I'll know it when I see it: Toward cognitively plausible recommendations

Susan L. Epstein and Eric Osisek
Hunter College and The Graduate Center of The City University of New York, New York, NY
susan.epstein@hunter.cuny.edu, eosisek@gc.cuny.edu

Abstract

This paper introduces *GURU*, a system that informatively and flexibly helps people find an item they are likely to enjoy. Our thesis is that a recommendation system can be cognitively plausible and collaborate with its users in real time. Like people, GURU retrieves supplementary data and considers similarity along multiple dimensions. Its representation is analogous to the ways that people recognize similarity, with overlapping sets of items and their prototypes. This paper provides empirical evidence from human subjects that GURU's approach to recommendation is both reasonable and effective for a large dataset of movies.

1 Introduction

A *recommender* (computer-based recommendation system) offers items intended to satisfy its human users. People, however, often find recommenders' suggestions trivial, arcane, or irrelevant (e.g., [Kille and Abel, 2015]). Human preferences are nuanced, distinctive, and may be difficult for their owner to express. This "I'll know it when I see it" perspective is particularly prevalent for items, such as movies, where users seek variety. Our thesis is that an effective recommender can be *cognitively plausible* (i.e., appear to reason as a person might) and collaborate with its users in real time. This paper introduces *GURU* (Gather Useful Recommendations for Users), a cognitively plausible movie recommender that interacts with people to find movies they are likely to enjoy.

The task of a recommender is to recognize an item attractive to a particular user. In cognitive psychology, the expectation that a new item will have a property (here, be enjoyed by the user) requires conceptual knowledge that supports inferences about that item. There is no evidence that human conceptual knowledge is expressible as a label or as a simple set of features. Indeed, psychologists report that human perception of similarity is far more complex than a pairwise metric [Tversky, 1977], and neuropsychological theories for the semantics of concepts increasingly incorporate multiple kinds of similarities (e.g., [Mahon and Caramazza, 2009]).

In response, our work adopts Wittgenstein's description of a concept as a set of items with a strong family resemblance [Wittgenstein, 1953]. GURU represents a concept as a *family*, a set of mutually similar items closely related to a prototypical item (its *seed*). Family similarity is calculated with respect to a *similarity rationale*, a metric on a set of features that together convey a particular aspect of an item. Here, for example, GURU generates separate sets of movie families under each of three similarity rationales: how much people enjoy movies (*ratings*), how they summarize them (*text*), and how they create them (*provenance*).

Given a similarity rationale, GURU creates overlapping families of items that resemble one another from the perspective of that rationale. Items in a family need only have feature values similar enough to one another's to produce a coherent, cohesive set that relates to its seed under the similarity rationale. Thus the same item can belong to many different families constructed under the same similarity rationale.

GURU uses families to make recommendations and to guide dialogues with the user. Human item choices, however, are often multimodal and diverge considerably from their expressed preferences [Edelman and Shahbazi, 2012]. GURU narrows the challenge of "I'll know it when I see it" by guiding the user to provide one hint: a movie she has enjoyed. Rather than consider prior history, GURU responds to an implicit request of the form "I want an x like y."

The next section of this paper discusses related work. Subsequent sections describe data acquisition, properties of the knowledge base, preprocessing, and concept construction as family generation. Finally, we report on experiments with human subjects and plans for future work.

2 Related Work

Given a set U of users and a set I of items, each represented as a vector of feature values, the task of a recommender is to select a subset of I that a particular $u \in U$ will enjoy. Most recommenders implement this as "predict u's rating for some subset of I and select the items with the highest predicted ratings." In collaboration with u, however, GURU sifts through families of items constructed from I.

The request for "an x like y" is reminiscent of case-based reasoning, where the system seeks similar examples. Movies are a weak-theory domain, one where the relationships among items are predominantly correlational. Most work on such domains seeks to partition I, and expects x to match particular features of the exemplars that represent its categories

(e.g., [Porter *et al.*, 1990]), the way genre-based suggestions do. The resultant categories seek to avoid *polymorphy* (unexplained variability). In contrast, GURU does not partition I, and tolerates polymorphy in the service of variety.

We emphasize that, although a family is a subset of items similar to one another under a specified metric, it is not a traditional cluster [Berkhin, 2006]. Families do not partition *I*, nor are they hierarchical. Although a family focuses on its seed, it typically overlaps with other families that may not even contain that seed. In this sense, families are more closely related to psychologists' categories, where similarity is essential and a prototype is a salient stimulus (e.g., [Goldstein, 1994; Rosch and Mervis, 1975]). Section 5 identifies criteria for family quality.

Traditional recommenders are content-based or do collaborative filtering. A *content-based* recommender identifies items in *I* most similar to (but distinct from) the user's previous choices. Common pairwise similarity rationales over *I* include cosine, adjusted cosine, and correlation [Sarwar *et al.*, 2001]. In contrast, *collaborative filtering* identifies subsets of users who rate many items similarly, and then recommends the most popular items among those users to the entire subset. For both kinds of recommenders, each new user is a *cold start*, that is, she must describe her preferences for the system to retain. GURU, however, assumes that users' preferences vary from one recommendation session to the next, so that every session is a cold start.

