Learning to Recognize Valuable Tags

Shilad Sen
Grouplens Research
University of Minnesota
ssen@cs.umn.edu

Jesse Vig
Grouplens Research
University of Minnesota
jvig@cs.umn.edu

John Riedl
Grouplens Research
University of Minnesota
riedl@cs.umn.edu

ABSTRACT

Many websites use tags as a mechanism for improving item metadata through collective user effort. Users of tagging systems often apply far more tags to an item than a system can display. These tags can range in quality from tags that capture a key facet of an item, to those that are subjective, irrelevant, or misleading. In this paper we explore tag selection algorithms that choose the tags that sites display. Based on 225,000 ratings and survey responses, we conduct offline analyses of 21 tag selection algorithms. We select the three best performing algorithms from our offline analysis, and deploy them live on the MovieLens website to 5,695 users for three months. Based on our results, we offer tagging system designers advice about tag selection algorithms.

ACM Classification Keywords

H.5.3 Information Interfaces and Presentation: Group and Organization Interfaces—*Collaborative computing*; H.5.2 Information Interfaces and Presentation: User Interfaces

General Terms

Design, Experimentation, Human Factors

Author Keywords

tagging, moderation, user interfaces

INTRODUCTION

Tags empower users to adapt the information space of their communities. Since tags can be created by any user, the number of tag contributions in a community scales with its size. Moreover, a system's vocabulary of tags evolves organically to reflect information concepts that taggers find most important. These advantages may not be without costs. Tagging systems may lack the quality control of expert-maintained content, and they may contribute to users' overall information overload. For example, Amazon.com[1] users have applied 705 distinct tags to the book "Liberal Fascism: The Secret History of the American Left" by Jonah Goldberg ranging from *conservative* to *editor promised cake*.

Since Amazon's screen space is limited, they only select twelve tags to display from among the 705 distinct tags . The twelve tags Amazon selects not only influence users' impressions of the book, they also influence users' perception of a community's tagging norms. This self-reinforcing cycle may explain why the most applied tags for "Liberal Faciscm" differ substantially between Amazon.com and the personal library cataloging site LibraryThing[2]. On Amazon the three most popular tags for the book are editorial tags that support liberal viewpoints: *wingnut welfare* (applied by 378 users), *I can has job mom* (285), and *editor promised cake* (229). On LibraryThing the top three top tags are *politics* (27), *history* (18), and *fascism* (9)[3]. After seeing these subjective tags, Amazon users may be more inclined to tag subjectively themselves. Since users generally prefer factual tags to subjective tags, over time this self-reinforcing cycle may worsen Amazon's tags and improve LibraryThing's tags [19].

Many tagging sites, including Amazon, Delicious[4], and LibraryThing use a simple method to order an item's tags: select the tags applied to the item most frequently. This intuitive method, which we call *num-item-apps*, is based on the assumption that the number of people who have added a tag to an item is a good estimate for how much other people will like to see that tag in the future. We call the method a tagging system uses to select and order the tags it displays its *tag selection algorithm*.

Despite the popularity of num-item-apps, other tag selection algorithms may also be effective. For example, if people search for high-value tags more often than low-value ones, systems that take into account the number of searches for a tag may improve upon num-item-apps. Moreover, num-item-apps does not apply to all domains. While a tag is a particular word or phrase, we define a *tag application* as an association between a tag and an item. Sites such as Amazon, Delicious, and LibraryThing that support individual tag applications enable users to add their own copy of a tag to an item. On the other hand, sites such as Flickr and YouTube that support shared tag applications prevent users from re-adding a tag that was already applied to an item by somebody else. Since each tag is applied to an item only once, a site that supports shared tag applications cannot use num-item-apps.

The design of tag selection algorithms is important for two reasons. First, tagging systems can often only display a small fraction of all the tags applied by users due to limited screen space. Users who are shown good tags are likely to be better informed and more satisfied. For example, Delicious users have applied over 1,000 distinct tags describing Amazon.com. Should Amazon be *shopping*, *myfav*, or *GRDE226*. Second tag selection algorithms can promote community norms. In the case of Goldberg's book, Amazon's tag selection algorithm featured and therefore promoted subjective tags. Although Amazon's example encourages poor tagging behavior, system designers should see this as an opportunity. Carefully engineered tag selection algorithms can create and promote positive community norms.

