Generating Transparent, Steerable Recommendations from Textual Descriptions of Items

Stephen Green
Sun Microsystems Labs
Burlington, MA
stephen.green@sun.com

Paul Lamere
The Echo Nest
Somerville, MA
paul@echonest.com

Jeffrey Alexander
Sun Microsystems Labs
Burlington, MA
jeffrey.alexander@sun.com

François Maillet
Dept of Computer Science
Université de Montréal
francois.maillet@sun.com

ABSTRACT

We propose a recommendation technique that works by collecting text descriptions of the items that we want to recommend and then using this *textual aura* to compute the similarity between items using techniques drawn from information retrieval. We show how this representation can be used to explain the similarities between items using terms from the textual aura and further how it can be used to steer the recommender. We'll describe a system that demonstrates these techniques and we'll detail some preliminary experiments aimed at evaluating the quality of the recommendations and the effectiveness of the explanations of item similarity.

Categories and Subject Descriptors

H.3.3 [**Information Search And Retrieval**]: Information Filtering

General Terms

Steerable recommender, Explainable recommender

Keywords

Steerable recommender, Explainable recommender

1. INTRODUCTION

One of the problems faced by current recommender systems is explaining why a particular item was recommended for a particular user. Tintarev and Masthoff [12] provide a good overview of why it is desirable to provide explanations in a recommender system. Among other things, they point out that good explanations can help inspire trust in a recommender, increase user satisfaction, and make it easier for users to find what they want. Vig *et al.* make a distinction between two kinds of explanations: *descriptions*, which reveal the actual mechanism used to make a recommendation and *justifications*, which use a conceptual model that may differ significantly from the underlying model used to generate the recommendations. Using descriptions rather than justifications will usually result in a system that is more *transparent* to the users, but this requires that the user be able to understand how the underlying model of the recommender works.

Along with the need for good explanations, we want to consider how users can influence the recommendations that a system is generating. In current recommender systems, if users are unsatisfied with recommendations, often their only way to change how recommendations are generated in the future is to provide thumbs-up or thumbs-down ratings to the system. Unfortunately, it is not usually apparent to the user how these gestures will affect future recommendations, since there is typically no immediate feedback. Tintarev and Masthoff [12] term this as a lack of *scrutability* in a recommender system.

In addition to all of this, many current recommender systems simply present a static list of recommended items in response to a user viewing a particular item. Our aim is to move towards exploratory interfaces that use recommendation techniques to help users find new items that they might like. This is an attempt to solve the *Find Good Items* task described by Herlocker *et al.* [7] in such a way that the exploration of the space is part of the user experience.

In this paper, we'll describe a system that builds a *textual aura* for items and then uses that aura to recommend similar items using textual similarity metrics taken from text information retrieval. In addition to providing a useful similarity metric, the textual aura can be used to explain to users why particular items were recommended based on the relationships between textual auras of similar items. The aura can also be used to *steer* the recommender, allowing users to explore how changes in the textual aura create different sets of recommended items.

We'll offer a brief description of a recommender system built around the idea of the textual aura and describe some initial evaluations that we have made to test the effectiveness of these recommendations.

Recommender Systems 2009, New York City

2. RELATED WORK

Tintarev and Masthoff's [12] survey of explanations in recommender systems provides an excellent description of the aims of explanations and what makes a good explanation. We will adopt their terminology wherever possible. Tintarev [11] provides a prototype explanation system for a movie recommender that is based on ratings and user-specified metadata. Zanardi and Capra [16] exploit tags and recommender system techniques to provide a social ranking algorithm that can be used for items that have been tagged. Their approach uses straight tag frequency, rather than more well-accepted term weighting measures. Nakamoto *et al.* [10] use tags applied to del.icio.us bookmarks to generate a representation for the Web sites pointed to by the URLs. They do EM clustering on the tagged objects in order to generate a number of topic clusters and then use the tags to provide explanations for the clusters.

Vig *et al.* [14] use social tags to generate descriptions of the recommendations generated by a collaborative filtering recommender. Although they explicitly forgo "keyword-style" approaches to generating recommendations, we believe that it is worthwhile to try using techniques that are known to provide good results in the traditional information retrieval space.

Wetzker *et al.* [15] use Probabilistic Latent Semantic Analysis (PLSA) to combine social tags and attention data in a hybrid recommender. While PLSA is a standard information retrieval technique, the dimensionality reduction that it generates leads to a representation for items that is difficult to use for explanations.

3. THE TEXTUAL AURA

The key aspect of our approach to generating transparent and steerable recommendations is to use an item representation that consists of text about the item. For some item types like books or blog posts, this could include the actual text of the item, but for most items, the representation will mainly consist of text crawled from the Web. In our representation, each item in the system is considered to be a document. The "text" of the document is simply the conglomeration of whatever text we can find that pertains to the item. Because we may collect text about the item from a number of sources, we divide the item's document into fields, each field representing the text from a particular source.

