Advances in Cognitive Systems 1 (2012) 1-18 Submitted 9/2012; published 12/2012

Similarity and Plausible Recommendations

Susan L. Epstein SUSAN.EPSTEIN@HUNTER.CUNY.EDU
Eric Osisek EOSISEK@GC.CUNY.EDU
Department of Computer Science, Hunter College and The Graduate Center, The City University of New York, New York, NY 10065 USA
Ben Hixon BHIXON@CS.WASHINGTON.EDU
Department of Computer Science & Engineering, University of Washington, Seattle, WA 98195 USA
Rebecca J. Passonneau BECKY@CS.COLUMBIA.EDU
Center for Computational Learning Systems, Columbia University, New York, NY 10115 USA

Abstract

Recommendations from a system to a person require thoughtful decisions. People prefer quick, relevant responses, ones that quickly lead to an acceptable choice. A system's recommendations, however, often rely on pairwise similarity between items or between users, based on a distance metric. This paper advocates a broader, set-based similarity based on text descriptions of known items, and on the items users have found appealing in the past. The text, autonomously gathered from the Web, proves to be a strong indicator of similarities among sets of items, as well as a source of useful diversity. Empirical results provide insight into the complexities of similarity, of user behavior, and of cognition in the system that formulates recommendations.

1. Introduction

When a human expert recommends an item to someone, she gauges its suitability from a variety of perspectives, including item popularity, the person's preferences, and the nature of the items available. At the core of recommendation, however, is similarity: similarity of items to one another, and similarity of users because they request similar items. Our long-range goal is a proficient and knowledgeable *recommender,* a system that discusses suggestions from its knowledge base with its user as if it were a human expert. A human expert, based on her experience and world knowledge, has a broad and deep perspective on how items she knows are similar to one another. We report here on automated recommenders whose "experience" is historical data on its users, and whose "world knowledge" is text descriptions written by people and available on the Web. Our thesis is that an intelligent recommender can autonomously construct meaningful sets of similar objects and can exploit them to produce plausible recommendations for its users. The novelty of our approach is the use of set-based, rather than pairwise, similarity. Our principal results are that set-based similarity can be efficiently detected, and can be used to produce *plausible* (i.e., likely to be acceptable) recommendations.

The cognitive challenge explored here is how a system represents and uses similarity. Systems traditionally gauge *pairwise similarity* with a distance metric on the feature vector values for two items. This in turn defines the *nearest neighbor* of an item, a second item whose feature vector values minimize its distance to those of the first. Feature values that identify items, however, do not always support the similarities important to people. Consider, for example, *Boots,* a US soldier's memoir of Vietnam. With a feature vector for title, author, and publication date, its nearest neighbors include three other books with the same title: a cowgirl romance novel, a fantasy about a magic talking cat, and a children's story about raingear. Any reader of one is unlikely to be interested in the others. A genre designation might help, but genres often prove too broad (e.g., history), too narrow (Oklahoma cowgirl-reporter romance), or too subjective (e.g., humor). Intuitively, what is lacking in the feature vector is values that represent what the books are about.

People habitually and adroitly agglomerate large collections of items into smaller, more manageable sets that are easier to reference and discuss. These sets are founded on similarity (Wierzbicka, 1985). In extensive experiments with people, however, human similarity judgments did not abide by the axioms of distance metrics (Tversky, 1977). Instead, people considered mutual similarity within a set. Our work uses pairwise similarities detected in people's item descriptions to build sets of items that share common features and yet offer interesting diversity.

This paper introduces *DRILL* (Dialogue for Recommendations In Large Libraries), which leverages text descriptions of the items in its knowledge base to build *clusters,* sets of mutually similar items. The system gauges similarity from human judgments: both user choice histories and text descriptions written for people by those familiar with the items. Ideally, description similarity would be based on the meaning of the text DRILL collects, but the semantics of text is a long-standing challenge in natural language processing. Instead, DRILL makes a good start at similarity detection from concise textual descriptions. It crawls the Web to gather descriptions of the database items, extensively prepares this data for use by a pairwise similarity metric, forms a weighted graph that represents those similarities, and then identifies clusters (sets of strongly, mutually similar items) within it. The system also hypothesizes plausible recommendations from its clusters, none of which is likely to contain both a children's story and a romance novel.

This work shows how multiple, expository, text-based item descriptions help identify user-appropriate items in a large database. It also shows that real-world users demonstrate pervasive discrepancies among what they request, what they claim to prefer, and what they choose. In such circumstances, DRILL's set-based recommendations prove more effective than recommendations based on pairwise distance alone.

This work is inspired by librarians' skill and breadth of knowledge, but it is not restricted to books. Like a good librarian, a recommender should structure representations that facilitate its task. Librarians seek to open new vistas for their patrons, to shift them gently between fiction and non-fiction, for example, or from one subgenre to another. Thus a recommender's clusters must be neither too broad (e.g., "book") nor too specific (e.g., "early 19th century space adventure"). A recommender confined to statistics or obvious feature values (e.g., author) is unlikely to strike the right balance between generality and specificity. Librarians often interview a user to determine what she has read, what books she liked, and what she liked about a particular book. As a result, their recommendations often range well beyond traditional genre or subject matter classification. Like a librarian, DRILL seeks to cast a wider net.

A recommender must ultimately address knowledge discovery, human choices, dialogue, natural language understanding of text, and user modeling; here we focus on the first two. "Dialogue" with DRILL is thus far simply a sequence of recommendations that ends when the user accepts one, and DRILL considers only word stems not associated with named entities. Our user model is realistic but simple; it is based on users' recorded preferences and previous choices.

The next section of this paper provides necessary background and related work. Section 3 describes DRILL, explains the construction of the large graphs DRILL explores, and outlines the algorithm DRILL uses to find clusters. Subsequent sections describe our experimental design, and present and discuss the results and future work.

2. Background and Related Work

DRILL is a recommender, tested here with a simulated user that models real-world data. This section provides background on other recommenders and their metrics, user simulation for dialogue, and our domain of investigation, the Andrew Heiskell Library.

2.1 Recommenders

Most work on recommendation systems has thus far been based on users' ratings (Bell & Koren, 2007) and interaction with the items (Amatriain & Basilico, 2012). Most recommenders collect *ratings* (numeric scores from users about how much they like individual items) and, based on them, suggest items the user is expected to like. Netflix (Bell & Koren, 2007), Pandora, and Amazon all have recommenders that suggest items to customers based on their ratings.

The principal paradigms for recommenders are collaborative filtering and content-based recommendation. *Collaborative filtering* is based on ratings, often on a scale from 1 to 5. It finds groups of users who have liked similar sets of items, and uses the average rating for item i from users similar to a user u to predict u's preference for i. (Collaborative filtering can also be used to identify similar items, based on ratings they received from the same users.) Users, however, assign ratings idiosyncratically, and there is no evidence that human experts process ratings to construct their recommendations. Because collaborative filtering requires ratings, and because enough ratings may be unavailable, DRILL does content-based recommendation.

