MovieExplorer: Building an Interactive Exploration Tool from Ratings and Latent Taste Spaces

Taavi T. Taijala
University of Minnesota
GroupLens Research
United States
taavi@taijala.com

Martijn C. Willemsen
Eindhoven University of Technology
School of Innovation Sciences
Netherlands
M.C.Willemsen@tue.nl

Joseph A. Konstan
University of Minnesota
GroupLens Research
United States
konstan@umn.edu

ABSTRACT

This [1] paper introduces and evaluates MovieExplorer, an interactive exploration tool designed to use the data available in a traditional ratings-based recommender system to provide an interactive interface more suited to user exploration and fulfillment of short-term recommendation needs. A field deployment with 1,950 users showed that users found the tool useful for a variety of exploration and short-term recommendation tasks, even preferring it to existing interfaces for several tasks. Experimentation with several design features found that the actual user navigation algorithms were significant (user satisfaction was lower for algorithms with faked controls) and that offering positive and negative feedback options led to increased feedback and user retention.

CCS CONCEPTS

• **Information systems → Information retrieval → Recommender systems • Information systems → Information retrieval → Users and interactive retrieval**

KEYWORDS

Information Exploration Tools, Interactive Recommendation

ACM Reference format:

T. Taijala, M. Willemsen, J. Konstan. 2018. In *Proceedings of ACM SAC Conference, Pau, France, April 9-13, 2018 (SAC'18)*, 10 pages. DOI: 10.1145/3167132.3167281

1 INTRODUCTION

In this paper, we introduce and evaluate MovieExplorer, an interactive exploration tool designed to integrate into a collaborative filtering recommender system. MovieExplorer leverages the latent taste space of a matrix-factorization-style recommender algorithm to create a user-navigable space. Our tool provides a simple interface that allows users to steer towards (and in one version away from) movies based their current needs and interests, enabling user-directed open-ended exploration of the movie space. We carry out an online field experiment with 1,950 users, evaluating the system as a whole, as well as specific design choices related to the type of feedback users can provide and the starting point for navigation.

The challenge of integrating exploration into recommender systems is not new. Early collaborative filtering systems focused primarily on predicting a user's ratings. As we review below, later researchers discovered that the set of items with the "highest predicted ratings" is often not the most useful set to recommend to a user. This set may not match the users' changing short-term needs and interests, and it runs the risk of guiding users into narrow product spaces or so called "filter bubbles." One reaction has been to adjust recommenders to consider additional factors beyond predicted rating, including diversity, novelty, and serendipity. Most of these systems still take a non-interactive approach to recommendation, although many do provide the option to explicitly search or filter by product attributes when such data is available (e.g. limiting recommendations to new movies or comedies).

Dialog-based and conversational recommenders were developed to offer users greater control over exploration. Many of these systems used case-based, attribute-driven approaches that allowed users to interactively refine their query by providing feedback on individual recommended items. Where they work well, these systems can be quite effective, but they tend to require substantial knowledge engineering.

In this work, we explore how to get the benefits of conversational interaction—in particular, rich incremental refinement of short-term preferences—while leveraging the benefits of a ratings-based collaborative filtering system. We are not the first to attempt this—some prior work we review below has allowed users to select individual recommended items as reflective of their short-term interests; however, we extend this existing work in four important ways:

1. We propose an algorithm that supports user navigation in a latent taste space.
2. We allow users to provide multiple elements of positive (and in one version negative) feedback during each interaction.
3. We explore and outline the design space for conversational collaborative filtering systems like ours.
4. We evaluate our system as a whole, as well as several key design features, in a large online field experiment.

SAC'18, April 9-13, 2018, Pau, France

DOI: 10.1145/3167132.3167281

The rest of this paper is organized as follows. Section 2 is a walkthrough and description of our tool. Section 3 reviews prior work and existing systems; section 4 explores the design space for such systems. Section 5 presents our study design, and section 6 reports and discusses the results of the study.

2 DESCRIPTION of MOVIEEXPLORER

2.1 Example Session

Meet Jane, a fictitious user looking for a movie to watch with her father, who doesn't like "frivolous" movies. She's normally a fan of romantic comedies, so her existing top-N recommendation list on MovieLens is poorly targeted to meet her current need. Instead, Jane turns to MovieExplorer to find a movie that both she and her father will enjoy.

In her first round, Jane receives ten movie suggestions. Having already seen and enjoyed it, she recognizes *Apollo 13* as the type of movie her father might also enjoy, so she tells the system to show her more movies more like it. She doesn't have strong opinions on the other nine movies.

Taking her feedback into account, it suggests ten new movies in the second round (top half of Figure 1). She recognizes a few she's seen or heard of, but none of them are quite right. She asks for movies more like *Saving Private Ryan*, *The Hunt for Red October*, and *Lincoln*, and less like *Toy Story 2*.

The system moves on to the third round (bottom half of Figure 1), where it suggests ten new movies. Notice that these movies not only reflect her previous feedback (which indicated a preference for serious titles), but also span a range of genres from history to war to science fiction. Perusing these movies, Jane comes across *Bridge of Spies*, which she vaguely remembers having previously heard about. After reading more about it and watching a trailer, she decides that *Bridge of Spies* seems like a good movie to watch with her father.

2.2 Algorithm

The core of our tool is an iterative algorithm that allows users to navigate a latent taste space, generated by a 30-dimensional nonnegative matrix factorization [29] of the MovieLens ratings dataset. The dataset contains nearly 25 million ratings from 250,000 users on 50,000 movies, collected over a period of 20 years, but we limit our tool to only the 10,000 most popular movies (those with approximately 90+ ratings), to ensure each movie's latent taste space representation is meaningful.

The algorithm proceeds in rounds, with the user starting from an initial system-chosen location in the latent taste space. In each round, the system selects a diverse set of ten highly-rated movies surrounding the user's current location and displays them. Depending on the version, the user can provide either positive only or positive and negative feedback (shown in Figure 1) for each of these items, which the system uses to refine the user's location in the latent taste space, moving towards movies that received positive feedback and away from ones that received negative feedback. This process repeats in each round. Java code for our algorithm is included in an online appendix.