The representation for an item is then a variant of the standard vector space representation used in document retrieval systems. An item is represented by a vector of length N, where N is the number of unique terms. Each component of this vector is a *term weight* that is meant to measure the importance of that term for the particular document.

We use one of the predominant term weighting measures, $tf \cdot idf$, where the weight of a term t in document d is calculated as follows:

$$w_{t,d} = \log(f_{t,d}) * \log(\frac{D}{f_t})$$

where $f_{t,d}$ is the frequency of term t in document d, f_t is the number of documents in which term t occurs, and D is the number of documents in the collection. In a nutshell, a term is considered to be important for a document if it occurs frequently in that document (the "term frequency") and infrequently in other documents in the collection (the "inverse document frequency").

The similarity between two documents is computed by taking the cosine of the angle between the two vectors representing the documents in this N dimensional space. Typically this is computed as the dot product of the length-normalized vectors for the documents:

$$\text{sim}(a, b) = \frac{a \cdot b}{||a||||b||}$$

Using length-normalized vectors for the documents means that longer documents (*i.e.*, items with a larger textual aura) do not always dominate shorter documents when computing document similarity.

The representation that we use for items is slightly more complicated than this: the system keeps track of separate vectors for each of the fields that make up the document so that we can calculate document similarity on any single field or any combination of fields. As well as building composite vectors for a single document, this representation also makes it very simple to build composite vectors for a number of items or to run clustering algorithms on the vectors to segment the interests of a user or group of users.

There are a number of term weighting and similarity schemes that have been proposed in the information retrieval literature. For example, Vig *et al.* [14] use Pearson's correlation coefficient for computing tag relevance, which is essentially a term weighting. Zanardi and Capra [16] also use Pearson's coefficient for a similar purpose. Zobel and Moffat [17] provide a comprehensive list of term weighting and similarity functions, and an analysis of the comparative performance using standard information retrieval evaluation collections.

One advantage of this vector space representation is that there are well known techniques for dealing with a number of the phenomena that are typically encountered when dealing with tags. For example, the issues of tag quality and tag redundancy encountered by Vig *et al.* [14] are usually handled quite well using the standard vector space model for document retrieval. The vector space model with $tf \cdot idf$ term weights handles high frequency terms easily. For example, at a social music site like Last.fm [2], virtually every artist has been tagged many times with the tag *rock*. If we were simply considering the frequency of tags, then *rock* appears to be an important tag for almost every artist, when in fact the tag offers very little information about an artist. Because *rock* has a high document frequency, it will tend to have a small inverse document frequency and will therefore tend to have a small term weight for any given artist.

Another advantage of the vector space model is that there are also well-known techniques for building and scaling systems that use such a representation. This means that we can build a system where the representations of items can be continually updated and the similarity of items can be computed on-demand, relieving the system of the need for periodic updates of an item-item similarity matrix.

Once one has made the jump to treating items as documents made up of the words in the textual aura, it becomes easy to do other things like treat individual social tags as documents whose words are the items to which the social tag has been applied. Thus a social tag is simply a document in a different A dimensional space (where A is the number of artists), and we can compute tag similarities in exactly the same way that we compute item similarities. Similarly, a user can be modeled as a document whose words are the social tags that the user has applied to items, allowing us to

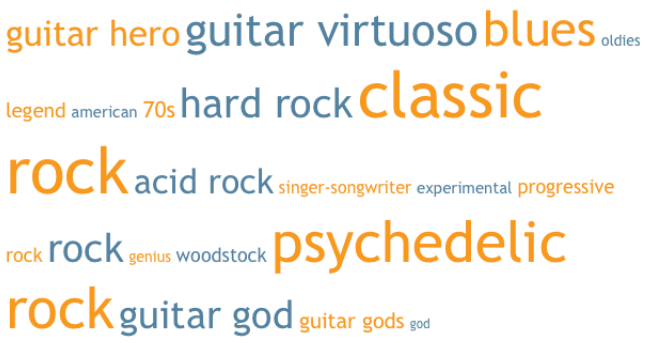

Figure 1: A portion of the textual aura for Jimi Hendrix

compute user-user similarity (as well as user-item similarity) in the same way. Indeed, we can treat the consumption of an item as a "word" in a field for the item's document and do something very much akin to collaborative filtering based on the item-item similarities for that field.

3.1 An Aura for Musical Artists

Our current work has focused on recommending musical artists. The data sources that we use to represent a musical artist include social tags that have been applied to the artist at Last.fm and the Wikipedia entry for the artist. Figure 1 shows a portion of the textual aura for Jimi Hendrix derived from the tags applied to him at Last.fm as a tag cloud. Unlike typical tag clouds, where the size of a tag in the cloud is proportional to the frequency of occurrence of the tag, here the size of a tag is proportional to it's term weight calculated using the scheme described above.