[1] http://www.amazon.com

[2] http://www.librarything.com
[3] as of July 20th, 2008
[4] http:/del.icio.us

IUI'09, February 8 - 11, 2009, Sanibel Island, Florida, USA.

In this paper we explore novel tag selection algorithms for displaying high-value tags and hiding low-value tags. The same tag may have different value for different users. For example, *my netflix queue* may be high-value to the user who applied the tag (it helps them organize their movies), but low-value for everybody else. We define a tag as high-value for a particular user and item if the user wants to see the tag alongside the item.

We structure our exploration of tag selection algorithms around four research questions:

RQ1: Which metrics should researchers use when evaluating tag selection algorithms? We begin our analyses by conducting an offline evaluation of tag selection algorithms based on a user survey about tag value. In these analyses, we simulate the behavior of tag selection algorithms for the survey responses. We distill the results from each algorithm to a single score using a *metric*. We explore several metrics to develop a tool for analytically comparing the effectiveness of different tag selection algorithms.

RQ2: How well do tag selection algorithms based on implicit user behavior perform? Ideally, tag selection algorithms would draw on natural behavior already present in tagging communities such as tag application, tag browsing, and tag searching. We explore the effectiveness of tag selection algorithms drawing on these *implict* signals of tag value.

RQ3: How well do tag selection algorithms based on explicit behavior perform? Many community maintained sites such as YouTube[5] and Digg[6] enable users to moderate contributions through explicit ratings. Inspired by these sites, we evaluate explicit moderation of tags using thumbs up and down ratings. Although these algorithms require more user effort than implicit algorithms, they may lead to improvements in tag selection algorithms that justify the extra user effort. We examine whether explicit ratings-based moderation mechanisms improve tag selection algorithms.

RQ4: How does the performance of each class of tag selection algorithms change as tag density increases? We evaluate our research questions using the MovieLens movie recommendation website. On a weekly basis, about 1,200 users visit MovieLens and create about 400 new tag applications. MovieLens users generate far less tagging activity than a site such as Delicious, where users generate 400 tag applications in a matter of minutes [7] Implicit tag selection algorithms may perform better on sites with higher tag density due to their reliance on user tagging activity. We explore the relationship between the performance of tag selection algorithms and the tag density of a site.

Finally, we deploy the best offline performer from each class of tag selection algorithm live on MovieLens. We evaluate users' reactions to each tag selection algorithm to see if the offline results reflect actual user preference in the live system.

Our work builds on our early tagging research exploring how and why tagging vocabularies evolve [20]. We also extend our more recent work on interface mechanisms for eliciting feedback about tags, and intuitive implicit and explicit signals of tag quality [19] in four key ways:

1. We explore novel metrics for offline analyses of tag selection algorithms.
2. We evaluate 17 implicit and explicit signals of tag value, including 10 novel signals.
3. We evaluate tag selection algorithms that use machine learning techniques to combine multiple signals of tag value.
4. We deploy the top performing tag selection algorithms from our offline analyses in a user-study.

RELATED WORK

Although public bookmarking systems such as Fab [3], Knowledge Pump [11], and Pharos [5] have been available since the 1990's, Millen et al. point to tagging as a key reason current social bookmarking systems have enjoyed greater success [18]. Our work furthers early studies of tagging communities by analyzing the value of tags created in an online community.

In early academic research on tagging communities, MacGregor and McCullogh [16] explore the relative merits of controlled versus evolved vocabularies, arguing that evolved ontologies engage users but lack the precision of their controlled counterparts. Golder and Huberman indicate that the proportions of tags applied to a given item in Delicious appear to stabilize over time, and suggest that community members may be influenced by what they see [12]. Cattuto furthers their work by presenting a generative model for users' tagging that predicts the rate at which both particular users and entire communities re-use tags [7]. Ames et al. study users' motivations for tagging, and find that users tag for both organizational and social purposes [1]. In earlier work, we show that the tags a user sees influence the tags they create themselves [20]. We also classify tags as generally factual, subjective, or personal (intended for the tag creator themselves), and find that users generally prefer factual tags over subjective tags and strongly dislike personal tags. Our research extends earlier work describing how and why users choose tags to the novel problem of how systems might select tags to show a user from among a large collection of tags that have been applied by other users.