To ameliorate the cold-start problem, recommenders can supplement ratings with information about I and *U*. Let *R* be the $|U| \times |I|$ ratings matrix whose *ui*th entry is user *u*'s rating for item i. *Matrix factorization* decomposes *R* into an upper triangular $|U| \times |A|$ matrix and an $|A| \times |I|$ matrix. The recommender that won the Netflix prize, *BellKor*, corrected its ratings for global and temporal biases and then did collaborative filtering with matrix factorization [Bell and Koren, 2007]. Because GURU assumes a cold start, it treats ratings with BellKor's global corrections but not its temporal ones.

Probabilistic matrix factorization (*PMF*) calculates the probability that a factorization is appropriate given the observed ratings, and searches for the most likely factorization [Salakhutdinov and Mnih, 2008]. Shan and Banerjee extracted topics from text summaries on the IMDb movie site, and then used expectation maximization to optimize a mixture of PMF models that predicted ratings [Blei *et al.*, 2003; Shan and Banerjee, 2010]. Because it requires ratings, PMF is inadequate for cold starts and also for *gray sheep*, users whose preferences are consistent with no user group.

GURU is a *hybrid* recommender, that is, it considers both content and agreement on ratings. On a much smaller scale, a hybrid on a query tree with associated rankings combined user preferences with item features [Das *et al.*, 2013]. A hybrid similar to Shan and Banerjee's cited work recommended papers to researchers based on their collection of items of interest [Charlin *et al.*, 2014].

Another important issue for recommenders is their ability to present a diverse set of items. A user who rejects *Terminator 2*, for example, is unlikely to welcome a recommendation for *Terminator 3*. To address this, one system k-clustered movies based on predicted ratings from a collaborative filtering system, where k movies fit the display screen [Boim *et al.*, 2011]. It recommended each cluster's priority medoid (top-rated item maximally similar to those in its cluster) and would branch to other clusters at the user's request. An extension of that work adapted k for the user with a probabilistic model of her observed behavior [Hasan *et al.*, 2014]. In contrast, GURU supports diversity through its creation of item families under different similarity rationales.

The work discussed thus far makes *one-shot* recommendations, that is, it presents a set of top-ranked items without further interaction. Content-based recommenders rely on user history in part because lengthy questionnaires or dialogues to elicit preferences are unreliable [Holbrook *et al.*, 2003]. Alternatively, a *conversational* recommender conducts an initial dialogue with the user to determine her current preferences. Such a dialogue can assist with both cold starts and gray sheep. To date, however, this approach has been applied only to relatively small datasets with predetermined questions and/or small feature vectors (e.g., [Mahmood and Ricci, 2009; Thompson *et al.*, 2004]). GURU is a conversational recommender that generates appropriate questions to elicit the user's current preferences.

3 Data

When work on GURU began, MovieLens had three publicly available movie datasets (https://movielens.org/); we use those three here. The largest dataset is a superset of the medium one, which is in turn a superset of the smallest. MovieLens data includes integer ratings in [1,5] from anonymized users, and at least one genre for each movie.

For the largest dataset, our web crawler collected supplementary data from five well-known, extensive websites: Rotten Tomatoes, IMDb, Turner Classic Movies, TV Guide, and Wikipedia. It retrieved text synopses (not reviews) and any available feature values for provenance: genres, stars, writers, directors, release year, and MPAA rating. (MPAA ratings are the Motion Pictures Association of America's discrete, ordered set of labels for child-appropriate content: G, PG, PG-13, R, NC-17.) We refer here to these three enhanced datasets as *SmallD, MediumD*, and *LargeD*. LargeD, with 10,681 movies, 71,567 users, and 10,000,054 ratings, provides a broad context within which we refined the ratings and synopses, as follows.

We converted ratings into expected ratings with Koren's method [Koren, 2010]. Let r_{ui} be the rating assigned by user user u to item i, and $\bar{r}$ be the mean of all ratings in the dataset. The *item bias* b_i for item i is the mean difference between $\bar{r}$ and the ratings ρ_i assigned to i by the users who rated it:

$$b_i = \frac{\sum_{\rho_i}(r_{ui} - \bar{r})}{|\rho_i|} \tag{1}$$

The *user bias* b_u for user u is the mean difference between the ratings R_u that u provided and $\bar{r} + b_i$:

$$b_u = \frac{\sum_{R_u}(r_{ui} - (\bar{r} + b_i))}{|R_u|} \tag{2}$$

For user u and movie i, the expected rating $\bar{r} + b_u + b_i$ adjusts for both user and item biases.

For text data, we concatenated all synopses for a single movie into a text document for it, and then preprocessed the resultant 9,244,636-word corpus. We removed all occurrences of 180 stop words, common words that do not contribute to the text's meaning (e.g., "the"). To discourage coincidental similarity between movies that reference the same proper noun (e.g., character names), we removed proper nouns with Stanford's named entity recognizer [Finkel et al., 2005]. Finally, we replaced each word with its stem [Porter, 1980]. The corpus of 10,681 documents then included 81,896 distinct *tokens* (word stems).