By tying the size of a tag in the cloud to the term weight, we are giving the user an indication of what kind of artists we will find when we look for similar artists. In this case, we will tend to find *guitar virtuoso*, *blues* influenced *psychedelic rock*, as these are the largest tags and therefore the tags with the most influence over the similarity calculation.

In addition to the basic information that we put into an item's aura, we have experimented with adding derived information. For example, when we crawl an artist's biography from Wikipedia, we index and store the biography as-is, but we also separately index and store only those words that occur in the biography that are also social tags that have been applied to the artist (we call these the *bio tags* for the artist).

We are not currently indexing reviews of the artists' work or Web pages or blog posts relating to the artist, but that is a straightforward addition to our data set and one that we are exploring. In addition, we have been working on automatically tagging audio files with social tags using machine learning techniques [6], but we have not incorporated this information into the textual aura for the items yet. We also need to consider the inclusion of tags generated by methods like the *Listen Game* [13].

3.2 Generating Recommendations

We can use the textual aura in a number of recommendation scenarios. We can find similar artists based on a seed artist's aura, or we can find artists that a user might like based on tags that they have applied or based on the combined aura of their favorite artists. We allow users to save tag clouds and use them to generate recommendations over time. All of these recommendations can be done in the presence of filters that will modify the set of recommended items.

Since the implementations of these recommendation scenarios are based on similar techniques, we'll illustrate the general approach by explaining the specific case of finding artists that are similar to a seed artist. First, we retrieve the document vector for the seed item. This may be a vector based on a single field of the item's document or a composite vector that incorporates information for a number of fields, possibly weighting the fields based on their importance. Depending on the circumstances, we may restrict the vector for the seed item to a proportion of the highest weighted terms, but in most instances we use all of the terms.

In any case, for each of the terms that occurs in the vector for the seed item, we retrieve the list of items that have that term in their textual aura. As each list is processed, we accumulate a per-item score that measures the similarity to the seed item. This gives us the set of items that have a non-zero similarity to the seed item. Using a typical text search engine for this operation means that it can be performed in well less than a second even for databases of millions of items, each of which may be described by hundreds or thousands of words.

Once this set of similar items has been built, we can select the top n most-similar items. This selection may be done in the presence of item filters. These filters can be used to impose restrictions on the set of items that will ultimately be recommended to a user.

For example, we can derive a popularity metric from Last.fm's listener counts and use this metric to divide the set of artists into *popular*, *mainstream*, *hipster*, or *rarities*. We can then use this division to filter the recommendations so that a user will only see recommendations for artists that everyone else is listening to (*popular*) or that hardly anyone else is listening to (*rarities*).

3.3 Explanations and Transparency

Using the textual aura of an item as its representation means that we can show the representation of an item to a user in a format that they can understand: a tag cloud like the one in Figure 1 where larger tags are more important. In our case, the importance is proportional to the weights associated with the terms in an artist's aura, rather than being proportional to the frequency of occurrence of the term, as is typical in tag clouds.

Given the auras for two items, and the cosine similarity measure discussed above, it is straightforward to determine how each of the terms in the textual aura for a seed item contributes to the similarity between the seed item and a recommended item. Given two items A and B, and the document vectors used to compute their similarity, we can build three new tag clouds that describe both how the items are related and how they differ.

We can build an *overlap* tag cloud from the terms that occur in the vectors for both items. The size of the tags in the cloud will be sized according to the proportion of the similarity between A and B that the term generates. Given a term t that occurs in both A and B, this proportion is:

$$\frac{w_{t,A} \cdot w_{t,B}}{\text{sim}(A, B)}$$

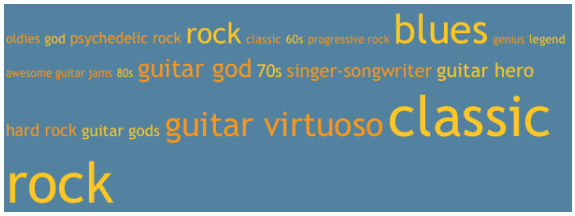

Figure 2: Overlap tag cloud for Jimi Hendrix and Eric Clapton

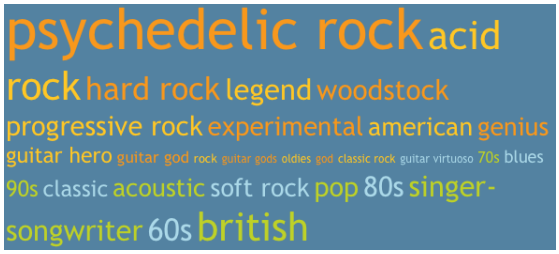

Figure 3: Difference tag cloud for Jimi Hendrix and Eric Clapton

As with the initial recommendation, we select the top n terms that are shared between the seed and recommended items and use those as the overlap tag cloud.