Several researchers have studied moderation in online communities. Cosley et al. find that "Wiki-like" systems that immediately display user contributions lead to more contribution than systems that require members to review contributions before they are displayed [8]. In other work, Cosley et al. show that intelligent task routing can be used to help users find tasks they might complete to improve the system [9]. Lampe and Resnick analyze the moderation system utilized on the online forum slashdot[8] [15]. They find that although the community perceives that forum moderations are generally fair, comments that are assigned low scores, or posted late in a conversation are often overlooked by moderators. Both Arnt and Zilberstein and Weimer et al. explore machine learning techniques for predicting moderation scores in online forums [2] [17]. Our research differs from the general work on moderation of contributions in that we focus on a type of contribution (tags) and test our results using a user-study.

[5]htttp://www.youtube.com
[6]htttp://www.digg.com
[7]Based on a recent examination of timestamps in Delicious's tag stream.
[8]http://www.slashdot.org

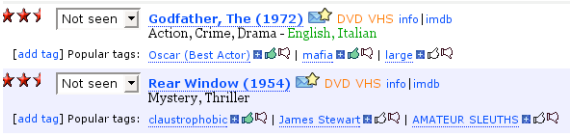

Fig. 1: *Tags on the MovieLens search results screen.*

Fig. 2: *Tags on the MovieLens movie details screen.*

In earlier work we explore user interfaces that help systems determine a tag's value [19]. In offline results we find that thumb rating feedback significantly improves a system's ability to identify good tags compared to simple implicit signals of tag quality. Our research extends this earlier work by exploring metrics for evaluating selection algorithms, testing ten new signals of tag quality , incorporating principled machine learning techniques, and conducting a user study to test offline results.

EXPERIMENTAL PLATFORM

As an experimental platform, we used the MovieLens movie recommender website. MovieLens members can tag movies, and use tags contributed by others in the community to find and evaluate movies. MovieLens users most commonly interact with tags through the search results screen and movie details screens. Figure 1 shows the search results screen, which lists movies matching a search query along with up to three community tags for every movie. Figure 2 shows the movie details screen, which provides detailed movie information including up to 30 of a movie's tags. Since we introduced tagging features to MovieLens in January 2006, MovieLens users have created 84,155 tag applications resulting in 13,558 distinct tags. 3582 users have applied at least one tag (12.1% of active users during the time period). Further details of MovieLens and the MovieLens tagging system can be found in [20].

In order to study explicit tag feedback, we introduced thumbs up and thumbs down tag ratings to the MovieLens community. The thumbs up and down rating icon appears next to all tags in the MovieLens system (fig.2). Since we introduced tag ratings in January 2007, 2105 users have contributed 150,031 ratings. 10,932 tags have been rated at least once (81% of all tags), and 4608 tags have been rated five or more times (34%). Further details of tag ratings are described in [19]. For brevity we refer to thumbs up and down tag ratings as "thumb ratings" in the rest of this paper.

OFFLINE EVALUATION

Experimental Methods

Although we believe that many users apply thumb ratings to indicate tag value, we wanted to collected a "gold-standard" set of tag value ratings for use in our analyses. To collect this dataset, we emailed 2,531 active MovieLens users and asked them to complete an online survey in which they provide feedback on tag quality. Users were asked to rate up to twenty tags applied to five movies on a five star scale.

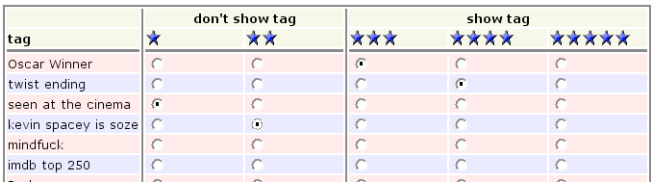

Fig. 3: *We asked users to rate tags for five movies on a one-to-five scale. We instructed them that MovieLens would only choose to show them tags rated 3, 4, or 5 stars.*

Figure 3 shows an example screen from the survey. As a point of reference, users were instructed that MovieLens would only display tags rated 3, 4, and 5 stars. 577 users responded to the survey (22.8% response rate) and rated at least one tag application. 546 users rated tags for all five movies. We gave users the option of continuing to rate tags after they completed rating tags for their first five movies. Users provided 74,987 one-to-five ratings. Two users provided more than 1,000 ratings, while 253 users provided 100 or more ratings. Users deemed 38% of rated tag applications worthy of display. The average tag rating was 2.17.