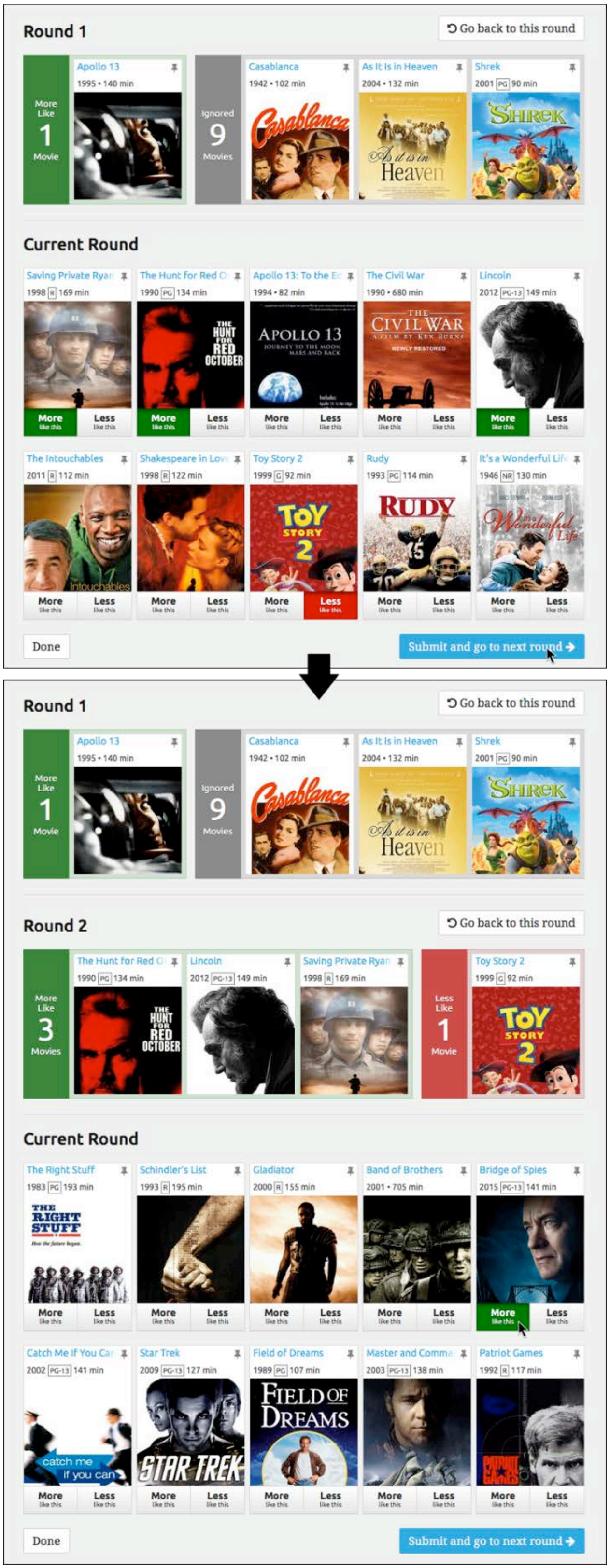

Figure 1. Jane's progression through MovieExplorer
Embedded thumbnails retrieved from the TMDB API.
https://www.themoviedb.org/

3 RELATED WORK

3.1 Conversational Recommender Systems

Conversational recommender systems use an interactive back-and-forth dialogue between the user and system to build ephemeral profiles of short-term preferences from which to make recommendations. They can react more quickly to changing user needs, at the cost of higher effort.

In critiquing systems, a common type of conversational recommender, this dialogue takes the form of the system making a recommendation and the user critiquing it along one or more dimensions (e.g. price, size, etc.). Burke et al. introduced some of the earliest examples of critiquing recommender systems with the FindMe collection of recommenders, which included the well-known Entrée restaurant recommender [1]. Early systems like Entrée were extremely limited—users only saw one primary recommendation at a time and were only allowed to choose among a small set of simple system-suggested critiques (e.g. cheaper, different cuisine, etc.).

Motivated by a desire to increase decision accuracy, improve choice confidence, and reduce cognitive effort [2], subsequent research on critiquing systems explored numerous extensions:

- Multiple items displayed per screen [2, 33].
- Compound system-suggested critiques [2, 24].
- Dynamic system-suggested critiques [17, 24].
- User-initiated critiques [2, 33].
- Tradeoffs between critiqued attributes [2, 33].

These systems all share a common drawback: to make their recommendations, they require extensive taxonomies of item attributes, which are labor-intensive to build and maintain. This has limited the real-world applications of critiquing systems, despite their benefits. A 2012 survey of critiquing systems mentions only two real world applications [2]: Amazon's "fix this recommendation", and MovieLens's "MovieTuner".

MovieTuner is noteworthy in that it allowed users to critique movies based on their relevance to various user-submitted tags [34]. In theory, using user-submitted data could help reduce the labor involved in building and maintaining such a system, but in practice, getting tagging data, especially for rarer titles, is hard.

In information retrieval, Cutting et al. propose Scatter/-Gather, a search tool employing the same iterative refinement strategy used in conversational recommenders [4]; however, the tool's reliance on textual data limits its applications.

Where meaningful attributes and metadata are available, traditional conversational recommender systems can provide sufficient support for short-term needs and exploration. Unfortunately, such data is often unavailable or of insufficient detail, quality, or coverage.

3.2 Collaborative Filtering Systems

Automated collaborative filtering, dating back to the GroupLens system [25, 27], was designed to use ratings from like-minded users in place of content or attribute data (later techniques improved on this by including other types of data when available). While originally measured by their success in predicting ratings, such systems quickly focused on the challenge of recommending items to users, often through what are known as "top-N lists."

These top-N lists are based on a user's long-term historical preferences, making for recommendations that are reasonable overall, but are poorly targeted to what the user needs or wants now. The literature on preference construction tells us that user preferences are highly context-dependent and change frequently [35]. Most systems provide no support for adapting their recommendations to meet these ever-changing short-term needs and interests. Furthermore, these top-N lists are non-interactive, preferring to *tell* the user what he or she will like, rather than enabling them to *explore* and *discover* new items. They too often focus on low-risk, low-reward "obvious" recommendations, which limits exposure to novel items and risks guiding users into undesirable "filter bubbles" [12, 18].