4 Item Graph Construction

An *item graph* is a weighted, undirected graph where each node represents a movie and an edge weight represents the similarity between the two nodes it connects. For each enhanced dataset, we built three complete item graphs, one for each of text, provenance, and ratings. Each node has a vector representation appropriate to its graph's rationale. In the LargeD *ratings graph*, for example, a node is is labeled by a vector of length 71,567 (the number of users), whose uth entry is the difference between user u's expected rating and actual rating for that movie, or 0 if u did not rate it. Ratings graph edge weights are pairwise cosine similarities.

In the *provenance graph* a node is labeled by four vectors whose lengths are determined by the number of relevant distinct values in LargeD. Provenance vectors are of the form $\langle g, p, y, x\rangle$ for genre vector g, participants vector p (stars, writers and directors), release year y, and MPAA rating x. Each vector may have multiple non-zero entries, that is, feature values are not unique. Vectors for genre and participants are boolean; vectors for release year and MPAA rating have discrete values. The similarity metric for provenance is

$$.25[T(g_1, g_2) + T(p_1, p_2) + s_y(y_1, y_2) + s_x(x_1, x_2)] \quad (3)$$

where T is the Tanimoto (a.k.a. Jaccard) coefficient

$$s_y(y_1, y_2) = \frac{10 - |y_1 - y_2|}{10} \text{ if } |y_1 - y_2| \leq 10, 0 \text{ otherwise}$$

$$s_x(x_1, x_2) = \begin{cases} 1.0 & \text{if } x_1 = x_2 \\ 0.5 & \text{if } x_1 \text{ and } x_2 \text{ are consecutive} \\ 0 & \text{otherwise} \end{cases}$$

The *term frequency tf(t,d)* of token t in document d is the number of times t occurs in d. In a corpus D of documents, the *tf-idf(t,d)* of t in d is [Zhao and Karypis, 2005]

$$\textit{tf-idf}(t, d) = \textit{tf}(t, d) \cdot log \frac{|D|}{|d' : t \in d'|} \quad (4)$$

For document d, *tf-idf* is high for a token frequent in d but uncommon in the rest of D; *tf-idf* is low for a token common across D or uncommon in d. In the *text graph* a movie is labeled by a vector of length 81,896 (the number of distinct tokens in our corpus). The kth value of that vector is the *tf-idf* of the kth token in the movie's synopsis document d. Text graph edge weights are pairwise cosine similarities.

To focus on stronger similarities in what were originally complete graphs, we eliminated edges whose weights denoted trivial or unreliable similarities. (Thresholds were 0.04 for the text graph and 0.38 for the provenance graph.) In the ratings graph, we dropped edges with negative weights and dropped edges between pairs of movies rated by fewer than 30 of the same users. Finally, in every item graph we pruned any nodes of degree 0. The upper portion of Table 1 describes the resultant item graphs for the 3900 movies in MediumD.

Table 1: The MediumD graphs and their families

Similarity rationale	Text	Provenance	Ratings
Item graph			
Nodes	3718	3839	3437
Edges	659,263	886,370	1,656,337
Density	9.54%	12.04%	28.05%
Average edge weight	0.050	0.431	0.110
Averages for families			
Nodes	54.32	35.48	18.71
Density	98.78%	98.85%	99.89%
Average edge weight	0.079	0.550	0.200

5 Family Construction and Evaluation

To model the conceptual relationships that underlie human similarity judgment, the *FF* (Family Finder) algorithm finds weighted subgraphs in an item graph. FF is an adaptation of an algorithm that partitioned a weighted constraint graph into dense, high-weight subgraphs [Li and Epstein, 2010]. FF uses Variable Neighborhood Search, a non-deterministic metaheuristic [Hansen and Mladenovic, 2003].

Algorithm 1: FF pseudocode to find a family F

Input: graph G, seed s, scoring function sf, greedy function g, swap function wf, termination function Q

$F \leftarrow \{s\}$
$G \leftarrow G - s$
$improved \leftarrow$ false
$iteration \leftarrow 1$
until Q = true
  $candidate \leftarrow$ node n in G that maximizes $g(n, F)$
  $provisional \leftarrow F \cup \{candidate\}$
  if *sf*$(provisional) >$ *sf*(F) /**greedy improvement**/
  then $F \leftarrow provisional$
    $G \leftarrow G - candidate$
    $improved \leftarrow$ true
    $iteration \leftarrow 1$
  else *swap-set* $\leftarrow$ *wf*(F)
    while *swap-set* $\neq \emptyset$ and Q = false
      $candidate \leftarrow$ *pop*(*swap-set*)
      $provisional \leftarrow swap(F, candidate)$
      if *sf*$(provisional) >$ *sf*(F) /**swap improvement**/
      then $F \leftarrow provisional$
        $G \leftarrow adjust(G, candidate)$
        $improved \leftarrow$ true
      else $improved \leftarrow$ false /**shake**/
        $F \leftarrow shake(F, iteration)$
        $iteration \leftarrow iteration + 1$
return F

Algorithm 1 is pseudocode for FF. It builds a family F in graph G from seed node s. On each iteration, FF tries to strengthen and enlarge F under its scoring function *sf*:

$$sf(F) = .6\mu_{EW}(F) + .4\log(\log size(F)) \quad (5)$$

where the log functions place F's mean edge weight μ_{EW} and number of nodes $size(F)$ on about the same scale. All ties are broken at random.