We considered a variety of machine learning classifiers as tag selection algorithms. Each classifier used features of a particular tag, user, and movie as input (e.g. the number of users who have applied a tag), and outputted the predicted probability that a tag is high-value. We experimented with a number of different classification methods, including decision trees, logistic regression, and neural networks using the WEKA machine learning toolkit. We found that support vector machines (SVMs) outperformed other classification methods.

Because SVMs are known to have difficulty with features with differing numerical scales, we log-transformed the features that best fit a log-normal distribution[9] Next, we z-score normalized all features and clipped values more than three standard deviations from the mean [10] We experimented with a number of different SVM kernels and implementations, and chose svm_perf [14] based on its linear kernel optimizations and diverse optimization criteria. We found that ROC area and top-n% optimization criteria outperformed the standard zero-one loss function. We found svm_perf to be somewhat to the choice of c, the tradeoff between maximizing the margin and minimizing errors, with $c = 50.0$ performing optimally. We use five-fold cross validation in all offline analyses.

Metrics

We experimented with three metrics that distill tag selection algorithm performance to a single score. Each metric evaluates whether a tag selection algorithm correctly classifies a survey rating as low value (two stars or less), or high value (three stars or more). This mapping from stars to value was marked on the key presented to users in the survey. In this section we explore:

[9]We ran a chi-squared test for a normal distribution before and after log transforming each feature. We transformed the seven features whose χ^2 improved five times or more after the transformation: tag-length, apps-per-movie, num-item-apps, num-apps, num-searches, num-users, and num-search-users.

[10]Less than 1% of values were clipped for most features. The highest level of clipping for any feature affected 2.5% of its values.

RQ1: Which metrics should researchers use when evaluating tag selection algorithms?

Herlocker et al place metrics for recommender systems into two categories: accuracy-based metrics, and precision-based metrics [13]. Inspired by this work, we explore the effectiveness of these metrics for tagging systems. *Classification accuracy* metrics examine the fraction of responses correctly classified as low or high value. *Precision* metrics examine the survey responses ranked highest by an algorithm, and measure the fraction that have high value. We consider one classification accuracy metric and two precision metrics:

Classification Accuracy (*class-acc*) measures the percentage of survey ratings that a tag selection algorithm classifies correctly as hide or display. Class-acc weighs each tag as equally important, regardless of whether the tag would be displayed by the selection algorithm or not.

Simulated Precision of an Item's Top-n (*item-top-n*) focuses on those tags a tag selection algorithm is most likely to display. Item-top-n groups survey responses by item and user, and then selects the most highly predicted n tags for each (user, item) pair. Item-top-n's score measures the percentage of the top n tags that are have high value. We set n to be three based on the number of tags MovieLens displays for a movie on the search result screen [11].

Precision of the Top-n% (*overall-top-n%*) is a relaxation of item-top-n. We wondered whether a simplified version of item-top-n would yield similar results. Instead of grouping results by item, overall-top-n% orders all survey ratings by their predicted value regardless of item, and measures the percentage of the top-ranked n% that have high-value. Based on the percentage of an item's tags typically displayed in MovieLens [12], we set n to be 25%.

For each of the 21 tag selection algorithm presented in the next section of this paper we calculate results using all three quality metrics. To compare metrics, we evaluated the Pearson correlation (similarity of numerical scores) and Spearman correlation (similarity of rankings) for the 21 scores between each pair of metrics. Although all three merics ranked tag selection algorithm performance similarly, the actual numerical scores produced by the metrics differed significantly (Fisher's z-test $p \leq 0.05$). Overall-top-25% and item-top-3 yielded a Pearson correlation of 0.98 while the two other pairs yielded 0.89 and 0.85. While the Spearman rank correlation between overall-top-25% and item-top3 was highest (0.99), the difference between it and the other metric pairs were less noticeable (0.96 and 0.97).

Since item-top-3 closely simulates actual tagging system behavior, we found its numerical values to be easily interpretable. An item-top-3 score of 0.65 indicates that roughly two out of the top three tags displayed for a movie are deemed displayable. We can similarly interpret overall-top-25%'s numerical scores based on its strong correlation with item-top-3. On the other hand, class-acc scores were quite difficult to interpret due to its weightings of all tags equally. A naive predictor that classifies all responses as low-value yields a classification accuracy of 61%, while the best tag selection algorithm we consider in this paper yields 70% classification accuracy. It seems difficult to translate classification accuracy into expected system behavior.