Recognizing these limitations, researchers and commercial systems have proposed various solutions:

- Context-aware recommender systems infer observable parts of a user's context (e.g. location) from sensor data [3, 5, 31]; however, a user's observable context only partially defines his or her short-term needs.
- Zhao et al. keep a user's top-N list fresh by promoting items from further down, increasing engagement [39].
- Netflix employs multiple top-N recommendation lists, each oriented around frequently changing, algorithmically-chosen topics, called "alt-genres" [13]. This approach transforms the traditional one-dimensional top-N list into a two-dimensional grid, giving users a wider range of items to choose from to meet their various needs and interests.
- Some systems use multi-armed bandits to build a more complete profile of a user's preferences [15, 22].
- Taramigkou et al. propose a music recommender that supports "guided exploration." Users indicate a type of music to explore towards, and the system uses tag data to build a playlist that gradually moves from the user's current tastes to the target type of music [30].
- Many others have focused on measuring and improving the quality of recommendations in top-N lists along user-centric dimensions, including diversity [18, 21, 32, 40], novelty [7, 9, 14, 21], and serendipity [7], showing improved overall user satisfaction.

The above systems are still largely non-interactive, which precludes any form of user-directed open-ended exploration of the item space. Herlocker et al. show that users value being able to browse and explore items in a recommender system, listing exploration as a task on which recommender systems should be evaluated [7]. Furthermore, Hoeffler et al. show that exploration is an important part of learning about a domain. They show that the breadth (rather than depth) of a user's experience in a domain is a positive predictor of the user's ability to "identify, predict, and appreciate higher-quality products" [8].

One straightforward solution is to allow users to directly control their top-N recommendations. Shafer et al. built the MetaLens system, which allows users to better match their short-term needs and interests by manipulating a set of predefined weights and filters [26]. Kveton and Berkovsky propose applying minimal interaction search to recommender systems by asking the user a series of content preference questions, which are used to refine their recommendations [11]. While an improvement over non-interactive systems, these approaches have the same limitation as conversational recommenders: they require a database of item attributes or content data.

3.3 Conversational Collaborative Filtering

Rafter and Smyth propose a system that combines conversational recommendation with user-user collaborative filtering [23]. Their system combines a temporary short-term preference profile with a user's standard long-term preference profile to recommend items. To refine their recommendations, users can pick a preferred item from the set, or no item at all. Picking an item adds it to the user's temporary short-term preference profile with positive feedback, while picking no items adds the entire set of recommendations to the profile with negative feedback. The system generates new recommendations from the updated profiles, and the process repeats. Kelly and Bridge extend this work with a diversity-aware algorithm which reduces the number of iterations required for a user to reach a specific item [10].

These approaches rely on user-user collaborative filtering, which has largely been replaced (due to poor performance) by newer approaches, such as item-item collaborative filtering and matrix factorization techniques. Our tool extends conversational collaborative filtering to modern matrix factorization techniques.

3.4 Navigating Information Spaces

MovieExplorer navigates over a set of 30 latent attributes generated by a nonnegative matrix factorization of ratings data [29]. This approach obviates the need for extensive taxonomies of item attributes, allowing us to use the same preference data used by collaborative filtering systems. Since matrix factorization is commonly used to power today's collaborative filtering systems, our tool is able to leverage existing infrastructure, reducing implementation complexity compared to stand-alone solutions.

Unfortunately, the critiquing approach to navigation used by many conversational recommenders does not work with latent attributes, since they are not easily explainable. Smyth and McGinty identify four common feedback strategies in conversational recommenders [28], two of which do not require explainable attributes:

- Preference-based feedback: The user selects a preferred item from a set. Rafter and Smyth employed this feedback strategy.
- Ratings-based feedback: The user rates items from a set.

Graus et al.'s work on alleviating the user cold-start problem provides an example of how the preference-based feedback strategy can be applied to support navigating a space of latent attributes. In their system, users repeatedly select "pivot" items to steer towards from a set of diverse options, allowing them to iteratively navigate the latent attribute space until they reach an ideal starting point [6].

Loepp et al. propose a feedback strategy specific to latent attributes which was not examined by Smyth and McGinty—directly asking users to express preferences for latent attributes by providing carefully chosen examples near each of a latent attribute's two extremes [16]. By comparing these two sets of examples, users can guess what the latent attribute represents and then express a preference for it. Unfortunately, this task is difficult for users, and Loepp et al. found that users had trouble distinguishing between more than five latent attributes.

We build on these works to create a conversational collaborative filtering system that supports navigating latent attributes. We choose to explore a ratings-based feedback strategy, as we feel it provids better control than the preference-based feedback strategy employed in previous work [6, 23].

However, the choice of feedback strategy is only one part of successful navigation. Navigating information spaces has been extensively researched in the literature on wayfinding, although the lessons learned there have not been widely applied to recommender systems. Wiener et al. present a taxonomy of the diverse array of tasks that users undertake when navigating information spaces [37], while others have looked at how to design systems to support these tasks [19]. We use this to inform our design of MovieExplorer.

4 DESIGN SPACE

Designing a conversational collaborative filtering system meant exploring a design space markedly different from that of traditional conversational recommenders. Since we are testing what we think is the first such tool to use a ratings-based feedback strategy to navigate through a space of latent attributes, we wanted to methodically explore the design space. In this section, we review 12 key design decisions we identified during development, describe the choices we made (summarized in Table 2), and explain our rationale for making them. We also present a number of alternative design choices, which we hope others will explore in future work.

4.1 Interaction Model

The interaction model reflects the nature of the user experience and the user interface. It covers how the user interacts with tool, including when and how they see items, what feedback they can provide, and what controls they have over the process.

4.1.1 Balancing Navigation and Exploration Needs. To properly support navigation and exploration, users need items that help them identify "where they are," items that let them "move in a direction," and novel items that let them explore (see Table 1). We considered interfaces that both physical/temporally separated these three item types, as well as integrated them in a single set.