Each iteration tries to improve $sf(F)$ by a greedy addition or a swap. A new node n selected from G need not be adjacent to s in G, but must maximize the greedy function g:

$$g(n, F) = .3SW(n, F) + .6\mu_{IW}(n, F) + .1\mu_{OW}(n, F) \quad (6)$$

where $SW(n, F)$ is the edge weight from n to F's seed s, μ_{IW} is n's mean edge weight to nodes inside F other than s, and μ_{OW} is n's mean edge weight to nodes outside F.

When FF finds no further greedy additions, it tries to *swap*, that is, to replace some node w in F with one or two nodes not in F. The swap function *wf* identifies $w \neq s$ with degree $|F| - 2$ or $|F| - 3$ in F, along with one or two replacements for w with degree ≥ 2 in F. If two nodes replace w, they must also be adjacent to one another. When a swap improves F's score, *adjust* returns w to G and removes the replacements.

When neither greedy additions nor swaps improve F's score, FF *shakes* (removes) nodes from F at random and returns them to G. Initially FF shakes out one node; that number increases each time additions and swaps do not improve F's score, and FF tries additions and swaps again. To halt FF, the termination function Q limits search time, number of iterations, and the maximum number of nodes to shake out.

Equation (6) emphasizes mutual similarity among non-seeds in F more strongly than similarity to s. This is because preliminary work indicated the presence of *magnets*, nodes that draw their large coterie of similar movies into F on subsequent greedy additions. In the examples of Figure 1, seeds are gray circles. The left graph is an acceptable family, but in the right graph a magnet (white circle) overshadows the seed. Without the constants in (6), magnets often overwhelmed families from dissimilar seeds until the families became nearly identical. Equation (6) downplays μ_{OW}, a measure of n's ability to support future family growth when it draws its neighbors along high-weight edges into F on subsequent iterations. If a magnet does join F, the *SW* term in (6) discourages the subsequent inclusion of neighbors of the magnet that are less similar to s than the magnet was.

Table 2 shows the 9-movie family built by FF from the ratings graph on 10,681 movies in LargeD for the seed *Sleepless in Seattle*. Qualitatively, fans of romantic comedies will recognize most of these movies. (All movie descriptions here are excerpts from synopses in LargeD.)

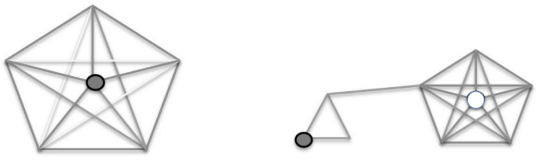

Figure 1: An acceptable family (left) and a family in thrall to a magnet (right). Heavier-weight edges are darker.

Table 2: Ratings family for *Sleepless in Seattle*

Sleepless in Seattle: a recently-widowed man's son calls a radio talk show...to find his father a partner
Pretty Woman : a man...needs an escort...and hires a beautiful prostitute...only to fall in love
You've Got Mail: a man and a woman who meet on-line in a chat room finally end up meeting one another
My Best Friend's Wedding: a woman realizes that friends can be lovers, but now has to convince the friend in question
Coyote Ugly: a sexy romantic comedy
How to Lose a Guy in 10 Days: executive and ladies' man...bets that he can make a woman fall in love with him in 10 days
Pearl Harbor: the story of two best friends...and their love lives as they go off to join the war
Under the Tuscan Sun: a divorced American writer ... moves to Italy in search of a...more romantic life
Head in the Clouds: a young Cambridge student gets involved with a budding photographer...their relationship gets severed

Our quantitative criteria for recommendation families are size, cohesiveness, focus on the seed, and non-randomness. Families should be large enough to be informative, but not so large that they cover more than a small fraction of the graph. The lower portion of Table 1 describes the families FF learned when seeded on each movie in MediumD's three item graphs. These families meet the size criterion; none was ever larger than 180. A family is *cohesive* if, compared to its item graph, it is denser and its edges are of uniformly higher weight. On MediumD, FF finds cohesive families; despite the relative sparsity of the item graphs in Table 1, the families themselves are all cliques or nearly so, and have considerably higher mean edge weights than their item graphs.

We used random sets [Newton *et al.*, 2007] to gauge the likelihood that FF's families were simply artifacts of random local search. Given the heavy computational burden this method entails, we tested only in SmallD, and compared the families to randomly selected subsets of the same size by their items' z-scores. A z-score of at least 2 is considered non-random. For the 1547 provenance families of size at least 3, we scored all 133,224 item occurrences. Every occurrence had a z-score of at least 4, which indicates that its participation in the set is significant. Moreover, the seeds' mean z-scores were substantially higher than that of the non-seeds: 13.03, compared to 7.72 for non-seed nodes. This test confirms that the families built by FF are not only non-random, but also highly interrelated and focused on their respective seeds. Results for the text families and the ratings families were equally strong. In other words, we would expect human subjects to accept movies in an FF family as related to one another under the similarity rationale that supports them.