[11]Search results account for 95% of all tag views

[12]About 25% of the possible tag applications are displayed.

	rankings	scores	tuning	effort
class-acc	high	low	low	low
overall-top-25%	high	high	medium	low
item-top-3	high	high	low	medium

Table 1: *Qualititive Goodness of Different Metrics. The columns list the a metric's quality of relative rankings for tag selection algorithms, the quality of numerical scores for algorithms, the difficulty of parameter tuning, and the difficulty of implementing the metric.*

Both overall-top-25% item-top-3 have disadvantages. Item-top-3 requires slightly more implementation effort due to its increased complexity. The choice of n in overall-top-n% can be difficult. We set it to be the fraction of all tag applications that would be displayed. Although this worked well for us, it is difficult to know whether this approach will succeed in other domains. Table 1 summarizes the strengths and weaknesses of each metric. Despite its increased implementation difficulty, we favor item-top-n due to its interpretability and ease of parameter selection.

In the following section describing our offline analyses we report item-top-3 results. We do not report overall-top-25% due to its similarity to item-top-3, nor do we report class-acc due to the difficulty in interpreting numerical scores.

Tag Selection Features

Tag selection algorithms determine which tags to show based on certain criteria. For example, num-item-apps orders tags by the number of times they have been applied to an item. Although this simple algorithm serves as a defacto industry standard, tag selection algorithms may order tags based on other criteria such as the number of times the tag has been applied across all items (num-apps) or the number of users who have searched for a tag (num-search-users). We call these criteria *features*, and in this section we describe the features we evaluate in this paper. In support of RQ2 and RQ3, we group features by the "class" of tag selection algorithm they correspond to: implicit or explicit.

When describing each feature, we label each feature with its specificity. Some features such as num-search-users are broadly applicable to all instances of a tag, returning identical results regardless of the item to which they are applied. We describe this broad level of specificity as *per-tag*. Other features, such as the num-item-apps, return different results for each (tag, item) pair. We describe the specificity of these narrower features as *per-item-tag*. Finally, we annotate features that are user-specific, such as a particular user's thumb rating as *per-tag-user* or *per-item-tag-user*. Narrow features capture a more specific usage of a particular tag (e.g. for a particular user, item or both), potentially offering a more precise signal of tag quality to tag selection algorithms. However, narrow features require a more specific behavior (e.g. a rating by a particular user instead of a rating by any user), and consequently need more activity to achieve the same level of coverage as broader features.

Implicit Features

Implicit features may improve the quality of displayed tags without additional user effort or interface modifications. Table 2 describes all 11 implicit tag selection features. We include two baselines: the defacto industry standard num-item-apps algorithm, and a random tag selection algorithm that arbitrarily orders tags. When presenting formulas for the features in Table 2 we use the following terminology: M is the collection of all movies, U is the collection of all users,

Table 2: *Description of features for tag selection. Implicit features are on the top half of the table and explicit features are on the bottom half.*