A *content-based* recommender suggests to a user items most similar to items she liked in the past, based on a pairwise similarity metric. Given a pair of feature-value vectors that describe two items, similarity metrics can be applied to them in several ways (Sarwar et al., 2001). *Cosine-based similarity* measures the similarity of the items as the cosine of the angle between their feature vectors. Identical feature vectors have cosine similarity 1; smaller values indicate less similarity. *Adjusted cosine similarity* reduces the impact of extreme mean values by subtracting the mean feature values before it calculates the cosine.

Although these similarity metrics were originally proposed for ratings data, they can also be applied to other feature representations. For example, in the set D of all documents, the *tf-idf* (term frequency–inverse document frequency) of word w in document d is

$$tf\text{-}idf\left(w,d\right) = tf\left(w,d\right)\cdot\log\frac{\left|D\right|}{\left|\left\{d'\in D \middle| w\in d'\right\}\right|} \qquad (1)$$

Here, the *term frequency tf(w,d)* is how often word w appears in document d, and the *document frequency* (the denominator in equation (1)) is the number of documents that contain word w. Thus *tf-idf* is high for a word that appears frequently in a document, if that word is not contained in many other documents, but is low for a word that appears in most documents. One may treat words in a document as features with *tf-idf* values, and use cosine-based similarity to detect document similarity (Zhao & Karypis, 2005). DRILL uses cosine-based *tf-idf* similarity based not on documents' content but on expository descriptions of that content. (Adjusted cosine similarity is unnecessary, since *tf-idf* already corrects for the norm.)

A content-based system uses a similarity metric to make recommendations. Two ways to produce recommendations from item similarity data are weighted sum and regression (Sarwar et al., 2001). A *weighted sum* recommender predicts a user's rating for an item as the average of her ratings on all other items, weighted by their similarity to the item under consideration. In a large database, however, it is unlikely that a user has rated every item other than the one for which the system is to predict a rating. This is known as the *sparse data problem*. To address it, a *regression* recommender first uses linear regression on all items the user has rated to predict her missing ratings. Then, to predict an individual unknown rating, it averages the real or predicted rating of every other item, weighted by its similarity to the requested item. The domain DRILL addresses in the work reported here, however, has no rating data at all.

A recommender for book purchases combined collaborative filtering with content-based recommendation paradigms from a two-layer graph (Huang et al., 2004; Huang et al., 2002). One layer had a node for each item, and an edge between similarly-rated items weighted by the items' similarity. The other layer had a node for each user, and an edge between similar users weighted by the similarity of the users' ratings. The two layers were connected by an edge between each user and each item she purchased. To make a recommendation for a user, a Hopfield net found items similar to those the user had previously purchased, and items purchased by users similar to her. This method did not, however, consider what a user had just requested when it suggested a replacement. If the requested item differed from what she usually rated highly, it still recommended the nearest neighbor to her usual choices.

Some recommenders have attempted to understand a user's preferences beyond what can be immediately computed from ratings data. A *conversational recommender* has a dialogue with the user to elicit and learn preference information intended to improve the recommendation process over time. For example, the Adaptive Place Advisor, a recommender for restaurants, asks questions such as "What kind of food would you like?" or "What price range?" to constrain the set of potential restaurants when many possibilities remain (Thompson et al., 2004). If no restaurant matches the user's constraints, the system asks permission to relax some constraint. During a dialogue, the Adaptive Place Advisor tracks both the user's expressed preferences and which preferences she is most likely to relax, and then applies them in an attempt to make better recommendations and shorten the dialogue. DRILL does not (yet) ask questions.

To the best of our knowledge, only one cognitive systems approach has previously been considered for a recommender (Maanen & Marewski, 2009). That work modeled a scientist's interests based on words that appeared in the abstracts of her publications, and assigned greater importance to words that appears frequently and in more recent publications. A word was also rated as more important if it often appeared alongside high importance words. Publications whose abstracts contained many high importance words were said to match the model of the scientist's interests, and the publications that best matched her interests were recommended. Although based on ACT-R, this recommender was outperformed by a simple take-the-best fast and frugal heuristic (Gigerenzer & Goldstein, 1999). While DRILL's text is like a set of abstracts, its set-based similarity allows for multiple modalities and diversity.

A recommendation's plausibility lies in its relevance and reasonableness to the user. Most user preference models for recommenders are used to enhance collaborative filtering (Rashid et al., 2002), but a recommender can also take advantage of user models. *Knowledge-based user models* elicit user preferences through questionnaires; *behavior-based user models* track patterns in user behavior. One recommender for academic research papers in an ontology, modeled users' Web browser behavior, and used ontological inference to link user behavior to ontology elements (Middleton et al., 2004). A framework proposed for the *user-model mediation problem* combined multiple data models of user preference into a single model (Berkovsky et al., 2008). More sophisticated models of user intent are primarily statistical. They consider knowledge sources such as Wikipedia (Hu et al., 2009), employ contextual information (White et al., 2009), or mine query logs with graphical models (Pantel et al., 2012). We test DRILL here with both knowledge-based and behavior-based users. Our focus, however, is principally on the development of the recommender, not the user model.

2.2 User Simulation in Dialogue

A *user simulator* is a program intended to behave as if it were a particular user while it engages in a dialogue with a recommender. User simulation is frequent in dialogue system development because it offers many practical advantages. When it produces realistic user behavior, a user simulator accelerates the design process. The first proposed user simulator for spoken dialogue development used a bigram model, where the likelihood of a simulated user action given a system utterance was the frequency of that system-action pair in a corpus of real dialogues (Eckert et al., 1997). Other, more sophisticated models have since been developed. *N*-gram simulators have incorporated features of the dialogue state (Georgila et al., 2006). Markov logic networks have modeled user dialogue intentions (Jung et al., 2011). Probabilistic simulators can include goals extracted as latent topics in a model generated from a set of real dialogues (Eshky et al., 2012).

There is, however, no standard way to compare the quality of simulated dialogue to dialogue with real users. One survey of quantitative evaluations for user simulations argued the importance of carefully designed and tested, realistic simulated dialogues, particularly in the absence of a corpus of real user-system dialogues for comparison (Schatzmann et al., 2005). The work reported here uses extensive real-world data to promote realism in our simulator, which is seeded with hundreds of histories that contain thousands of actual users' choices.

