extensions	.tiff, .tif (D)
Current Version	6 (P)
Current Version Release Date	03 Jun 1992 (P)
Software License	Proprietary software (D)
Software	QuickView Plus, Acrobat, AutoCAD, CorelDraw, Freemaker, GoLive, Illustrator, Photoshop, Powerpoint (P), SimpleText, Seashore, Imagine (D)
Software Homepage	`http://adobe.com/photoshop(D)`
Operating System	PC, Mac OS X, Microsoft Windows (D)
Genre	Image (Raster) (P), Image file format (I), SimpleText - Text editor, Adobe Photoshop - Raster graphics editor (D)
Open Format	none (P)
Standards	ISO 12639:2004 (W)
Vendors	Aldus, Adobe Systems, Apple Computer, now Apple Inc., Microsoft (D), Adobe Systems Incorporated (P), Aldus Corporation (P)
VENDOR DESCRIPTION	
Organization Name	Adobe Systems
Country	United States (P)
Foundation date	Dec 1982 (F)
Number of Employees	6068 (Jan 2007), 8660 (2009)(F), 9,117 (2010)(W)
Revenue	3,579,890,000 US$ (Nov 28, 2008) (F)
Homepage	`http://adobe.com/photoshop(F)`

Table 1: File format and vendor description. (Information sources P = Pronom, D = DBPedia, F = Freebase, W = Wikipedia)

3.1 Domain Information Aggregation

Differently to the e-commerce domain where KBRs import detailed item descriptions from product catalogs there is no such catalog for computer file formats. The Unified Digital Format Registry (UDFR)[14] project was started in 2009 by a group of Universities and GLAM institutions with the aim of building a single, shared technical registry for file formats based on a semantic web and linked data approach. The project is based on the Pronom database which provides basic information about a large number of file formats and will be extended by data on migration pathways and available software/tools. The registry should be available from the beginning of 2012. As Pronom data is not rich enough to build a recommendation and reasoning mechanism for preservation scenarios of file formats on top, we collect additional information sources and aggregate them into a single homogeneous property representation in the recommender's knowledge base. DiPRec uses two types of operations for aggregating domain information:

- data unification: the data representation retrieved from different knowledge bases is unified and combined under the DiPRecs property model definition. For example, the number of software tools supporting a given file format is calculated over different data sources. The individual object's namespace, the transformation process of values, the query on how to extract a given record, etc. are preserved and are part of the property's model representation.

- property composition: more abstract properties which require a hierarchical composition are computed by aggregating basic properties by weighted numbers. The model on property definition is meant to be kept very simple. For example "supported by major vendors" will check if at least one of the software companies is considered to fulfill this requirement by combining the properties like "NUMBER OF EMPLOYEES", "VENDOR REVENUE"). See Table 2.

When aggregating domain information we are interrogating external knowledge sources like DbPedia and Freebase which manage huge amounts of linked open data triples. This allows us to extract fragmental descriptions on file formats, software applications and vendors supporting given file formats (see Table 1). DbPedia allows to post sophisticated queries using SPARQL query and OWL ontology languages [13] for retrieving data available in Wikipedia. Freebase [15] is a practical, scalable semantic database for structured knowledge and is mainly composed and maintained by community members. Public read/write access to Freebase is allowed through an graph-based query API using the Metaweb Query Language (MQL) [6]. PRONOM data is released as linked open data and is accessible through a public SPARQL endpoint.

AGGREGATED PROPERTIES	
File format related	
Is supported by major software vendors?	yes
Is an open file format?	no
Is widely supported by current web browsers?	yes
Which versions officially supported by vendor?	6.0
Which versions are frequently used?	6.0
Image file compression supported?	yes
Preservation related metadata	
Is creator information available?	yes/no
Is publisher information available?	yes/no
Is digital rights information available?	yes/no
Is file migration allowed?	yes/no
Object creation date?	datetime
Is an object preview available?	URL

Table 2: Sample compound properties.

[14] `http://www.udfr.org/`

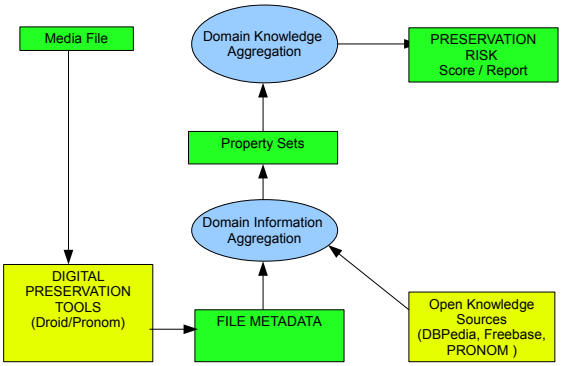


Figure 2: Domain knowledge aggregation process.

3.2 Domain Knowledge Aggregation

Pronom as presented before is a viable resource for anyone requiring impartial and definitive information about the file formats, software products and other related data. Extremely valuable to the DiPRec recommender is the information related to the file conversion tools based on a given PUID. Therefore we employ the Droid [15] characterization service for automatically extracting technical metadata and identifying file formats from physical media files. This metadata is then used in conjunction within the domain knowledge aggregation process presented in the Fig. 2

The risk analysis module is in charge of evaluating information previously aggregated in the DiPRec knowledge base for a given record at hand over following (exemplary) dimensions of digital preservation:

- Web accessibility: Dissemination copies are published and accessible on e.g. the content provider's web portal. There should be previews of objects (e.g. thumbnails for images, video summaries, short intro for audio files) and 'rich' object descriptions to increase their visibility and retrieval. The chosen file representations should render in the latest browsers without plugin support and cope with modern features (e.g. pseudo streaming, progressive image display, HTML5, X3D, etc). Content is made available through different exploitation channels.

- Archiving and costs: The decision of following a specific institutional preservation policy for a given technology is heavily influenced by given hardware and budget constraints. Future exploitations on the costs for content exploitation need to be predicted and taken into account.

Other scenarios may include:

- Provenance metadata
- Data exchange and collaborative data enrichment
- Publishing and digital rights management

The definition of preservation dimensions is not orthogonal and therefore certain properties might be involved more than once when computing different risk score. Due to management and maintenance reasons properties are also grouped by sets and a property may belong to one or more property sets. The extent to which a property belongs to a property set and consequently contributes to the risk computation over a given dimension is modeled through the introduction of specific weighting factors (see Equation 1).

The value of the overall risk score for a given collection of objects is computed as a weighted sum over all digital preservation dimensions:

$$R_i = \sum_{ps \in PS_i} w_{ps,i} * \sum_{p \in PROP_{ps}} w_{p,ps} * d(p, PFV(p)) \quad (1)$$

Where R_i represents the preservation risk computed over the dimension i. ps represents the index of the current property set within all sets associated to the dimension i. The $w_{(ps,i)}$ is the weight of the contribution of the property set ps to the dimension i. Similarly, p stands for the index of current properties within the list of properties available in the given property set $PROP_{ps}$. $w_{p,ps}$ denotes the importance of a property p for the property set ps. The distance between the current property and the defined - 'preservation conform' - value for this property is represented through $d(p, PFV(p))$.

3.3 User requirements elicitation

DiPRec is designed to work as a multi-purpose digital preservation support tool which can be used in various scenarios by different types of customers. For examples the tool may support content providers in analyzing the 'preservation friendliness' of their infrastructure, their archiving solutions or the visibility of their artifacts published in the Europeana portal. Recommendations are always to be seen in the context in which the digital objects are used. Within the scope of the Assets project there is the common interest to offer public access to digital assets through the Europeana portal (i.e. web discovery), to provide advance search functionality (i.e. description richness and preservation of provenance information) as well as the topic of the data archiving dimension.

As a result of the requirements elicitation process user profiles are created. A set of multiple choice questions is used to distinguish the relevant dimensions of available preservation objectives. According to different levels of complexity, role and required domain knowledge the system offers a subset of questions which are well understood and the best available choice for a user to express his needs. Fig. 3 presents sample workflow which could be used to determine a given user profile. For example a private user (ut = *private person*) with a solid level of IT knowledge (itk = *expert*) will be asked about preferred encodings and compression types of the digital content, while others would define attributes about storage limitations and upload samples of a given collection.

3.4 Recommendation computation

Differently to classic KBRs where the application's scope is very well delimited in terms of selecting the best matching items in a list of known possibilities, the DiPRec system relies on expressing an institutional preservation context in form of user requirements that are combined with the knowledge acquired about the long term accessibility threatening. We employ tools to evaluate the content of a given collection from a technical point of view and to generate fine grained preservation risk scores. When records are identified to have vulnerabilities on certain preservation dimensions a

[15] `http://sourceforge.net/projects/droid/`

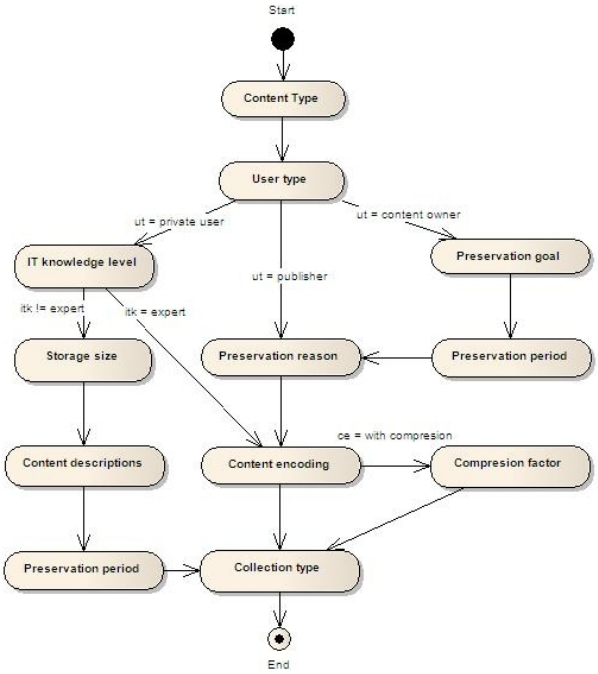


Figure 3: User requirements elicitation workflow

rule based engine as JBoss Drools[16] is used to propose appropriate preservation actions. The set of available business rules are defined by domain experts in form of simple IF-THEN-ELSE rules. These rules are neither complete nor meant to be non-overlapping. Unified tool access for processing executable preservation plans is provided through the Assets preservation normalisation framework which is able to invoke the tools with exactly defined settings and parameter configurations.

IF ($r_{ac} > 0.5$ AND i_a == true AND i_{wa} == true AND $open\_format$ == FALSE)
THEN migrate($preservation\_format$)

IF ($content\_type$ == IMAGE)
THEN preservation_format = (JPEG/2000:1, TIFF/6:0.8)

IF ($file\_format$ == TIFF/5 AND $preservation\_format$ == JPEG/2000)

$$THEN migration\_tool = IMAGE\_MAGICK \quad (2)$$

The preservation recommendations are computed using the constraint solving problems (CSP) theory [8, 11]. Constraints are defined within the preservation actions knowledge base, the CSP context is defined by user profiles and the preservation risks are identified for the given data collection. The recommendations are represented in form of preservation actions. For example, the set of business rules defined above combined with a user profile indicating interest in the dimension of archiving and web accessibility will lead to the following recommendation when analyzing a collection of images in TIFF format:
$migrate(TIFF/5, JPEG/2000, IMAGE\_MAGICK)$
In free text translation, the recommendation will suggest the migration of the files available in $TIFF/5$ format to $JPEG/2000$ by using the $IMAGE\_MAGICK$ software with standard settings.

[16]http://www.jboss.org/drools

4. EVALUATION

The evaluation of the first prototype of DiPRec was conducted within the scope of the Assets project. Ten partners of the project consortium provided metadata and binary content (10 collections with a total size of 516GB contained in 368067 media files) for supporting the development and testing of services developed within the scope of the project. The first step in the evaluation process was the identification of file formats, definition of property sets and the aggregation of the domain knowledge available in open knowledge bases on these file formats.

The Table 3 lists the distribution of file formats by content type. Even the experimental data was taken from a small number of content providers, we discovered a variety of 18 formats in 38 different versions used for encoding the digital content.

Content Type	File Format	# Versions	# Files
TEXT	TXT	1	4
TEXT	DOC	1	16
TEXT	XML	1	20101
TEXT	HTML	1	1205
IMAGE	JPG	8	323332
IMAGE	PSD	1	3
IMAGE	PNG	4	1228
IMAGE	BMP	2	141
IMAGE	GIF	2	1066
IMAGE	TIFF	4	4
IMAGE	PDF	16	25008
AUDIO	MP3	1	3634
VIDEO	FLV	1	9468
VIDEO	MPEG4	1	935
VIDEO	MPEG1	1	3074
VIDEO	MPEG3	1	3074
3D	PLY	1	50
3D	DAE	1	307

Table 3: Distribution of file formats in Assets collections.

The Digital Record Object Identification tool (DROID) version 5, signature file 45 was executed through the Assets preservation normalisation tool suite and was able to successfully identify file formats in 95 percent of the cases through its binary signature method except of the 3D model objects which have not yet been collected by Pronom. Appropriate information on all of the file formats was contained in DbPedia and Freebase and the domain knowledge acquisition process was completed by successfully computing the preservation risk analysis scores.

The second part of the evaluation consisted in computing recommendations for the given content. Therefore, we created a user profile for content providers that are interested in making their content accesible through Europeana. Within this context, the content providers manifest interest for the web accessibility digital preservation dimension.

The highest diversity of file formats was found in the image collections. The recommendation to migrate these files to the JPEG 2000 format didn not get a high priority

and will be performed within the next period of scheduled storage migration. The Image Magick tool was the recommended choise for performing this transformation action. The whole audio content available in Assets was provided in the mp3 format and no recommendation was made for transforming audio collections. The most restrictive constraints for web accessibility are defined for the video content. The pseudostreaming protocol is an advanced technological solution used for distributing information over the web. It allows the user to interact with the media-player and to quickly navigate within the content without the needed to download the entire media file. This protocol is supported by two file formats: flash video (FLV) and MPEG4 with H2.64 video encoding. It has native support in HTML5 and is used in HTML4 with an adequate browser plugin. A part of the Assets content is already available in FLV format and another part is available in MPEG1 or MPEG2. The DiPRec resulting recommendation is to migrate the content to FLV by using the ffmpeg [17] tool.

5. CONCLUSION

Within this paper we introduced the DiPRec recommender system, an expert support tool in the domain of digital long-term preservation for GLAMs. An important contribution of this papers is the exploitation of an open linked data approach for constructing the recommender's knowledge base built upon open registries as DbPedia and Pronom. Since the knowledge acquisition, aggregation and unification process is fully automated it is easy to upgrade the recommender's knowledge base.

We looked at preservation planning which is the process of specifying clearly defined and relevant trees of objectives in a defined preservation dimension and evaluating them within a given (institutional) context to generate well-documented decisions. DiPRec is able to advance the process with inferred community knowledge and reduces the degree of manual evaluation processes or require technical expertise in this process.

Am important concern related to the KBRs is the trust in the provided recommendations. This is especially relevant for the digital preservation domain where we deal with a large amount of multimedia material and the execution of the preservation actions is associated with considerable costs. Within this paper we did not examine the completeness, correctness and quality degree of the underlying data. We however argue that data from open knowledge bases like DbPedia or Freebase could protect from biases introduced by the economical interests of professional companies by its underlying community approach.

The tool has been designed by reusing our past experience in building knowledge based and case based recommender systems [23, 8] and combining it with the expertise of creation long-term preservation infrastructure and applications [14, 1]. Based on this work the Assets normalisation tool suite is able to automate the process of object identification and characterisation and therefore directly integrates within the property evaluation, risk analysis and recommendation process for a given record. We presented a first evaluation of digital content provided by national libraries and archives through the Assets project where the underlying concepts of the DiPRec approach were proven to work adequately.

[17] http://www.ffmpeg.org/

6. ACKNOWLEDGMENTS

This work was partially supported by the EU project "ASSETS - Advanced Search Services and Enhanced Technological Solutions for the European Digital Library" (CIP-ICT PSP-2009-3, Grant Agreement n. 250527).

7. REFERENCES

[1] Aitken, B., Helwig, P., Jackson, A., Lindley, A., Nicchiarelli, E., Ross, S.: The planets testbed: Science for digital preservation. Code4Lib 1(3) (2008), http://journal.code4lib.org/articles/83

[2] Becker, C., Kulovits, H., Guttenbrunner, M., Strodl, S., Rauber, A., Hofman, H.: Systematic planning for digital preservation: evaluating potential strategies and building preservation plans. International Journal on Digital Libraries 10(4), 133–157 (2009), http://dblp.uni-trier.de/db/journals/jodl/jodl10.html#BeckerKGSRH09

[3] Becker, C., Kulovits, H., Rauber, A., Hofman, H.: Plato: a service-oriented decision support system for preservation planning. In: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries. pp. 367–370. ACM, New York (2008), http://publik.tuwien.ac.at/files/PubDat_170832.pdf, vortrag: 8th ACM/IEEE-CS joint conference on Digital libraries (JCDL 2008), Pittsburgh, Pennsylvania; 2008-06-16 – 2008-06-20

[4] Becker, C., Rauber, A.: Four cases, three solutions: Preservation plans for images. Tech. rep., Vienna University of Technology, Vienna, Austria (April 2011)

[5] Becker, C., Rauber, A., Heydegger, V., Schnasse, J., Thaller, M.: A generic xml language for characterising objects to support digital preservation. In: SAC '08: Proceedings of the 2008 ACM symposium on Applied computing. pp. 402–406. ACM, New York, NY, USA (2008)

[6] Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. Int. J. Semantic Web Inf. Syst. 5(3), 1–22 (2009)

[7] Burke, R.D.: Hybrid web recommender systems. In: The Adaptive Web. pp. 377–408 (2007)

[8] Felfernig, A., Gordea, S.: Ai technologies supporting effective development processes for knowledge-based recommender applications. In: SEKE. pp. 372–379 (2005)

[9] Heitmann, B., Hayes, C.: C.: Using linked data to build open, collaborative recommender systems. In: In: AAAI Spring Symposium: Linked Data Meets Artificial IntelligenceŠ. (2010

[10] Heitmann, B., Hayes, C.: Enabling case-based reasoning on the web of data. In: The WebCBR Workshop on Reasoning from Experiences on the Web (2010)

[11] Jannach, D., Zanker, M., Fuchs, M.: Constraint-based recommendation in tourism: A multiperspective case study. J. of IT & Tourism 11(2), 139–155 (2009)

[12] Jannach, D., Zanker, M., Jessenitschnig, M., Seidler, O.: Developing a conversational travel advisor with advisor suite. In: ENTER'07. pp. 43–52 (2007)

[13] Jens, L., Jörg, S., Sören, A.: Discovering unknown connections -the dbpedia relationship finder. In: Proceedings of the 1st Conference on Social Semantic

Web (CSSW). vol. P-113, pp. 99–109. Gesellschaft für Informatik, Leipzig, Germany (2007)

[14] King, R., Schmidt, R., Jackson, A., Wilson, C., Steeg, F.: The planets interoperability framework: An infrastructure for digital preservation actions. In: ECDL09 Proceedings of the 13th European conference on Research and advanced technology for digital libraries. vol. 5714/2009, pp. 425–428. Springer-Verlag (2009), http://dx.doi.org/10.1007/978-3-642-04346-8_50

[15] Kurt, B., Colin, E., Praveen, P., Tim, S., Jamie, T.: Freebase: a collaboratively created graph database for structuring human knowledge. In: SIGMOD '08 Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1247–1249. ACM, New York, NY, USA (2008)