6 Empirical Design and Results

Our experiments recruit human subjects online, through email and bulletin boards, with no reward for participation.

After informed consent, the subject answers questions while GURU records both her answers and the clickstream. Finally, the subject may volunteer demographic information.

A recommender should guide its user efficiently. Because people prefer prototypes when they reference a set of items [Rosch and Mervis, 1975] and also name prototypes first in discussion [Mervis *et al.*, 1976], we give GURU a computational head start: 92 prototypes for genres. The 92 were chosen from among movies in MediumD that had been rated at least 30 times. In each genre, the 3 movies that most often received a 5-star rating, and so were likely to be recognized and enjoyed, became prototypes. (Not every genre was rated often enough to contribute three prototypes.) GURU uses these 92 movies to estimate the subject's current preferences.

To begin a session in any of our experiments, the subject selects a *starter*, a movie she considers enjoyable. In the starter dialogue, GURU offers a randomly selected, randomly ordered list of five prototypes, and asks the subject to select one she has seen and enjoyed. If she selects "None of these," GURU provides a brief synopsis for each of them and asks which she thinks she might enjoy. (Longer synopses are also clickable, concealed by a spoiler warning.) The subject may read any number of synopses before she selects a starter or requests a new set of five from which to choose. Once the subject selects a starter, GURU generates its *prospects*, all movies in any family that includes the starter. The starter need not be the family's seed, and the family may have been constructed under any of the three similarity rationales.

6.1 Experiments 1, 2, and 3

The first three experiments used FF's families for MediumD. GURU asked, one at a time, about a set of randomly ordered movies. For each movie, GURU offered a synopsis and, on request, a longer synopsis concealed by a spoiler warning. On a Likert scale from 5 (extremely enjoyable) to 1 (not enjoyable at all), subjects were asked to indicate the degree to which they had enjoyed the movie (if they had seen it) or expected to enjoy it based on the synopsis. In each experiment, a paired sample t-test showed no significant difference in the Likert values our subjects assigned to pairs of movies under the same condition, that is, subjects were consistent in their ratings of movies generated under the same rationale.

Experiment 1 tested whether a subject would be more likely to consider movies enjoyable if they were in the same family as her starter. GURU offered eight movies drawn at random: two from MediumD, and two from each of the starter's text prospects, its ratings prospects, and its provenance prospects. Under a t-test, our 143 subjects preferred movies similar to their starter under each similarity rationale to randomly selected movies ($t = 143, p < 0.01, \mu = 2.5, 2.769, 2.867, 2.769$). Subjects did not, however, prefer movies suggested under one similarity to those suggested under any other ($t = 143, p = 0.05$).

Experiment 2 tested whether a subject would be more likely to consider movies enjoyable under one pair of similarity rationales than another. Let L_r be the set of all prospects under rationales R_1 or R_2 (or both). We scored $m \in L_r$ with

$$v(m, r, R_1, R_2) = w(m, r, R_1) \cdot F(m, r, R_1) + w(m, r, R_2) \cdot F(m, r, R_2) \quad (7)$$

where $w(m, r, R)$ is the weight of the edge between m and the starter r in the item graph for rationale R, and $F(m, r, R)$ is the number of families under rationale R that contain both m and r. Without repetition, GURU sampled movies similar to starter r from L_r with probabilities based on the v scores computed in (7). GURU offered 102 subjects 9 movies each: 3 from each pair of similarity rationales. Likert scores for movies similar under ratings and provenance were higher than for movies similar under ratings and text ($t = 102, p = 0.05, \mu = 3.086, 3.039$).

Experiment 3 tested whether a subject would be more likely to consider movies enjoyable if they were nearest neighbors of the starter r in the ratings graph (the traditional approach to recommendation) than if they were related to r under one similarity rationale. GURU offered 66 subjects 8 movies: the 2 nearest neighbors under ratings and 2 from each pair of rationales, selected under equation (7) as in Experiment 2. An ANOVA indicated no significant differences in Likert scores from 66 subjects on any of those pairs (p = 0.05). This test is currently underpowered; we continue to recruit more subjects to see if non-statistically significant trends will become significant.

6.2 Experiment 4

Our hypothesis in Experiment 4 is that good recommendations can arise from families with above-average similarity to a movie identified by the subject as enjoyable. Let $P(r)$ be the set of all *prospect families*, those, under any of the three rationales, that contain the subject's starter r. The movies in $P(r)$ are potentially relevant but typically so many that a full list would overwhelm the user. For example, *Field of Dreams* belongs to 41 families and has 429 prospects.

GURU generates questions, one at a time, to narrow $P(r)$ rapidly in response to the subject's current preferences. GURU asks about a text or provenance feature f^*. For example, if f^* is a text token, "Would you like to watch a movie about dogs?" or, if f^* is a participant, "Would you like to watch a movie written by Michael Crichton?" Other provenance features ask the subject to select any number from a short list of values. If f^* is genre, GURU lists the 10 genre values with the highest acuity; if f^* is the movie's year, GURU lists all decades in the dataset; and if f^* is MPAA rating, it lists all MPAA ratings. Every question has a "No preference" option.