MovieExplorer generates a single, integrated set of items designed to support all three of the above user needs. The items are chosen to surround the user's current location (at varying distances) in a manner that spans nearby tastes, allowing the user to navigate freely in all directions. We make various adjustments to ensure both recognizable and novel items are shown.

4.1.2 Number of Items Displayed. How many items should be displayed on each screen? Options include single items, pairs of items, some other number, and a "never-ending" list of items.

MovieExplorer displays 10 items per screen. We chose this number because it provides users with a sufficient number of items to use for navigation without being overwhelming. Displaying more items also improves the chances that a user will recognize some (but not all) of them, which is important for helping the user to both provide meaningful feedback and situate himself or herself in the latent taste space, while still allowing for the discovery of novel items.

4.1.3 Amount and Type of Feedback. What input can the user provide on each screen of recommendations? Options include:

- Selecting a "pivot item" to steer towards (or away from).
- Rating some or all of the items (e.g. star ratings, thumbs up/down, etc.).
- Other types of qualitative feedback such as "I've seen it," "I'm not familiar with it," etc.

MovieExplorer experiments with both unary (positive-only) and binary (positive-negative) feedback. Users can apply feedback to any number of items. We felt that using a single "pivot item," as in Graus et al.'s work [6], was too coarse and that star ratings might be confused with long-term ratings.

Table 1. Item Types and Purposes

	Situational Items	**Navigational Items**	**Recommended Items**
Helps the user to…	"know where they are."	"move in different directions."	"explore novel items."
Desirable Properties	• Recognizable • Near the user's current location	• Recognizable • Surround the user's current location, but at sufficient distances that differences are readily apparent	• Novel • Near the user's current location

4.1.4 Refreshing the Items Displayed. Does the user or system control when new items are added to the screen? Are old items removed? Which ones? Here are some possible approaches:

- Each time the user provides feedback on an item, it is removed from the screen with new item replacing it.
- When the user is done providing feedback, they click a submit button which removes any items inconsistent with their feedback, replacing them with items that are consistent.
- A never-ending set of items slowly scrolls across the screen with old items being removed from one side and replaced by new items on the other side.

With MovieExplorer, when users click the "submit" button, the current set of items is moved up the page to the history section (where they no longer accept feedback), and a new set of items are displayed in their place. In early designs, we found this approach to be easy for users to understand and use.

4.1.5 User Control. How much control over the process does the user have?

MovieExplorer provides a limited set of user controls to ensure a more homogenous experience between the users in our study. Users are only able to start a new session; go back to a previous point in a session; provide item feedback; submit their feedback and refresh the list; or add items to their wishlist.

4.2 Navigational Model

The navigational model concerns how the system states and user states are represented and updated during navigation.

4.2.1 Location Model. How does the system encode and store a user's current "location" in the latent attribute space (or more generally, encode and store the user's state)? A vector is an obvious choice when using a latent attribute space, but other choices are possible, including multiple vectors [36], latent attribute thresholds, or sets of good/bad items.

MovieExplorer uses a vector space to represent the item space and individual vectors to represent both items and the user's current location. We normalize the user's current location to the unit sphere, making it directional only.

4.2.2 Starting Location. What is the system's default starting location? Is it chosen by the system or the user? If chosen by the system, does it change from use to use or is it always the same? Is the starting location personalized to each user?

MovieExplorer experiments with using three different system-chosen starting locations, with users evenly split between the three:

- Nonpersonalized: the average of all users' long-term preference vectors.
- Long-term preference: the current user's long-term preference vector.
- Short-term preference: the average of the vectors for the current user's last five highly rated items (4+ stars).

4.2.3 Constraining vs. Non-constraining. Does each round of feedback permanently constrain the parts of the item space a user can subsequently reach, like in Scatter/Gather [4]?

MovieExplorer uses a non-constraining approach. We felt that this approach provided better support for open-ended exploration, since users could more easily recover from mistakes, or simply change their minds and reverse direction.

Table 2. Design parameters chosen for MovieExplorer

Balancing navigation and exploration needs	Select a single set of items balanced between ones that help the user situate themselves in the space of items, move in different directions, and explore novel items.
Number of items displayed	10 items at a time.
Amount and type of feedback *(experimental condition)*	• Users provide positive/no feedback on any number of items. • Users provide positive/negative/no feedback on any number of items.
Refreshing the items displayed	User-controlled, all items refreshed.
User control	Users can start over, go back to a previous round, set item feedback, submit item feedback, and wishlist an item.
Location model	A vector space with individual vectors representing both items and the user's current location.
Starting location *(experimental condition)*	• Nonpersonalized: the average of all users' long-term preference vectors. • Long-term preference: the current user's long-term preference vector. • Short-term preference: the average of the vectors for the current user's last five highly rated items (4+ stars).
Constraining vs non-constraining	Non-constraining.
Persistent preferences	No use of long-term user preferences, except as a starting location for users in one of the experimental conditions.
Determinism	Randomly exclude some items from candidate pool during item selection to promote freshness.
Diversity vs. Proximity	Limit item selection to top-N items with the highest relevance, then greedily select the most diverse ones.
Familiarity vs. Novelty	Prefer popular, highly-rated items during item selection, but ignore previously shown and some rated items.

4.2.4 Persistent Preferences. Do a user's previous uses of the tool affect their future results? Are the long-term preferences of users used by the system?

MovieExplorer ignores previous uses and long-term preferences, except in one experimental condition where a user's long-term preferences are used as his or her starting location. We chose this approach to ensure that users would be free to navigate anywhere in the item space, regardless of past uses or long-term preferences.

4.3 Item Selection

Item selection concerns how the system picks what items to show to the user on each screen. We present several important considerations for system designers.

4.3.1 Determinism. A deterministic item selection process may increase monotony, reduce serendipity, and lead to lower user satisfaction when measured over multiple uses. Designers may add randomness to prevent this, with the caveat that too much randomness may be detrimental to the user experience.

MovieExplorer injects randomness into item selection by randomly excluding some of the items near the user's current location from the candidate pool from which items are selected. We felt this provided the optimal balance between accuracy of results and increased novelty during repeated uses.