[16] Lindley, A., Jackson, A.N., Aitken, B.: A collaborative research environment for digital preservation - the planets testbed. Enabling Technologies, IEEE International Workshops on 0, 197–202 (2010)

[17] Nonaka, I., Takeuchi, H.: The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press (May 1995)

[18] Orit Edelstein, Michael Factor, R.K.T.R.E.S.P.T.: Evolving domains, problems and solutions for long term digital preservation. iPRES 2011 - 8th International Converence on Preservation of Digital Objects (2011)

[19] Ricci, F., Werthner, H.: Case base querying for travel planning recommendation. Journal of IT & Tourism 4(3-4), 215–226 (2001), http://dblp.uni-trier.de/db/journals/jitt/jitt4.html#RicciW01

[20] Shabir, N., Clarke, C.: Using linked data as a basis for a learning resource recommendation system. In: 1st International Workshop on Semantic Web Applications for Learning and Teaching Support in Higher Education (SemHE'09) (September 2009), http://eprints.ecs.soton.ac.uk/18053/

[21] Strodl, S., Becker, C., Neumayer, R., Rauber, A.: How to choose a digital preservation strategy: evaluating a preservation planning procedure. In: JCDL '07: Proceedings of the 2007 conference on digital libraries. pp. 29–38. ACM, New York, NY, USA (2007), http://doi.acm.org/10.1145/1255175.1255181

[22] Sven Schlarb, Edith Michaelar, M.K.A.L.B.A.S.R.A.J.: A case study on performing a complex file-format migration experiment using the planets testbed. IS&T Archiving Conference 7, 58–63 (2010)

[23] Zanker, M., Gordea, S., Jessenitschnig, M., Schnabl, M.: A hybrid similarity concept for browsing semi-structured product items. In: EC-Web. pp. 21–30 (2006)

[24] Zanker, M., Jessenitschnig, M., Jannach, D., Gordea, S.: Comparing recommendation strategies in a commercial context. IEEE Intelligent Systems 22(3), 69–73 (2007)

Automated Ontology Evolution as a Basis for User-Adaptive Recommender Interfaces

Elmar P. Wach
STI Innsbruck, University of Innsbruck/
Elmar/P/Wach eCommerce Consulting
Hummelsbüttler Hauptstraße 43
22339 Hamburg
+49 172 713 6928

elmar.wach@sti2.at,
wach@elmarpwach.com

ABSTRACT

This research proposes an automated OWL product domain ontology (PDO) evolution (without a human inspection) based on given user feedback and enhancing an existing ontology evolution concept. Its manual activities are eliminated by formulating an adaptation strategy for the conceptual aspects of an automated PDO evolution and establishing a feedback cycle. The adaptation strategy consists of a feedback transformation strategy and a PDO evolution strategy and decides when and how to evolve by evaluating the impact of the evolution on the application. An evolution heuristic and evolution strategies are utilised. The adaptation strategy was validated/ firstly "instantiated" by applying it to a real-world conversational content-based e-commerce recommender system as use case. The evolved PDO is going to be evaluated with an experiment and validated with the use case as well.

Categories and Subject Descriptors

D.2.10 [**Software Engineering**]: Design – *Methodologies*. H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *relevance feedback*. H.3.5 [**Information Storage and Retrieval**]: On-line Information Services – *commercial services, web-based services*. I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods – *representations (procedural and rule-based), semantic networks*. I.2.6 [**Artificial Intelligence**]: Learning – *concept learning, knowledge acquisition*. K.4.3 [**Computers and Society**]: Organizational Impacts – *automation*.

General Terms

Management, Measurement, Experimentation, Standardization.

Keywords

Ontology Evolution, Recommender Systems, Self-Adapting Information Systems, Heuristics.

1. INTRODUCTION

Recommender systems in e-commerce applications have become business relevant in filtering the vast information available in e-shops (and the Internet) to present useful recommendations to the user. As the range of products and customer needs and preferences change, it is necessary to adapt the recommendation process. Doing that manually is inefficient and usually very expensive.

Recommenders based on product domain ontologies[1] (PDO) can extract questions about the product characteristics and features to investigate the user preference and eventually recommend products that match the needs of the user. By changing the PDO, such a recommender generates different questions and/ or their order and herewith adapts the recommender interface to the user preference. Hence, an automated adaptation of the recommendation process can be realised by automatically evolving the PDO[2]. The high cost of the manual adaptation of the recommendation process and the underlying PDO can herewith be minimised.

This research proposes an automated OWL PDO evolution (without a human inspection) based on given user feedback[3] and enhancing an existing ontology evolution concept. Its manual activities are eliminated by formulating an adaptation strategy for the conceptual aspects of an automated PDO evolution and establishing a feedback cycle. Automatically evolving the PDO is more efficient and less expensive than manually doing it. The present research tackles an automated process for the first time (to the best knowledge of the author).

Figure 1 depicts the starting basis schematically.

In the data modelling layer the OWL PDO evolution is induced by different kinds of user feedback, i.e. from external and internal data sources. When evolving the PDO, it can be necessary to adapt instance data (i.e. products) as well in order to keep them correctly annotated. Afterwards, the new PDO version including associated instance data is provided to the application layer. There

[1] A product domain ontology (PDO) is defined as the formal, explicit specification of a shared conceptualisation of a product description based on OWL DL; this definition is derived from [6]

[2] Ontology evolution is defined as the timely adaptation of a PDO by preserving its consistency (a PDO is consistent if and only if it preserves the OWL DL constraints); this definition is derived from [7] and [16]

[3] In order to focus this research on developing an automated ontology evolution, the feedback is given

UCERSTI 2, October 23, 2011, Chicago, IL, USA

and in the external data sources, the effect of the PDO evolution is evaluated and again reported to the data modelling layer which concludes the feedback cycle.

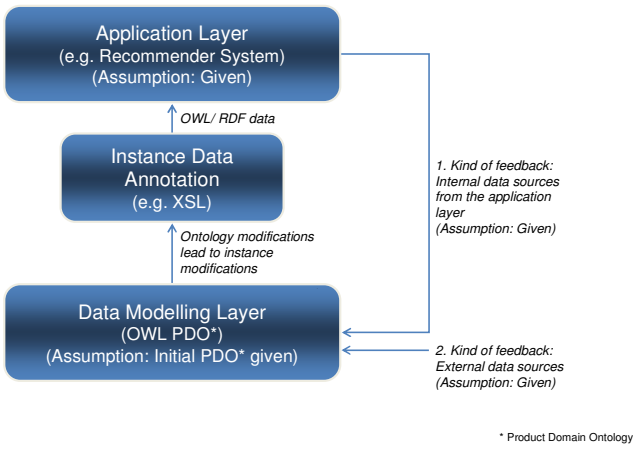


Figure 1. PDO evolution induced by user feedback

The main research question is: How can an automated[4] product domain ontology evolution be realised based on feedback?

2. RELATED WORK

Previous approaches in the topic of this research can be found in concepts for ontology evolution like formulated frameworks for ontology evolution.

[13] focused on the evolution process and have defined six phases consisting of capturing, representation, semantics of change (i.e. a rich description about the semantic role of an ontology entity in order to get more information for solving inconsistencies), implementation, propagation, and validation of ontology changes. This process is implemented in the KAON[5] framework and the Ontologging[6] system. Evolution strategies have been formulated defining elementary and composite changes for executing a change request and eventually deciding the evolution path. [9] focused on detecting ontology changes and have defined five components relating the different change representations to each other. They have proposed a component-based framework for ontology evolution supporting data transformation between two ontology versions, update of remote ontologies, consistent reasoning, verification and approval of ontology changes, and data access to an old ontology via the new one. [14] focused on the user interaction and have provided a usage-based approach implemented in the OntoManager[7] system. The conceptual architecture is based on the MAPE model (Monitor – Analyse – Plan – Execute). The activities of a user are captured in a semantic log and are instances of a user log ontology. The log data is aggregated and visualised helping an ontology manager in adapting the ontology. Eventually, the ontology evolution process guarantees a transfer from one ontology version to another while preserving consistency. [8] focused on handling inconsistency in changing ontologies and have defined a framework consisting of four approaches addressing the consistent ontology evolution, the repairing of inconsistencies, the reasoning with inconsistent ontologies, and multi-version reasoning. For the first three approaches consistency algorithms have been formulated. A consistent ontology evolution is ensured by removing axioms that are structurally connected with the conflicting axioms. [11] focused on collaborative environments and have developed a set of Protégé[8] plugins to support different ontology evolution scenarios. Those include synchronous (i.e. online)/ asynchronous ontology editing, continuous editing/ periodic archiving (i.e. versions), curation (i.e. inspection by a human)/ no curation, and monitored (i.e. record of changes)/ non-monitored ontology changes. The central element is a change and annotation ontology (ChAO) which gathers and provides information about the ontology changes including meta-information like the author and timestamp. [10] introduced a general framework answering the essential questions of what can be changed in an ontology and how each change should be implemented. It is split in five steps comprising the ontology model selection, supported operations, consistency model (i.e. integrity rules), inconsistency resolution, and action selection based on a preference ordering. [18] proposed Evolva, a framework and tool for the whole ontology evolution cycle which decreases user input by making use of background knowledge like lexical databases, online ontologies and unstructured Web documents. It consists of the components information discovery (i.e. extracts content from external data sources manually specified), data validation (i.e. identifies new terms and checks the quality), ontology changes (i.e. integrates the new information to the ontology), evolution validation (i.e. handles conflicts), and evolution management (i.e. manually controlling the evolution (modifying, filtering), records changes and propagates them to dependent ontologies).

Due to the specific challenges of the present research like the automated ontology evolution process, none of the frameworks discussed can be completely used as basis, e.g. all frameworks include a step for the human inspection of the ontology changes before they are executed. The closest work to the research in this paper is [13] – in the six phase evolution process, two steps include manual activities, namely (i) "implementation" in which the implications of an ontology change are presented to the user and have to be approved by her before execution, and (ii) "validation" in which performed changes can get manually validated. The research in this paper aims at eliminating both manual steps in [13] with the adaptation strategy and its implementation. To automate (i), the ontology evolution is conceptualised and implemented as a complete feedback cycle. An insufficient ontology change is indicated by decreased metrics and gets revised according to the evolution strategy chosen. Hence, the ontology changes do not have to get manually approved before execution. To automate (ii), the PDO changes are predefined and application-oriented. Hence, only valid changes are executed, and nobody has to manually validate them.

3. APPROACH AND PROPOSED SOLUTION

The aim of this research is to combine the use of PDO with processing user feedback. The work focuses on how the given

[4] Without human inspection

[5] http://kaon.semanticweb.org

[6] European Commission project IST-2000-28293

[7] German BMBF project SemIPort (08C5939) and European Commission project Ontologging

[8] http://protege.stanford.edu

feedback can lead to a self-improvement of the semantic application by adapting the PDO. In this context self-improvement means that by automatically processing user feedback and evolving the PDO, the defined key performance indicators (KPI) of the application will increase.

The use case is a real-world conversational content-based e-commerce recommender system based on PDO that semantically describe the products offered in e-commerce applications according to GoodRelations[9]. Four types of PDO changes are defined with the following impact on the user dialogue in the recommender system:

- Switching individuals (i.e. properties are related to other individuals within the same class): This leads to a different clustering of the questions
- Switching datatype property ranges (i.e. properties get Boolean ranges instead of string ranges and vice versa (where applicable)): This leads to textual modifications of the questions
- Switching annotation properties label and comment (i.e. properties get different labels and comments extracted from another information source): This leads to textual modifications of the questions (and maybe a need-based sales approach instead of a technology-prone one)
- Changing annotation property priority (i.e. different priority values): This leads to a different ranking of the questions and skips the ones with low priorities

In this paper the PDO change switching individuals is used as an example (confer section 4.2). A digital camera has a feature HDMI. This PDO change defines in which feature-related section the question is nestled whether the camera should offer HDMI.

The success and thus the KPI of an e-commerce recommender are usually defined by the click-out rate (i.e. clicks-to-recommendations) or conversion rate (i.e. customers-to-recommender users). The user gives feedback to the quality of a product recommendation in following the recommendation (i.e. click-out) or even buying the product (i.e. conversion).

In the approach a six step adaptation strategy for the conceptual aspects of an automated PDO evolution has been formulated and a feedback cycle established. The adaptation strategy answers the questions when and how to evolve the PDO by evaluating the impact of the evolution in the precedent feedback cycle. The first question defines the (temporal and causal) trigger initiating the PDO change. Basically, this is receiving and transforming the feedback into ontology input and will be addressed with the feedback transformation strategy. The second question defines the changing of the PDO with annotated instances. This is evolving the PDO and will be addressed with the PDO evolution strategy. Due to space limitations and the focus on realising a user-centric evaluation, the adaptation strategy is not elaborated in this paper. The strategy is used to concisely describe the application for which the automated PDO evolution should be implemented and the impacts of PDO changes on the application behaviour. The interested reader is referred to [17].

[9] www.purl.org/goodrelations

3.1 Evolution Heuristic and Evolution Strategies

The automated ontology evolution is realised by utilising an evolution heuristic and evolution strategies. Those are defined in the fifth step of the adaptation strategy "Decide the adequate PDO evolution". The impact of the PDO change is measured in the Feedback Transformer (confer section 3.2) component by calculating the Success Trend ST for the new user feedback from the application layer and external data sources. The ST is analysed by a heuristic that defines the PDO change to be executed. A heuristic is a strategy that uses accessible and loosely applicable information to solve a problem of a human being or a machine [12] and leads to a solution of a complex problem with simplified conceptual aspects or reduced computation power. [3] mentioned first the term metaheuristic for a computational method that makes few or no assumptions about the problem being optimised and introduced the tabu search metaheuristic [4]. The tabu search enhances a local search (i.e. iteratively improving a criterion in the search space) metaheuristic by using "taboos" – a solution is not executed again according to the criteria defined in the tabu list. The philosophy when utilising a heuristic should be that the highest precedent ST defines the next PDO change to always choose the best evolution. The relevant characteristics of the heuristic have initially to be defined, confer section 4.1. This manual effort is rewarded with a greater conceptual flexibility resulting in an evolution that is more application-oriented. The relevant metrics have to be defined and the calculations formulated.

The PDO evolution is decided based on the ST. In case the feedback includes information extracted from the PDO (e.g. property-based feedback), the subsequent evolution (i.e. type of PDO change) is defined by implementing the ST in the same representation as before (e.g. ontological entity, range), and neither statistical means nor a heuristic has to be applied.

This research proposes to additionally formulate evolution strategies that decide the general evolution behaviour (e.g. executing the same type of PDO change or a rollback) by correlating the types of PDO changes needed to the ST calculated. Additionally, the path for determining the initial ST has to be defined, e.g. the order of the different types of PDO changes and for which PDO they are executed (i.e. ramp-up of the evolution strategies). The philosophy should be that the development (and its strength) of the precedent ST defines the next type of PDO change to distinguish different evolution impacts.

A positive ST means a positive trend (i.e. an increase) of the metrics, a negative the opposite. The larger the figure is, the stronger the development of the metrics (in either direction) from the precedent to the current cycle has been. So, there are two criteria (i.e. ST and its strength) to decide about the next type of PDO change. Basically, there can be two resulting user behaviours in the e-commerce recommender system:

- The user is satisfied with the product recommendation and clicks to see the detail page or order it; in that case the metrics increase, but it still has to be decided if a change should be made
- The user is not satisfied with the product recommendation and leaves the recommender; the metrics decrease, though we do not know why she was not pleased, and a PDO change is advisable

In the first case, one can argue either way – a change is luring to even further increase the metrics. On the other hand, one could keep everything as it is and wait for the next feedback. The latter case is more urging for a change. It has still to be decided if it is a change or just a rollback to retrieve the previous setting. So, it is advisable to define evolution strategies reflecting different behaviours with associated types of PDO changes. In the following, these strategies are predefined and discussed.

Risky Evolution:

An evolution is induced in either case, i.e. a positive or a negative trend. Different types of PDO changes than in the precedent feedback cycle are executed. This behaviour tries to radically improve the metrics by all means and can be described as "always evolve differently". The decision criteria are as follows:

- Increase of the KPI (i.e. $0 \leq ST \leq 1$)
- Decrease of the KPI (i.e. $-1 \leq ST < 0$)

Progressive Evolution:

An evolution depends on the leap in the ST between two consecutive feedback cycles and can be fine-tuned with a threshold defining the trend significance (i.e. the increase of the ST between the precedent and the current cycle). In case of a significant positive trend, the same type of PDO change as in the precedent feedback cycle is executed. In case of a moderately positive trend, a different type of PDO change than in the precedent feedback cycle is executed. In case of a negative trend, it is optional to either do a different type of PDO change than in the precedent feedback cycle or a rollback (to be selected in the administration interface of the Adaptation Manager). This behaviour tries to repeat a significant increase by the same means but gives also the option to revert a negative development. It can be described as "learn from the past". Additionally, the "risk" of the evolution can be adjusted with the threshold. The higher it is the more unlikely the same type of PDO change as in the precedent feedback cycle is executed, and the strategy is tuned towards the Risky Evolution (with a higher threshold). Initially, the threshold is defined to be 20%[10] and can be changed in the administration interface as well. The decision criteria are as follows:

- Significant increase of the KPI (for the beginning, the threshold is defined to be 20%, i.e. $0{,}2 \leq ST \leq 1$)
- Moderate increase of the KPI (i.e. $0 \leq ST < 0{,}2$)
- Decrease of the KPI (i.e. $-1 \leq ST < 0$)

Safe Evolution:

An evolution is induced only by a negative trend. In that case, a rollback is executed. This behaviour tries only to revert a negative development. It can be described as "only revert negative trends". The decision criteria are as follows:

- Increase of the KPI (i.e. $0 \leq ST \leq 1$)
- Decrease of the KPI (i.e. $-1 \leq ST < 0$)

[10] Increase of the ST by 20 basis points between the precedent and the current feedback cycle

Rollback:
This "strategy" reverts the PDO changes from the precedent feedback cycle (i.e. rolling back to the precedent PDO version) and is based on any reason or decision of the manager. It is executed only once but can be manually chosen multiple times. The behaviour can be described as "undo the PDO changes".

The evolution strategies introduced above are considered as basic categories. They can be fine-tuned with regard to the associated types of PDO changes as well as the threshold defining the trend significance. Table 1 sums up the predefined evolution strategies, decision criteria (ST), and the type of PDO changes to be executed in the feedback cycle.

Table 1. Evolution strategy, Success Trend ST, and associated type of PDO change

Evolution Strategy	Decision Criteria	Type of PDO Change
Risky Evolution ("always evolve differently")	$-1 \leq ST \leq 1$	Different than before
Progressive Evolution ("learn from the past")	$0{,}2^* \leq ST \leq 1$ $0 \leq ST < 0{,}2^*$ $-1 \leq ST < 0$	Same as before Different than before Different than before or Rollback
Safe Evolution ("only revert negative trends")	$0 \leq ST \leq 1$ $-1 \leq ST < 0$	None Rollback
Rollback ("undo the PDO changes")	Manually	Rollback

* Increase of the ST by 20 basis points between the precedent and the current feedback cycle

Each evolution strategy besides Rollback ensures an adaptive change of the PDO and thus the recommender interface. By selecting a strategy in the administration interface, the business manager decides how fundamental the evolution will be.

3.2 Implementing the Strategy by Programming an Application

By following the principles of adaptive systems [2], the adaptation strategy is implemented in a new adaptation layer (confer figure 2) consisting of components in which the user feedback gets transformed (i.e. Feedback Transformer) and the respective actions are decided and initiated (i.e. Adaptation Manager). This system creates an evolved PDO with associated instances.