2.3 The Andrew Heiskell Braille and Talking Book Library

The Andrew Heiskell Braille and Talking Book Library is a branch of The New York Public Library system and a Regional Library of the National Library Service. Heiskell's more than 5000 active patrons order books at least once each week by telephone, and receive them by mail. A patron often orders as many as 20 books during a single conversation with a librarian. Patrons do not rate books; they check them out and keep them as long as they wish. As a result, there is no clear indication of a patron's opinion about any book. Most users, however, do have general preferences for authors and genres recorded in the database

The work reported here is based on a copy of Heiskell's database, sanitized to protect the privacy of its patrons. Our copy includes 71,166 books by 28,045 authors. Some books are available in large print or in Braille; this work considers only the 53,616 recorded books. (There are 711 books for which there are recordings by different readers; they have different catalog numbers.) We have circulation data on 499 of Heiskell's patrons. Because the book descriptions DRILL employs are in English, and because the motivation for reading in more than one language may be different, we restricted consideration to the 414 patrons who checked out at least one recorded book between 1975 and 2008, and requested only books in English. Here we consider only the 15,749 books that are recorded in English and were chosen by at least one of those 414 users. We also confirmed that each user had on record at Heiskell at least one preferred author and at least one preferred genre that could be satisfied by one of the 15,749 items.

3. Knowledge Discovery and Book Graphs

This section describes how we filtered the original book database, and augmented it with text gleaned during extensive Web-crawling. The resultant text was preprocessed, and then represented as token vectors with *tf-idf* values. This data was used to construct a large graph within which a novel algorithm then sought sets of books related to one another by their descriptions. To emphasize that our approach is domain-independent, the remainder of the paper refers to items chosen by users, rather than books checked out by patrons.

3.1 Gathering Data

An important aspect of our work is its focus on expository, descriptive text (e.g., synopses rather than reviews) about the items in the database. We assembled text descriptions for the 15,749 items. The Heiskell library database includes a short description for 98% of these items ($\mu = 48.78$ words, $\sigma = 7.40$). We supplemented those descriptions with others collected by a simple Web crawler that extracted item descriptions from the HTML in Freie University's RDF Book Mashup (Bizer & Gaub, 2012) and a commercial bookseller's website (Barnes & Noble, 2012). Most other websites contain descriptions identical to those available on these two sites, and were therefore omitted. There were 1,479,933 words in the full description corpus.

Some Heiskell descriptions were about the item's manufacture or condition, rather than its content. To remove them, we inspected and then eliminated all Heiskell descriptions for an item that were identical to that for a different item (e.g., "Book" or "Commercial speed, 2-track, no braille or large print labels"). After the removal of duplicate descriptions, there were 25,682 descriptions for the 15,749 items. Of these descriptions, 25.76% came from the RDF Book Mashup, 11.51% from Barnes & Noble, and 62.74% from Heiskell. Items had from 0 (79 items) to 10 (1 item) descriptions ($\mu = 1.70$, $\sigma = 0.91$). Each description now contained from 1 to 2617 words ($\mu = 93.97$, $\sigma = 92.54$), nearly twice what was originally available from Heiskell.

3.2 Building the Graphs

To assess the similarity of pairs of items based on the words in their text descriptions, we first used openNLP's open source tokenizer (http://opennlp.apache.org/) to separate an item's descriptions into individual words (*tokens*). A *stop word* (e.g., "it," "an," "the") is one that encodes little information for our purposes. We removed stop words from these tokens, using a list of 180 words. Next, we used Stanford University's named entity recognizer to remove names, locations, and organizations from the descriptions (Finkel et al., 2005). This avoided misleadingly high similarity scores attributable purely to named entities, particularly unusual character names. Finally, we applied a stemming algorithm (Porter, 1980) to the remaining tokens. It replaced such related words as "murder," "murdering," and "murdered" with a single token. This text processing reduced the 191,466 distinct words in the corpus to 61,855 tokens.

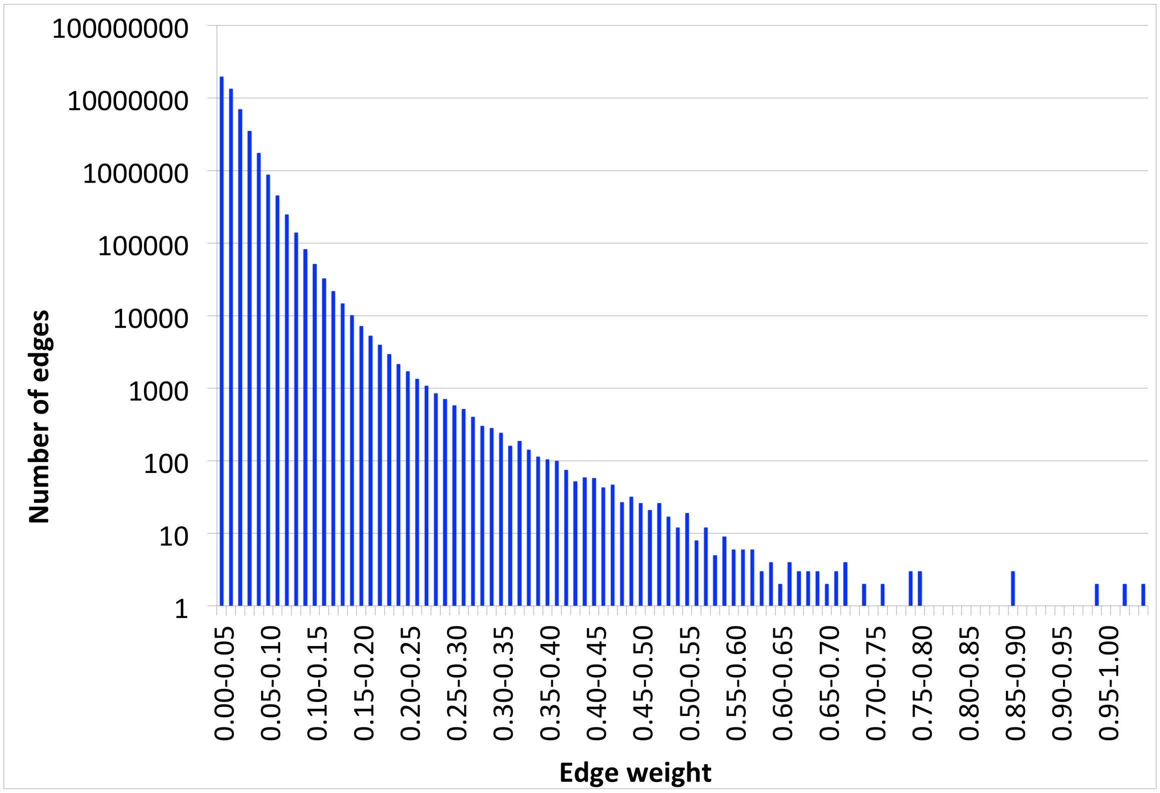

Figure 1. Similarity scores (edge weights) on a logarithmic scale for pairs of items in the word graph.