feature	specificity	motivating hypothesis	description	formula	top 5 results
random	n/a	n/a	Randomly orders tags.	$random()$	n/a
num-item-apps	per-item-tag	Tags applied to a particular item by more users are more relevant.	Orders tags by the number of times they have been applied to a particular item.	$\|A_{m,t}\|$	*prison* (The Shawshank Redemption), *quentin tarantino* (Pulp Fiction), *anime* (Spirited Away), *time travel* (Twelve Monkeys), *holocaust* (Schindler's List)
num-apps	per-tag	Tags applied more times overall across items are more relevant.	Orders tags by the number of times they have been applied across all items.	$\|A_t\|$	*classic, tumey's dvds, less than 300 ratings, 70mm, r*
num-users	per-tag	Tags applied overall across items by more users are more relevant.	Orders tags by the number of users who have applied the tag across all items.	$\|\{user(ta) : ta \in A_t\}\|$	*classic, comedy, action, sci-fi, fantasy, funny*
num-searches	per-tag	Tags searched for more times are more relevant.	Orders tags by the number of searches for the tag.	$\|S_t\|$	*comedy, classic, oscar (best picture), seen more than once, action*
num-search-users	per-tag	Tags searched for by more users are more relevant.	Orders tags by the number of users who searched for the tag.	$\|\{user(ts) : ts \in S_t\}\|$	*comedy, oscar, classic, nudity (topless), sci-fi.*
tag-share	per-item-tag	Tags that account for a larger fraction of an item's tag applications are more relevant.	Orders tags by the fraction of the item's tags applications that are for the tag.	$\frac{\|A_{t,m}\|}{\|A_m\|}$	*anime* (Perfect Blue), *why the terrorists hate us* (Bratz: The Movie), *serial killer* (Copycat), *jane austen* (Persuasion), *tom hanks* (The Money Pit)
avg-fraction-items-tagged	per-tag	Tags whose creators apply the tag many times are more likely to be list-making tags, and less relevant for the community as a whole.	Orders by the average, across all users, of the fraction of all the items tagged by the user that have the tag.	$mean_{u \in U} \frac{\|A_{t,u}\|}{\|A_u\|}$	*dvdhyllssa, dvd, tumey's dvds own, oppl, sven's to see list*
apps-per-item	per-tag	Tags applied more often to the items to which they are applied are more relevant.	Orders tags by the average number of times they are applied to their items.	$\frac{\|A_t\|}{\|\{item(ta):ta \in A_t\}\|}$	*pixar, mozart, johnny cash, quentin tarantino, monty python*
num-tag-words	per-tag	Tags with many words are less desirable.	Orders tags by the number of words in them.	$num\_words(t)$	5% of all tags are one word, 38% are two words, 11% are three words, and 13% are four or more words. Mini-reviews or comments account for many of the longest tags.
tag-length	per-tag	Tags with very few letters are less desirable.	Orders tags by the number of letters in them.	$num\_characters(t)$	The five most used tags with at most three letters: *r, dvd, own, war, vhs*
feature	**specificity**	**motivating hypothesis**	**description**	**formula**	**top 5 results**
global-avg	per-tag	Tags that are rated more highly have higher value.	Orders tags by the bayesian estimate of the fraction of thumbs up ratings.	$bayes\_frac\_up(R_t)$	*addiction, submarine, golden palm, superhero, james bond*
global-app-avg	per-tag-item	Specific applications of tags to a movie that are rated highly have higher value.	Orders tag applications by the bayesian estimate of the fraction of thumbs up ratings.	$bayes\_frac\_up(R_{t,m})$	*mafia* (The Godfather), *pixar* (Toy Story), *james bond* (Casino Royale), *time travel* (Back to the Future), *pixar* (Ratatouille)
user-average	per-tag-user	Users that rate a tag highly value that tag.	Orders tags by the bayesian estimate of the user's fraction of thumbs up ratings for the tag.	$bayes\_frac\_up(R_{t,u})$	not applicable (user specific)
user-rating	per-tag-item-user	Users that rate a tag application thumbs up value that tag application.	Orders tag applications by the user's rating for that specific application.	$R_{t,u,m}$	not applicable (user specific)
normalized-global-avg	per-tag	Tags searched for by more users are more relevant.	Orders tags by the number of users who searched for the tag.	$mean\left(\sum_{u \in U} bayes\_frac\_up(T_{u,m})\right)$	*superhero, studio ghibli, submarine, roat trip, martial arts*
reputation	per-tag	Tags applied by users who create highly rated tags will have high value.	See description in text.	See description in text.	*dinosaurs, serial killer, martial arts, space, archeology*
hierarchical-value	per-item-tag	See description in text.	See description in text.	See description in text.	*rome* (Gladiator), *dinosaur* (Jurassic Park), *adolf hitler* (Downfall - Der untergana), *genocide* (Hotel Rwanda), *excellent script* (Juno)

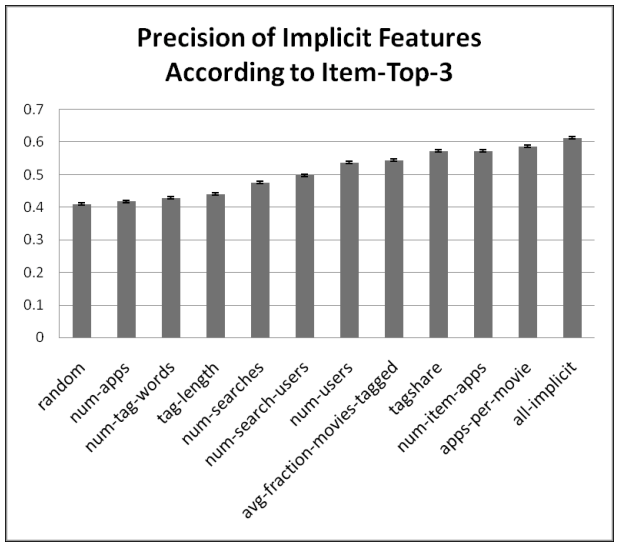

Fig. 4: *The performance of implicit features according to item-top-3. Differences between all neighboring pairs are statistically significant at the 0.05 level except for num-item-apps and tagshare, which achieve equal scores.*