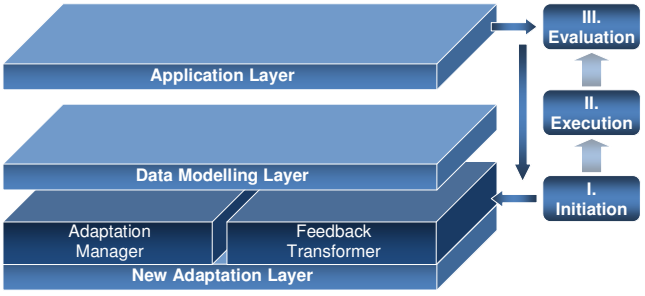


Figure 2. PDO evolution cycle with a new adaptation layer

The whole evolution cycle is based on the generic change process model [1] consisting of three iterative phases and defining four activities:

1. Phase "initiation" – Activities: Requesting the change and analysing/ planning the change
2. Phase "execution" – Activity: Implementing the change
3. Phase "evaluation" – Activity: Verifying/ validating the change

The three layers (i.e. application layer, data modelling layer, and adaptation layer) interact during the three phases of the generic change process model forming the basis of the automated PDO evolution process.

In the first phase "initiation" the different kinds of user feedback are delivered to the adaptation layer and thus a PDO change requested. As the PDO is the backbone of a semantic application, the feedback is assumed to be RDF data. This feedback is converted to ontology input by the Feedback Transformer according to the feedback transformation strategy. The Feedback Transformer accesses the user feedback channels programmatically via SPARQL endpoints and identifies the PDO affected with SPARQL SELECT statements. Eventually, the Feedback Transformer calculates the Success Trends ST for each feedback channel, e.g. by a simple value transformation or by calculating the relative frequencies of the property values in the feedback. Then, the PDO evolution is prepared by identifying the next PDO change with the transformed feedback by the Adaptation Manager. The system has to decide which evolution actions to take according to the PDO evolution heuristic and strategy. The Adaptation Manager analyses the transformed feedback with a tabu search metaheuristic that chooses the PDO change with the highest ST. The tabu criteria are implemented for each type of feedback. Additionally, the predefined evolution strategies (i.e. Risky Evolution, Progressive Evolution, Safe Evolution, Rollback) are implemented and ramped-up. For determining the initial ST, the different types of PDO changes are sequentially executed in an alphabetical order with an exemplary PDO. These values are then valid as starting basis for all PDO. After this phase, the evolution strategy decides whether the (i) same or (ii) another type of PDO change is executed. In (i), a PDO change within the same type of PDO change is executed and ST(t+1) calculated, except a tabu criterion defined by the evolution heuristic is met. In this case, another type of PDO change is executed in contrary to the evolution strategy. In (ii), the type of PDO change and the PDO change to be executed are determined by the evolution heuristic, and ST(t+1) is calculated.

In the second phase "execution" the changes get implemented in the data modelling layer directed by the PDO evolution heuristic and strategy and by retaining a consistent PDO including correctly annotated instance data. In the Adaptation Manager the predefined PDO changes (for the use case they are switching individuals, switching datatype property ranges, switching annotation properties label and comment, changing annotation property priority, confer section 3.) are implemented and thus ensure a consistent ontology evolution. They are executed with SPARQL CONSTRUCT rules or programmatically. Eventually, the versioning is implemented according to the change-based concept and utilising an ontology with annotated logs. The new PDO version with associated instances is provided to the application layer.

The third phase "evaluation" concludes the feedback cycle by measuring the impact of the change. This is done by calculating adequate metrics relating the currently evaluated feedback from the application layer and external data sources reported to the adaptation layer to the precedent feedback.

The process from the feedback type to the resulting type of PDO change is depicted in the activity diagram in figure 3.

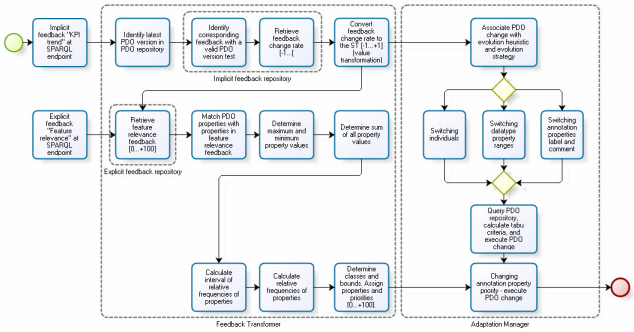


Figure 3. Activity diagram feedback type to type of PDO change

4. EVALUATION AND VALIDATION

The adaptation strategy has been validated/ "instantiated" by applying it to the use case. As this recommender is already used in live applications, it is a real-world scenario. In a conversational approach the actions and modifications done in the adaptation layer mainly lead to a changed user dialogue.

Implicit user feedback is derived from user interactions in the application layer and gathered by unobtrusively monitoring user needs. Explicit user feedback is gathered by extracting information from various websites. Both feedback channels deliver RDF data via separate SPARQL endpoints programmatically accessible.

Applying the adaptation strategy could be done quite smoothly. Only minor aspects of the strategy were clarified, restructured, and reformulated. After having applied the strategy, the use case was concisely described and conceived by the ontology engineer. Moreover, the result formed the basis of the technical specification and thus the development of the adaptation layer.

Due to space limitations the "instantiation" of the adaptation strategy is not completely elaborated in this paper. In the following the evolution heuristic based on tabu search is introduced (excluding its ramp-up).

4.1 Characteristics of the Evolution Heuristic

The evolution heuristic determines the PDO change to be executed. As the evolution strategies define if the same type of PDO change is repeated or another one is executed, the type has still to be determined in the latter case as well as the PDO change (e.g. switching the property weight from the individual WeightAndDimension to the individual GeneralCharacteristics). For this, a tabu search metaheuristic is utilised with the following characteristics: (i) Always the impact of the evolution in the precedent feedback cycle is evaluated, (ii) only one implicit PDO change is executed per cycle, and (iii) "greedy" approach: The

evolution heuristic chooses the PDO change with the highest ST. There are two types of ST for determining the PDO change to be executed: (i) $ST_{f\_pdo\_change\_x}$ is the ST for the forward PDO change x, and (ii) $ST_{b\_pdo\_change\_x}$ is the ST for the backward PDO change x (i.e. reverts the forward change). Forward PDO changes to be executed are determined with the highest $ST_{f\_pdo\_change\_x}$, backward PDO changes with the highest $ST_{b\_pdo\_change\_x}$.

In the following the tabu criteria are defined.

4.1.1 Specific Tabu Criteria sw and ch

The specific tabu criteria are specifically calculated for each type of PDO change.

4.1.1.1 Allowed Number of Horizontal Switches sw

With sw one (set of) ontological entity of a PDO within the same type of PDO change is switched, e.g. a PDO change of one (set of) property or (set of) individual – most of times there is only one switch possible like changing the individual, the property range, or the annotation properties label and comment, and the next change would be reverting that change. This tabu is defined as follows:

$$sw = \begin{cases} 0, \text{ case: } p=1 \wedge c_{fix}=0 \\ 2+c_{fix}^2/2-c_{fix}, \text{ case: } p=1 \wedge c_{fix}=2*k,\ c_{fix}, k \in \mathbb{N}\setminus\{0\} \\ 1+c_{fix}*(c_{fix}-1)/2, \text{ case: } p=1 \wedge c_{fix}=2*k-1,\ k \in \mathbb{N}\setminus\{0\} \\ 1+p^2/2-p, \text{ case: } p>1 \wedge p=2*k,\ p \in \mathbb{N}\setminus\{0,1\},\ k \in \mathbb{N}\setminus\{0\} \\ p*(p-1)/2, \text{ case: } p>1 \wedge p=2*k-1,\ p \in \mathbb{N}\setminus\{0,1\},\ k \in \mathbb{N}\setminus\{0\} \end{cases} \quad (\mathbf{1})$$

(c_{fix} being the number of fixed candidates within a type of PDO change (i.e. to these candidates can be switched), p being the number of pools of sets of entities (e.g. each source for the properties is a pool like string ranges, Boolean ranges, DBpedia, or WordNet; p can be changed for each type of PDO change in the administration interface); a pool p can be switched on the level of ontological entity (s') or completely (s), i.e. all sets of ontological entities are switched at once (can be changed for each type of PDO change in the administration interface, in case of more than one data pool p), k being a natural number to indicate an even ($c_{fix} = 2 * k$, $p = 2 * k$) or odd ($c_{fix} = 2 * k - 1$, $p = 2 * k - 1$) number of fixed candidates or pools: The case for the even c_{fix} or p equates to an Eulerian trail, the case for the odd c_{fix} or p to an Eulerian circuit).

Result is the number of allowed switches sw. In case s is already connected to c_{fix} (e.g. s - c_{fix} = 1), the second and third case in (**1**) are lessen by this one "impossible" switch (i.e. sw_{fix} = sw - 1).

4.1.1.2 Allowed Number of Vertical PDO Change Iterations ch

With ch successive sw switches within the same type of PDO change are executed, i.e. the next (sets of) ontological entities are going to be switched. This tabu is defined as follows:

$$ch = \begin{cases} (s-ch_{fix})/n; \text{ case: } p=1,\ n \in \mathbb{N}\setminus\{0\},\ s,\ ch_{fix} \in \mathbb{N},\ s \geq ch_{fix} \\ s'/n, \text{ case: } p>1 \wedge s' \subset s \text{ (i.e. single sets)},\ n \in \mathbb{N}\setminus\{0\},\ s' \in \mathbb{N} \\ \text{Not applicable, case: } p>1 \wedge s' \equiv s \text{ (i.e. all sets at once)} \end{cases} \quad (\mathbf{2})$$

ch is truncated to the natural number.

(s being all sets of ontological entities within a type of PDO change (e.g. all sets of individuals, all sets of properties, all sets of annotation properties label and comment), s' being a single set of ontological entities within a type of PDO change (e.g. specific properties) to be switched to another pool, n being the fraction of the "free" sets (i.e. not connected to a c_{fix}) of entities within a type of PDO change allowed to be switched (e.g. n = 1: All free sets of entities, n = 2: Half of the free sets, etc.; n can be changed for each type of PDO change in the administration interface)).

Result is the number of allowed PDO change iterations ch. Analogous to the case distinction of the horizontal switches sw and sw_{fix}, ch is splitted in the first case in (**2**) into s is not connected to c_{fix} before switching (ch), and s is already connected to c_{fix} before switching (ch_{fix}).

4.1.2 General Tabu Criterion gt

To avoid an uniform optimisation and cycles, the PDO changes within the same type of PDO change are consecutively executed only as often as there are different types T of PDO changes not induced by a feedback based on a PDO extraction (here: Three times, T = 3, i.e. switching individuals, switching datatype property ranges, and switching annotation properties label and comment).

In case a type of PDO change has less than T PDO changes, the general tabu criterion gt is met when all PDO changes within the respective type of PDO change have been executed.

To calculate the general tabu criterion gt, the overall number of switches sw and ch executed has to be respected. Hence, this tabu is valid when having executed either all sw and ch switches within

the respective type of PDO change (case: Number of all switches ≤ T) or the number of switches executed within the same type of PDO change equals T (here: $T = 3$) (case: Number of all switches > T); this tabu is defined as follows:

$$gt = \begin{cases} sw*ch+(sw-1)*ch_{fix} \leq T, \text{ case: } p=1, sw, ch, ch_{fix}, T \in \mathbb{N} \\ sw*ch \leq T, \text{ case: } p>1 \wedge s' \subset s \text{ (i.e. single sets)}, sw, ch, T \in \mathbb{N} \quad (\mathbf{3}) \\ sw \leq T, \text{ case: } p>1 \wedge s' \equiv s \text{ (i.e. all sets at once)}, sw, T \in \mathbb{N} \end{cases}$$

Result is the number of allowed PDO changes gt. The PDO changes are sequentially executed and added to the tabu list. In case the tabu gt or T is met, another type of PDO change is going to be executed.

In case another type of PDO change is executed, the overall oldest tabu is deleted from the tabu list.

After the ramp-up and in case the general tabu criterion gt or T is met (here: The same type of PDO change shall be consecutively executed for the fourth time), the PDO change with the highest ST in another type of PDO change is going to be executed and ST(t+1) calculated.

In case the "allowed number of horizontal switches" sw is met, the PDO change with the second highest ST within the same type of PDO change is executed and ST(t+1) calculated.

4.2 Example Calculation of the Tabu Criteria

The tabu criteria are exemplarily calculated for the type of PDO change switching individuals. It has one data pool ($p = 1$, i.e. one set of individuals); p is manually entered in the Administration Interface. A digital camera has the following sets of properties and individuals {s, I}: {faceDetection, Features}, {weight, WeightAndDimension}, {videofunction, GeneralCharacteristics}, {HDMI, Ports}, {opticalZoomFactor, LensFeatures}, and {touchscreen, Display}. So, the question if the camera should offer HDMI is nestled between the port-related features of the camera. By observing the relationships, it is obvious that not all combinations make sense, e.g. HDMI cannot belong to WeightAndDimension, but it could belong to Features or GeneralCharacteristics. When switching the HDMI property to another individual, e.g. from Ports to GeneralCharacteristics, the question after HDMI could be placed aside the question for the video function which could make more sense from a customer point of view. The Feedback Transformer identifies the general individuals (i.e. c_{fix}) by parsing the strings. In the example the two individuals mentioned above are of general meaning, i.e. $c_{fix} = 2$.

- Specific tabu criterion "allowed number of horizontal switches" sw:

  (**1**), second case, with $c_{fix} = 2$:

  $sw = 2$ (case: s is not connected to c_{fix} before switching) and $sw_{fix} = 1$ (case: s is already connected to c_{fix} before switching)

  Result: The specific tabu criterion sw is met with two switches or one switch; in this case, the next set of individuals is going to be switched.

- Specific tabu criterion "allowed number of vertical PDO change iterations" ch:

  (**2**), first case, with $c_{fix} = 2$, $n = 2$ (i.e. half of the "free" sets; "free" meaning not connected to c_{fix} before switching), $s = 6$ (i.e. properties):

  $ch = 2$ (**3**)

  Result: The specific tabu criterion ch is met with switching two sets of individuals allowed to be switched.

- General tabu criterion gt:

  (**3**), first case:

  $gt = 6 \leq T$

  Result: The general tabu criterion gt is met with switching the minimum of six sets of individuals to c_{fix} and T; as $T = 3$ (i.e. three types of PDO changes not induced by a feedback based on a PDO extraction), the tabu is met with three individual switches; in this case, another type of PDO change is going to be executed

This means in case of a high ST for the switch of HDMI from Ports to GeneralCharacteristics, this switch will be within the first three individual switches and get executed. In case it is not, the question for HDMI will remain aside the port-related questions.

4.3 Future Work: Evaluation and Validation of the Adaptation Layer

The adaptation layer is going to be evaluated by conducting an experiment with approximately thirty ontology experts who evaluate the ontology evolution. The automatically evolved PDO is going to be compared with a manually evolved one by setting up and evaluating an experiment with ontology experts who analyse the feedback delivered and decide the PDO changes to be executed. Eventually, the PDO resulted from this manual evolution is compared with the automatically evolved one regarding the evaluation criteria consistency, completeness, conciseness, expandability, and sensitiveness [5].

The adaptation layer is going to be validated by programming the layer and measuring the effects in the e-commerce recommender system. Its success is defined by the click-out rate (i.e. clicks-to-recommendations; the user follows the recommendation by clicking on the product recommended) which measures the impact of the PDO evolution induced by the implicit and explicit user feedback.

The validation scenario will be to analyse and evaluate the impact of the PDO evolution with regard to the respective KPI reported to the adaptation layer after having accomplished the defined number of recommendation processes by utilising the formulated evolution strategies, i.e. Risky, Progressive, and Safe Evolution. In each feedback cycle the transformed feedback (i.e. ST) gets reported to the Adaptation Manager. The feedback is PDO-based

or PDO- and property-based. According to the feedback reported, the PDO evolves. The new PDO version is provided to the data modelling layer and the application layer, and eventually an adapted recommender interface is presented to the customer. The feedback circle of the automated system concludes with re-evaluating the KPI after having again accomplished the defined number of recommendation processes.

The intended results are a highly user-adaptive system and eventually better recommendations given to the customer leading to an increase of the defined KPI. The expected business impacts are a higher customer satisfaction and loyalty and eventually increased revenue for the provider of the e-commerce application (and the recommender system).

5. CONCLUSION

The need for automatically updating and evolving ontologies is urging in today's usage scenarios. Here, it is the basis for creating a user-adaptive recommender interface. The present research tackles an automated process for the first time (to the best knowledge of the author). The reason for that can be found in the ontology definition "formal, explicit specification of a shared conceptualisation" [6]. "Shared" means the knowledge contained in an ontology is consensual, i.e. it has been accepted by a group of people [15]. Entailed from that, one can argue that by processing feedback in an ontology and evolving it, it is no longer a shared conceptualisation but an application-specific data model. On the other hand, it is still shared by the group of people who are using the application. It may even be argued that the ontology has been optimised for the usage of that group (in a specific context or application) and thus is a new way of interpreting ontologies: They can also be a specifically tailored and usage-based knowledge representation derived from an initial ontology – an ontology view, preserving most of the advantages like the support of automatically processing information. Thus, this changed way of conceiving ontologies could facilitate the adoption and spread of using this powerful representation mechanism in the real world, as it is easier to accomplish consensus within a smaller group of people than a larger one.

In this research the PDO are based on GoodRelations and evolve within that upper ontology. This ontology as well as the "subsumed" PDO conforms to the ontology definition by [6]. The PDO are application-specific and evolve according to the needs of their users. Hence, they offer the advantages of both worlds.

In the next steps of this research the adaptation layer is going to be evaluated and validated.

6. ACKNOWLEDGMENTS

The research presented in this paper is funded by the Austrian Research Promotion Agency (FFG) and the Federal Ministry of Transport, Innovation, and Technology (BMVIT) under the FIT-IT "Semantic Systems" program (contract number *825061*).

7. REFERENCES

[1] Bennett, K. H. and Rajlich, V. T. 2000. Software maintenance and evolution: A roadmap, *Proceedings of the Conference on the Future of Software Engineering*, pp. 73-87.

[2] Broy, M. et al. 2009. Formalizing the notion of adaptive system behavior, *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC '09)*, pp. 1029-1033.

[3] Glover, F. W. 1986. Future paths for integer programming and links to artificial intelligence, *Comput. Oper. Res.*, Volume 13, pp. 533-549.

[4] Glover, F. W. and Laguna, M. 1997. *Tabu Search*, Kluwer Academic Publishers.

[5] Gómez-Pérez, A. 2001. Evaluation of ontologies, *International Journal of Intelligent Systems*, Volume 16, pp. 391-409.

[6] Gruber, T. R. 1993. Toward principles for the design of ontologies used for knowledge sharing, *Formal ontology in conceptual analysis and knowledge representation*, Kluwer Academic Publishers.

[7] Haase, P. and Stojanovic, L. 2005. Consistent evolution of OWL ontologies, *Proceedings of the 2nd European Semantic Web Conference (ESWC 2005)*, pp. 182 - 197.

[8] Haase, P. et al. 2005. A framework for handling inconsistency in changing ontologies, *Proceedings of the 2005 International Semantic Web Conference (ISWC05)*, pp. 353-367.