In DRILL, an *item graph* for a database has a node for each item. The *word graph* is an item graph based on similarity of the items' descriptions. Each item's feature vector is the *tf-idf* values of all tokens in its descriptions. The similarity score for each of the 124,007,626 (${}_{15749}C_2$) pairs of distinct items was calculated as the cosine similarity of their *tf-idf* feature vectors. To examine the score distribution, scores were binned as in Figure 1. To restrict attention to reasonable similarity, only scores of at least 0.03 were retained. Each remaining score served as the weight for an edge inserted between its corresponding pair of items. The result was a weighted item graph on 15,749 nodes and 10,726,660 edges, with *density* (fraction of possible edges) 8.65%, where two items whose descriptions contain more words with identical stems have a higher-weighted edge between them.

The *circulation graph* is a second item graph on the same 15,749 nodes. Each item is represented as a 414–bit vector, where the *i*th bit is true if and only if the *i*th user once chose that item. The edge weights are the cosine similarity of the two items' bit vectors; a higher edge weight indicates that the two items were chosen by more of the same users. These edge weights were binned as in Figure 2, and only edges with a similarity score of at least 0.16 were retained. Each remaining score served as the weight for an edge inserted between its corresponding pair of items. There were 11,284,018 edges (density 9.10%).

3.3 Finding the Clusters

Foretell is an algorithm that searches a graph for dense subgraphs with high edge weights. (A cluster may, but is not required to, be a clique.) Foretell is based on Variable Neighborhood Search (*VNS*), a local search metaheuristic that moves through a sequence of increasingly large neighborhoods (Hansen et al., 2004). Foretell was originally developed to identify the most difficult portions of a constraint graph for constraint satisfaction solvers (Epstein & Wallace, 2006; Li & Epstein, 2010). Foretell has also found biologically-meaningful clusters of genes that interact heavily in protein-protein-interaction networks (Epstein et al., 2012). Here we use its new, standalone version to find clusters of similar items.

Simplified, high-level pseudocode for Foretell as it is applied here appears in Table 1. Under the direction of a control file, Foretell begins with a *seed*, a first node for the cluster. Each time Foretell calls it, VNS attempts to improve the score of *candidate,* the cluster with the best score so far. On each iteration except the first, VNS shakes *candidate*, that is, it randomly selects any *c* nodes (except the seed) and removes them from *candidate*. (Shaking provides search neighborhoods of varying size.) Next, VNS calls *VND* (Variable Neighborhood Descent) which tries to improve *candidate* in two ways. First, VND greedily adds nodes to *candidate*, along with any edges they have to nodes already in *candidate*. Second, VND tries to *swap* (interchange) a node, or a pair of adjacent nodes, not in *candidate* for one node currently in *candidate,* or to add a node

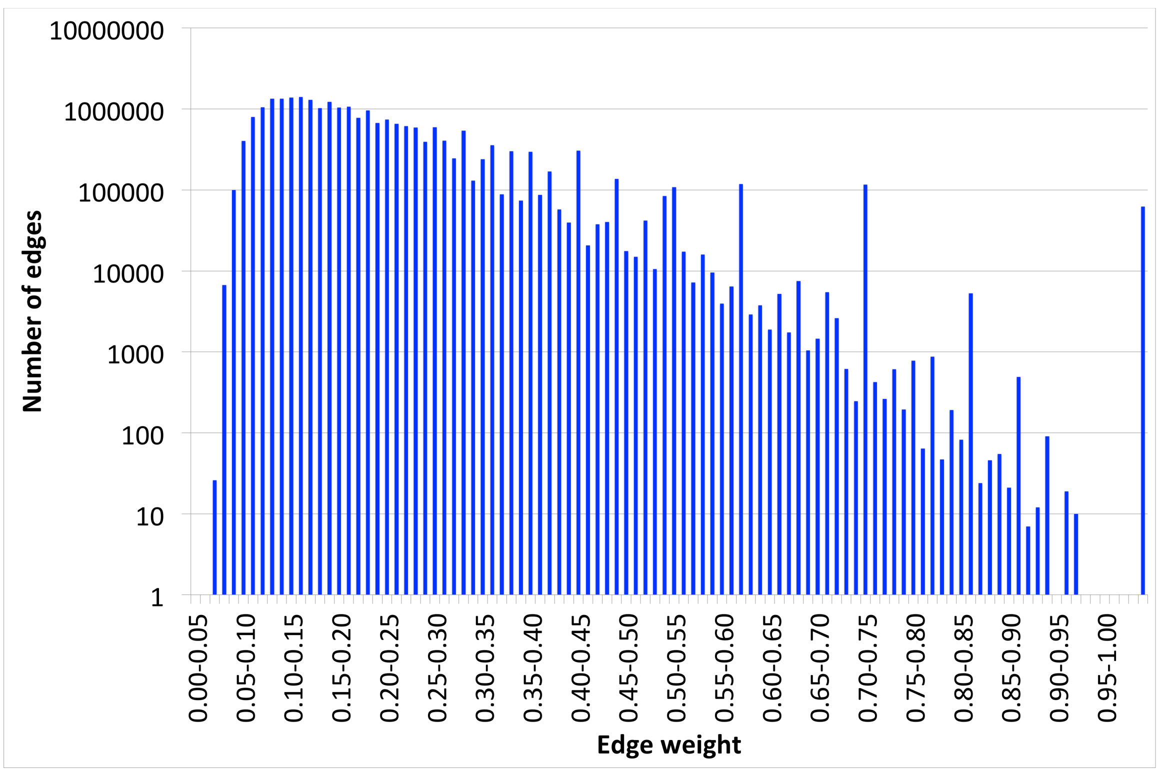

Figure 2. Similarity scores (edge weights) on a logarithmic scale for pairs of items in the circulation graph. Items chosen by a single user cause the rightmost high value.

adjacent to all but one or two of those in *candidate.* When a VNS iteration improves *candidate,* the revision becomes the new *cluster,* and *c* is reset to 1. When VND does not improve *candidate*, *c* is incremented. Termination conditions limit the time devoted to a single cluster and the number of VNS iterations, but allow the upper bound on *c* to rise as *candidate* grows.

In this manner, Foretell gradually grows the best-scoring cluster it can around the seed, randomly discards part of the cluster, and then tries to grow it again, until it exhausts its resources or can do no better. A cluster seeded on a node in an item graph is a subgraph that contains the node and a collection of items that are not only similar to the node itself, but also to one another. Thus, in the word graph a cluster seeded on a book represents a set of books with mutually similar *tf-idf* values, while in the circulation graph a cluster seeded on a book represents a set of books with common readers. To confirm that clusters can provide plausible recommendations, we devised the following experiments.