4.3.1 Diversity vs. Proximity. Diversity in recommendations has been shown to increase user satisfaction and reduce choice difficulty [38, 40]. Sufficient diversity is also necessary for effective navigation in all directions. On the other hand, the items closer to the user's current location are more likely to be relevant.

MovieExplorer tries to balance diversity and proximity in order to suggest movies both relevant to the user and useful for navigation. We select a candidate pool to items close to the user's current location for proximity, and then we greedily select items from this pool with the largest minimum distance from the items already selected to ensure diversity (as in Willemsen et al. [38]).

4.3.1 Familiarity vs. Novelty. Familiar items allow users to more easily situate themselves in the item space and provide navigational feedback more quickly, while novel items make better recommendations and are an essential part of exploration.

When calculating how close an item is to the user's current location, we factor in the item's vector magnitude (an indication of quality) to shift the candidate pool slightly towards popular, highly-rated items, which tend to be more familiar to users. On the other hand, we randomly choose to ignore some rated items to increase novelty. In addition, we ignore items that have already been displayed to the user in previous rounds, resulting in increased novelty in later rounds.

5 EXPERIMENTAL EVALUATION

We deployed our tool into the existing MovieLens website. We randomly assigned users into a between-subjects experiment, collecting survey data after first use, and usage data over a follow up 35-day period.

5.1 Research Questions

RQ1: Is this tool valuable to users? What tasks do they find it useful for? How does it compare to top-N recommendation?
RQ2: What is the effect of the navigational algorithm?
RQ3: What is the effect of allowing users to provide negative feedback on items in addition to positive feedback?
RQ4: What is the effect of the different starting locations?

5.2 Recruitment and Subject Population

We promoted the MovieExplorer tool using a banner on the MovieLens homepage and had 1,950 users try it over a period of 25 days. Users who clicked on the banner were randomized into one of 18 experimental conditions and given a short set of instructions explaining how to use the tool. Users were allowed to use the tool as much as they liked. When navigating away from the tool after first use, users were asked to take a survey. Users who did not respond were asked again after subsequent uses.

5.3 Experimental Conditions

We conducted a 3x2x3 between-subjects experiment.

5.3.1 Navigational Algorithm. Our first experimental condition compares our navigational algorithm against two baseline "sham" algorithms that do not respond to user feedback.

- 80% (n=1,562) were assigned to the real navigational algorithm. This unequal assignment was designed to minimize exposure of users to a potentially undesirable experience, while still providing sufficient statistical power.
- 10% (n=161*) were assigned to an algorithm that ignored user feedback and selected successive sets of 10 items from a top-N recommendation list generated from the same non-negative matrix factorization used by the real algorithm. **Users assigned to this condition with insufficient rating data were excluded from the study, reducing the sample size.*
- 10% (n=227) were assigned to an algorithm that ignored user feedback and simply selected items at random.

5.3.2 Type of Feedback. We compared two types:

- Unary positive-only feedback (50%).
- Binary positive-negative feedback (50%).

5.3.3 Starting Location. We compared three starting locations:

- Nonpersonalized: an average of all users' long-term preference vectors (33%).
- Long-term preference: the current user's long-term preference vector (33%).
- Short-term preference: the average of the vectors for the current user's last five highly rated items (33%).

Users in the latter two conditions with insufficient rating data were excluded from the study. These conditions had no effect on users in the baseline algorithm conditions.

5.4 Data Collection.

During a 35-day observation period, we logged participants' usage and interaction data for both the MovieExplorer tool and the rest of the MovieLens website. The data collected includes:

- Timestamps for user actions, including visiting the MovieLens website, starting new sessions, going back to previous rounds, or submitting feedback on a round.
- The user's vector location in each round, the movies displayed and the user's subsequent feedback on them.
- Movies a user hovers over, clicks on, wishlists, or rates.

5.5 Survey Design.

Our study also included an optional survey presented to users at the end of their first use (and after subsequent uses if the user did not respond or opt out). The survey included the questions below.

Satisfaction and predicted use: We presented eight questions, in three parts, to assess users' overall satisfaction and intent to return. First, we asked users to rate the tool on a 7-point scale on the dimensions below derived from QUIS [20]. For analysis, the questions were combined into a single "combined QUIS score."

- Terrible vs. Wonderful
- Difficult vs. Easy
- Frustrating vs. Satisfying
- Dull vs. Stimulating
- Rigid vs. Flexible

Next, we asked users to rate their agreement to the statements below on a 5-point likert scale. For analysis, the questions were combined into a single "combined general satisfaction score."

- "I will probably use this feature again"
- "I would recommend this feature to a friend"
- "I think this feature is a major improvement to MovieLens"

Finally, users were asked to predict how often they would use the tool when they visited MovieLens in the future: almost every time, some of the time, only occasionally, or not at all.

Task usefulness and preference: Users were asked to

Table 3. Participation and usage summary

	n	Survey Resp.	Logins Used	Sessions	Rounds per Session
Real algorithm	*1,562*	529	1.7	2.8	8.6
Top-N algorithm	*161*	61	1.5	2.5	9.5
Random algorithm	*227*	51	1.5	3.5	8.1
For real algorithm					
Positive-only feedback	*766*	230	1.6	2.8	8.6
Positive-negative feedback	*796*	299	1.7	2.7	8.7
For real algorithm					
Nonpersonalized starting location	*564*	186	1.5	2.8	8.5
Long-term preference starting location	*454*	151	1.7	2.8	8.0
Short-term preference starting location	*544*	192	1.7	2.7	9.3

The table above shows the usage summary for users in different experimental conditions. Not all survey responses were fully completed. Logins used counts the average number of MovieLens login session in which the tool was used. Users are automatically logged out after being inactive for one hour. Sessions counts the average number of independent uses of the tool (i.e. each time a user starts over at the starting location).