[9] Klein, M. and Noy N. F. 2003. A component-based framework for ontology evolution, *Proceedings of the IJCAI-03 Workshop on Ontologies and Distributed Systems*.

[10] Konstantinidis, G. et al. 2007. Ontology evolution: A framework and its application to RDF, *Proceedings of the Joint ODBIS & SWDB Workshop on Semantic Web, Ontologies, Databases*.

[11] Noy, N. F. et al. 2006. A framework for ontology evolution in collaborative environments, Proceedings of the 2005 International Semantic Web Conference (ISWC05), pp. 544-558.

[12] Pearl, J. 1983. Heuristics: Intelligent search strategies for computer problem solving, Addison-Wesley.

[13] Stojanovic, L. et al. 2002. User-driven ontology evolution management, *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW '02)*, pp. 285-300.

[14] Stojanovic, N. et al. 2003. The OntoManager – a system for the usage-based ontology management, *LNCS 2888*, pp. 858-875.

[15] Studer, R. et al. 1998. Knowledge engineering: Principles and methods, *Data & Knowledge Engineering*, Volume 25, Number 1-2, pp. 161-198.

[16] Suárez-Figueroa, M. C. and Gómez-Pérez, A. 2008. Towards a glossary of activities in the ontology engineering field, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC '08)*.

[17] Wach, E. P., 2011. Automated ontology evolution for an e-commerce recommender, *Proceedings of the 14th International Conference on Business Information Systems (BIS 2011)*, in press.

[18] Zablith, F. 2009. Evolva: A comprehensive approach to ontology evolution, *Proceedings of the 6th European Semantic Web Conference (ESWC 2009)*.

A User-centric Evaluation of Recommender Algorithms for an Event Recommendation System

Simon Dooms
Wica-INTEC, IBBT-Ghent University
G. Crommenlaan 8 box 201
B-9050 Ghent, Belgium
Simon.Dooms@UGent.be

Toon De Pessemier
Wica-INTEC, IBBT-Ghent University
G. Crommenlaan 8 box 201
B-9050 Ghent, Belgium
Toon.DePessemier@UGent.be

Luc Martens
Wica-INTEC, IBBT-Ghent University
G. Crommenlaan 8 box 201
B-9050 Ghent, Belgium
Luc1.Martens@UGent.be

ABSTRACT

While several approaches to event recommendation already exist, a comparison study including different algorithms remains absent. We have set up an online user-centric based evaluation experiment to find a recommendation algorithm that improves user satisfaction for a popular Belgian cultural events website. Both implicit and explicit feedback in the form of user interactions with the website were logged over a period of 41 days, serving as the input for 5 popular recommendation approaches. By means of a questionnaire users were asked to rate different qualitative aspects of the recommender system including accuracy, novelty, diversity, satisfaction, and trust.

Results show that a hybrid of a user-based collaborative filtering and content-based approach outperforms the other algorithms on almost every qualitative metric. Correlation values between the answers in the questionnaire seem to indicate that both accuracy and transparency are correlated the most with general user satisfaction of the recommender system.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; H5.2 [**User Interfaces**]: User-centered design

General Terms

Algorithms, Experimentation, Human Factors.

Keywords

Recommender systems, events, user-centric evaluation, experiment, correlation, recommendation algorithms.

1. INTRODUCTION

More and more recommender systems are being integrated with web based platforms that suffer from information overload. By personalizing content based on user preferences, recommender systems assist in selecting relevant items on these websites. In this paper, we focus on evaluating recommendations for a Belgian cultural events website. This website contains the details of more than 30,000 near future and ongoing cultural activities including movie releases, theater shows, exhibitions, fairs and many others.

In the research domain of recommender systems, numerous studies have focused on recommending movies. They have been studied thoroughly and many best practices are known. The area of event recommendations on the other hand is relatively new. Events are so called *one-and-only* items [5], which makes them harder to recommend. While other types of items generally remain available (and thus recommendable) for longer periods of time, this is not the case for events. They take place at a specific moment in time and place to become irrelevant very quickly afterwards.

Some approaches towards event recommendation do exist. For the Pittsburgh area, a cultural event recommender was build around trust relations [8]. Friends could be explicitly and implicitly rated for trust ranging from 'trust strongly' to 'block'. A recommender system for academic events [7] focused more on social network analysis (SNA) in combination with collaborative filtering (CF) and finally Cornelis et al. [3] described a hybrid event recommendation approach where both aspects of CF and content-based algorithms were employed. To our knowledge however, event recommendation algorithms were never compared in a user-centric designed experiment with a focus on optimal user satisfaction.

For a comparison of algorithms often offline metrics like RMSE, MAE or precision and recall are calculated. These kinds of metrics allow automated and objective comparison of the accuracy of the algorithms but they alone can not guarantee user satisfaction in the end [9]. As shown in [2], the use of different offline metrics can even lead to a different outcome of the 'best' algorithm for the job. Hayes et al. [6] state that real user satisfaction can only be measured in an online context. We want to improve the user satisfaction for real-life users of the event website and are therefore opting for an online user-centric evaluation of different recommendation algorithms.

2. EXPERIMENT SETUP

To find the recommendation algorithm that results in the highest user satisfaction, we have set up a user-centric evaluation experiment. For a period of 41 days, we monitored both implicit and explicit user feedback in the form of user interactions with the event website. We used the collected feedback as input for 5 different recommendation algorithms, each of which generated a list of recommendations for every user. Bollen et al. [1] hypothesizes that a set of somewhere between seven and ten items would be ideal in the sense that it can be quite varied but still manageable for the users. The users therefore received a randomly chosen recommendation list containing 8 events together with an online questionnaire. They were asked to rate different aspects about the quality of their given recommendations.

In the following subsections, we elaborate on the specifics of the experiment such as the feedback collection, the rec-

Feedback activity		Feedback value
Click on	'I like this'	1.0
Share on	Facebook/Twitter	0.9
Click on	Itinerary	0.6
Click on	Print	0.6
Click on	'Go by bus/train'	0.6
Click on	'Show more details'	0.5
Click on	'Show more dates'	0.5
Mail to	a friend	0.4
Browse to	an event	0.3

Table 1: The distinct activities that were collected as user feedback together with the feedback value indicating the interest of an individual user for a specific event ranging from 1.0 (very interested) to 0.3 (slightly interested).

ommendation algorithms, how we randomized the users, and the questionnaire.

2.1 Feedback collection

Feedback collection is a very important aspect of the recommendation process. Since the final recommendations can only be as good as the quality of their input, collecting as much high quality feedback as possible is of paramount importance. Previous feedback experiments we ran on the website [4] showed that collecting explicit feedback (in the form of explicit ratings) is very hard, since users do not rate often. Clicking and browsing through the event information pages are on the other hand activities that were abundantly logged. For optimal results, we ultimately combined implicit and explicit user feedback gathered during the run of the experiment.

Since explicit ratings are typically provided after an event has been visited, algorithms based on collaborative filtering would be useless. It therefore makes sense to utilize also implicit feedback indicators like printing the event's information, which can be collected before the event has taken place. In total 11 distinct feedback activities were combined into a feedback value that expressed the interest of a user for a specific event.

The different activities are listed in Table 1 together with their resulting feedback values which were intuitively determined. The $max()$ function is used to accumulate multiple feedback values in case a user provided feedback in more than one way for the same event.

2.2 Recommendation Algorithms

To assess the influence of the recommendation algorithm on the experience of the end-user, 5 different algorithms are used in this experiment. Each user, unaware of the different algorithms, is randomly assigned to one of the 5 groups receiving recommendations generated by one of these algorithms as described in Section 2.3.

As a baseline suggestion mechanism, the random recommender (RAND), which generates recommendations by performing a random sampling of the available events, is used. The only requirement of these random recommendations is that the event is still available (i.e. it is still possible for the user to attend the event). The evaluation of these random recommendations allows to investigate if users can distinguish random events from personalized recommendations, and if so, the relative (accuracy) improvement of more intelligent algorithms over random recommendations.

Metadata field	Weight
Artist	1.0
Category	0.7
Keyword	0.2

Table 2: The metadata fields used by the content-based recommendation algorithm with their weights indicating their relative importance.

Because of its widespread use and general applicability, standard collaborative filtering (CF) is chosen as the second algorithm of the experiment. We opted for the user-based nearest neighbor version of the algorithm (UBCF) because of the higher user-user overlap compared to the item-item overlap. Neighbors were defined as being users with a minimum overlap of 1 event in their feedback profiles but had to be at least 5% similar according to the cosine similarity metric.

The third algorithm evaluated in this experiment is singular value decomposition (SVD) [11], a well-known matrix factorization technique that addresses the problems of synonymy, polysemy, sparsity, and scalability for large datasets. Based on preceding simulations on an offline dataset with historical data of the website, the parameters of the algorithm were determined: 100 initial steps were used to train the model and the number of features was set at 70.

Considering the transiency of events and the ability of content-based (CB) algorithms to recommend items before they received any feedback, a CB algorithm was chosen as the fourth algorithm. This algorithm matches the event metadata, which contain the title, the categories, the artist(s), and keywords originating from a textual description of the event, to the personal preferences of the user, which are composed by means of these metadata and the user feedback gathered during the experiment. A weighting value is assigned to the various metadata fields (see Table 2), thereby attaching a relative importance to the fields during the matching process (e.g., a user preference for an artist is more important than a user preference for a keyword of the description). The employed keyword extraction mechanism is based on a term frequency-inverse document frequency (tf-idf) weighting scheme, and includes features as stemming and filtering stop words.

Since pure CB algorithms might produce recommendations with a limited diversity [9], and CF techniques might produce suboptimal results due to a large amount of unrated items (cold start problem), a hybrid algorithm (CB+UBCF), combining features of both CB and CF techniques, completes the list. This fifth algorithm combines the best personal sugestions produced by the CF with the best sugestions originating from the CB algorithm, thereby generating a merged list of hybrid recommendations for every user. This algorithm acts on the resulting recommendation lists produced by the CF and CB recommender, and does not change the internal working of these individual algorithms. Both lists are interwoven while alternately switching their order such that both lists have their best recommendation on top in 50% of the cases.

For each algorithm, the final event recommendations are checked for their availability and familiarity with the user.

Events that are not available for attendance anymore, or events that the user has already explored (by viewing the webpage, or clicking the link) are replaced in the recommendation list.

2.3 Randomizing Users

Since certain users have provided only a limited amount of feedback during the experiment, not all recommendation algorithms were able to generate personal suggestions for these users. CF algorithms, for instance, can only identify neighbors for users who have overlapping feedback with other users (i.e. provided feedback on the same event as another user). Without these neighbors, CF algorithms are not able to produce recommendations. Therefore, users with a limited profile, hindering (some of) the algorithms to generate (enough) recommendations for that user, are treated separately in the analysis. Many of these users are not very active on the website or did not finish the evaluation procedure as described in Section 2.4. This group of cold-start users received recommendations from a randomly assigned algorithm that was able to generate recommendations for that user based on the limited profile. Since the random recommender can produce suggestions even without user feedback, at least 1 algorithm was able to generate a recommendation list for every user. The comparative evaluation of the 5 algorithms however, is based on the remaining users. Each of these users is randomly assigned to 1 of the 5 algorithms which generates personal suggestions for that user. This way, the 5 algorithms, as described in Section 2.2, are evaluated by a number of randomly selected users.

2.4 Evaluation Procedure

While prediction accuracy of ratings used to be the only evaluation criteria for recommender systems, during recent years optimizing the user experience has increasingly gained interest in the evaluation procedure. Existing research has proposed a set of criteria detailing the characteristics that constitute a satisfying and effective recommender system from the user's point of view. To combine these criteria into a more comprehensive model which can be used to evaluate the perceived qualities of recommender systems, Pu et al. have developed an evaluation framework for recommender systems [10]. This framework aims to assess the perceived qualities of recommenders such as their usefulness, usability, interface and interaction qualities, user satisfaction of the systems and the influence of these qualities on users' behavioral intentions including their intention to tell their friends about the system, the purchase of the products recommended to them, and the return to the system in the future. Therefore, we adopted (part of) this framework to measure users' subjective attitudes based on their experience towards the event recommender and the various algorithms tested during our experiment. Via an online questionnaire, test users were asked to answer 14 questions on 5-point Likert scale from "strongly disagree" (1) to "strongly agree" (5) regarding aspects as recommendation accuracy, novelty, diversity, satisfaction and trust of the system. We selected the following 8 most relevant questions for this research regarding various aspects of the event recommendation system.

Q1 The items recommended to me matched my interests.

Q2 Some of the recommended items are familiar to me.

Algorithm	*#Users*
CB	43
CB+UBCF	36
RAND	45
SVD	36
UBCF	33

Table 3: The 5 algorithms compared in this experiment and the number of users that actually completed the questionnaire about their recommendation lists.

Q4 The recommender system helps me discover new products.

Q5 The items recommended to me are similar to each other (reverse scale).

Q7 I didn't understand why the items were recommended to me (reverse scale).

Q8 Overall, I am satisfied with the recommender.

Q10 The recommender can be trusted.

Q13 I would attend some of the events recommended, given the opportunity.

3. RESULTS

We allowed all users of the event website to participate in our experiment and encouraged them to do so by means of e-mail and a banner on the site. In total 612 users responded positively to our request. After a period of feedback logging, as described in section 2.1, they were randomly distributed across the 5 recommendation algorithms which calculated for each of them a list of 8 recommendations. After the recommendations were made available on the website, users were asked by mail to fill out the accompanying online questionnaire as described in section 2.4.

Of the 612 users who were interested in the experiment, 232 actually completed the online questionnaire regarding their recommendations. After removal of fake samples (i.e., users who answered every question with the same value) and users with incomplete (feedback) profiles, 193 users remained. They had by average 22 consumptions (i.e., expressed feedback values for events) and 84% of them had 5 or more consumptions. The final distribution of the users across the algorithms is displayed in Table 3.

Figure 1 shows the averaged results of the answers provided by the 193 users in this experiment for the 8 questions we described in section 2.4 and for each algorithm.

Evaluating the answers to the questionnaire showed that the hybrid recommender (CB+UBCF) achieved the best averaged results to all questions, except for question Q5, which asked the user to evaluate the similarity of the recommendations (i.e. diversity). For question Q5 the random recommender obtained the best results in terms of diversity, since random suggestions are rarely similar to each other. The CF algorithm was the runner-up in the evaluation and achieved a second place after the hybrid recommender for almost all questions (again except for Q5, where CF was the fourth after the random recommender, the hybrid recommender and SVD).

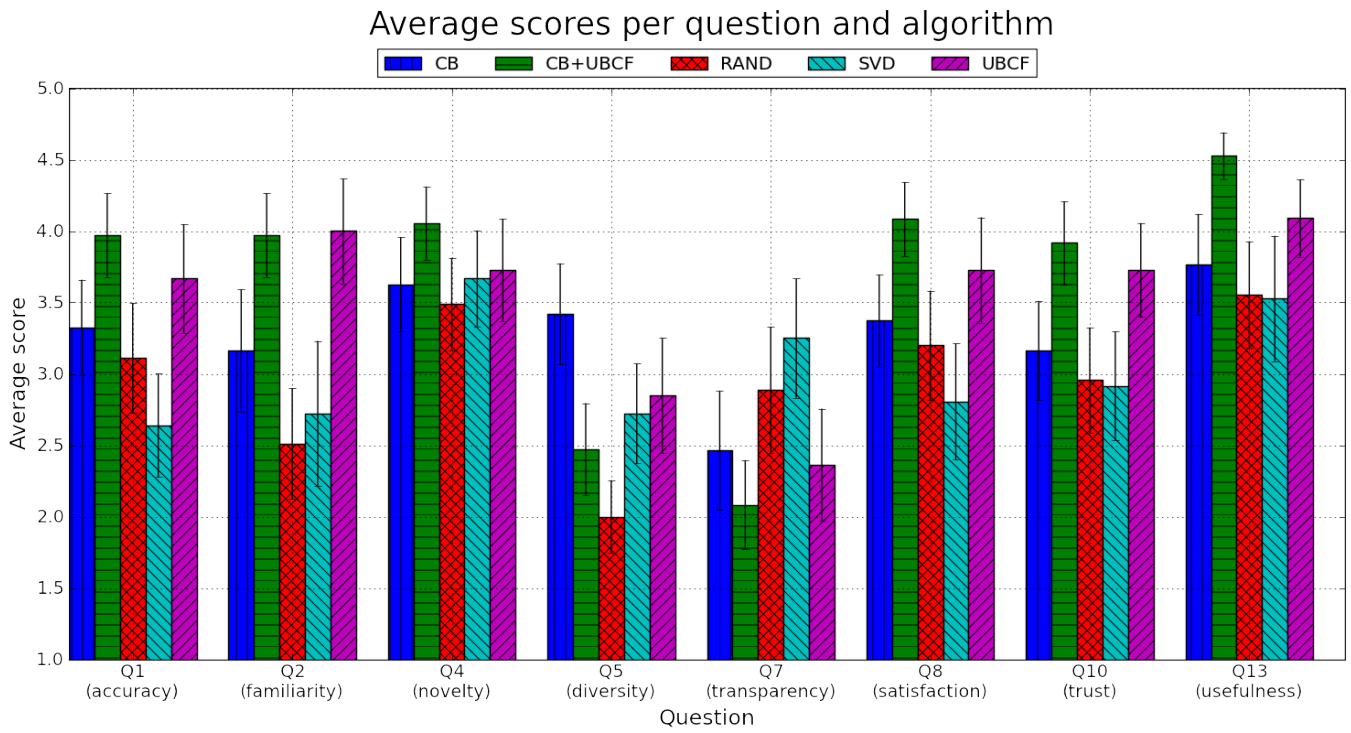


Figure 1: The averaged result of the answers (5-point Likert scale from "strongly disagree" (1) to "strongly agree" (5)) of the evaluation questionnaire for each algorithm and questions Q1, Q2, Q4, Q5, Q7, Q8, Q10 and Q13. The error bars indicate the 95% confidence interval. Note that questions Q5 and Q7 were in reverse scale.

The success of the hybrid recommenders is not only clearly visible when comparing the average scores for each question (Figure 1), but also showed to be statistically significantly better than every other algorithm (except for the CF recommender) according to a Wilcoxon rank test ($p < 0.05$) for the majority of the questions (Q1, Q2, Q8, Q10 and Q13). Table 4 shows the algorithms and questions for which statistically significant differences could be noted according to this non-parametric statistical hypothesis test.

The average performance of SVD was a bit disappointing by achieving the worst results for questions Q1, Q7, Q8, and the second worst results (after the random recommender) for questions Q2, Q4, Q10, Q11, and Q13. So surprisingly the SVD algorithm performs (averagely) worse than the random method on some fundamental questions like for example Q8 which addresses the general user satisfaction. We note however that the difference in values between SVD and the RAND algorithm was not found to be statistically significant except for question Q5.