4. Empirical Framework

This section describes experiments in which a recommender suggests items to our simulated Automated User, *AUdrey*, which communicates with XML (eXtensible Markup Language) (Jung et al., 2009). Although the resultant dialogue is not in natural language, XML is a robust source from which to generate natural language, that is, sentences can be readily construed from its templates (Wilcock, 2003). The use of XML relieves our investigation from the errors endemic to natural language understanding, and allows us to focus on recommendations. The increasing

Table 1. High-level pseudocode for *Foretell* and VNS. See the text for further details.

```
Foretell(seed, control)
Initialize cluster as graph with one node, seed, and no edges
do until termination conditions met
   cluster ←VNS(cluster, neighborhood, control)

VNS(cluster, neighborhood, control)
candidate ← copy(cluster)
bestScore ← clusterScorer(cluster)
c ← 1
do until termination conditions met
   if c ≠ 1
      shake(candidate, c)
   do until termination conditions met
      improve candidate with VND
   if score of improved candidate is higher than bestScore
      cluster ← candidate
      bestScore ← clusterScorer(cluster)
      c ← 1
   else   c ← c + 1
          candidate ← copy(cluster)
return cluster
```

presence of on-line chat support and Web-based browsing agents also makes this a reasonable approach.

Each experiment evaluates a recommender's ability to offer items that match a user's known preferences, recorded at Heiskell as favorite authors and favorite genres. For each (quite simple) dialogue, AUdrey takes as its *persona,* one of the 414 users in the database. This persona includes a user identifier, the items the user has chosen in the past, and the user's known preferences, recorded at Heiskell as favorite authors and favorite genres. The dialogue scenario is as follows: the recommender asks AUdrey for its identity, AUdrey provides identification for its persona, AUdrey randomly selects and requests an item its persona once chose, and the recommender reports that the item is unavailable. Then the recommender offers alternative items one at a time until AUdrey accepts one. Every experiment used the same 100 personae randomly chosen from the database.

The recommenders tested here used either pairwise or set-based similarity. In preparation offline, DRILL calculates the pairwise *word-based* and *circulation-based similarity* between each pair of items, and constructs the item graphs, as described in Section 3. It then uses Foretell, with each book in turn as a seed, to build clusters in the word graph and clusters in the circulation graph. For pairwise similarity, recommenders *NN-w* and *NN-c* offer AUdrey the requested item's nearest neighbors in decreasing order of word-based or circulation-based similarity, respectively. For set-based similarity, we tested several variants of DRILL. They all reference sets that represent books with similar text descriptions, sets that represent books with similar circulation histories, or both.

The experimental parameters are the recommender and how AUdrey decides whether or not to accept a recommendation. We tested three acceptance conditions. Under *accept-by-author* AUdrey accepts only items written by one the current persona's favorite authors, under *accept-by-genre* it accepts those in at least one of its favorite genres, and under *accept-by-history* it accepts only items the persona once chose. Every recommender knows the users' identities, and the items' authors, titles and text descriptions.

Every DRILL recommender uses set-based similarity to assemble *prospects* (plausible recommendations), and then offers them one at a time in decreasing order of pairwise similarity to the requested item. A DRILL recommender is labeled either *w* or *c.* A *w* indicates that its initial prospects are the set of all items that appear in any word-based cluster with the requested item; a *c* indicates that it uses circulation-based clusters for its initial prospects instead. In addition, an *f* indicates that it filters prospects by recorded user preferences: by the persona's author preferences in accept-by-genre experiments, and by the persona's genre preferences in accept-by-author experiments. Once a DRILL recommender exhausts its initial prospects, the default is to offer items from outside the cluster at random, unless a suffix indicates otherwise. A + indicates that, after the initial prospects, additional prospects are those in any cluster with the item most recently chosen by the user but not yet offered. (As necessary, sets of prospects are generated from the next most recently chosen item. If it exhausts the persona's circulation history, DRILL+ will resort to randomly-selected items.) An *n* indicates that, after the initial prospects, additional ones are the most similar, selected as the corresponding NN would. (+*n* behaves first as + and then as *n*.) Depending upon their suffixes, DRILL recommenders know either circulation-based or word-based similarities and clusters, DRILL+ recommenders know circulation history, and DRILL-*f* recommenders know the persona's preferences that are not in the acceptance condition. (Note that accept-by-history is applicable only to DRILL-*w*, because the other recommenders have access to the user's choice history.)

5. Results

The salience of DRILL's clusters is gauged here in two ways. First, we inspect the contents of some clusters to understand their diversity and variation. Second, we report on whether clusters over a large set of items lead to recommendations acceptable to AUdrey.

5.1 Clusters

In the word graph, Foretell found sets of books similar not only to each seed, but similar to one another as well. Under a 20-second per cluster cutoff on 15,749 nodes and nearly 11 million edges, Foretell generated one cluster seeded on each node. The clusters averaged 24.46 items (σ = 2.56). Their edge weights (μ = 0.05, σ = 0.02) were no different from those of the word graph (μ = 0.05, σ = 0.25). These clusters were often cliques or nearly so; cluster density averaged 94.85%, despite 8.65% density in the graph as a whole.

In the circulation graph, Foretell quickly found larger and denser clusters, again with high edge weights. The circulation graph on 15,749 nodes has more than 11 million edges. Within the same 20-second cutoff, Foretell found clusters with mean size 53.18 (σ = 10.05) and density 99.61%, that is, they were almost always cliques. In a graph with mean edge weight 0.26 (σ = 0.10), circulation-based cluster weights were higher 0.28 (σ = 0.10, $p < .001$).

Inspection of individual clusters offers insights into expository text-based descriptions and what might be recommended from them. For example, consider the unusually small cluster (size 14) from the word graph that was seeded on *The Jackie Robinson Reader*. All but one book is about baseball or another sport, but their range is considerable. Some are fiction. Some focus on minority players, others on young ones. A second cluster was seeded on *The Letter*, an award-winning romance novel; its 31 books are all love stories. The cluster includes other classic romance novels (e.g., Steel's *Bittersweet*) and prize-winning, more traditional novels (e.g., Straub's *Mr. X*). Some are dramatic, others are funny, and several are about love from beyond the grave. There are even a few self-help books on how to find love.

Finally, because Foretell is non-deterministic, we examined the consistency with which it formulates clusters. We ran it ten times for 20 seconds each on the word graph, with *Harry Potter and the Order of the Phoenix* as the seed. The resultant ten clusters ranged in size from 26 to 29 and were nearly cliques (density μ = 96.40%, σ = 0.22), again with mean edge weight 0.06 (σ = 0.24). Of the 45 different items that appeared in these ten clusters, 19 appeared in all 10, and 26 appeared in more than five. Furthermore, Spearman's rank correlation coefficient on the edge weights to the seed in these 10 clusters was 0.96. This indicates that, whether or not precisely the same nodes were present, the similarity of nodes to the seed was strongly consistent from one run to the next.

In summary, Foretell's clusters consistently capture sets of items with similarities that are significantly more prevalent than those recorded by pairwise similarity. For circulation-based clusters, these similarities are also above average. Moreover, the contents of these clusters appear to reflect the breadth and coherence of a human expert.