Figure 2 shows a representation of an overlap tag cloud for seed artist Jimi Hendrix and recommended artist Eric Clapton. The overlap cloud clearly shows that Clapton was recommended because he is a *classic rock guitar virtuoso* who plays the *blues*. The overlap cloud for the recommended artist The Doors shows that while they are also described as *blues*-playing *classic rock*, there is a much bigger influence from *psychedelic rock*, which is missing from the Clapton overlap cloud.

This explanation of the similarity between two items is directly related to the mechanism by which the similarity was computed. In the terms used by Vig *et al.* [14], this is a description of the recommendation not a justification. In the terms used by Tintarev and Masthoff [12], these explanations make the system transparent: the user can use the explanation we provide to understand how and why a particular item was recommended.

In addition to the overlap cloud, we can generate a *difference* cloud that shows the terms in A and B that are not in the set of overlapping terms. Because these sets of terms are disjoint, the size of the terms in these clouds is proportional to the term weights in the clouds for A or B. The difference cloud can be used to indicate to a user how a recommended item differs from the seed item, which may lead them into an attempt to steer the recommendation towards a particular aspect.

For example, the difference cloud between Jimi Hendrix and Eric Clapton shown in Figure 3 shows us that Hendrix has *psychedelic rock* as a very important tag, while this tag is missing for Clapton. It also shows us that *80s* and *90s* are important tags for Clapton and missing for Hendrix, indicating that Clapton likely produced new music during those decades.

Having these sets of terms provide an explanation for a recommendation requires that the user understand what a particular tag means. For example, a tag like *death metal* has a particular meaning to a certain community. Because we treat tags as items, we can easily provide links to information related to a tag. In particular, we can provide a set of related tags by performing a tag similarity computation and displaying the most-similar tags as a tag cloud like the one shown in Figure 4. We can also easily provide a small set of items that have the tag in their textual aura as an important tag.

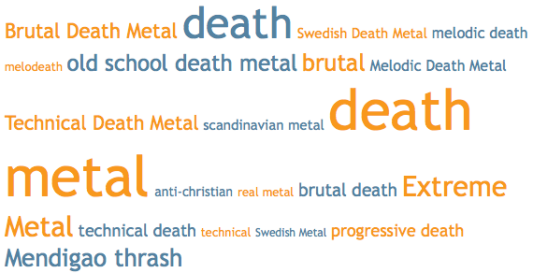

Figure 4: Tag similarity cloud for *death metal*

3.4 Steerable Recommendations

In addition to providing a straightforward way of generating explanations of the recommendations made, our approach offers an equally straightforward way of *steering* the recommendations. As we have said, the recommendations generated from a seed item are based on the document vectors in a high dimensional space, which can be presented to the user as a tag cloud.

We allow the user to directly manipulate the document vector associated with an item and use their manipulations to interactively generate new sets of recommended items on the fly. If the vector for an item is presented as a tag cloud, then the user can manipulate this representation by dragging tags so that they increase or decrease in size.

For example, let's say that a user starts with the tag cloud for Jimi Hendrix shown in Figure 1. If the user enlarges the tag *psychedelic rock*, increasing its importance, then the set of recommended artists changes to include artists like Steppenwolf and Iron Butterfly, for which this tag is much more important. Alternatively, the user can decrease the importance of that tag and increase the importance of the *guitar virtuoso* tag to find artists that are more like Hendrix along this dimension. This leads to artists like Robin Trower and Stevie Ray Vaughn being recommended as more similar.

As with any other recommendation, we can provide an explanation like the ones described above that detail why a particular item was recommended. While the user is interacting with the item representation in this way, it is crucial to provide them with an easy way to add new tags to the cloud as they discover new facets that they find intriguing. In our current system, when we display the information for a recommended artist we provide a way for the user to add any of that artist's tags to the tag cloud currently being used to steer the recommendations. We also offer a hinted search for tags that the user might like to add.

The user can also make particular tags *sticky* or *negative*. A sticky tag is one that must occur in the textual aura of a recommended artist. A sticky tag will still contribute some portion of the similarity score, since it appears in both the

seed item and in the recommended item. A negative tag is one that must not occur in the textual aura of a recommended artist. Negative tags are used as a filter to remove artists whose aura contains the tag from the results of the similarity computation.

The overall effect of the steerability provided to the users is that they can begin to answer questions like "Can I find an artist like Britney Spears but with an indie vibe?" or "Can I find an artist like The Beatles, but recording in this decade?"