To find f^*, GURU represents each family F_j by its *centroid* c_j, an artificial movie whose feature values are the means of the respective features' values in F_j. Initially, the *utility* of a family $F \in P(r)$ is r's mean edge weight in F. A question about feature f^* is intended to maximally distinguish among all prospects in the *useful families* $U(r) \subseteq P(r)$, those with above average utility. In $U(r)$, f^* is the feature that maximizes the average difference between its values $f(c_j)$ in the centroids and its mean value μ_f across $U(r)$:

$$f^* = \arg\max_f \frac{1}{|U(r)|} \sum_{j=1}^{|U(r)|} |f(c_j) - \mu_f| \quad (8)$$

On each iteration, GURU identifies $U(r)$, calculates which feature f^* to ask the subject about, and then, if the subject

expresses a preference, refines $P(r)$ and recalculates $U(r)$ based on her answer. In response to a preference for or against f^*, GURU eliminates prospects that violate it, recalculates the centroids of reduced families, and adjusts the utility of each family F_j by $f^*(c_j)$, as an increment for a "yes" answer or a decrement for a "no." Iteration halts when $|U(r)| \leq 3$, fewer than 7 prospects remain, or an expressed preference does not reduce $P(r)$.

GURU then recommends one prospect at random from each useful family, in descending order by their utility, until 6 prospects have been chosen or none remain. If the subject has seen the movie, she is asked how much she enjoyed it. Otherwise, she sees a brief synopsis and can request a longer one before she evaluates how much she would expect to enjoy the recommendation. Both questions use the Likert scale from the earlier experiments.

Table 3 is a sample real-time, online dialogue. With only 5 questions, GURU rapidly narrowed $U(r)$ from 69 families and 309 movies down to 23 families, 3 of which were useful, and offered the subject the 5 movies shown in Table 3. (One synopsis indeed described Godzilla as a "dinosaur-like beast.") Experiment 4 is now in preliminary testing.

Table 3: Sample GURU dialogue for the starter *Jurassic Park*.

The user's replies are in boldface.

Do you want to watch a movie about a dinosaur?
Yes
What kind of movie do you want to watch?
Please check all that apply [*list of 10 genres here*]
Action, Science Fiction
Do you want to watch a movie directed by Steven Spielberg?
No
Do you care about the release year of the movie?
No
Does a movie's MPAA rating matter to you (e.g., PG-13)?
Yes
Please select all acceptable ratings: G,PG,PG-13,R,NC-17
PG, PG-13, R

GURU now recommends one movie at a time and asks how much the user enjoyed it or would expect to enjoy it.

I recommend *Carnosaur 3: Primal Species* because it is an action movie about a dinosaur
Description please
...terrorists get a surprise when their cargo turns out to contain living dinosaurs...now have to...prevent the extinction of the human species
That would be quite enjoyable
I recommend *Lost World: Jurassic Park* because it is a science fiction movie about a dinosaur
I have seen it; it was moderately enjoyable
I recommend *Baby... Secret of the Lost Legend* because it is a science fiction movie about a dinosaur
Description please
Paleontologist and her husband discover a mother and baby brontosaurus...try to protect them from hunters who want to capture them.
That would be moderately enjoyable
I recommend *Carnosaur 2* because it is an action movie about a dinosaur
Description please
Extinction is a thing of the past. [*User requests a longer description, omitted to conserve space*.]
That would be quite enjoyable
I recommend *Godzilla* because it is an action movie and a science fiction movie about a dinosaur
I have seen it; it was extremely enjoyable

7 Discussion

7.1 Responsiveness and Plausibility

Human uncertainty, impatience, and unreliability preclude the ideal approach to reactive recommendation: ask the user which features she cares about at this moment, their relative importance, and what their values should be, and then comb the dataset for good matches. Instead, GURU uses a starter to capture the user's attitude, and uses families to create a richer structure that supports multiple similarities while it casts a broad but thoughtful net.

The automatic acquisition of knowledge from unstructured data is at the heart of our approach. GURU crawls the web for descriptive information about the items in its dataset, applies similarity metrics to construct graphs on those items, and develops an interrelated family around each item in each graph.

Refusal to commit to any of the starter's feature values allows GURU to avoid the pitfalls of more traditional recommenders. Without forced matching on features like genre, GURU provides breadth. Without reliance on nearest neighbors, GURU can offer more than merely a sequel or a prequel. Nor does GURU ignore the user's current mood to focus on popularity, as collaborative filtering might. For example, one movie in Table 3 was the second-nearest neighbor of *Jurassic Park* under ratings, but none of the others appeared among its top 30 nearest neighbors. Instead, GURU looks for similarities under its rationales and prefers recommendations with support from multiple rationales.

GURU's advice is compatible with what we know about human representation of and conversation about concepts. Unlike a semantic network or an artificial neural network, a small set of prospects closely related to the starter and accompanied by descriptive text is easy to consider. Although one could display an entire family, we believe that even a list of more than six movies would drive away most users. GURU's simple explanation for each recommendation is the features that the recommended movie shares with the user's starter. Combined with text descriptions from the dataset, this highlights family resemblances for the user.