5.2 Dialogues

Acceptance conditions and personae vary in how difficult they are to satisfy. Together they define a *target,* a set of items satisfactory to the user. Our users' preferred genres and authors, however, define targets that vary considerably in size, because some genres are more specific and some authors are better represented. Users' recorded author preferences average 0.40% of all items and never exceed 1.85%, but genre preferences average 33.84% and range as high as 88.94%. Historical preferences represent extensive borrowing over many years; they average 2.18% and range as high as 9.37%.

Table 2 reports *recommendation efficiency,* the average number of offers in a dialogue until AUdrey accepts one. There were, however, considerable discrepancies between users' requests and their recorded preferences. Overall, only 32.24% of all items circulated to our 100 users matched their recorded genre preferences, and 10.97% matched their recorded author preferences. Table 2 therefore distinguishes between *consistent* requests (that match at least one preference in the persona's acceptance criterion) and *inconsistent* ones (that match no preference). (For accept-by-history, all requests are consistent.) To compensate for target size, we also calculated *recommendation precision*, the average number of offers in a dialogue before AUdrey accepts one, weighted by the likelihood that a random recommendation would succeed on each persona. (Data omitted.) All comparisons here are statistically significant at least at $p < 0.05$.

For the smaller targets posed by preferred authors, circulation-based recommenders are more effective than word-based ones, but only when the request is consistent with the persona's recorded preferences. In any other circumstances, word-based clusters are a safe, and often a better, choice. For requests inconsistent with preferred authors, recommendations from DRILL+-*w* and DRILL+*wf* were the most efficient and precise; no others had both qualities. For preferred genres, for all requests, and for inconsistent requests, the word-based recommenders are more efficient and more precise than all the circulation-based recommenders. Indeed the circulation-based recommenders are notably less efficient and less precise than random. For consistent requests accepted by genre, however, many recommenders are more efficient and precise than random, with no significant difference among them. Finally, for requests accepted by history, DRILL-*w* is more

Table 2. Recommendation efficiency during attempts to match the recorded preferences for AUdrey's persona. These values do not consider task difficulty, and smaller values are better. Bold values are better than *Baseline* (the expected number of offers for random recommendations). A shaded value is (equally) best in its column.

	Accept by author			Accept by genre			Accept by history
	All data	Consistent	Inconsistent	All data	Consistent	Inconsistent	All data
Baseline	1201.23	422.96	1395.79	6.37	2.94	13.32	289.35
DRILL-*w*	**737.29**	**27.65**	914.70	8.50	**2.27**	21.15	**178.61**
DRILL-*wf*	966.46	344.05	1122.06	8.79	**2.00**	22.58	—
DRILL-*wn*	**708.21**	**69.20**	867.96	9.59	**2.25**	24.48	—
DRILL+-*w*	**435.83**	**139.40**	**509.94**	12.99	**2.27**	34.76	—
DRILL+-*wf*	**399.10**	**92.45**	**475.76**	12.79	**2.00**	34.70	—
DRILL+-*wn*	**746.36**	**201.70**	882.53	19.36	**2.25**	54.09	—
NN-*w*	**792.04**	**58.60**	975.40	7.59	2.38	18.15	—
DRILL-*c*	779.33	**3.10**	973.39	51.03	2.07	150.42	—
DRILL-*cf*	760.94	**29.05**	943.91	50.36	**1.30**	149.97	—
DRILL-*cn*	**552.75**	**3.10**	**690.16**	57.50	2.07	170.03	—
DRILL+-*c*	**549.30**	**3.10**	**685.85**	60.79	2.07	180.00	—
DRILL+-*cf*	**576.88**	**29.05**	**713.84**	60.00	**1.30**	179.08	—
NN-*c*	**550.98**	**3.45**	**687.87**	76.01	2.04	226.18	—

efficient and more precise than random.

In summary, the efficiency of a recommender depends not only on the size of the target its acceptance condition presents, but also on whether or not the request is consistent with the user's recorded preferences and historical behavior. Unless the target is small and the request consistent, word-based clusters provide the most plausible recommendations.

6. Discussion

Given the results above, how should a recommender proceed? That depends upon the target. If a user is easy to satisfy (a large target, as with accept-by-genre) and presents an inconsistent request, then a random suggestion could suffice — given our users' recorded genre preferences, on average at least one in three offers should be accepted about half the time. If a user's request is consistent for a large target, or if a user is more difficult to satisfy (presents a small target, as with accept-by-author), the recommender should use some variant of DRILL. If the system knows that a particular user's requests are generally consistent (certainly a learnable feature), the recommender should use DRILL-*c*. If, on the other hand, the recommender knows that this user's requests are generally inconsistent, it should use either DRILL-*w* or DRILL+-*w*. Without any indication of consistency, given the prevalence of inconsistent requests and the *c* recommenders' poor performance on them, our data indicates that a word-based DRILL is the best option tested.

Clusters succeed, we believe, because their set-based similarity offers a more cohesive set of prospects. In contrast, the individual items identified by NN are similar to the requested book but not necessarily at all similar to one another. DRILL assembles its prospects based on clusters, which are intended to model the kind of similarity Tversky detected in people (Tversky, 1977). Word-based clusters capture similarities in the way one person would describe a book to another; circulation-based clusters capture similarities with respect to the readers a book attracts. Furthermore, because DRILL uses every cluster in which the requested book appears, it is able to address similarity along multiple dimensions. For example, a humorous sports book would draw from both its genres.

Intuitively, a recommender pursues the dialogue's target, here a set of similar items posed by the persona and the acceptance condition. Clusters model sets of similar items, and as the user's target narrows, DRILL is more effective. Although Foretell rarely gathers together many items by the same author, it assembles sets of items whose text descriptions are a useful indication of their similarity from the perspectives of the original synopses. The removal of named entities often separates highly-related items, such as those in a series, but that relationship would be readily available from the database. Instead, the absence of named entities allows items to cluster in some broader sense. To measure intra-cluster similarity, we compared all pairs of items in each cluster. For author, similarity was 1 if they had the same author, else 0. For genre, we averaged cosine similarity between the boolean genre vectors for each item. Given the relatively few books written by any author ($\mu = 1.95$, $\sigma = 3.01$), average author similarity was expectedly low, about 0.001. By genre, however, average intra-cluster similarity was 0.28. This indicates that items clustered by their text descriptions are somewhat likely to be of the same genre, and are therefore predictive of the user's genre preference, whether or not it is recorded in the database. (We note, however, that genres are notoriously subjective, both in their definition and their assignment.)

Most people want diversity; indeed, Netflix now deliberately fosters diversity in its recommendations (Amatriain & Basilico, 2012). User simulation for a recommender therefore requires variety, implemented here by AUdrey's use of preferences for genre and author from real-world users at Heiskell. The recorded preferences here, however, often had little bearing on actual behavior; work in other domains has confirmed that human behavior elsewhere is similarly inconsistent. This makes evaluation of recommenders difficult.