In Tintarev and Masthoff's [12] terms, steerable recommendations make the system *scrutable* for the users. The steerability provides a way for users to modify the recommendations to correct the system, and makes the explanation part of the cycle that they describe. This steerability can also make the use of the system fun, providing *satisfaction* and, in our experience, usefulness.

Providing steerable recommendations means that we are moving quickly away from static lists of items displayed with a seed item and into an exploration of the space of items. We believe that these represent distinctly different activities that users will switch between depending on their needs, in a way similar to the difference between search and browse behaviors that present in traditional search engines (see [9] for more on this distinction).

4. AN IMPLEMENTATION OF STEERABLE RECOMMENDATIONS

As part of the AURA project in Sun Labs, we have developed the Music Explaura [3], an interface for finding new musical artists that incorporates all of the techniques described above. Users can start out by searching for a known artist, by searching for an artist with a particular tag in their aura, or by searching for a tag and seeing similar tags. Figure 5 shows the artist page for Jimi Hendrix.

The aim here is to provide as much context as possible for the user. In addition to the tag cloud for the artist (based on Last.fm tags by default), we display a portion of the artist's biography from Wikipedia, videos crawled from YouTube, photos crawled from Flickr, albums crawled from Amazon, and events crawled from Upcoming. In the left-hand column, the top 15 most-similar artists to the seed artist are displayed.

For each of the similar artists we display a few of the top terms from their textual aura and their popularity (also based on data from Last.fm.) The user can play music for the seed artist or for any of the recommended artists. There are links to the overlap and difference tag clouds for each of the recommended artists, and at any time the user can begin steering the recommendations by clicking one of the steering icons.

The back-end of the system is a distributed data store that is designed to support these kinds of recommendation activities. Items are distributed to nodes in the data store based on a hash of their primary key. Each of the nodes in the data store has a key-value store that is used for storing and quickly retrieving the item data as well as the attention data (*e.g.*, User u listened to track t). The nodes also have a search index that indexes and stores the textual aura for the items. The search index supports querying operations as well as the textual similarity and filtering operations that drive the recommendations.

System	Average Rating	Relative Precision	Novelty
Sun aura	4.02	0.49	0.31
CF-10M	3.68	-0.02	0.24
CF-2M	3.50	-0.16	0.28
Sun-CF-12K	3.48	-0.38	0.26
CF-1M	3.26	-0.47	0.23
CF-100K	2.59	-1.01	0.32
Expert	2.06	-1.17	0.58
Music Critic	2.00	-1.26	0.49
CF-1M	1.82	-1.18	0.38
Hyb-1M	0.89	- 3.31	0.57
CF-10K	0.82	-4.32	0.47
CF-10K	-2.39	-13.18	0.48

Table 1: Survey Results. CF-X represents a collaborative filtering-based system with an estimated number of users X

The data store supporting the application shown in Figure 5 has 16 nodes and has information for about 30,000 musical artists, amounting to approximately 6.5GB of data in total. This data store is capable of supporting 14,000 concurrent users doing typical recommendation tasks (finding similar items, fetching item data, adding attention data) with an average response time at the client of less than 500ms. The code for the data store is available as open source from The AURA Project [1].

5. EXPLORATORY EVALUATIONS

5.1 Recommendation Quality

To get a better understanding of how well the aura-based recommendations perform, we conducted a web-based user survey that allowed us to compare the user reactions to recommendations generated by a number of different recommenders. We compared two of our research algorithms - our aura-based recommender and a more traditional collaborative filtering recommender to nine commercial recommenders. The aura-based system used a data set [8] consisting of about 7 million tags (with 100,000 unique tags) that had been applied to 21,000 artists. The CF-based system generated item-item recommendations based on the listening habits of 12,000 Last.fm listeners. The nine commercial recommenders evaluated consist of seven CF-based recommenders, one expert-based recommender and one hybrid (combining CF with content-based recommendation). We also included the recommendations of five professional critics from the music review site Pitchfork [5].

To evaluate the recommenders we chose the simple recommendation task of finding artists that are similar to a single seed artist. This was the only recommendation scenario that was supported by all recommenders in the survey. We chose five seed artists: The Beatles, Emerson Lake and Palmer, Deerhoof, Miles Davis and Arcade Fire. For each recommender in the study, we retrieved the top eight most similar artists. We then conducted a Web-based survey to rank the quality of each recommended artist. The survey asked each participant in the survey to indicate how well a given recommended artist answers the question "If you like the seed artist you might like X." Participants could answer "Excel-