and T is the collection of all tags. A is the collection of all tag applications. $user(ta)$, $tag(ta)$, and $movie(ta)$ denote the user, tag, and movie associated with a tag application. As shorthand notation, we denote the subset of A applied to a particular movie $A_u = \{ta \in A : movie(ta) = m\}$ as A_m. We similarly note applications applied by a particular user, and for a particular tag as A_u and A_t. We describe the set of all tag searches as S with similar subset notation. Below, we describe some additional details of the features.

The tagshare of a particular tag for a movie is the fraction of a movie's tag applications that are for the particular tag. For example, in MovieLens the movie "The Visitor" has been tagged with *immigrants* (by 2 users), *New York City* (1), *PG13* (1), and *R* (1). Since there are a total of 5 tag applications, *immigrant*'s tagshare would be $2/5 = 0.4$, while *R* would have a tagshare of 0.2. Additionally, we smoothed each tagshare value by adding a constant of .0465 to the numerator and 5.0 to the denominator. We determined these constants by using an empirical Bayes methodology[13][6]. After smoothing, tagshare for *immigrant* and *R* would be $2.465/10 = 0.2465$ and $1.465/10 = 0.1465$ respectively. This smoothing reflects the possibility that the two initial applications of immigrants may be due to chance.

Avg-fraction-movies-tagged measures the average, across all users, of the fraction of all the movies tagged by the user that have the tag. For example, if a user created tag applications dog (applied to movies 1,2), cat (movies 2 and 3), and mouse (movie 4), the fraction of the user's tagged movies that are tagged with dog are $2/4 = 0.5$. We smoothed this value by adding a constant of 0.6 to the numerator, and 5.0 to the denominator (we determined these constants using an empirical Bayes methodology [6]). After smoothing the value of dog for our hypothetical user is (2+0.6) / (4+5.0) = 0.29.

Figure 4 compares the item-top-3 performance of tag selection algorithms based on each of the implicit features. 95% Confidence intervals are ±.0036 (n=74987, $p \leq 0.05$)[14]. Differences between all neighboring pairs are statistically significant at the 0.05 level except for num-item-apps and tagshare, which achieve equal scores. All features outperform the random baseline (0.41). Num-item-apps, the most common feature among real-world systems, performs well, outperforming all features except for apps-per-movie.

Some features were skewed by a small number of users. For example, since num-apps orders tags by the number of applications, it predicts equal value for a tag that is used 3 times by 10 users and one that is used 30 times by one user. Because a few users can heavily influence num-apps, two personal list-making tags rank among the 10 most often applied tags: *tumey's dvds* and *less than 300 ratings*. These personal tags are generally disliked by people other than their creator[20]. A system can reduce the number of personal tags by normalizing each influence over num-apps. A simple method for normalization is to count the number of users who apply a tag instead of the number of applications. Likely due to this normalization num-users and num-search-users outperform their non-normalized counterparts num-apps and num-searches .537 to .418 and .498 to .475 respectively according to item-top-3.

Figure 4 also includes a tag selection algorithm based on a combination of all implicit features (all-implicit). We constructed the all-implicit algorithm identically to the single feature tag selection algorithms, but used multiple features as input into the SVM classifier. We found that a combination of five features (num-tag-users, num-item-apps, tag-words, tag-length, average-fraction-movies-tagged, apps-per-item) outperformed any single feature, achieving an item-top-3 score of 0.613. We determined the five top features by first selecting the top performing single feature, and iteratively adding the feature that most improved the currently set of best performers. Adding more than five features did not provide a significant improvement.

We can now answer:

RQ2: How well do tag selection algorithms based on implicit user behavior perform?

To summarize our findings: num-item-apps, the feature most popular among real-world system designers, performs well among the implicit features. Apps-per-movie performs best among all implicit features, with a slight but significant edge over num-item-apps. The top performing tag selection algorithm is all-implicit, the combination of implicit features. We will return to RQ2 in the following section to study our algorithms in a live community.