7.2 Generality

For clarity, this paper has described GURU simply as a movie recommender, but its automated acquisition of knowledge is more generally applicable. *DRILL* (Dialogue for Recommendations in Large Libraries) was a book recommender for patrons of a real-world library [Epstein *et al.*, 2012]. DRILL gathered descriptions for 53,616 books from the librarians' synopses, a commercial book seller's website, and an exten-

sive book mashup [Bizer and Gaub, 2012]. DRILL differed from GURU in several significant ways.

Without readers' ratings, DRILL relied on circulation data made available by the library: the borrowing history and expressed genre preferences of 414 anonymized patrons. Because book provenance data (e.g., editors, researchers) is less widely available, DRILL had only two similarity metrics rather than three, and thus generated only two sets of families: text and circulation. DRILL's text graph was similar to GURU's, and its text families were often qualitatively as compelling as GURU's. For example, 13 of the 14 books in the text family for the biography *The Jackie Robinson Reader* were about baseball or another sport, but they also included fiction and books on minority players and on young athletes.

DRILL's circulation graph, however, was a boolean vector on the 414 patrons, and thus considerably less nuanced than GURU's ratings graph. Moreover, because the library's patrons were unavailable, evaluation had to be by simulation from patrons' borrowing histories: if DRILL suggested a book that the patron had checked out, that was considered a successful recommendation. The quantity of readily available movie data, particularly provenance, and the greater willingness of human volunteers to discuss movies with a computer resulted in GURU. Nonetheless, DRILL demonstrated that it is possible to harness existing unstructured descriptive item data for another domain, along with opinions and behavioral data, to construct meaningful relationships among items.

7.3 Alternatives and Future Work

PageRank detects and ranks similarity within a large weighted graph [Brin and Page, 1998]. If applied here, once it converged it would provide a pairwise view of the likelihood that one movie would recommend another under some similarity rationale. Item graphs, however, are considerably denser than those to which PageRank is typically applied. FF can also set a time limit, score a family as it evolves, and quantify explicitly the qualities valued in a family.

GURU promotes diversity. In different sessions, it will recommend different movies to a user who chooses the same starter, unless she also provides the same preferences and GURU makes the same random choices in $U(r)$. Although GURU still allows families to exploit magnets for diversity, the user's starter and GURU's questions balance a user's current preferences against the strength of a magnet.

The use of three nested datasets was crucial here. LargeD provided a broad context within which to norm ratings and to identify a substantial token vocabulary. SmallD (with 1682 movies, 943 users, and 100,000 ratings) allowed us to demonstrate, exhaustively, that despite FF's non-determinism, the families it generates are non-random. SmallD was also the test-bed where we investigated only a few values for two other computationally exorbitant tasks: pruning thresholds for the item graphs, and the constants in equations (5) and (6). Both were applied unchanged to MediumD and LargeD.

Although we used MediumD to accelerate online testing, we expect to run Experiment 4 on LargeD in a laboratory setting. Preliminary tests suggest that, as the dataset enlarges, family size does not necessarily increase. Instead, each family focuses more strongly on its seed — both its score under equation (5) and the mean edge weight to its seed increase, while the standard deviation of its edge weight decreases.

A significant issue in recommendation is that both the user set U and the item set I are dynamic. (This was one reason Netflix never put BellKor into production [Amatriain and Basilico, 2012].) New users and items appear, known users and items become unavailable, and new descriptive features (e.g., a new director) arise for items across time. Thus, any attempt to classify U or I by a predetermined set of features must be temporary.

Improvements in family scores in the larger datasets, however, bode well for an incremental version of FF, one that would activate the web crawler to extract data for a new movie nm. (The text item graph is large enough to defer full re-computation of *tf-idf* values to infrequent, offline updates. Ratings could be updated at the same time. Given nm, the system could create three families seeded on it, add nm to each existing family for which it satisfied equation (6), and reactivate FF on those families for further additions and swaps. If a movie were to become unavailable, the system would shake it from each family that contained it, and refresh that family with a single iteration of FF. Moreover, much of this computation could proceed in parallel.

Unlike traditional recommenders, GURU does not attempt to predict ratings. Instead, it suggests a small set of movies the user is likely to enjoy, based on a real-time dialogue that elicits spur-of-the-moment preferences. Future work includes further investigation of the constants in equations (5) and (6) with human subjects. Given that people do not weight features equally when they gauge similarity [Weisberg, 2012], it may also be possible to refine the provenance edge-weight formula in equation (3).

As DRILL demonstrated, GURU's approach is not restricted to movies. It is suitable for consumable and recreational items (e.g., books, videos, clothing, food) that offer many choices. Other similarity rationales (e.g., accessibility, critics' opinions) are also readily incorporated. GURU is a first step in the use of family resemblance to enhance human cognition with an expert assistant who shares people's worldview. GURU collaborates with its user to help make choices in an item-rich environment, so that the user will indeed know it when she sees it.

References

[Amatriain and Basilico, 2012] X. Amatriain and J. Basilico. Netflix recommendations: Beyond the 5 stars (part 1). `http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html`, 2012.