We looked more closer into this observation and plotted a histogram (Figure 2) of the different values (1 to 5) for the answers provided for question Q8. A clear distinction between the histogram of the SVD algorithm and the histograms of the other algorithms (CB and RAND shown in the figure) can be seen. Whereas for CB and RAND most values are grouped towards one side of the histogram (i.e. the higher values), this is not the case for the SVD. It turns out that the opinions about the general satisfaction of the SVD algorithm where somewhat divided between good and bad with no apparent winning answer. These noteworthy

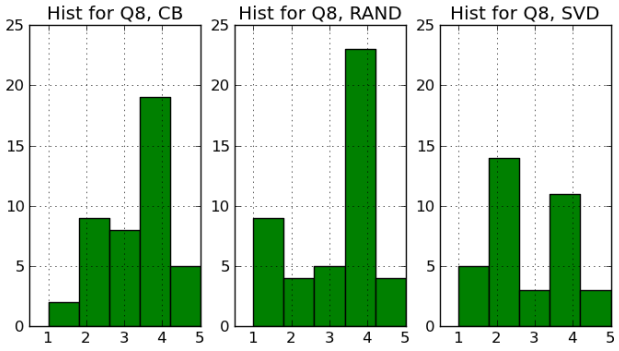


Figure 2: The histogram of the values (1 to 5) that were given to question Q8 for algorithm CB (left), RAND (middle) and SVD (right).

rating values for the SVD recommender are not only visible in the results of Q8, but also for other questions like Q2 and Q5. These findings indicate that SVD works well for many users, but also provides inaccurate recommendations for a considerable number of other users. These inaccurate recommendations may be due to a limited amount of user feedback and therefore sketchy user profiles.

Figure 1 seems to indicate that some of the answers to the questions are highly correlated. One clear example is question Q1 about whether or not the recommended items

	CB	CB+UBCF	RAND	SVD	UBCF
CB	-	Q1, Q2, Q5, Q8, Q10, Q13	Q2, Q5	Q1, Q5, Q7, Q8	Q2, Q5, Q10
CB+UBCF	Q1, Q2, Q5, Q8, Q10, Q13	-	Q1, Q2, Q4, Q5, Q7, Q8, Q10, Q13	Q1, Q2, Q7, Q8, Q10, Q13	Q13
RAND	Q2, Q5	Q1, Q2, Q4, Q5, Q7, Q8, Q10, Q13	-	Q5	Q2, Q5, Q10
SVD	Q1, Q5, Q7, Q8	Q1, Q2, Q7, Q8, Q10, Q13	Q5	-	Q1, Q2, Q7, Q8, Q10
UBCF	Q2, Q5, Q10	Q13	Q2, Q5, Q10	Q1, Q2, Q7, Q8, Q10	-

Table 4: The complete matrix of statistically significant differences between the algorithms on all the questions using the Wilcoxon rank test on a confidence level of 0.95. Note that the matrix is symmetric.

matched the user's interest and question Q8 which asked about the general user satisfaction. As obvious as this correlation may be, other correlated questions may not be so easy to detect by inspecting a graph with averaged results and so we calculated the complete correlation matrix for every question over all the algorithms using the two-tailed Pearson correlation metric (Table 5).

From the correlation values two similar trends can be noticed for questions Q8 and Q10 dealing with respectively the user satisfaction and trust of the system. The answers to these questions are highly correlated (very significant $p < 0.01$) with almost every other question except for Q5 (diversity). We must be careful not to confuse correlation with causality, but still data indicates the strong relation between user satisfaction and recommendation accuracy and transparency.

This strong relation may be another reason why SVD performed very badly in the experiment. Its inner workings are the most obscure and least obvious to the user and therefore also the least transparent.

Another interesting observation lies in the correlation values of question Q5. The answers to this diversity question are almost completely unrelated to every other question (i.e., low correlation values which are not significant $p > 0.05$). It seems like the users of the experiment did not value the diversity of a recommendation list as much as the other aspects of the recommendation system. If we look at the average results (Figure 1) of the diversity question (lower is more diverse) we can see this idea confirmed. The ordering of how diverse the recommendation lists produced by the algorithms were, is in no way reflected in the general user satisfaction or trust of the system.

To gain some deeper insight into the influence of the qualitative attributes towards each other, we performed a simple linear regression analysis. By trying to predict an attribute by using all the other ones as input to the regression function, a hint of causality may be revealed. As regression method we used multiple stepwise regression. We used a combination of the forward and backward selection approach, which step by step tries to add new variables (or remove existing ones) to its model that have the highest marginal relative influence on the dependent variable. The following lines express the regression results. We indicated what attributes were added to the model by means of an arrow notation. Between brackets we also indicated the coefficient of determination R^2. This coefficient indicates what percentage of the variance in the dependent variable can be explained by the model. R^2 will be 1 for a perfect fit and 0 when no linear relationship could be found.

Q1 $\leftarrow$ Q7, Q8, Q10, Q13 ($R^2 = 0.7131$)

Q2 $\leftarrow$ Q7, Q10, Q13 ($R^2 = 0.2195$)

Q4 $\leftarrow$ Q10, Q13 ($R^2 = 0.326$)

Q5 $\leftarrow$ Q1, Q13 ($R^2 = 0.02295$)

Q7 $\leftarrow$ Q1, Q2, Q8, Q10 ($R^2 = 0.6095$)

Q8 $\leftarrow$ Q1, Q7, Q10, Q13 ($R^2 = 0.747$)

Q10 $\leftarrow$ Q1, Q2, Q4, Q7, Q8, Q13 ($R^2 = 0.7625$)

Q13 $\leftarrow$ Q1, Q2, Q4, Q5, Q8, Q10 ($R^2 = 0.6395$)

The most interesting regression result is the line were Q8 (satisfaction) is predicted by Q1, Q7, Q10 and Q13. This result further strengthens our belief that accuracy (Q1) and transparency (Q7) are the main influencers of user satisfaction in our experiment (we consider Q10 and Q13 rather as results of satisfaction than real influencers but they are of course also connected).

Table 6 shows the coverage of the algorithms in terms of the number of users it was able to produce recommendations for. In our experiment we noticed an average coverage of 66% excluding the random recommender.

Algorithm	Coverage (%)
CB	69%
CB+UBCF	66%
RAND	100%
SVD	66%
UBCF	65%

Table 6: The 5 algorithms compared in this experiment and their coverage in terms of the number of users for which they were able to generate a recommendation list of minimum 8 items.

Next to this online and user-centric experiment, we also ran some offline tests and compared them to the real opinions of the users. We calculated the recommendations on a training set that randomly contained 80% of the collected feedback in the experiment. Using the leftover 20% as the

	Q1 (accuracy)	Q2 (familiarity)	Q4 (novelty)	Q5 (diversity)	Q7 (tranparency)	Q8 (satisfaction)	Q10 (trust)	Q13 (usefulness)
Q1	1	.431	.459	.012	-.731	.767	.783	.718
Q2	.431	1	.227	.036	-.405	.387	.429	.415
Q4	.459	.227	1	-.037	-.424	.496	.516	.542
Q5	.012	.036	-.037	1	0.16	-.008	.001	-.096
Q7	-.731	-.405	-.424	.016	1	-.722	-.707	-.622
Q8	.767	.387	.496	-.008	-.722	1	.829	.712
Q10	.783	.429	.516	.001	-.707	.829	1	.725
Q13	.718	.415	.542	-.096	-.622	.712	.725	1

Table 5: The complete correlation matrix for the answers to the 8 most relevant questions on the online questionnaire. The applied metric is the Pearson correlation and so values are distributed between -1.0 (negatively correlated) and 1.0 (positively correlated). Note that the matrix is symmetric and questions Q5 and Q7 were in reverse scale.

Algorithm	Precision (%)	Recall (%)	F1 (%)
CB	0.462	2.109	0.758
CB+UBCF	1.173	4.377	1.850
RAND	0.003	0.015	0.005
SVD	0.573	2.272	0.915
UBCF	1.359	4.817	2.119

Table 7: The accuracy of the recommendation algorithms in terms of precision, recall and F1-measure based on an offline analysis.

test set, the accuracy of every algorithm was calculated over all users in terms of precision, recall and F1-measure (Table 7). This procedure was repeated 10 times to average out any random effects.

By comparing the offline and online results in our experiment we noticed a small change in the ranking of the algorithms. In terms of precision the UBCF approach came out best followed by respectively CB+UBCF, SVD, CB and RAND. While the hybrid approach performed best in the online analysis, this is not the case for the offline tests. Note that also SVD and CB have swapped places in the ranking. SVD showed slightly better at predicting user behaviour than the CB algorithm. A possible explanation (for the inverse online results) is that users in the online test may have valued the transparency of the CB algorithm over its (objective) accuracy. Our offline evaluation test further underlines the shortcomings of these procedures. In our experiment we had over 30,000 items that were available for recommendation and on average only 22 consumptions per user. The extreme low precision and recall values are the result of this extreme sparsity problem.

It would have been interesting to be able to correlate the accuracy values obtained by offline analysis with the subjective accuracy values provided by the users. Experiments however showed very fluctuating results with on the one hand users with close to zero precision and on the other hand some users with relative high precision values. These results could therefore not be properly matched against the online gathered results.

4. DISCUSSION

The results clearly indicate the hybrid recommendation algorithm (CB+UBCF) as the overall best algorithm for optimizing the user satisfaction in our event recommendation system. The runner-up for this position would definitely be the UBCF algorithm followed by the CB algorithm. This comes as no surprise considering that the hybrid algorithm is mere a combination of these UBCF and CB algorithms. Since the UBCF algorithm is second best, it looks like this algorithm is the most responsible for the success of the hybrid. While the weights of both algorithms were equal in this experiment (i.e., the 4 best recommendations of each list were selected to be combined in the hybrid list), it would be interesting to see how the results evolve if these weights would be tuned more in favour of the CF approach (e.g., $5 * UBCF + 3 * CB$).

Because we collected both implicit and explicit feedback to serve as input for the recommendation algorithms, there were no restrictions as to what algorithms we were able to use. Implicit feedback that was logged before an event took place allowed the use of CF algorithms and the availability of item metadata enabled content-based approaches. Only in this ideal situation a hybrid CB+UBCF algorithm can serve an event recommendation system.

The slightly changed coverage is another issue that may come up when a hybrid algorithm like this is deployed. While the separate CB and UBCF algorithms had respectively coverages of 69% and 65%, the hybrid combination served 66% of the users. We can explain this increase of 1% towards the UBCF by noting that the hybrid algorithm requires a minimum of only 4 recommendations (versus 8 normally) to be able to provide the users with a recommendation list.

5. CONCLUSIONS

For a Belgian cultural events website we wanted to find a recommendation algorithm that improves the user experience in terms of user satisfaction and trust. Since offline evaluation metrics are inadequate for this task, we have set up an online and user-centric evaluation experiment with 5 popular and common recommendation algorithms i.e. CB, CB+UBCF, RAND, SVD and UBCF. We logged both implicit and explicit feedback data in the form of weighted user interactions with the event website over a period of 41 days. We extracted the users for which every algorithm was able to generate at least 8 recommendations and presented each of these users with a recommendation list randomly chosen from one of the 5 recommendation algorithms. Users were asked to fill out an online questionnaire that addressed qualitative aspects of their recommendation lists including accuracy, novelty, diversity, satisfaction, and trust.

Results clearly showed that the CB+UBCF algorithm, which is a combination of both the recommendations of CB and UBCF, outperforms (or is equally as good in the case of question Q2 and the UBCF algorithm) every other algorithm except for the diversity aspect. In terms of diversity the random recommendations turned out best, which of course makes perfectly good sense. Inspection of the correlation values between the answers of the questions revealed however that diversity is in no way correlated with user satisfaction, trust or for that matter any other qualitative aspect we investigated. The recommendation accuracy and transparency on the other hand were the two qualitative aspects highest correlated with the user satisfaction and showed promising predictors in the regression analysis.

The SVD algorithm came out last in the ranking of the algorithms and was statistically even indistinguishable from the random recommender for most of the questions except for again the diversity question (Q5). A histogram of the values for SVD and question Q8 puts this into context by revealing an almost black and white opinion pattern expressed by the users in the experiment.

6. FUTURE WORK

While we were able to investigate numerous different qualitative aspect about each algorithm individually, the experiment did not allow us, apart from indicating a best and worst algorithm, to construct an overall ranking of the recommendation algorithms. Each user ended up evaluating just one algorithm. As our future work, we intend to extend this experiment with a focus group allowing to elaborate on the reasoning behind some of the answers users provided and compare subjective rankings of the algorithms.

We also plan to extend our regression analysis to come up with a causal path model that will allow us to have a better understanding as to how the different algorithms influence the overall satisfaction.

7. ACKNOWLEDGMENTS

The research activities that have been described in this paper were funded by a PhD grant to Simon Dooms of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT Vlaanderen) and a PhD grant to Toon De Pessemier of the Fund for Scientific Research-Flanders (FWO Vlaanderen). We would like to thank CultuurNet Vlaanderen for the effort and support they were willing to provide for deploying the experiment described in this paper.

8. REFERENCES

[1] D. Bollen, B. Knijnenburg, M. Willemsen, and M. Graus. Understanding choice overload in recommender systems. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 63–70. ACM, 2010.

[2] E. Campochiaro, R. Casatta, P. Cremonesi, and R. Turrin. Do metrics make recommender algorithms? In *Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops*, WAINA '09, pages 648–653, Washington, DC, USA, 2009. IEEE Computer Society.

[3] C. Cornelis, X. Guo, J. Lu, and G. Zhang. A fuzzy relational approach to event recommendation. In *Proceedings of the Indian International Conference on Artificial Intelligence*, 2005.

[4] S. Dooms, T. De Pessemier, and L. Martens. An online evaluation of explicit feedback mechanisms for recommender systems. In *Proceedings of the 7th International Conference on Web Information Systems and Technologies (WEBIST)*, 2011.

[5] X. Guo, G. Zhang, E. Chew, and S. Burdon. A hybrid recommendation approach for one-and-only items. *AI 2005: Advances in Artificial Intelligence*, pages 457–466, 2005.

[6] C. Hayes, P. Massa, P. Avesani, and P. Cunningham. An on-line evaluation framework for recommender systems. In *Workshop on Personalization and Recommendation in E-Commerce*. Citeseer, 2002.

[7] R. Klamma, P. Cuong, and Y. Cao. You never walk alone: Recommending academic events based on social network analysis. *Complex Sciences*, pages 657–670, 2009.

[8] D. Lee. Pittcult: trust-based cultural event recommender. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 311–314. ACM, 2008.

[9] S. McNee, J. Riedl, and J. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems*, page 1101. ACM, 2006.

[10] P. Pu and L. Chen. A user-centric evaluation framework of recommender systems. In *Proc. ACM RecSys 2010 Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI)*, 2010.

[11] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, and M. U. M. D. O. C. SCIENCE. *Application of dimensionality reduction in recommender system-a case study*. Citeseer, 2000.

Evaluating Rank Accuracy based on Incomplete Pairwise Preferences

Brian Ackerman
Arizona State University
backerman@asu.edu

Yi Chen
Arizona State University
yi@asu.edu

ABSTRACT

Existing methods to measure the rank accuracy of a recommender system assume the ground truth is either a set of user ratings or a total ordered list of items given by the user with possible ties. However, in many applications we are only able to obtain implicit user feedback, which does not provide such comprehensive information, but only gives a set of pairwise preferences among items. Generally such pairwise preferences are not complete, and thus may not deduce a total order of items. In this paper, we propose a novel method to evaluate rank accuracy, *expected discounted rank correlation*, which addresses the unique challenges of handling incomplete pairwise preferences in ground truth and also puts an emphasis on properly ranking items that users most prefer.

Categories and Subject Descriptors

H.3.4 [**Storage and Retrieval**]: Systems and Software - Performance evaluation—*efficiency and effectiveness*

General Terms

Measurement, Performance

Keywords

Recommender systems, rank accuracy, pairwise preference, evaluation

1. INTRODUCTION

Recommender systems have proven their value in many applications where it is advantageous to personalize a user's experience. Currently, recommender systems power some of the world's most visited websites. Users visit websites like Amazon, Netflix, and Urbanspoon when they are looking for suggestions on items that may be of interest. These websites can suggest relevant items from a large collection of items and output a ranked list ordered by the predicted relevance to the user to aid in a user's decision making process.

Given the prevalence of various recommender systems, a critical question becomes their evaluation. Evaluation measures are used to compare different techniques of recommendation and give researchers an objective for optimization. Existing evaluation measures may take a set of user ratings as the ground truth [2, 4], or a total ordered list of items given by the user [9], with an option to support ties [3, 8]. Based on the weight given to items, there are two types of evaluation. Reference rank accuracy measures the similarity between the ranked list output by a system and the list of user's preferences, such as Spearman's ρ [8] and Kendall's τ [3]. There are also variations of area under the curve measures, such as the one used in [7], which are closely related to a general loss function. Utility rank accuracy measures give a stronger weight to the items that have a higher relevance according to the user, such as normalized discounted cumulative gain [2], which is widely adopted [1, 5, 6].

Ground truth often has to be obtained through explicit user feedback. In some applications, we may solicit users to give ratings on items and use such ratings as relevance scores, such as data obtained from Netflix and MovieLens. The rating data provides accurate user opinions on items, and can also provide a total order of user's preferences on items. However, such data requires explicit feedback, and may be considered as burdensome for some users.

On the other hand, there is great potential for a recommender system, and search engines, to leverage implicit user feedback which can also be used for evaluation. After a user issues a query, a set of items will be recommended. Among them, a user may select some to check for more details, and finally may choose one or more of the products to buy or places to visit. Such user interaction provides implicit user feedback on an item's relevance. We can consider the set of items purchased or visited to be preferred to the set of selected items, which in turn is preferred to the set of items that do not receive any user interaction. Alternatively, we may also measure the actual time that a user spends examining an item to infer finer grained preferences.

After collecting data in the form of implicit user feedback, we must also be able to evaluate recommendation techniques based on the data. Using the data, we may not have the actual rating of items, but a set of pairwise preferences among items. Note that generally *pairwise preferences may not be complete*. That is to say there may exist two items which a user's preference is unknown. For instance, we don't know a user's preference between an item that is presented to the user in response to a query and an item that is not shown to the user. In this case, a total order among a set of items cannot be deduced. In this paper, we only consider evaluating preferences that are not contradictory. We simply allow for the aggregation of preferences for different user sessions assuming the preferences are consistent. It is worth noting that considering contextual factors involved with a user session would also be advantageous, but is outside the scope

Table 1: Example Pairwise Preference Data

	Preferences
Ground Truth	$A \succ C$, $A \succ D$, $A \succ E$, $C \succ D$, $B \succ D$
Prediction	$C \succ A$, $C \succ B$, $C \succ D$ $A \succ E$, $B \succ E$, $D \succ E$

of this paper. These contextual factors could include the set of items the user's were recommended, items previously purchases or visited by the user, and how the items were displayed to the user. It is also possible to find cyclic preferences (e.g. $A \succ B$, $B \succ C$, $C \succ A$). However, we also do not consider these cases.

To evaluate recommender systems whose user preferences are obtained through implicit user feedback, we study evaluation methods for a general type of recommender system that takes a set of *possibly incomplete pairwise preferences* as input for training and for ground truth, and outputs a ranked list of items for recommendation. After examining why existing evaluation methods may fail, we proposed a novel rank accuracy measure, expected discounted rank correlation (EDRC), to handle incomplete pairwise preferences. EDRC handles two unique challenges in this setting, the lack of total ranking order on items, and possible unknown preferences between two items. EDRC also takes a discounted model that gives bigger penalty for wrong prediction on the more preferred items. To the best of our knowledge, this is the first attempt for designing an evaluation measure for ground truth and system prediction that is a set of incomplete pairwise preferences.

The remainder of this paper is outlined as follows: after we introduce the problem setting in Section 2, we briefly review two commonly used evaluation measures. Section 4 proposes a new method to measure rank accuracy for incomplete pairwise preferences, and Section 5 concludes the paper.

2. PROBLEM SETTING

In this work, we design an evaluation measure for a recommender system that takes ranking data in the form of pairwise preferences as input, trains on the ranking data to infer relevance values for each item in a test set, and then outputs a ranking order for the items based on their predicted relevance values. For the test set, we consider the ground truth to be a set of pairwise preferences collected from implicit user feedback.