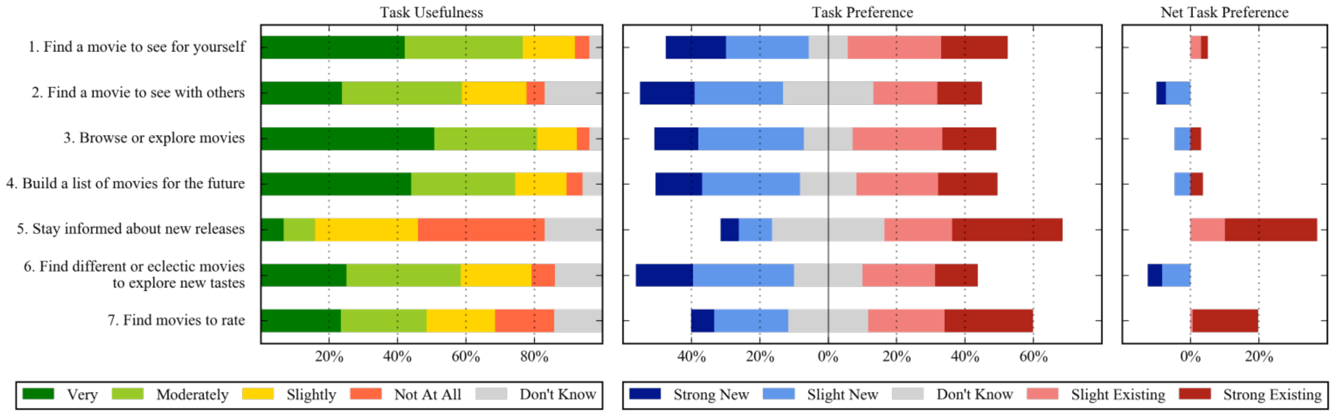

Figure 2. Task usefulness and preference for users in the real algorithm condition (n=359)

evaluate usefulness of MovieExplorer and preference compared with existing MovieLens on severn tasks (as shown in Figure 2). Note that MovieExplorer was not designed to support tasks 5 and 7, which were included as a sanity check.

6 RESULTS AND DISCUSSION

Table 3 provides a summary of user participation and usage for the experiment. Remaining results are organized by research question.

RQ1: *Is this tool valuable to users? What tasks do they find it useful for? How does it compare to top-N recommendation?*

Overall assessment: For the real algorithm, users reported an average combined QUIS score of 23.5 (significantly higher than the neutral value of 20; p<0.001) and an average combined general satisfaction score of 9.8 (significantly higher than the neutral value of 9; p<0.001), indicating positive user experience.

Per-task assessment: Figure 2 shows that more than 50% of users found MovieExplorer very or moderately useful for the five tasks it was designed to support (1, 2, 3, 4, and 6). A substantial number of users preferred MovieExplorer to the familiar MovieLens interface for all tasks, with a majority preferring it for tasks 2, 3, 4, and 6.

Intent to use and actual use: Most respondents would use the tool at least semi-frequently in the future, with 16% saying they'd use it almost every time and 45% saying they'd use some of the time they used MovieLens. Over the 35-day observational period, we found that a third of users in the real algorithm condition returned at least once. On average, these users used the tool in 1.95 (28.8%) of their subsequent MovieLens visits.

Result: Users in the real algorithm condition report a positive user experience with our tool, finding it useful and preferable to an existing "top-N" recommender various tasks.

RQ2: *What is the effect of the navigational algorithm?*

In Table 4 we compare our real navigational algorithm to two "sham" baseline algorithms (top-N and random) on the combined QUIS and general satisfaction scales. Users in the real algorithm condition report a significantly more positive user experience, suggesting users can perceive navigational differences.

Table 5–7 show that users in the real algorithm condition are more likely to return, provide more positive and less negative feedback per round, and rate more movies than users in the two "sham" conditions. This suggests that purposeful navigation significantly affects user experience and behavior.

Result: We find that purposeful navigation is an important part of user experience (i.e. user satisfaction and retention).

RQ3: *What is the effect of allowing users to provide negative feedback on items in addition to positive feedback?*

For the real algorithm, we found no substantial differences in survey responses or rating/wishlisting behavior between users in the two feedback conditions; however, Table 5 shows that users in the positive-negative feedback condition were significantly more likely to return to use the tool 1+, 2+, and 3+ times. In addition, Table 7 shows that users in the positive-negative feedback condition provided both significantly more total feedback per round as expected (3.5 vs. 2.1, p<0.001) and significantly more positive feedback per round as well (2.5 vs 2.1, p<0.001). We compared feedback conditions for the two "sham" algorithms, where results were unaffected by feedback, and found a similar result, suggesting that the increased positive feedback in the positive-negative condition is a reaction to the interface rather than better navigational control.

With the increase in total feedback per round, we expected that users in the positive-negative feedback condition would

Table 4. User satisfaction

	Combined QUIS likert scale			Combined general satisfaction likert scale		
	n	Mean score		*n*	Mean score	
Real algorithm	*367*	23.5		*387*	9.8	
Top-N algorithm	*48*	21.0	p=0.015	*50*	8.7	p=0.020
Random Algorithm	*31*	18.5	p<0.001	*34*	7.6	p<0.001

The table above shows the average scores for the combined QUIS likert scale (ranging from 5–35 inclusive, neutral value of 20) and the combined general satisfaction likert scale (ranging from 3 to 15 inclusive, neutral value of 9). Both scales are intended to measure overall user satisfaction with the tool.

For each column above, we performed two two-sided t-tests, comparing the real algorithm condition with the top-N algorithm condition and the random algorithm condition. We applied Holm-Bonferroni corrections to correct p-values for the two comparisons between algorithms. Bracketed results are significant.

spend more time per round; however, we found no significant difference between the two conditions.

Result: Offering positive-negative feedback results in significantly increased return rates, with no significant changes in user satisfaction or rating/wishlisting behavior.

RQ4: *What is the effect of the different starting locations?*

For the real algorithm, the three starting locations showed similar user satisfaction, but showed different properties and user behaviors. Table 6 shows that the nonpersonalized and short-term preference starting location conditions provided more items that users subsequently rated or wishlisted, compared to the long-term preference condition. In addition, the nonpersonalized starting location appears to be particularly well-suited for reminding users of previously-seen-but-unrated items (as measured by ratings made within three hours [18]). We attribute this to users in this condition being shown items significantly more popular than those shown to users in the other two conditions (mean 12.8k ratings vs. 9.5k and 9.8k; p<0.001 for both comparisons). Future work might explore using a popularity filter to better untangle the effects of these different starting locations.