[Bell and Koren, 2007] R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Proceedings of the Seventh IEEE International Conference on Data Mining*, pages 43–52, 2007.

[Berkhin, 2006] P. Berkhin. A survey of clustering data mining techniques. In J. Kogan, C. Nicholas, and M. Teboulle, editors, *Grouping Multidimensional Data*, pages 25–72. Springer, 2006.

[Bizer and Gaub, 2012] C. Bizer and T. Gaub. Freie University's RDF book mashup. `http://www4.wiwiss.fu-berlin.de/bizer/bookmashup/`, 2012.

[Blei *et al.*, 2003] D.M. Blei, A.Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[Boim *et al.*, 2011] R. Boim, T. Milo, and S. Novgorodov. Diversification and refinement in collaborative filtering recommender. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 739–744. ACM, 2011.

[Brin and Page, 1998] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30:107–117, 1998.

[Charlin *et al.*, 2014] L. Charlin, R.S. Zemel, and H. Larochelle. Leveraging user libraries to bootstrap collaborative filtering. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 173–182, 2014.

[Das *et al.*, 2013] M. Das, G. D. F. Morales, A. Gionis, and I. Weber. Learning to question: Leveraging user preferences for shopping advice. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 203–211. ACM, 2013.

[Edelman and Shahbazi, 2012] S. Edelman and R. Shahbazi. Renewing the respect for similarity. *Frontiers in Computational Neuroscience*, 6:1–19, 2012.

[Epstein *et al.*, 2012] S.L. Epstein, E. Osisek, and R.J. Passonneau. Similarity and plausible recommendations. *Advances in Cognitive Systems*, 2:185–202, 2012.

[Goldstein, 1994] R. Goldstein. The role of similarity in categorization: providing a groundwork. *Cognition*, 52:125–157, 1994.

[Hansen and Mladenovic, 2003] P. Hansen and N. Mladenovic. Variable neighborhood search. In F. W. Glover and G. A. Kochenberger, editors, *Handbook of Metaheuristics*, pages 145–184. Springer, Berlin, 2003.

[Hasan *et al.*, 2014] M. Hasan, A. Kashyap, V. Hristicis, and V. Tsotras. User effort minimization through adaptive diversification. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 203–212. ACM, 2014.

[Holbrook *et al.*, 2003] A.L. Holbrook, M.C. Green, and J.A. Krosnick. Telephone versus face-to-face interviewing of national probability samples with long questionnaires: Comparisons of respondent satisficing and social desirability response bias. *Public Opinion Quarterly*, 67:79–125, 2003.

[Kille and Abel, 2015] B. Kille and F. Abel. Engaging the crowd for better job recommendations. In *Proceedings of ACM RecSys CrowdRec 2015 Workshop*. ACM, 2015.

[Koren, 2010] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53:89–97, 2010.

[Li and Epstein, 2010] X. Li and S.L. Epstein. Learning cluster-based structure to solve constraint satisfaction problems. *Annals of Mathematics and Artificial Intelligence.*, 60(1):91–117, 2010.

[Mahmood and Ricci, 2009] T. Mahmood and F. Ricci. Improving recommender systems with adaptive conversational strategies. In *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia*, 2009.

[Mahon and Caramazza, 2009] B.Z. Mahon and A. Caramazza. Concepts and categories: A cognitive neuropsychological perspective. *Annual Review of Psychology*, 60:27–51, 2009.

[Mervis *et al.*, 1976] C.B. Mervis, J. Catlin, and E. Rosch. Relationships among goodness-of-example: category norms and word frequency. *Bulletin of the Psychonomic Society*, 7:268–284, 1976.

[Newton *et al.*, 2007] M.A. Newton, F.A. Quintana, J.A. den Boon, S. Sengupta, and P. Ahlquist. Random-set methods identify distinct aspects of the enrichment signal in gene-set analysis. *Annals of Applied Statistics*, 1(1):85–106, 2007.

[Porter *et al.*, 1990] B.W. Porter, R. Bareiss, and R.C. Holte. Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, 45(1-2):229–263, 1990.

[Rosch and Mervis, 1975] E. Rosch and C.B. Mervis. Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology*, 7:573–605, 1975.

[Salakhutdinov and Mnih, 2008] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Proceedings of Advances in Neural Information Processing Systems 20 (NIPS 07)*. ACM Press, 2008.

[Sarwar *et al.*, 2001] B. Sarwar, G. Karypis, j. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*, 2001.

[Shan and Banerjee, 2010] H. Shan and A. Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM '10)*. IEEE Computer Society, 2010.

[Thompson *et al.*, 2004] C. A. Thompson, M. H. Gker, and P. Langley. A personalized system for conversational recommendations. *Journal of Artificial Intelligence Research*, pages 393–428, 2004.

[Tversky, 1977] A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

[Weisberg, 2012] M. Weisberg. Getting serious about similarity. *Philosophy of Science*, 79(5):785–794, 2012.

[Wittgenstein, 1953] Ludwig Wittgenstein. *Philosophical Investigations*. Blackwell, Oxford, UK, 1953.

[Zhao and Karypis, 2005] Y. Zhao and G. Karypis. Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery*, 10:141–168, 2005.