We define a pairwise preference as a user's comparison between two items. We denote a pairwise preference as $A \succ B$ meaning item A is preferred to item B. Clearly, from a ranked list of items, we can derive a set of pairwise preferences. For example, we consider three items A, B, and C. The system may predict the order of these items to be A, B, C. Then the derived set of pairwise preferences is: $\{ A \succ B, A \succ C, B \succ C \}$. On the other hand, we may not always be able to derive a total order of items based on a set of pairwise preferences. For example, consider the set of pairwise preferences in the ground truth in Table 1 is insufficient to form a total order. For instance, we do not know the preference between A and B. We define a set of pairwise preferences to be *complete* if there exists a pairwise preference between every two items involved in the set, or a pairwise preference can be inferred. Otherwise, it is a set of *incomplete pairwise preferences*. As discussed in Section 1, incomplete pairwise preferences are common for user preferences that are obtained implicitly. This is true for both the training dataset and test dataset which contains the ground truth or user's preferences.

Table 2: Example Rating and Preference Data

$\mathcal{I}$	$rel(i)$	$index(i)$	$\widehat{index}(i)$
A	5	1	2
B	4	2	1
C	3	3	3
D	2	4	4

Since existing evaluation metrics are inapplicable for incomplete pairwise preferences in the ground truth, we propose a measure for rank accuracy where the ground truth is a set of possibly incomplete pairwise preferences. Since we can derive a set of pairwise preferences from the ranked list output, the core of the evaluation is to measure the similarity between two sets of pairwise preferences, one for system output, and the other from the user.

3. EXISTING EVALUATION METHODS

Before we discuss our newly proposed method of evaluation, we look at two existing methods, nDCG and AP correlation. For both methods we show a brief example and then discuss cases where each does not work for our problem setting.

3.1 Normalized Discounted Cumulative Gain

Cumulated gain-based evaluation was proposed by Jarvelin and Kekalainen in 2002 [2] to evaluate rank accuracy when the ground truth is a set of user ratings. The intuition is that it is more important to correctly predict highly relevant items than marginally relevant ones.

nDCG is formally defined in Equation 1. We denote $\mathcal{I}$ as a set of items suggested by the system, $rel(i)$ is the relevance of item i according to the user, $index(i)$ is the index of item i sorting the items based on the user's relevance score, and $\widehat{index}(i)$ is the index of item i sorted by the system's prediction.

$$nDCG = \frac{1}{Z} \cdot \left[\sum_{i \in \mathcal{I}} \frac{2^{rel(i)} - 1}{log_2(1 + index(i))} \right] \quad (1)$$

Z gives the maximum possible discounted cumulative gain if the items were correctly sorted, and is used as a normalization to ensure the result is between 0 and 1.

$$Z = \sum_{i \in \mathcal{I}} \frac{2^{rel(i)} - 1}{log_2(1 + \widehat{index}(i))} \quad (2)$$

For example, we look at the sample data in Table 2. If we look at item A, $rel(A) = 5$, $index(A) = 1$, and $\widehat{index}(A) = 2$. Below is the full sample calculation for $nDCG$.

$$nDCG = \frac{1}{Z} \cdot \left[\frac{2^4 - 1}{log_2(2)} + \frac{2^5 - 1}{log_2(3)} + \frac{2^3 - 1}{log_2(4)} + \frac{2^2 - 1}{log_2(5)} \right]$$

$$Z = \frac{2^5 - 1}{log_2(2)} + \frac{2^4 - 1}{log_2(3)} + \frac{2^3 - 1}{log_2(4)} + \frac{2^2 - 1}{log_2(5)}$$

This gives a value of .870 for the example.

nDCG assumes that we know an actual relevance for each item. It does not directly support the case when an item is preferred to another item, but the magnitude of the preference is unknown. When there is a total order on item preferences, we may assign rating scores with equal magnitude and then use nDCG. However, the absence of total order will result in such score assignments to be impractical, thus showing inapplicability of nDCG.

3.2 AP Correlation

AP (average precision) correlation (τ_{ap}) was proposed by Yilmaz et al. [9] as a modification to Kendall's τ which penalizes mistakes made for highly relevant items more than less relevant items. AP correlation finds the precision between two total orders at each index in the list and then takes the average of these values, as defined in Equation 3.

$$\tau_{ap} = \frac{2}{N-1} \cdot \left[\sum_{i \in \mathcal{I}} \frac{C(i)}{index(i) - 1} \right] - 1 \qquad (3)$$

N is the number of ranked items in the list and $C(i)$ is the number of items at an index less than $index(i)$ that are correctly ranked according to the ground truth. Consider the data in Table 2. For item A, $C(A) = 0$ because A comes after B in the prediction, but A is preferred to B in the ground truth. Below is a sample calculation.

$$\tau_{ap} \quad = \quad \frac{2}{4-1} \cdot \left[\frac{0}{1} + \frac{2}{2} + \frac{3}{3} \right] - 1 \quad = \quad \frac{1}{3}$$

AP correlation is measured on scale of -1 to +1, where -1 means the lists are in reverse order and +1 means the list are the same.

AP correlation assumes that each list, the ground truth and the system's prediction, gives a total order of items. There is no simple modification of AP correlation to support partial orders. Two challenges must be addressed when thinking about how to evaluate rank accuracy based on incomplete pairwise preferences. How does one assign a rank to an item when a total list cannot be constructed? In Equation 3, $index(i)$ is the rank index in the list, but when a total order is unknown, such as the sample data in Table 1, a new method is needed to give a rank index. Second, how does one consider the cases where a user's preference between items is unknown? In that example, we don't know user's preference between items A and B. How to evaluate a system that makes a predication $A \succ B$?

4. EXPECTED DISCOUNTED RANK CORRELATION

We propose *expected discounted rank correlation (EDRC)* to measure the similarity between two sets of pairwise preferences. Given a set of ground truth pairwise preferences from the user $\mathcal{G}$, and a set of predicted pairwise preferences output by the system $\hat{\mathcal{G}}$, EDRC calculates the expected correlation the two sets. Note that we may have user preferences on a large set of items based on many different user sessions. However, we only consider preferences relating items currently recommended by the system. That is, the set of items in $\mathcal{G}$ and $\hat{\mathcal{G}}$ are the same.

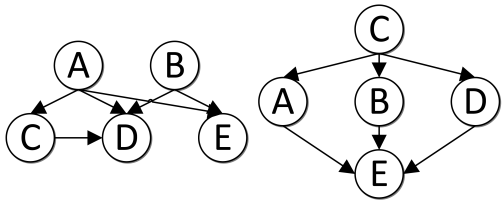


Figure 1: Topological Order based on Ground Truth (left) and System Prediction (right) in Table 1

Similar to AP correlation and nDCG, EDRC emphasizes preserving the order of the user's most preferred items and enforcing a smaller penalty for less preferred items. Different from nDCG whose ground truth is a set of relevance scores, the ground truth supported by EDRC is a set of pairwise preferences. Different from AP correlation which requires complete pairwise preferences, EDRC allows incomplete pairwise preferences.

As discussed in Section 3.2, the presence of incomplete pairwise preferences entails two challenges: the lack of total order among items in the ground truth, and unknown preferences between two items. Next, we look at the first problem.

Assigning Rank and Computing Weight. In the spirit of discounted gain, we want to give different *discount* (weight) for different items. If we know the rank of an item, $R(v)$, in the ground truth, we may set the weight linearly, logarithmically, or exponentially. However, the problem is how to compute the rank of an item given incomplete pairwise preferences. From a set of pairwise preferences as ground truth, we first derive a topological order among the items. Consider a graph where a vertex represents an item, and a directed edge represents a pairwise preference. The item corresponding to the source vertex is preferred to the item corresponding to the target vertex. In the following discussion, we use item and vertex interchangeably.

For example, the ground truth in Table 2 can be represented by the graph in Figure 1, where the fact that item A is preferred to item C is shown by a directed edge from vertices A to C. In this graph, an item is preferred to any item reachable following a directed path. For example, A is preferred to both C and D in the ground truth of Table 1.

Now we discuss how to leverage the graph to compute item rank. We initiate the rank of every vertex to be 1 and traverse the graph beginning for the start nodes. Whenever we follow an edge from a vertex u to v, we update v's rank to $max(R(v), R(u) + 1)$. Note that here we consider every user specified pairwise preference as equally important, and is assigned a unit weight of 1. Thus we keep the maximum score of node among the different paths that can reach this node from a start node. The procedure is defined in the procedure SetRanks($\mathcal{G}$) in Algorithm 1.

The ranks for the items in the graph of Figure 1 are as follows: $R(A) = 1$, $R(B) = 1$, $R(C) = 2$, $R(D) = 3$ and $R(E) = 2$. As we can see, nodes A and B have the same rank, since we don't know user's preferences between the two.

Algorithm 1 Setting Rank Value

SetRanks($\mathcal{G}$)
1: $\mathcal{S}$ = all nodes in $\mathcal{G}$ without an incoming edge
2: **for all** $v \in \mathcal{S}$ **do**
3: $R(v) \leftarrow 1$; Visit(v, 1)
4: **end for**

Visit(v, $rank_{in}$)
1: $R(v) \leftarrow \max(R(v), rank_{in} + 1)$
2: **for all** $w \in OUT(v, \mathcal{G})$ **do**
3: Visit(w, $R(v)$)
4: **end for**

Then we define the discount term, $D(v)$, which can take various forms depending on what type of discount is desired. For example, for a simple linear discount, $D(v) = R(v)$. For exponential discount, $D(v) = 2^{R(v)}$ and for logarithmic discount, $D(v) = log_2(1 + R(v))$.

Handling Unknown Preferences between Items. Another problem is computing the *score*, $C(v)$, of an item in the system's output in the presence of incomplete pairwise preferences in the ground truth. We propose the following formula to define the score of an item where $OUT(v, \mathcal{G}^+)$ is the set of outgoing edges from v in the transitive closure, $\mathcal{G}^+$, of $\mathcal{G}$, and W, the set of items that are more preferred or have an unknown relationship with v where $W = V(\mathcal{G}) \setminus [\{v\} \cup OUT(v, \mathcal{G}^+)]$. EP checks for the *expected score* regarding the relationship between v and all items in W.

$$C(v) = \sum_{w \in W} EP(v, w)$$

For a pair of items (v, w), suppose v preferred over w, that is, $v \succ w$ in $\mathcal{G}$. There are three cases. We have $v \succ w$ in $\hat{\mathcal{G}}$. In this case, $EP(v, w) = 1$. The second case, we have $w \succ v$ in $\hat{\mathcal{G}}$. Since the system prediction contradicts to the ground truth, we have $EP(v, w) = 0$. The third case, the system cannot predict a preference between v and w. Then we need to compute the expected score. By default we may assume there is 50% likelihood for $v \succ w$ and 50% likelihood for $w \succ v$. We then let $EP(v, w) = .5$. Alternatively, we may have a more accurate likelihood estimation based on collaborative filtering. Assuming we have a set of equally similar users, if 70% of the users have $v \succ w$ and 30% of similar users have $w \succ v$, then the expected score of $v \succ w$ is 0.7. For example, below are samples for C and EP based on Table 1.
$C(C) = EP(C, A) + EP(C, B) + EP(C, E) = 0 + .5 + .5 = 1$
$C(D) = EP(D, A) + EP(D, B) + EP(D, C) + EP(D, E) = .5 + .5 + 1 + .5 = 2.5$
$C(E) = EP(E, A) + EP(E, B) + EP(E, C) + EP(E, D) = 1 + 1 + .5 + .5 = 3$

Putting Things Together. Based on the discussion of how to compute a score for an item, $C(v)$, and the discount (weight) of an item, $D(v)$, we now put these together for an evaluation measure EDRC. We denote the set of all vertices in $\mathcal{G}$ without an incoming edge as $\mathcal{S}$.

$$EDRC(\mathcal{G}, \hat{\mathcal{G}}) = \frac{2}{Z} \cdot \left[\sum_{v \in V(\mathcal{G}) \setminus \mathcal{S}} \frac{C(v)}{D(v)} \right] - 1$$

Here, Z is a normalization factor to ensure the value is between +1 and -1.

$$Z = \sum_{v \in V(\mathcal{G}) \setminus \mathcal{S}} \frac{|W|}{R(v)}$$

Considering the example data in Table 1, we now show how to put together the sample calculation for EDRC using a linear discount method.

$$EDRC(\mathcal{G}, \hat{\mathcal{G}}) = \frac{2}{\frac{29}{6}} \cdot \left[\frac{1}{2} + \frac{2.5}{3} + \frac{3}{2} \right] - 1 = \frac{5}{29}$$

When both the ground truth and prediction is a complete set of pairwise preferences and the discount term is $D(v) = R(v) - 1$, the values for EDRC and AP correlation will be the same.

5. CONCLUSION

We consider a problem setting where the input from the user and the system's prediction are sets of possibly incomplete pairwise preferences. Based on this setting, we discuss the limitations of two evaluation methods, nDCG and AP correlation. Then we propose a new rank correlation metric, expected discounted rank correlation (EDRC), that compares two sets of pairwise preferences, one from the ground truth and one from the system prediction. This new measure has wide applications for evaluating recommender systems whose input data and ground truth are obtained from implicit user feedback.

Acknowledgements

This work was sponsored in part by an NSF CAREER Award IIS-0845647 and IIS-0915438. This work was also supported in part by an IBM faculty award.

6. REFERENCES

[1] L. Baltrunas, T. Makcinskas, and F. Ricci. Group recommendations with rank aggregation and collaborative filtering. In *Proceedings of the 4th ACM conference on Recommender systems*, 2010.
[2] K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, October 2002.
[3] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.
[4] R. Kumar and S. Vassilvitskii. Generalized distances between rankings. In *Proceedings of the 19th International World Wide Web Conference*, 2010.
[5] N. N. Liu and Q. Yang. Eigenrank: A ranking-oriented approach to collaborative filtering. In *Proceedings of the 31st Annual International ACM SIGIR Conference*, 2008.
[6] S.-T. Park and W. Chu. Pairwise preference regression for cold-start recommendation. In *Proceedings of the 3rd ACM conference on Recommender systems*, 2009.
[7] S. Rendle, C. Freudenthaler, Z. Gantner, and S.-T. Lars. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009.
[8] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.
[9] E. Yilmaz, J. A. Aslam, and S. Roberston. A new rank correlation coefficient for information retrieval. In *Proceedings of the 31st Annual International ACM SIGIR Conference*, 2008.

Setting Goals and Choosing Metrics for Recommender System Evaluations

Gunnar Schröder
T-Systems Multimedia Solutions GmbH
Riesaer Straße 5
01129 Dresden, Germany
gunnar.schroeder@t-systems.com

Maik Thiele, Wolfgang Lehner
Dresden University of Technology
Faculty of Computer Science, Database Technology Group
01062 Dresden, Germany
{maik.thiele,wolfgang.lehner}@tu-dresden.de

ABSTRACT

Recommender systems have become an important personalization technique on the web and are widely used especially in e-commerce applications. However, operators of web shops and other platforms are challenged by the large variety of available algorithms and the multitude of their possible parameterizations. Since the quality of the recommendations that are given can have a significant business impact, the selection of a recommender system should be made based on well-founded evaluation data. The literature on recommender system evaluation offers a large variety of evaluation metrics but provides little guidance on how to choose among them. This paper focuses on the often neglected aspect of clearly defining the goal of an evaluation and how this goal relates to the selection of an appropriate metric. We discuss several well-known accuracy metrics and analyze how these reflect different evaluation goals. Furthermore we present some less well-known metrics as well as a variation of the area under the curve measure that are particularly suitable for the evaluation of recommender systems in e-commerce applications.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

recommender systems, e-commerce, evaluation, metrics, measure, area under the curve, auc, informedness, markedness, matthews correlation, precision, recall, roc

1. INTRODUCTION

There is a large variety of algorithms for recommender systems that were published in research and have been implemented by the industry. Almost every author claims that a particular algorithm or implementation is superior to another in a certain respect, which makes it difficult to choose among them. Currently a comprehensive and objective comparison of existing recommender systems is hard to find and the available results are sometimes contradictory for different data sets or metrics. How efficient and successful a specific recommender system is also depends on the specific purpose of a recommender system and the characteristics of the domain it is applied to. It is very unlikely that there is a single best solution for any domain and context. If we want to increase the effectiveness of recommendations we have to determine the best fit for a given scenario through thorough evaluation of available algorithms and parameterizations.
This is particularly important for the usage of recommender systems in e-commerce applications where the choice of algorithms can have a significant business impact. In a web shop a better recommender system can have a direct effect on the company's revenue since the recommendations can significantly influence the users' buying decisions [15].
The research that is presented in this paper is derived from an evaluation of various recommendation algorithms that we conducted for a large German e-commerce portal. In order to achieve meaningful evaluation results we developed an evaluation methodology and went through a wide range of literature on the evaluation of recommender systems and information retrieval systems and developed a framework for the evaluation of recommender systems.
In this paper we will focus on the specific evaluation demands of recommender systems in e-commerce applications. We discuss the importance of defining a sufficiently detailed goal and analyze which aspects of the recommender's user interface and the used preference data influence a reasonable choice of metrics. We give an overview of applicable accuracy metrics, explain how to choose among the large variety and highlight some metrics that are particularly well-suited to the evaluation of recommender systems. In order to discuss accuracy metrics in detail a discussion of non-accuracy measures has to be omitted due to space constraints.

2. RELATED WORK

Over time numerous quantitative metrics and qualitative techniques for offline and online evaluations of recommender systems have been published in research. We will start by giving a short overview of some of the most important publications that are relevant to this paper.
Herlocker et al. provide a comprehensive overview of the existing methods for evaluating collaborative filtering systems [9]. Although they focus on collaborative filtering methods, many of the presented evaluation approaches, metrics and techniques are applicable to other types of recommender

systems as well. They conducted an insightful study on the correlation of different metrics and concluded that the analyzed metrics can be subdivided in three major classes.
Olmo and Gaudioso build on this survey and derive a framework for recommender systems that divides recommender systems into a *filter* and a *guide* component [4]. Their aim is to separate the calculation of recommendations from their presentation. They propose to use metrics that focus on the fact whether the recommendations presented by the system are actually followed by the users of the recommender system and suggest to pay more attention to the presentation of recommendations as well as the respective objective of recommender systems.
Cremonesi and Lentini present an evaluation methodology for collaborative filtering recommender systems [2]. They use mean squared error, root mean squared error (RMSE) and mean absolute error (MAE) as well as the classification accuracy metrics precision, recall, f-measure and ROC graphs to compare two collaborative filtering algorithms using the MovieLens[1] data set and a further movie data set obtained from an IPTV service provider.
Konstan et al. summarize findings about the usage of automated recommender systems for information-seeking tasks [12]. They list many problems and challenges in evaluating recommender systems and emphasize the importance of standardized evaluation methodologies, metrics and test data sets for progress in research.
Kohavi et al. provide a survey and practical guide for conducting controlled experiments on the web [11]. In a follow-up paper Crook et al. describe common pitfalls they experienced and emphasize the importance of choosing an *overall evaluation criterion* that truly reflects business goals [3].
Jannach et al. provide a chapter on evaluating recommender systems in their book [10]. They review the current state of research and survey the approaches taken in published papers on the evaluation of recommender systems.
Shani and Gunawardana contributed a chapter on evaluating recommender systems to the handbook by Ricci et al. [14] and describe the most important aspects in conducting offline and online experiments as well as user studies. They outline basic approaches for all three types of evaluations, some important accuracy metrics and discuss various other properties of recommenders that should be considered when evaluating recommenders.
A further valuable source for metrics and measures is the literature on information retrieval. It is, by its very nature, concerned with the quality of search results and provides insight into evaluation methodologies and metrics (e.g.[5]).