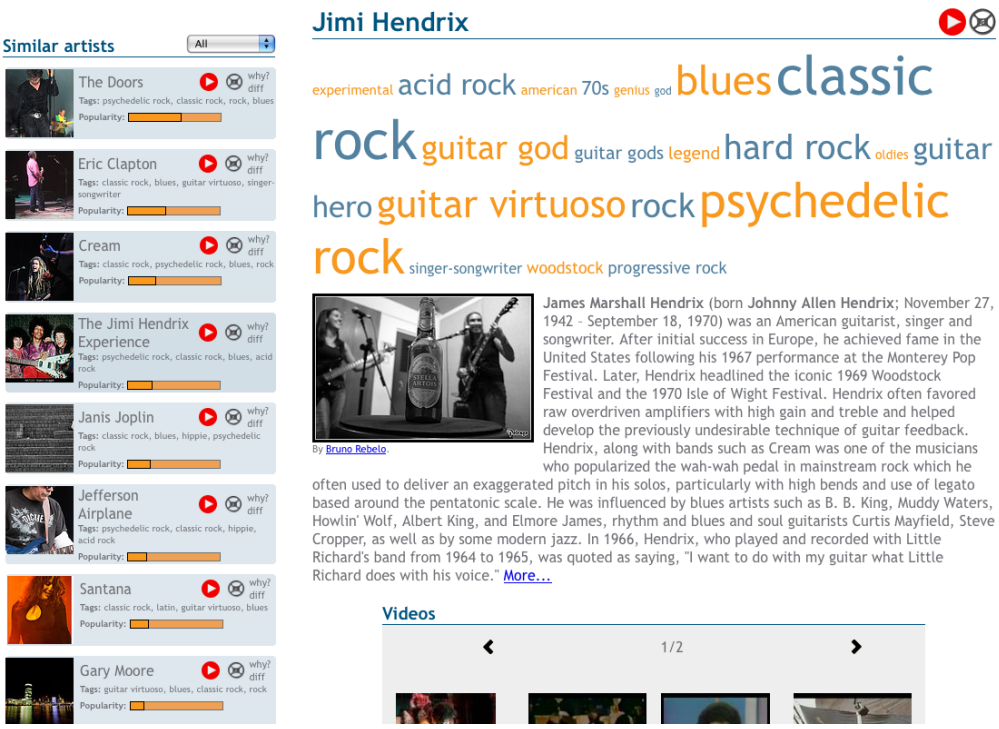

Figure 5: Explaura interface for Jimi Hendrix

lent" "Good" "Don't Know" "Fair" or "Poor". Two hundred individuals participated in the survey, contributing a total of over ten thousand recommendation rankings.

We used the survey rankings to calculate three scores for each recommender:

Average Rating The average score for all recommendations based on the point assignment of 5, 1, 0, -1 and -5 respectively for each Excellent, Good, Don't Know, Fair and Poor rating. This score provides a ranking of the overall quality of the recommendations.

Relative Precision The average score for all recommendations based on the point assignment of 1, 1, 0, -1 and -25 respectively for each Excellent, Good, Don't Know, Fair and Poor rating. This score provides a ranking for a recommender's ability to reject poor recommendations, which is an important characteristic of a recommender that is used for tasks such as playlist generation.

Novelty The fraction of recommendations that are unique to a recommender. For instance, a value of 0.31 indicates that 31 percent of recommendations were unique to the particular recommender. Recommenders with low novelty scores tend to give predictable recommendations that are not useful for music discovery.

Table 1 shows the results of the survey. Some observations about the results: CF-based systems with larger numbers of users tend to have higher average ratings and relative precision. Somewhat surprisingly, human-based recommenders (Expert and Music Critic) do not rate as well as larger CF-based recommenders, but do tend give much more novel recommendations. Poorly rated recommenders tend to have a higher novelty score. The aura-based recommender provides good average rating and relative precision results while still providing somewhat novel recommendations.

It is important not to draw too many conclusions from this exploratory evaluation. The recommendation task was a simple one using popular artists, the number of participants was relatively small and the participants were self-selected. Nevertheless, the survey does confirm that the aura-based approach to recommendation is a viable approach yielding results that are competitive with current commercial systems.

5.2 The Music Explaura

In order to gain preliminary insight into the effectiveness of the textual aura as a basis for generating good recommendations, explaining recommendations, and affording users direct access to the steering of recommendations, we conducted a small scale qualitative usability evaluation using a cross-section of potential end-users.

In this pilot study, we wanted to evaluate how much users like and agree with the recommendations and whether they feel that the interface provides a more effective means by which to engage with and actively explore recommendations.

5.2.1 Participant Interviews

We asked users to evaluate the Music Explaura's recommendations with respect to the familiarity and accuracy of the recommendations, their satisfaction with the recommendations, how novel the recommended artists were, the transparency and trustworthiness of the recommender, and the steerability of the recommendations.

Ten participants interested in music and with at least some music recommender familiarity, representing a cross

section of listener types were recruited from the student population of Bentley University and musician forums on craigslist. Participants were all web-savvy, regular Internet users. One-on-one, in-depth UI evaluations were conducted by three facilitators.