User preferences are often multi-modal; for example, a user may enjoy both science fiction and cookbooks. To understand how people distribute their choices within large target sets, we examined each user's consistency across her circulation history in two ways. The cosine similarity of the boolean vectors that record the item genre(s) in a chronologically consecutive pair of items chosen by a user chose was 0.41, while the similarity for all pairs of items a user has ever chosen was 0.37 ($p < .01$). Thus users are somewhat more consistent over consecutive choices than in their overall choices. Recall that when DRILL+ exhausts the cluster seeded on the requested item, it gathers prospects from clusters in which the most recently chosen item lies, with recourse to immediately preceding chosen items. This explains why DRILL+ performs less well than DRILL on inconsistent requests accepted by genre. If the user has chosen k items, but the prospects from the kth item are misleading, DRILL+ gathers prospects from the k-1st item, which are more likely to be similar to the prospects from the kth than to the request itself. As a result, prospects from immediately preceding choices are likely to be similarly unacceptable.

There is a delicate balance here between minimum similarity and cluster size. Recall that a cutoff determined which edges item graphs retained. Because edge weights in the circulation graph average 0.28, compared to 0.05 in the word graph, we also tested a sparser graph (cutoff 0.15, density 0.2%) that eliminated additional low-similarity relationships. It consisted of a single connected component plus 1124 isolated nodes, around which there can be no clusters. Foretell formed substantially smaller clusters in this graph ($\mu = 9.17$, $\sigma = 2.66$). Smaller clusters, or no word-based cluster at all (for six of our 100 dialogues' isolated-node requests), provide fewer initial prospects before DRILL resorts to random offers or DRILL+ goes to the circulation history. (Data omitted.)

A recommender should offer a manageable set of plausible alternatives when a request is inherently unsatisfiable (e.g., another book in a completed series) or is satisfied by an intolerably long list of items. DRILL currently creates a sequence of offers too long for human tolerance. We tallied *dialogue success* as the frequency with which AUdrey accepted an offer within a *tolerance threshold t,* the number of offers it received before it would have abandoned the dialogue. Consistent requests were easier to satisfy within $t = 3$ by circulation-based recommenders. Inconsistent ones were rarely satisfied, but more often by genre under DRILL+-*w* and NN-*w*. Expository text under this more stringent requirement remains the best approach.

DRILL-*w*'s ability to predict what a user once chose (accept-by-history) based only on the word-graph cluster for a request means that descriptions of the requested item are similar to descriptions of what the user chose in the past. Although the style and tone of a book could be excellent predictive similarities, those are well beyond the current reach of natural language processing. For example, sentiment analysis, an active area of natural language processing research, relies on specialized lexicons, such as the Dictionary of Affect in Language (DAL) created by a psychologist to enrich the feature representation of text (Whissell, 2009). When DAL's lexical dimensions of positivity, activation, and imagery were applied to text descriptions for similar objects from another domain, however, text descriptions with high DAL similarity were associated with items that human judges considered quite different.

Both the *n* and the *f* recommenders have yet to deliver on their promise. The *n* recommenders were motivated by the high standard deviations that still deny apparently lower values in Table 2 any statistical significance. Because *n* recommenders eliminate random selection after the prospects are exhausted, they were expected to provide a measure of clarity. Instead, on large targets, randomness often proved a better choice. The intent of the *f* recommenders was to exploit the kind of knowledge librarians solicit and use, for example, to consider whether even a prize-winning horror book is a plausible offer for a reader of romance novels. We expect that a larger sample will ultimately support this premise, but for now the evidence remains anecdotal.

Small targets are the most challenging, and that is where a combination of salient features proves most effective. Mere feature similarity based on traditional item attributes must fail when a user exhausts the items with a preferred value (e.g., an author), but word-based similarity can continue to generate plausible prospects. Moreover, when DRILL+ tries first by word-based similarity and then by circulation similarity, its performance under accept-by-author improves for all requests and for inconsistent requests.

DRILL is an ongoing project. The original version of Foretell created disjoint clusters, chose its own seeds, and preferred large, dense clusters — high-weight edges were only a secondary consideration. The new version accepts from its control file other functions that specify how to score a cluster, in what respect VND should be greedy, and when to terminate search on a cluster. We have begun to test versions that emphasize high-weight edges and density more than size during cluster formation, and incorporate additional domain knowledge.

Current work extends DRILL to have more sophisticated interaction with its users, including the ability to discuss the nature of a cluster and perhaps a prototype. Future work will study the impact of a minimum score on cluster size and salience, and cluster on other metrics (e.g., such as Library of Congress subject headings). We intend to investigate synonym identification through WordNet to treat related stems (e.g., "kill" and "murder") as similar tokens. We also plan to evaluate cluster salience with human users, and to integrate other features of items (e.g., author, copyright year, and the named entities thus far extracted but ignored), particularly to order prospects. Other important avenues of exploration are the combination of prospects from clusters seeded on the same item but constructed under different metrics, detection of clusters in a graph whose nodes are users (for which the circulation graph is a prototype), the simulation of human users with more elaborate preference models, and preference elicitation during a dialogue for use with *f* recommenders.

Experimental design in this area raises significant issues about how to construct evaluation metrics for recommendation and how to apply them to both real and simulated users. This work highlights the importance of target size and inconsistent human behavior in evaluation. The role of weak similarities is ill understood, and also requires further study.

7. Conclusion

Clearly, recommenders must contend with users who are not only impatient but also multimodal, idiosyncratic, and often inconsistent with both their own recorded preferences and recent behavior. To address this conflict between recommendation breadth (variety among offers) and efficiency, DRILL casts a wider net — it draws its prospects from multiple clusters to represent multifaceted items and to support users with eclectic preferences. Human experts often recommend similar shifts to users who have exhausted a set of closely related items; DRILL's clusters allow it to simulate such an expert's plausible recommendations.

Human perception of similarity differs significantly from the pairwise distance between feature vectors on which many systems rely (Tversky, 1977). DRILL's novel approach exploits pairwise relationships to detect set-based similarity with a breadth and flexibility that collaborative filtering and content-based recommenders lack. In particular, DRILL's clusters for expository descriptions gleaned from the Web prove surprisingly prescient. We believe that they move toward categories that people perceive because they are based on human-generated text and human behavior. In a domain of substantial scale and complexity, the word-based clusters prove sufficiently varied to satisfy a range of targets presented by simulated users.