The evaluation of recommender systems has emerged as an important topic as more and more algorithms and techniques for recommenders systems are presented. Many different evaluation metrics can be applied in evaluations, although some (e.g. RMSE, MAE and precision) are more frequently used than others. However, authors rarely justify their particular choice and little guidance is offered on how to choose a metric for a specific purpose. Some notable exceptions are the comparison of metrics given by Herlocker et al. [9] and the overview by Jannach et al. [10]. In this paper we try to provide the reader with a better understanding of the existing accuracy metrics and how to apply them in order to evaluate recommenders for specific goals.

[1] `http://www.grouplens.org/node/73`

3. SETTING THE EVALUATION GOAL

The first step of an evaluation should be to define its goal as precisely as possible. Although this seems rather obvious, we would like to emphasize this step since it is often not executed with sufficient care in evaluations in general. We can only choose one or several adequate metrics and interpret their results, if we have set the purpose of our evaluation beforehand. The results of popular evaluation metrics such as RMSE, precision or F_1-measure do not necessarily lead us to the best algorithm for our purpose, as we try to illustrate later on. Only if the set of metrics accurately reflects the specific objectives that are connected to the recommender system in our evaluation and the concrete usage scenario, can we be sure that the obtained results are useful [11, 3]. Moreover, the specific interface choice for the recommender system and the usage patterns that are to be expected should be taken into account. Such a precise evaluation goal could be: Find the recommendation algorithm and parameterization that leads to the highest overall turnover on a specific e-commerce web site, if four product recommendations are displayed as a vertical list below the currently displayed product.

3.1 Objectives for Recommender Usage

Having a clear conception of the objectives we want to achieve by designing, implementing or applying a recommender system is a worthwhile effort, even if we recognize that it may be too difficult or expensive to measure them accurately. In many cases we may discover that only the results of a large scale field study would accurately reflect our evaluation goal. Nevertheless a defined goal will help us in selecting the most appropriate metric or set of metrics that allows us to obtain the most precise measurement of the recommender's suitability for the usage scenario which can be achieved with a reasonable effort.
Common reasons for implementing a recommender system are the desire to improve user satisfaction and to increase the economic success of a platform. Although both goals are interrelated they may be competing in some scenarios. As an example, in e-commerce a recommender may either determine the top recommendations based on the best price-performance ratio for the customer but it may also show the products that are likely to lead to the highest revenue for the business. For this purpose commercial recommenders for web shops often consider a reward attribute for items that models how much the company profits from a sale of a certain item. This information can be used e.g. in combination with classification accuracy metrics (see Section 4.2).
A recommender system in a live environment usually has to be optimized with regard to various other objectives that are related to the technical performance and the system's life cycle such as responsiveness, scalability, peak load, reliability, ramp-up efforts, maintainability, extensibility and cost of ownership [10]. These aspects have a strong influence on the decision among a set of algorithms and implementations and put further constraints on a choice. Further aspects that extend beyond the accuracy of a recommender are coverage, novelty, serendipity, confidence, persuasiveness, trust, robustness, security and many more (cf. [9, 14]). Measuring these diverse qualities of a recommender system is a large field that is beyond the scope of the presented work. We will confine our analysis to accuracy metrics and attempt to analyze how different measures reflect varying objectives.

3.2 Analyzing the Recommender System and its Data

In order to further specify our evaluation goal we have to take a closer look at the recommender's task, its interface and the utilized data.

What is the recommender's task and how does the system interact with the user? In our opinion the most common usage scenarios for recommender systems are *prediction*, *ranking* and *classification* tasks which manifest themselves in different types of user interfaces.

In a *prediction* task the focus lies on predicting ratings for unrated items (e.g. movies, music or news articles) of a user and showing these predictions to him. For example, a movie rental company might show predicted ratings to users in order to aid them in their decision making.

In a *ranking* task the recommender tries to determine an order of items, often with the purpose of creating a top-k list of items. This task is particularly common in e-commerce applications, where a top-k or unlimited ordered list of recommended products is shown in a sidebar or on a dedicated page. But also web portals for news, content and multimedia make heavy use of top-k lists. A further interesting ranking task for a recommender is to order a limited set of search results according the user's predicted preference.

In a *classification* task the recommender determines a set with a limited (often fixed) number of recommended items with no particular order among them implied. E.g. articles or products that are predicted to be of interest to a user are highlighted or annotated on a web page. A top-k list of recommended items can be seen as a classification task as well, especially if the number of items is very small and the order is less prominent e.g. recommended items are arranged in a grid or scrollable in two directions in a circular fashion (cf. Amazon).

Recommender systems are further used to determine *similar items* or *similar users*. E.g. in web shops on a product detail page usually a list of similar products is shown or a news web site often lists articles similar to the one currently shown. Recommending similar users is of particular interest for social web applications such as Facebook. These tasks which focus on the items and users themselves instead of the user's ratings, can be seen as ranking or classification tasks depending on the chosen interface.

Realizing for which of these tasks the recommender system is used and what the user interface looks like is important for choosing the most appropriate metric. The number of displayed recommendations and their visual arrangement (To which extent does it convey a ranking?) should also be considered before choosing a metric. In many usage scenarios more than one of these tasks has to be fulfilled by a recommender system. Therefore it may be sensible to apply one accuracy metric for each of the usage scenarios to find a reasonable compromise or to even consider using different recommendation algorithms for each task.

What kind of user preference data is used by the recommender? The preferences of a user can be gathered either through *explicit* or *implicit* ratings. While an explicit rating is made as a deliberate statement by a user who has the intention to express his or her opinion, an implicit rating is deducted from actions of the user that had a different primary goal. If, for example, a user rates a movie on a web site, this is a direct expression of his or her opinion and preferences, so we consider it an explicit rating. Purchasing a product or clicking on a link to display an article usually has a different primary goal than expressing a positive rating, so these actions are considered an implicit rating.

The user ratings are usually collected using either a *numerical*, a *binary* or a *unary* rating scale. Although other preference models such as textual reviews are conceivable they are rarely used in today's recommender systems.

A *numerical* rating is represented by a number from either a discrete or a continuous rating scale, in most cases with a limited range. Typical examples of a discrete rating scale are ratings on a scale from zero to five stars or Likert response scales that are commonly used in questionnaires. To be precise, these rating scales are actually *ordinal* scales, a fact which is ignored by predictive accuracy metrics (cf. 4.1) that make intensive use of ratios. An example of a continuous rating scale could be a slider that is set by a user and translated to a real value.

A *binary* rating scale allows users to assign items to two different classes (like/dislike). YouTube, for example, allows users to rate movies with either thumb up or thumb down.

A *unary* rating, by contrast, allows users to assign items only to a single class, which is in most cases positive (e.g. like). A prominent example of an explicit unary rating is Facebook's "Like"-button. Implicit unary ratings can be purchased products in a web shop or clicked links on a news page.

The important difference between binary and unary ratings is that unary ratings offer no distinction between disliked and unrated items. With unary ratings we cannot tell whether a user actually dislikes an item or simply does not know the item or does not bother to rate it. In a large web shop, such as Amazon, a customer will only be aware of a small portion of the product catalog. Furthermore, other factors such as limited budget, limited time, external constraints, and products the user already owns determine whether a customer will actually buy a product he or she likes. Being aware of this difference and the implied biases is important when conducting an evaluation and interpreting the results of various metrics.

4. OVERVIEW OF EVALUATION METRICS

Evaluation metrics for recommender systems can be divided into four major classes [9, 4]: 1) Predictive accuracy metrics, 2) Classification accuracy metrics, 3) Rank accuracy metrics and 4) Non-accuracy metrics. Since we confine our analysis to accuracy metrics we will omit the discussion of this class and suggest to consult the literature (e.g. [9, 14]).

4.1 Predictive Accuracy Metrics

Predictive accuracy or *rating prediction* metrics embark on the question of how close the ratings estimated by a recommender are to the true user ratings. This type of measures is very popular for the evaluation of non-binary ratings. It is most appropriate for usage scenarios in which an accurate prediction of the ratings for all items is of high importance. The most important representatives of this class are *mean absolute error* (MAE), *mean squared error* (MSE), *root mean squared error* (RMSE) and *normalized mean absolute error* (NMAE) (cf. [9, 8]). MSE and RMSE use the squared deviations and thus emphasize larger errors in comparison to the MAE metric. MAE and RMSE describe the error in the same units as the computed values, while MSE yields

squared units. NMAE normalizes the MAE metric to the range of the respective rating scale in order to make results comparable among recommenders with varying rating scales. The RMSE metric has been used in the Netflix competition in order to determine the improvement in comparison to the Cinematch algorithm as well as the prize winner. This was a significant source of discussion over the course of the competition.

These predictive accuracy error metrics are frequently used for the evaluation of recommender systems since they are easy to compute and understand. They are well-studied and are also applied in many contexts other than recommender systems. However, they do not necessarily correspond directly to the most popular usage scenarios for recommender systems. There are few cases where users are in fact interested in the overall prediction accuracy. Recommender systems are more commonly used to display a limited list of top ranked items or the set of all items that have been rated above a certain threshold. Many recommendation algorithms are able to provide more accurate statements about a limited set of items that the user either likes or dislikes. The estimations for many other items are rather inaccurate but often also significantly less important to users.

This applies in particular to e-commerce applications where we are usually more concerned with suggesting some products to customers that they will like in comparison to estimating the most accurate ratings for the large amount of items that customers would never purchase.

4.2 Classification Accuracy Metrics

Classification accuracy metrics try to assess the *successful decision making capacity* (SDMC) of recommendation algorithms. They measure the amount of correct and incorrect classifications as relevant or irrelevant items that are made by the recommender system and are therefore useful for user tasks such as finding good items. SDMC metrics ignore the exact rating or ranking of items as only the correct or incorrect classification is measured [9, 4]. This type of metric is particularly suitable for applications in e-commerce that try to convince users of making certain decisions such as purchasing products or services.

A comprehensive overview of classic, basic information retrieval metrics, such as recall and precision and further improved metrics, is given in the survey by Powers [13]. The following short summary is based on his terminology and descriptions for these metrics. In section 5.1 we will discuss three less well-known metrics that were described in this paper as well and that we deem very useful. To the best of our knowledge they have not been used for the evaluation of recommender systems so far.

The common basic information retrieval (IR) metrics are calculated from the number of items that are either relevant or irrelevant and either contained in the recommendation set of a user or not. These numbers can be clearly arranged in a contingency table that is sometimes also called the *confusion matrix* (see Table 1).

Precision or *true positive accuracy* (also *confidence* in data mining) is calculated as the ratio of recommended items that are relevant to the total number of recommended items:

$$\text{precision} = \text{tpa} = \frac{tp}{tp + fp}$$

This is the probability that a recommended item corresponds to the user's preferences. The behavior of this (and the following) IR metrics can be observed in Figure 1 which shows a potential ranking of four relevant items in a set of ten items by a recommender. In examples (a) to (j) the length k of the top-k list varies, while the item order remains fixed.

	Relevant	Irrelevant	Total
Recommended	tp	fp	$tp + fp$
Not Recommended	fn	tn	$fn + tn$
Total	$tp + fn$	$fp + tn$	N

Table 1: A contingency table or confusion matrix that accumulates the numbers of true/false positive/negative recommendations.

Recall or *true positive rate* (also called *sensitivity* in psychology) is calculated as the ratio of recommended items that are relevant to the total number of relevant items:

$$\text{recall} = \text{tpr} = \frac{tp}{tp + fn}$$

This is the probability that a relevant item is recommended. The two measures, precision and recall, are inversely related which we notice if we vary the size of the set of recommendations (cf. Fig. 1). In most cases, increasing the size of the recommendation set will increase recall but decrease precision. Because of this mutual dependence it makes sense to consider precision and recall in conjunction with two other metrics that are called *fallout* and *miss rate*.

Fallout or *false positive rate* is calculated as the ratio of recommended items that are irrelevant to the total number of irrelevant items:

$$\text{fallout} = \text{fpr} = \frac{fp}{fp + tn}$$

This is the probability that an irrelevant item is recommended.

Miss rate or *false negative rate* is calculated as the ratio of items not recommended but actually relevant to the total number of relevant items:

$$\text{missRate} = \text{fnr} = \frac{fn}{tp + fn}$$

This is the probability that a relevant item is not recommended.

Just as precision and recall describe the recommender's performance regarding the true positives, two similar metrics can be defined that measure how the algorithm behaves regarding true negatives. Since this corresponds to interchanging the true and false values of the underlying data for the precision and recall metric, they are called *inverse precision* and *inverse recall*.

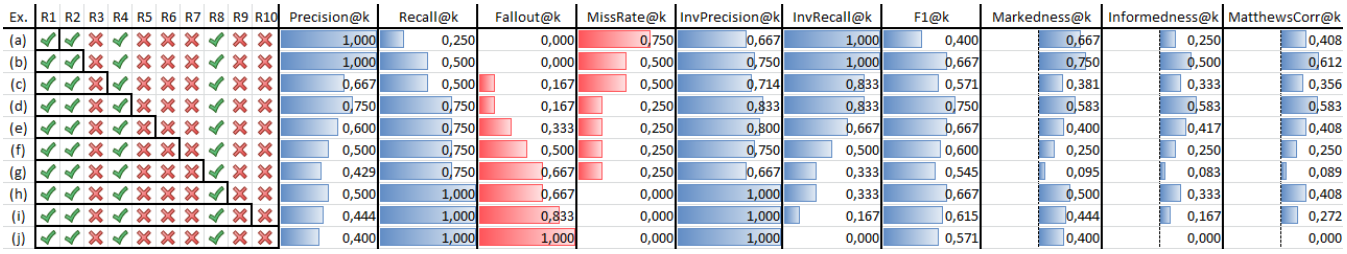
Inverse precision or *true negative accuracy* is calculated as the ratio of items not recommended that are indeed irrelevant to the total number of not recommended items:

$$\text{inversePrecision} = \text{tna} = \frac{tn}{fn + tn}$$

This is the probability that an item which is not recommended is indeed irrelevant.

Inverse recall or *true negative rate* (also called *specificity*) is calculated as the ratio of items not recommended that are really irrelevant to the total number of irrelevant items:

$$\text{inverseRecall} = \text{tnr} = \frac{tn}{fp + tn} = 1 - \text{fpr}$$



Ex.	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Precision@k	Recall@k	Fallout@k	MissRate@k	InvPrecision@k	InvRecall@k	F1@k	Markedness@k	Informedness@k	MatthewsCorr@k
(a)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	1,000	0,250	0,000	0,750	0,667	1,000	0,400	0,667	0,250	0,408
(b)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	1,000	0,500	0,000	0,500	0,750	1,000	0,667	0,750	0,500	0,612
(c)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,667	0,500	0,167	0,500	0,714	0,833	0,571	0,381	0,333	0,356
(d)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,750	0,750	0,167	0,250	0,833	0,833	0,750	0,583	0,583	0,583
(e)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,600	0,750	0,333	0,250	0,800	0,667	0,667	0,400	0,417	0,408
(f)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,500	0,750	0,500	0,250	0,750	0,500	0,600	0,250	0,250	0,250
(g)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,429	0,750	0,667	0,250	0,667	0,333	0,545	0,095	0,083	0,089
(h)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,500	1,000	0,667	0,000	1,000	0,333	0,667	0,500	0,333	0,408
(i)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,444	1,000	0,833	0,000	1,000	0,167	0,615	0,444	0,167	0,272
(j)	✓	✓	✗	✓	✗	✗	✗	✓	✗	✗	0,400	1,000	1,000	0,000	1,000	0,000	0,571	0,400	0,000	0,000

Figure 1: Varying the length k of a top-k list for a possible recommender's ranking of ten items.

This is the probability that an irrelevant item is indeed not recommended.

The *F_1-measure* and *F_β-measure* try to combine precision and recall into a single score by calculating different types of means of both metrics. The F_1-measure or F_1-score is calculated as the standard *harmonic mean* of precision and recall:

$$\mathrm{F}_1 = \frac{2}{\frac{1}{\mathrm{precision}} + \frac{1}{\mathrm{recall}}} = \frac{2 \cdot \mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

Moreover there is a large variety of metrics that are derived from these basic information retrieval metrics [9, 5]. *Mean average precision* (MAP) is a popular metric for search engines and is applied, for example, to report results at the Text Retrieval Conference (TREC) [5]. It takes each relevant item and calculates the precision of the recommendation set with the size that corresponds to the rank of the relevant item. Then the arithmetic mean of all these precisions is formed.

$$\mathrm{AP} = \frac{\sum_{r=1}^{N} (P(r) \times \mathrm{rel}(r))}{\text{number of relevant documents}}$$

Afterwards we calculate the arithmetic mean of the average precisions of all users to get the final mean average precision:

$$\mathrm{MAP} = \frac{\sum_{u=1}^{M} \mathrm{AP}_u}{M}$$

Various other means including the geometric mean (GMAP), harmonic mean (HMAP) and quadratic mean (QMAP) can be applied instead. All of these measures emphasize true positives at the top of a ranking list. This behaviour can be observed in Figure 2 (a) to (j) where the position of a single relevant item within a ranking of ten items is varied.

Further derived measures that we will discuss later are *ROC curves* and the *area under the curve* (AUC) measure.

4.3 Rank Accuracy Metrics

A *rank accuracy* or *ranking prediction* metric measures the ability of a recommender to estimate the correct order of items concerning the user's preference, which is called the measurement of *rank correlation* in statistics. Therefore this type of measure is most adequate if the user is presented with a long ordered list of items recommended to him. A rank prediction metric uses only the relative ordering of preference values so that is independent of the exact values that are estimated by a recommender. For example, a recommender that constantly estimates items' ratings to be lower than the true user preferences, would still achieve a perfect score as long as the ranking is correct.

For the usage of rank accuracy metrics, it is important to know whether they measure total or partial orderings. Most rank accuracy metrics such as *Kendall's tau* and *Spearman's rho* compare two total orderings. The problem with these measures is that in most cases we are not provided with a full ranking of all items. Many recommendation algorithms can generate only a partial list of items that are most likely to be preferred by a user. All other items would have to be concatenated to the list in an arbitrary order. Furthermore, the recommendation list can contain groups of items with a similar rating that can appear in varying orders. The same applies to the true preferences of a user. In order to create a full ranking of the items all preference values for the user have to be known. Since the user might express the same rating for several items the list will again contain groups of items that can appear in an arbitrary order. The largest problem is posed by items for which no user rating is known. These items could in fact hold an arbitrary place within the ranking. Sometimes it is assumed that ratings for items that are preferred are known, so that the unknown items are concatenated to the end of the list. In general, however, the unknown items could as well contain items that would appear within the top ranks if rated by the user [9].

The bottom line is that in most cases a rank metric for partial orderings would be more appropriate for comparing recommendation lists that are produced by recommenders to item rankings from known user preferences. One possibility is to use an arbitrary total ordering that complies with the partial ordering, though the results will become less meaningful when the number and size of item groups with identical rating increases. An evaluation might state that a certain ranking is inferior even though only items within groups of identical ratings switched their places. A better solution is to compare all or a subset of the total orderings that comply with the partial orderings and average the results. However, this can easily become combinatorially challenging when both rankings are in fact partial orderings. Fagin et al. discuss several derived metrics for comparing partial orderings that could be applied for recommender systems [6]. Bansal and Fernandez-Baca provide runtime efficient implementations of these metrics [1].