Surprisingly, Table 5 suggests that users in the long-term preference starting location condition returned to use the tool more in the future, despite finding fewer desirable items.

Result: Users in the nonpersonalized and short-term preference starting location conditions were the most productive as measured by rating/wishlisting behavior, while users in the long-term preference condition were more likely to return. To help users discover novel items, we would choose to use either the nonpersonalized or short-term preference starting locations.

7 CONCLUSION

We presented a new conversational recommender algorithm based on matrix factorization to better support exploration. We explore the design space for such tools and deployed and evaluated a prototype in an online experiment.

We found that users find our tool useful, and even prefer it to an existing "top-N" recommender, for a variety of short-term needs and exploration related tasks. We also explored certain design choices and come to some clear conclusions:

Table 6. Return rate

	n	1+ times		2+ times		3+ times	
Real algorithm	*1,562*	33.8%	p=0.018	14.2%	p=0.032	6.9%	
Top-N algorithm	*161*	30.4%		11.2%		3.7%	
Random algorithm	*227*	25.1%		8.4%		4.8%	
For real algorithm							
Positive-only feedback	*766*	30.8%	p=0.014	12.3%	p=0.031	5.5%	p=0.029
Positive-negative feedback	*796*	36.7%		16.1%		8.3%	
For real algorithm							
Nonpersonalized starting location	*564*	29.1%	p=0.004	13.5%		6.4%	
Long-term preference starting location	*454*	38.8%		15.4%		7.0%	
Short-term preference starting location	*544*	34.6%		14.0%		7.4%	

The table above shows the percentage of users who returned to use the tool in the given minimum numbers of subsequent login sessions. Users are automatically logged out from the MovieLens website after being inactive for one hour.

For each column, we performed a series of chi-square tests for equality of proportions. We compared the real algorithm condition with the top-N algorithm condition and the random algorithm condition; the two feedback conditions; and the three pairwise combinations of starting location conditions. We applied Holm-Bonferroni corrections to correct p-values for the two comparisons between algorithms and the three pairwise comparisons between starting locations. Bracketed results are significant.

Table 5. Rating and wishlisting behavior

	n	Rated within 3 hours		Rated after 3 hours		*n*	Wishlisted	
Real algorithm	*1,562*	1.57	p<0.001 p<0.001	3.07	p<0.001 p<0.001	*1,079*	3.12	p<0.001
Top-N algorithm	*161*	0.43		1.41		*117*	3.47	
Random algorithm	*227*	0.40		0.55		*134*	1.10	
For real algorithm								
Positive-only feedback	*766*	1.54		3.34		*526*	3.30	
Positive-negative feedback	*796*	1.61		2.80		*553*	2.94	
For real algorithm								
Nonpersonalized starting location	*564*	2.41	p<0.001	3.44	p=0.011	*353*	3.38	
Long-term preference starting location	*454*	0.55	p<0.001	2.13		*338*	2.22	p=0.045
Short-term preference starting location	*544*	1.56		3.46		*388*	3.65	

The table above shows how many unique items the average user rated within 3 hours, rated after 3 hours, and wishlisted after having seen them in the MovieExplorer tool. The 3-hour cutoff was chosen in previous work to separate ratings made after being reminded about a previously seen movie from ratings made after watching a movie in response to a recommendation [18].

For each of these three behaviors, we performed a series of two-sided t-tests. We compared the real algorithm condition with the top-N algorithm condition and the random algorithm condition; the two feedback conditions; and the three pairwise combinations of starting location conditions. We applied Holm-Bonferroni corrections to correct p-values for the two comparisons between algorithms and the three pairwise comparisons between starting locations. Bracketed results are significant.

Note that the sample sizes for the "wishlisted" behavior are smaller, as we excluded users who had never used the wishlist feature on MovieLens from those calculations.

Table 7. Feedback per round

	Positive-only feedback			Positive-negative feedback				
	Positive			Positive			Negative	
Real algorithm	2.1	p<0.001	p<0.001	2.5	p<0.001	p<0.001	1.0	p=0.003
Top-N algorithm	1.1			1.4			1.5	
Random algorithm	0.7			1.0			1.9	

The table above shows the average number of movies that received positive and negative feedback per round, calculated as the average of user averages, to ensure each user's experience carries the same weight, regardless of how much they used the tool.

For each column above, we performed two two-sided t-tests, comparing the real algorithm condition with the top-N algorithm condition and the random algorithm condition. We applied Holm-Bonferroni corrections to correct p-values for the two comparisons between algorithms. Significant results are indicated by brackets.

- Purposeful navigation is an important component of an exploration tool. Users can tell the difference between algorithms that respond to their feedback and those that don't, and they strongly prefer the former.
- Allowing users to provide both positive and negative feedback makes them more likely to return.
- Starting users at their long-term preferences results in them rating and wishlisting fewer items; however, they are more likely to return. This means the optimal starting location will depend on a system's goals.

MovieExplorer is a proof-of-concept system. Having laid out the design space and opportunity for this type of exploration tool, we look forward to future work which explores the design space experimentally in other ways and other domains.

ACKNOWLEDGMENTS

We would like to thank F. Maxwell Harper and Qian Zhao for their help with the implementation of the MovieExplorer system and algorithm. This material is based upon work supported by a National Science Foundation Graduate Research Fellowship under Grant No. 00039202.

REFERENCES

[1] R. D. Burke, K. J. Hammond, and B. C. Yound. 1997. The FindMe approach to assisted browsing. IEEE Expert 12, 4 (July 1997), 32–40.

[2] Li Chen and Pearl Pu. 2012. Critiquing-based recommenders: survey and emerging trends. User Model. User-Adapt. Inter-act. 22, 1 (2012), 125–150.

[3] Keith Cheverst, et al. 2000. Developing a context-aware electronic tourist guide: some issues and experiences. In Proc. CHI '00, 17–24.