Interviews began with each participant briefly discussing their current methods for seeking out and discovering new music as well as their familiarity with music recommender systems, including Amazon, iTunes, Last.fm, and Pandora [4]. They were encouraged to follow a talk-aloud protocol as they explored music related to a seed artist of their own choosing using a sequence of 3 different recommenders. All participants evaluated Last.fm and the Explaura. Nine of the ten evaluated Pandora and one of the ten evaluated Amazon.

For each site they evaluated, participants were asked about their initial impressions of the recommendations generated for the artist they selected. They were asked to rate how well they liked the site's results, and to discuss whether the recommendations were relevant and interesting. They were also asked whether the recommendations made sense to them and how much they felt that they understood why the system recommended those particular artists. Finally, they were asked to discuss how they would explore beyond the initial recommendations.

The interviews concluded with summary questions regarding the participants' overall evaluation of the Explaura versus other recommenders they had used, their intent to return to the site, and their expectation that they would use the recommender in the future. Lastly, they were asked to rate the interface's ease of use and to provide an overall rating of its quality as a solution that met their needs.

5.2.2 Quality of Recommendations

The Explaura interface presents a number of interaction paradigms very different from familiar interaction conventions, creating a pronounced learning curve for new users. In order to support users in focusing on evaluating the relative merits of the exploration options, the facilitators were more actively involved in demonstrating and explaining the Explaura's features and functionality than they were in guiding users through comparison sites. This may have altered users perceptions of facilitator interest or involvement in this interface versus comparison sites, motivating more positive responses.

As many users commented, the evaluation of a recommender takes some time. Users expect, in particular with recommenders that are more exploratory in nature, to spend a fair amount of time interacting with and probing these sites before coming to a reliable conclusion about their efficacy and value. Given the time constraints of the research, we were unable to gauge how users impressions might have evolved with more opportunity to engage in a process of discovery.

Given these confounding factors, we cannot know for sure how accurately participants were able to isolate relevant recommender attributes in answering questions related to different dimensions of satisfaction. We therefore focused our analysis and discussion of the interviews on participants' revealing comments and qualitative observations, which we find to be the most reliable reflections of their reactions.

Users' ratings of different sites' recommendations appeared to largely correspond to how much they agreed with the recommendation of artists that were familiar to them. Included in this assessment of the quality of the recommendations was often an assessment of rank ("I don't know if I agree with this ranking.")

We asked users to comment on how many of the recommended artists were new to them. Because each participant explored the interfaces using a personally selected seed artist, the ratio of new to unfamiliar artists among initial recommendations on each site varied. Where the sites produced appreciably different recommendations, most users did find that Explaura's recommendations were the more varied and unfamiliar. Reaction to higher rates of unfamiliarity in Explaura's recommendation list was mixed. While some showed a strong admiration for higher rates of unfamiliar recommendation, others stated that they were only looking for familiar artists.

While the preference for familiar or novel artists in the initial recommendations list varied considerably, virtually all participants expressed appreciation for the drop-down filter which allowed them to set the level of popularity of recommendation in the results list. Users affirmed that the desire to explore artists more well known or obscure will vary by circumstance and expressed strong positive reactions to this unique option.

5.2.3 Explanations

The three recommenders provided three different levels of transparency. Last.fm offers no explanations, Pandora provides complex, natural language descriptions of similarities between recommended tracks, and the Explaura presents the similarities between artists in the form of tag clouds.

Most Last.fm users initially maintained that they did understand why artists were recommended. In order to get at a more accurate assessment of the system's transparency, a follow up question was posed which encouraged participants to consider the similarity scale rating on the Last.fm similar artists page, and to have them explain what that similarity scale might mean and on what it might be based. Here, some users hesitated and agreed that they did not know, but many still ventured a guess as to what it probably meant. Users of Pandora often said that they would never look at explanations on their own.

Ten of ten users agreed, though not always immediately (some required prompting to investigate the similarity tag clouds) that they understood why Explaura recommended the items it did, and that the list of recommendations made sense to them.

When asked to describe the relative value of the explanations provided by Pandora and the Explaura, some users suggested that they preferred the tag cloud's accessibility ("You don't have to have a masters in music,"). Other listener types, typically enthusiasts and savants, suggested that Pandora's explanations were inconsistent: sometimes interesting and insightful and sometimes too vague. Within the small scope of this qualitative research, no clear preference could be discerned for explanation type.

It was clear that the value of the tag clouds as an explanation is highly vulnerable to any perception of inaccuracy or redundancy in the tag cloud. Also, sparse tag clouds produce more confusion than understanding. One user commented, "I would suggest that the clouds should have more than two words. This doesn't mean anything." This points to a need to provide a certain number of terms for all of the

artists, which may require autotagging artists.