The results reported here make clear that more selective users necessitate a more intelligent recommender. In particular, DRILL+ gauges similarity with a combination of word-based and circulation-based knowledge, which better addresses highly-selective users and inconsistent requests. Moreover, DRILL's structured representations could readily support dialogue about similarities among items, based on the word stems that underlie their similarity scores, and provide a rich evaluation platform for cognitive measures of similarity in people. DRILL is a first step toward a system that collaborates with its user to detect and respond to her interests. As ultimately envisioned, DRILL is set-based, interactive, proactive, and self-aware.

This research moves toward the autonomous construction of metaknowledge derived from word-based similarity of expository text. As such, it is applicable to any domain with sufficient expository text that describes items of interest, including movies, music, and vendors' catalog items. It does not require ratings. Its clustering is not limited to similarity as represented by text or historical choice; it can be applied to other item graphs as well. Meanwhile, curated, clustered text descriptions for database items, particularly when combined with a history of choices, here prove a useful indicator of the preferences of (simulated) human users.

Acknowledgements

This research was supported in part by The National Science Foundation under awards IIS-084966, IIS-0745369, and IIS-0744904. Ben Hixon worked on this project while a student at Hunter College. We thank the many librarians who so generously shared their expertise, and Sofia Grayevsky, Peter Valdez, and Xi Yun for assistance with Foretell.

References

Amatriain, X., & Basilico, J. (2012). Netflix recommendations: Beyond the 5 stars (Part 1). Retrieved 11/15/2012, from http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html.

Barnes, & Noble. (2012), Retrieved 11/15/2012, from http://www.barnesandnoble.com/.

Bell, R. M., & Koren, Y. (2007). Scalable collaborative filtering with jointly derived neighborhood interpolation weights. *Proceedings of the Seventh IEEE International Conference on Data Mining* (pp. 43-52). IEEE Conference Publications.

Berkovsky, S., Kuflik, T., & Ricci, F. (2008). Mediation of user models for enhanced personalization in recommender systems. *User Modeling and User-adapted Interaction*, *18*, 245-286.

Bizer, C., & Gaub, T. (2012). Freie University's RDF book mashup, Retrieved 11/15/2012, from http://www4.wiwiss.fu-berlin.de/bizer/bookmashup/)

Eckert, W., Levin, E., & Pieraccini, R. (1997). User modeling for spoken dialogue system evaluation. *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding* (pp. 80-87). IEEE Conference Publications.

Epstein, S. L., Li, X., Valdez, P., Grayevsky, S., Osisek, E., Yun, X., & Xie, L. (2012). Discovering protein clusters *Proceedings of the AAAI Fall Symposium on Discovery Informatics: The Role of AI Research in Innovating Scientific Processes* (pp. 21-28). AAAI Press.

Epstein, S. L., & Wallace, R. J. (2006). Finding crucial subproblems to focus global search. *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence* (pp. 151-159). IEEE Conference Publications.

Eshky, A., Allison, B., & Steedman, M. (2012). Generative goal-driven user simulation for dialog management. *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (pp. 71-81). Association for Computational Linguistics.

Finkel, J. R., Grenager, T., & Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. (pp. 363-370). Association for Computational Linguistics.

Georgila, K., Henderson, J., & Lemon, O. (2006). User simulation for spoken dialogue systems: Learning and evaluation. *Proceedings of the Ninth International Conference on Spoken Language Processing* (pp. 1065-1068). Curran Associates, Inc.

Gigerenzer, G., & Goldstein, D. G. (1999). Betting on one good reason: The take the best heuristic. In G. Gigerenzer, P. M. Todd & A. R. Group (Eds.), *Simple heuristics that make us smart.* New York: Oxford University Press.

Hansen, P., Mladenovic, N., & Urosevic, D. (2004). Variable neighborhood search for the maximum clique. *Discrete Applied Mathematics*, *145*, 117-125.

Hu, J., Wang, G., Lochovsky, F., Sun, J.-t., & Chen, Z. (2009). Understanding user's query intent with Wikipedia. *Proceedings of the Eighteenth International Conference on the World Wide Web* (pp. 471-480). Association for Computing Machinery.

Huang, Z., Chung, W., & Chen, H. (2004). A graph model for e-commerce recommender systems. *Journal of the American Society for Information Science and Technology*, *55*, 259-274.

Huang, Z., Chung, W., Ong, T.-H., & Chen, H. (2002). A graph-based recommender system for digital library. *Proceedings of the Joint Conference on Digital Libraries* (pp. 65-73). Association for Computing Machinery.

Jung, S., Lee, C., Kim, K., Jeong, M., & Lee, G. G. (2009). Data-driven user simulation for automated evaluation of spoken dialog systems. *Computer Speech and Language, 23*, 479-509.

Jung, S., Lee, C., Kim, K., Lee, D., & Lee, G. G. (2011). Hybrid user intention modeling to diversify dialog simulations. *Computer Speech & Language, 25,* 307-326.

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

Maanen, L. v., & Marewski, J. N. (2009). Recommender systems for literature selection: A competition between decision making and memory models. *Proceedings of the 31st Annual Meeting of the Cognitive Science Society*. Cognitive Science Society.

Middleton, S. E., Shadbolt, N. R., & Roure, D. C. D. (2004). Ontological user profiling in recommender systems. *ACM Transactions on Information Systems, 22*, 54-88.

Pantel, P., Lin, T., & Gamon, M. (2012). Mining entity types from query logs via user intent modeling. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics* (pp. 563-571).

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*, 130−137.

Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A., & Riedl, J. (2002). Getting to know you: Learning new user preferences in recommender systems. *Proceedings of the Seventh International Conference on Intelligent User Interfaces* (pp. 127-134). Association for Computing Machinery.

Sarwar, B., Karypis, G., Konstan, j., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the Tenth International Conference on the World Wide Web* (pp. 285-295). Association for Computing Machinery.

Schatzmann, J., Georgila, K., & Young, S. (2005). Quantitative evaluation of user simulation techniques for spoken dialogue systems. *Proceedings of the SIGDial Workshop' 05* (pp. 45–54). Association for Computational Linguistics.

Thompson, C. A., Göker, M. H., & Langley, P. (2004). A personalized system for conversational recommendations. *Journal of Artificial Intelligence Research, 21*, 393-428.

Tversky, A. (1977). Features of similarity. *Psychological Review, 84,* 327-352.

Whissell, C. (2009). Using the revised dictionary of affect in language to quantify the emotional undertones of samples of natural language. *Psychological Reports, 105,* 1-13.

White, R. W., Bailey, P., & Chen, L. (2009). Predicting user interests from contextual information. *Proceedings of the 32nd international ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 363-370). Association for Computing Machinery.

Wierzbicka, A. (1985). *Lexicography and conceptual analysis.* Ann Arbor, MI: Karoma Publishers.

Wilcock, G. (2003). Integrating natural language generation with xml web technology. *Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics, Conference Companion* (pp. 247-250). Association for Computational Linguistics.

Zhao, Y., & Karypis, G. (2005). Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery, 10,* 141-168.