[4] Douglass R. Cutting et al. 1992. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In Proc. SIGIR '92, 318–329.

[5] Nicolas Ducheneaut et al. 2009. Collaborative filtering is not enough? Experiments with a mixed-model recommender for leisure activities. User Model. Adapt. Pers. (2009), 295–306.

[6] Mark P. Graus and Martijn C. Willemsen. 2015. Improving the User Experience During Cold Start Through Choice-Based Preference Elicitation. In Proc. RecSys '15, 273–276.

[7] Bruce P. Douglass. 1998. Statecarts in use: structured analysis and object-Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. TOIS 22, 1 (2004), 5–53.

[8] Steve Hoeffler, Dan Ariely, Pat West, and Rod Duclos. 2013. Preference exploration and learning: the role of intensiveness and extensiveness of experience. J. Consum. Psychol. 23, 3 (July 2013), 330–340.

[9] Komal Kapoor, et al. 2015. "I Like to Explore Sometimes": Adapting to Dynamic User Novelty Preferences. In Proc. RecSys '15, 19–26.

[10] John Paul Kelly and Derek Bridge. 2006. Enhancing the diversity of conversational collaborative recommendations: a comparison. Artif. Intell. Rev. 25, 1 (2006), 79–95

[11] Branislav Kveton and Shlomo Berkovsky. 2015. Minimal interaction search in recommender systems. In Proc. IUI '15, 236–246.

[12] Neal Lathia, et al. 2010. Temporal diversity in recommender systems. In Proc. SIGIR '10, 210–217.

[13] Emily Lawrence. 2015. Everything is a Recommendation Netflix, Altgenres and the Construction of Taste. Knowl. Organ. 42, 5 (2015).

[14] Kibeom Lee, Woon Seung Yeo, and Kyogu Lee. 2010. Music recommendation in the personal long tail. Proc WOMRAD ACM RecSys (2010).

[15] Lihong Li et al. 2010. A contextual-bandit approach to personalized news article recommendation. In Proc. WWW '10, 661–670.

[16] Benedikt Loepp, Tim Hussein, and Jüergen Ziegler. 2014. Choice-based preference elicitation for collaborative filtering recommender systems. In Proc. CHI '14, 3085–3094.

[17] Kevin McCarthy et al. 2005. Experiments in Dynamic Critiquing. In Proc. IUI '05, 175–182.

[18] Tien T. Nguyen et al. 2014. Exploring the Filter Bubble: The Effect of Using Recommender Systems on Content Diversity. In Proc. WWW '14, 677–686.

[19] Laurence Nigay and Frédéric Vernier. 1998. Design method of interaction techniques for large information spaces. In Proceedings of the working conference on Advanced visual interfaces, 37–46.

[20] Kent L. Norman et al. 1989. Questionnaire for user interaction satisfaction. HCI Lab Coll. Park Univ. Md..

[21] Pearl Pu, Li Chen, and Rong Hu. 2011. A User-centric Evaluation Framework for Recommender Systems. In Proc. RecSys '11, 157–164.

[22] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning diverse rankings with multi-armed bandits. In Proc. ICML '08, 784–791.

[23] Rachael Rafter and Barry Smyth. 2005. Conversational Collaborative Recommendation – An Experimental Analysis. Artif. Intell. Rev. 24, 3–4 (November 2005), 301–318.

[24] James Reilly et al. 2004. Dynamic critiquing. In ECCBR, 763–777.

[25] Paul Resnick et al. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In Proc. CSCW '94, 175–186.

[26] J. Ben Schafer, Joseph A. Konstan, and John Riedl. 2002. Meta-recommendation systems: user-controlled integration of diverse recommendations. In Proc. CIKM '02, 43–51.

[27] Upendra Shardanand and Pattie Maes. 1995. Social information filtering: algorithms for automating "word of mouth." In Proc. CHI '95, 210–217.

[28] Barry Smyth and Lorraine McGinty. 2003. An analysis of feedback strategies in conversational recommenders. In the Fourteenth Irish Artificial Intelligence and Cognitive Science Conference (AICS 2003).

[29] Gábor Takács et al. 2008. Investigation of various matrix factorization methods for large recommender systems. In Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition, 6.

[30] Maria Taramigkou, et al. 2013. Escape the Bubble: Guided Exploration of Music Preferences for Serendipity and Novelty. In Proc. RecSys '13, 335–338.

[31] Mark Van Setten, Stanislav Pokraev, and Johan Koolwaaij. 2004. Context-aware recommendations in the mobile tourist application COMPASS. In Adaptive hypermedia and adaptive web-based systems, 515–548.

[32] Saúl Vargas and Pablo Castells. 2011. Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems. In Proc. RecSys '11, 109–116.

[33] Paolo Viappiani, Boi Faltings, and Pearl Pu. 2006. Evaluating preference-based search tools: a tale of two approaches. In Proc. AAAI '06, 205–211.

[34] Jesse Vig, Shilad Sen, and John Riedl. 2011. Navigating the tag genome. In Proc. IUI '11, 93–102.

[35] Caleb Warren, A. Peter McGraw, and Leaf Van Boven. 2011. Values and preferences: defining preference construction. Wiley Interdiscip. Rev. Cogn. Sci. 2, 2 (March 2011), 193–205.

[36] Jason Weston, Ron J. Weiss, and Hector Yee. 2013. Nonlinear latent factorization by embedding multiple user interests. In Proc. RecSys '13, 65–68.

[37] Jan M. Wiener, Simon J. Büchner, and Christoph Hölscher. 2009. Taxonomy of human wayfinding tasks: A knowledge-based approach. Spat. Cogn. Comput. 9, 2 (2009), 152–165.

[38] Martijn C. Willemsen et al. 2011. Using latent features diversification to reduce choice difficulty in recommendation lists. RecSys '11, (2011), 14–20.

[39] Qian Zhao et al. 2017. Toward Better Interactions in Recommender Systems: Cycling and Serpentining Approaches for Top-N Item Lists. In Proc. CSCW '17, 1444–1453.

[40] Cai-Nicolas Ziegler et al. 2005. Improving Recommendation Lists Through Topic Diversification. In Proc. WWW '05, 22–32.