We think that rank accuracy metrics are very well suited for e-commerce applications if they allow to compare partial rankings in a meaningful way. In our implementation we used a variation of the Kendall's tau metric for partial rankings on boolean classifications. That is, we compare the partial ranking provided by recommended and not recommended items with the partial ranking of relevant and irrelevant items. This boolean Kendall's tau is in fact highly correlated to the AUC metric (e.g. Fig. 2).

5. IMPROVED METRICS FOR RECOMMENDER SYSTEMS

5.1 Informedness, Markedness and Matthews Correlation

A major problem with the frequently used metrics precision, recall and F_1-measure is that they suffer from severe biases. The outcome of these measures not only depends on the accuracy of the recommender (or classifier) but also very much on the ratio of relevant items. In order to illustrate this we imagine an ideal recommender and its inverse and apply them to generate top-k lists of four items for varying ratios of relevant items (see Fig. 3). The examples show that the informatory value of all three measures varies. In extreme cases with a highly skewed ratio its value can be even misleading (cf. Fig. 3 e – h).
To solve this problem Powers [13] introduced three new metrics that try avoid these biases by integrating the inverse precision and inverse recall respectively.
Markedness combines precision and inverse precision into a single measure and expresses how marked the classifications of a recommender are in comparison to chance:

$$\begin{aligned}\text{markedness} &= \text{precision} + \text{inversePrecision} - 1 \\ &= \frac{tp}{tp+fp} + \frac{tn}{fn+tn} - 1\end{aligned}$$

Informedness combines recall and inverse recall into a single measure and expresses how informed the classifications of a recommender are in comparison to chance:

$$\begin{aligned}\text{informedness} &= \text{recall} + \text{inverseRecall} - 1 \\ &= \frac{tp}{tp+fn} + \frac{tn}{fp+tn} - 1\end{aligned}$$

Both markedness and informedness return values in the range $[-1, 1]$.
The *Matthews Correlation* combines the informedness and markedness measures into a single metric by calculating their geometric mean:

$$\begin{aligned}\text{correlation} &= \frac{(tp \cdot tn) - (fp \cdot fn)}{\sqrt{(tp+fn) \cdot (fp+tn) \cdot (tp+fp) \cdot (fn+tn)}} \\ &= \pm\sqrt{\text{informedness} \cdot \text{markedness}}\end{aligned}$$

The range of the Matthews Correlation is $[-1, 1]$, so the sign in the second representation of the formula actually depends on the respective signs for markedness and informedness.
We propose to replace the measures precision and recall by markedness and informedness for most evaluation purposes in order to avoid being misled by underlying biases. As we can see in Figures 1 and 3, markedness, informedness and Matthews Correlation are significantly more helpful in choosing an adequate size for a top-k recommendation list. Furthermore derived metrics such as mean average precision (MAP) could as well be replaced by their respective equivalents (e.g. mean average markedness).

5.2 Limited Area Under the Curve

ROC curves provide a graphical representation for the performance of a recommender system, an information retrieval system or any other type of binary classifier. A ROC curve plots recall (true positive rate) against fallout (false positive rate) for increasing recommendation set size. An

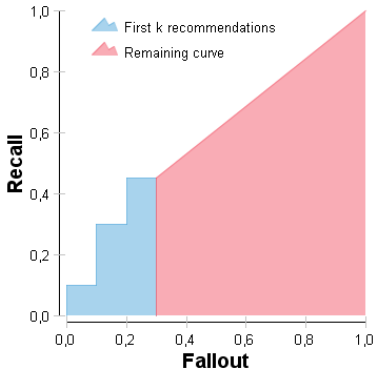
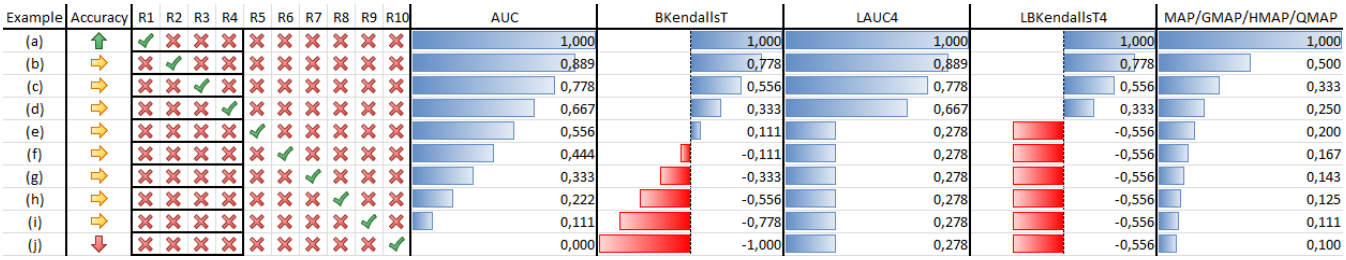


Figure 5: An example for the ROC curve used by our variant of the AUC measure.

in-depth discussion of ROC curves can be found in the paper by Fawcett [7].
A perfect recommender would yield a ROC curve that goes straight up towards 1.0 recall and 0.0 fallout until all relevant items are retrieved. Afterwards it would go straight right towards 1.0 fallout while the remaining irrelevant items follow. The obvious aim is consequently to maximize the area under the ROC curve. The *area under the curve* (AUC) can therefore be used as a single measure for the overall quality of a recommender system. However, a frequently uttered point of criticism is that users are often more interested in the items at the top of recommendation lists but that the AUC measure is equally affected by swaps at the top or the bottom of a recommendation list (cf. Figures 2 and 4 e – h). This may be a disadvantage if we are mainly interested in finding the top ranked items and thus care mostly for the first part of the ROC graph. Therefore, in addition to the standard AUC measure, we implemented a slight variation of the AUC measure. This *limited area under the curve* (LAUC) measure uses the same approach but instead of measuring the complete area, it takes only the area under the first part of the curve into account. This is the part of the curve which is formed by the first k recommendations.
However, measuring this partial area is not easy due to several problems:

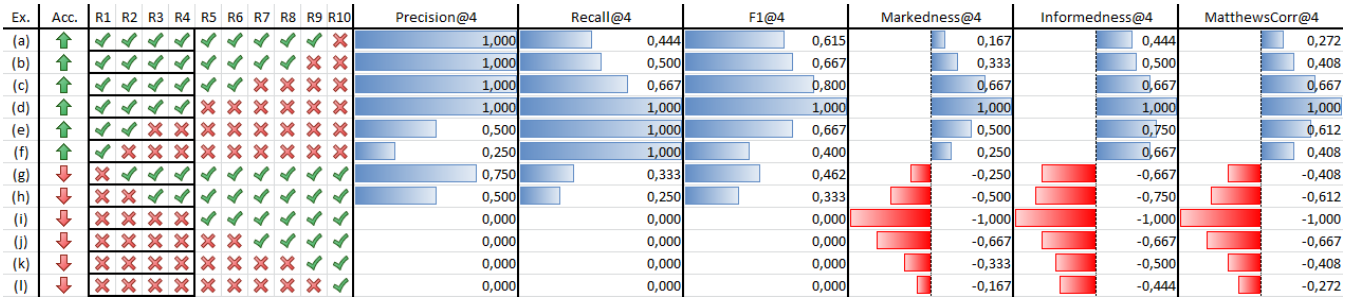
- The measure has to be normalized in a certain way to be comparable.
- The normalized measure should return one if all relevant items are retrieved first by the recommendation algorithm and fit within the recommendation list.
- If the recommendation list contains only relevant items, then the area under the curve is in fact zero. Nevertheless the normalized measure should still return one.
- Relevant items that are retrieved at the end of the list with no irrelevant items following do not add to the area under the limited curve.

A good solution for these problems is to generate just a limited recommendation list that only contains a fixed number of items and to assume that all other relevant items will be distributed uniformly over the rest of the ranking list until all items are retrieved. This means that we calculate the AUC measure for the first part of the ROC curve in the standard way until the first k recommendations have been retrieved. Then we take the end point of the ROC curve



Example	Accuracy	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	AUC	BKendallsT	LAUC4	LBKendallsT4	MAP/GMAP/HMAP/QMAP
(a)	↑	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	1,000	1,000	1,000	1,000	1,000
(b)	→	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	0,889	0,778	0,889	0,778	0,500
(c)	→	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	0,778	0,556	0,778	0,556	0,333
(d)	→	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	0,667	0,333	0,667	0,333	0,250
(e)	→	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	0,556	0,111	0,278	-0,556	0,200
(f)	→	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	0,444	-0,111	0,278	-0,556	0,167
(g)	→	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	0,333	-0,333	0,278	-0,556	0,143
(h)	→	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	0,222	-0,556	0,278	-0,556	0,125
(i)	→	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	0,111	-0,778	0,278	-0,556	0,111
(j)	↓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	0,000	-1,000	0,278	-0,556	0,100

Figure 2: Varying the position of a single relevant item on a four out of ten recommendation list.



Ex.	Acc.	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Precision@4	Recall@4	F1@4	Markedness@4	Informedness@4	MatthewsCorr@4
(a)	↑	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	1,000	0,444	0,615	0,167	0,444	0,272
(b)	↑	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	1,000	0,500	0,667	0,333	0,500	0,408
(c)	↑	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	1,000	0,667	0,800	0,667	0,667	0,667
(d)	↑	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	1,000	1,000	1,000	1,000	1,000	1,000
(e)	↑	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	0,500	1,000	0,667	0,500	0,750	0,612
(f)	↑	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	0,250	1,000	0,400	0,250	0,667	0,408
(g)	↓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	0,750	0,333	0,462	-0,250	-0,667	-0,408
(h)	↓	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	0,500	0,250	0,333	-0,500	-0,750	-0,612
(i)	↓	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	0,000	0,000	0,000	-1,000	-1,000	-1,000
(j)	↓	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	0,000	0,000	0,000	-0,667	-0,667	-0,667
(k)	↓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	0,000	0,000	0,000	-0,333	-0,500	-0,408
(l)	↓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	0,000	0,000	0,000	-0,167	-0,444	-0,272

Figure 3: Varying the ratio of relevant items in a four out of ten recommendation list and applying two recommenders with perfect and inverse accuracy.

formed by the first k recommendations and draw a straight line to the upper right corner (see Fig. 5). The area under the curve that is situated to the right of the curve formed by the top-k list thus is a simple trapezoid.
The resulting LAUC measure is very useful for the evaluation of recommender systems that are applied to generate top-k lists of items (cf. Figures 2 and 4):

- The measure returns one if all relevant items are retrieved within the top-k list.
- The measure becomes minimal if no relevant items are retrieved within the top-k list. If all irrelevant items are retrieved first and fit within the top-k list the measure returns zero.
- A top-k list that contains more relevant items will yield a higher score than a list with less relevant items, except if the length of the list is close to the total number of items. In this case the order of relevant and irrelevant items within the recommendation list would have a higher influence on the overall score.
- If a relevant item moves towards the top of the list the measure increases.
- A swap at the top or bottom of the top-k list has the same effect on the measure's value.
- All changes beyond the end of the top-k list are ignored by the measure.

A similar variation of the boolean Kendall's tau, which considers only the top-k recommendations, is another useful and highly correlated metric (e.g. Fig. 2).

6. SOME GUIDELINES FOR CHOOSING A METRIC

In order to choose an appropriate evaluation metric for a given recommendation scenario, it is helpful to answer the following questions:

Is there a distinction between rated and unrated items? If all items are implicitly rated predictive accuracy metrics are not applicable, because there are no unrated items for which we can predict a rating and measure the accuracy.

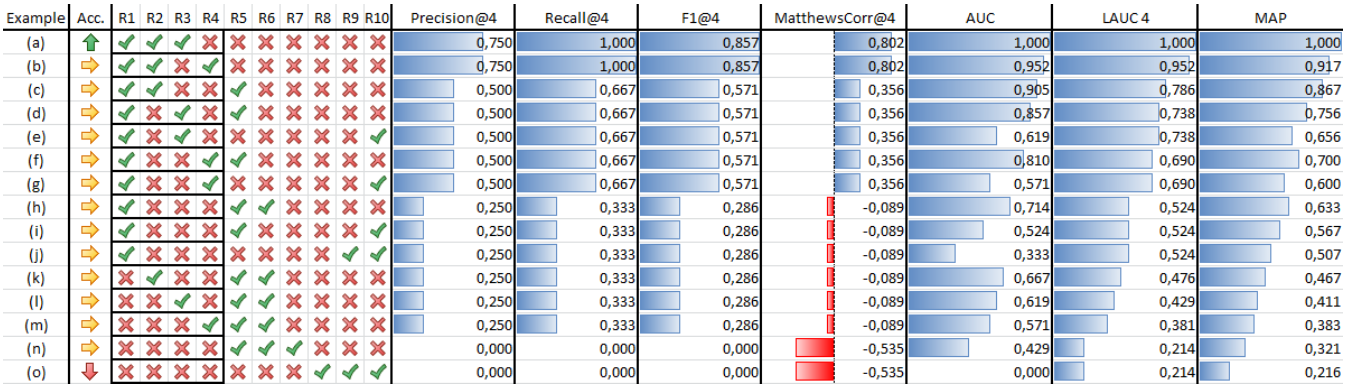
Are items rated on a numerical or a binary scale? A binary rating scale usually suggests a classification or ranking task.

Are users interested in rating predictions or only in top-ranked items? If users only care about top-ranked (or lowest-ranked) items and not about individual rating scores for items this suggests a classification or ranking task. Although an algorithm may use predicted ratings internally, it has to succeed in estimating the top-ranked (or lowest-ranked) items and a higher error on rating predictions for items is acceptable as long as top-ranked (or lowest-ranked) items are identified correctly. Therefore an evaluation should focus on classification or ranking metrics.

Is a limited list of top-ranked items shown? If yes, a metric that measures the overall predictive accuracy or overall ranking accuracy is not appropriate. The exact rating predictions and ranking of other items are irrelevant to users and should not be considered by the metric.

Do the recommended items have or imply an order? Users will usually consider recommendations in a certain order, in particular if many recommendations are shown. If this is the case, basic information retrieval metrics such as precision, recall, markedness and informedness are not sufficient since they ignore the order among the recommended items. A metric that considers the order of recommended items as well is more appropriate for this purpose.

How fast does the user's interest in lower ranked items decay? The metric should reflect the user's decay in interest. MAP and GMAP, for example, emphasize the first recommendations in contrast to the AUC or Kendall's tau measure that weighs swaps in lower and higher ranks in the



Example	Acc.	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Precision@4	Recall@4	F1@4	MatthewsCorr@4	AUC	LAUC 4	MAP
(a)	↑	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	0,750	1,000	0,857	0,802	1,000	1,000	1,000
(b)	→	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	0,750	1,000	0,857	0,802	0,952	0,952	0,917
(c)	→	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	0,500	0,667	0,571	0,356	0,905	0,786	0,867
(d)	→	✓	✗	✓	✗	✓	✗	✗	✗	✗	✗	0,500	0,667	0,571	0,356	0,857	0,738	0,756
(e)	→	✓	✗	✓	✗	✗	✗	✗	✗	✗	✓	0,500	0,667	0,571	0,356	0,619	0,738	0,656
(f)	→	✓	✗	✗	✓	✓	✗	✗	✗	✗	✗	0,500	0,667	0,571	0,356	0,810	0,690	0,700
(g)	→	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	0,500	0,667	0,571	0,356	0,571	0,690	0,600
(h)	→	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,714	0,524	0,633
(i)	→	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	0,250	0,333	0,286	-0,089	0,524	0,524	0,567
(j)	→	✓	✗	✗	✗	✗	✗	✗	✗	✓	✓	0,250	0,333	0,286	-0,089	0,333	0,524	0,507
(k)	→	✗	✓	✗	✗	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,667	0,476	0,467
(l)	→	✗	✗	✓	✗	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,619	0,429	0,411
(m)	→	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	0,250	0,333	0,286	-0,089	0,571	0,381	0,383
(n)	→	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗	0,000	0,000	0,000	-0,535	0,429	0,214	0,321
(o)	↓	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	0,000	0,000	0,000	-0,535	0,000	0,214	0,216

Figure 4: Varying the position of three relevant items on a four out of ten recommendation list.

same way (cf. Fig. 2). If users hardly look at the ranking of lower ranked items, a classification or ranking error for lower ranked items becomes irrelevant.

7. CONCLUSION

In this paper we analyzed the relation between objectives for applying recommender systems and metrics used for their evaluation. We emphasized the importance of defining a precise goal for the evaluation and discussed how the data used by the recommender as well as its user interface affect the specific task for the recommender. We gave an overview of the three main classes of accuracy related evaluation metrics and discussed their applicability for different types of recommender systems. In order to illustrate the specific advantages and disadvantages of various metrics discussed, we compared them using several informative examples. We proposed to utilize markedness, informedness and Matthews correlation as classification metrics since they are superior to precision, recall and F_1-measure for most purposes. We presented a new variation of the area under the curve measure that is particularly suited for top-k recommendations which are used in many e-commerce applications. This limited area under the curve measure combines classification and ranking accuracy to create a better measure for this purpose. Furthermore, we provided some crisp guidelines that help to choose an appropriate evaluation metric for a specific usage scenario.

8. ACKNOWLEDGMENTS

The work presented in this paper is funded by the European Social Fund and the Free State of Saxony under the grant agreement number 080954843.

9. REFERENCES

[1] M. S. Bansal and D. Fernández-Baca. Computing distances between partial rankings. *Information Processing Letters*, 109(4):238 – 241, 2009.

[2] P. Cremonesi, R. Turrin, E. Lentini, and M. Matteucci. An evaluation methodology for collaborative recommender systems. In *AXMEDIS*, pages 224–231, Washington, DC, USA, 2008.

[3] T. Crook, B. Frasca, R. Kohavi, and R. Longbotham. Seven pitfalls to avoid when running controlled experiments on the web. In *Proceedings of the 15th ACM SIGKDD*, pages 1105–1114, 2009.

[4] F. H. del Olmo and E. Gaudioso. Evaluation of recommender systems: A new approach. *Expert Systems with Applications*, 35(3):790–804, October 2008.

[5] Eighteenth Text REtrieval Conference. Appendix a: Common evaluation measures. In *The Eighteenth Text REtrieval Conference (TREC 2009) Proceedings*, 2009.

[6] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee. Comparing partial rankings. *SIAM Journal on Discrete Mathematics*, 20(3):628–648, 2006.

[7] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[8] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transaction on Information Systems*, 22(1):5–53, January 2004.

[10] D. Jannach, M. Zanker, A. Felfernig, and G. Friedich. *Recommender Systems: An Introduction*. Cambridge University Press, 1 edition, 9 2010.

[11] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne. Controlled experiments on the web: survey and practical guide. *Data Min. Knowl. Discov.*, 18:140–181, February 2009.

[12] J. A. Konstan, S. M. McNee, C.-N. Ziegler, R. Torres, N. Kapoor, and J. Riedl. Lessons on applying automated recommender systems to information-seeking tasks. In *AAAI*, 2006.

[13] D. M. Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness and correlation. Technical report, School of Informatics and Engineering, Flinders University Adelaide, South Australia, 2007.

[14] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors. *Recommender Systems Handbook*. Springer, Berlin, 1st edition, 10 2010.

[15] M. Zanker, M. Bricman, S. Gordea, D. Jannach, and M. Jessenitschnig. Persuasive online-selling in quality and taste domains. In *Electronic Commerce and Web Technologies*, pages 51–60, 2006.