Almost none of the participants immediately grasped the meaning of the tag clouds. The general first impression is of confusion. Furthermore, no one expected to be able to manipulate the cloud. The mechanism of interaction is highly complex (tags can be made sticky, negative or deleted, tags can change size), which leads to a steep learning curve for new users.

Despite these problems, the users continually expressed the desire to limit and redefine the scope of an exploration. When the concept is presented, virtually all users express surprise, interest, and pleasure at the idea that they can do something with the results. However, after experimenting with tag cloud manipulation, positive reactions of the promise of the concept shift almost invariably to comments regarding qualified interest, for example, "I like it, but not the way it is now." or "The tag cloud needs to change."

6. FUTURE WORK

The current system uses the traditional bag-of-words model for the tags. While this has provided some worthwhile results, it seems clear that we should be clustering terms like *canada* and *canadian* so that their influence can be combined when generating recommendations and explanations. At the very least, combining such terms should lead to less confusion in users.

We're interested in generating more language-like descriptions of the similarities and differences between items. It seems like it should be possible to use the term weightings along with language resources like WordNet to generate Pandora-like descriptions of the similarity tag clouds, which may make them more approachable for new users.

Our ultimate aim is to provide hybrid recommendations that include the influence of the textual aura as well as that of collaborative filtering approaches. An obvious problem here is how to decide which approach should have more influence for any given user or item.

7. CONCLUSIONS

The textual aura provides a simple representation for items that can produce novel recommendations while providing a clear path to a Web-scale recommender system. The initial evaluation of the quality of the recommendations provided by an aura-based recommender provides strong evidence that the technique has merit, if we can solve some of the user interface problems.

Our initial usability study for the Music Explaura showed that the tag cloud representation for the artist can be confusing at first viewing. While the users expressed a real desire to explore the recommendation space via interaction with the system, the current execution needs further usability testing and interface design refinement to enhance user acceptance of the model.

Acknowledgments

Thanks to Professor Terry Skelton of Bentley University.

8. ADDITIONAL AUTHORS

Susanna Kirk (Bentley University, `kirk_susa@bentley.edu`), Jessica Holt (Bentley University, `holt_jess@bentley.edu`), Jackie Bourque (Bentley University, `bourque_jacq@bentley.edu`), Xiao-Wen Mak (Bentley University, `mak_xiao@bentley.edu`)

9. REFERENCES

[1] The AURA Project. `http://kenai.com/projects/aura`.

[2] Last.fm. `http://last.fm`.

[3] The Music Explaura. `http://music.tastekeeper.com`.

[4] Pandora. `http://www.pandora.com`.

[5] Pitchfork magazine. `http://kenai.com/projects/aura`.

[6] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 385–392. MIT Press, Cambridge, MA, 2008.

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

[8] P. Lamere. Last.fm artist tags 2007 data set. `http://tinyurl.com/6ry8ph`.

[9] G. Marchionini. Exploratory search: from finding to understanding. *Commun. ACM*, 49(4):41–46, 2006.

[10] R. Y. Nakamoto, S. Nakajima, J. Miyazaki, S. Uemura, H. Kato, and Y. Inagaki. Reasonable tag-based collaborative filtering for social tagging systems. In *WICOW '08: Proceeding of the 2nd ACM workshop on Information credibility on the web*, pages 11–18, New York, NY, USA, 2008. ACM.

[11] N. Tintarev. Explanations of recommendations. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 203–206, New York, NY, USA, 2007. ACM.

[12] N. Tintarev and J. Masthoff. A survey of explanations in recommender systems. In *Data Engineering Workshop, 2007 IEEE 23rd International Conference on*, pages 801–810, April 2007.

[13] D. Turnbull, R. Liu, L. Barrington, and G. Lanckriet. A game-based approach for collecting semantic annotations of music. In *Proceedings of the 8th International Conference on Music Information Retrieval*, pages 535–538, September 2007.

[14] J. Vig, S. Sen, and J. Riedl. Tagsplanations: explaining recommendations using tags. In *IUI '09: Proceedingsc of the 13th international conference on Intelligent user interfaces*, pages 47–56, New York, NY, USA, 2008. ACM.

[15] R. Wetzker, W. Umbrath, and A. Said. A hybrid approach to item recommendation in folksonomies. In *ESAIR '09: Proceedings of the WSDM '09 Workshop on Exploiting Semantic Annotations in Information Retrieval*, pages 25–29, New York, NY, USA, 2009. ACM.

[16] V. Zanardi and L. Capra. Social ranking: uncovering relevant content using tag-based recommender systems. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 51–58, New York, NY, USA, 2008. ACM.

[17] J. Zobel and A. Moffat. Exploring the similarity space. *SIGIR Forum*, 32(1):18–34, 1998.