Explicit Features

We now shift our focus to explore explicit features based on the thumbs up and thumbs down ratings. When presenting formulas for the explicit features we extend the terminology from the previous section. R represents the collection of all thumb ratings for tags. As shorthand notation, we denote the subset of R applied to a particular movie $\{tr \in R : movie(tr) = m\}$ as R_m. We similarly note applications applied by a particular user, and for a particular tag as R_u and R_t.

[13]We treated the fraction of a movie's tag applications that are a particular tag as a bernoulli variable with a beta conjugate prior.

[14]95% confidence intervals range from ±.0036 for scores near .5, ±.0034 for those scores furthest away from .5

We use $bayes\_frac\_up()$ to refer to a bayesian estimate of the fraction of a collection of ratings that are thumbs up. We found it important to use a bayesian estimate to adjust for collections with few ratings. Ideally, instead of using a bayesian approach we would collect a a large collection of ratings for a particular entity (i.e. the tag *funny*). We call the result we would get by collection a very large number of ratings a tag's "true" fraction of thumbs up ratings. Unfortunately, the MovieLens user base is too small to collect this much data. Suppose instead that we have only received one thumb rating for *funny* and it was positive. We could naively estimate that the true fraction of thumbs up was 100%. However, based on our experience in MovieLens, that seems very unlikely. The bayesian paradigm provides a method for analytically encoding this uncertainty. In order to calculate the bayesian estimate, we treat the thumbs up and down ratings as a Bernoulli random variable with a beta conjugate prior [10]. Using a gamma distribution for hyperparameters[15] we selected the beta conjugate prior most likely to produce the data. We repeated this procedure for four groupings of thumb data: by tag ($\alpha = 0.54$, $\beta = 1.74$), by user ($\alpha = 0.76$, $\beta = 2.15$), by user-tag ($\alpha = 0.27$, $\beta = 0.96$), and by movie-tag ($\alpha = 0.41$, $\beta = 1.42$). As an example, suppose a particular user has rated the tag *zombies* thumbs up three times and thumbs down once. We use the user-tag parameters ($\alpha = 0.27$, $\beta = 0.96$) to calculate a Bayesian estimate of the true fraction of thumbs up ratings: $\frac{3+0.27}{3+1+0.96} = 0.66$.

Table 2 describes all 7 explicit tag selection features. We now give details for several of the explicit features.

Normalized-global-avg is a user normalized version of *global-avg*. We average each user's *user-avg*, the bayesian estimates for a user's average rating for a tag.

Reputation estimates the value of a tag based on the reputation of the users who have applied the tag. The reputation of a user is computed as the average value of all the tags the user has applied. We use normalized-global-avg[16] as a measure of a tag's value.

We also experimented with three composite algorithms composed of multiple features. The first composite algorithm was a heuristic combining explicit features that is easily implementable by system designers called *hierarchical-value*. The underlying principle of hierarchical-value is to begin with broad features early in a tag's lifecycle when little data is available about the tag. As more data is available about a tag, we shift to more specific features. The value of hierarchical-value for a tag application *ta* follows:

- If the user rated the tag, return $user\_avg(ta)$
- Otherwise, return a linear combination of $global\_app\_avg$, $normalized\_global\_avg$, and $reputation$ as follows:

$$\frac{\begin{array}{l}4.0 \cdot reputation(ta) + \\ 2.0 \cdot |R_{t,m}| \cdot global\_app\_avg(ta) + \\ 1.0 \cdot |R_t| \cdot normalized\_global\_avg(ta)\end{array}}{4.0 + 2.0 \cdot |R_{t,m}| + 1.0 \cdot |R_t|}$$

Hierarchical-value weights each of the three component features according to the amount of evidence for each of them. Since our evidence is thumb ratings, the weighting is a constant multiplied by the number of thumb ratings for the tag or item-tag. [17]

In addition to hierarchical-value, we included two tag-selection algorithms that use SVMs similarly to all-implicit. All-explicit is a combination of all simple explicit features. All-implicit-explicit is a combination of all implicit and explicit features. As opposed to the expert-tuned heuristic nature of hierarchical value, all-explicit and all-implicit-explicit use an SVM to automatically optimize a weighting for features. However, since we only consider linear kernels[18], the SVM cannot express relationships structures as complex as hierarchical-value. Thus, the SVM composites choose automatic linear optimization for many features over hand-engineered non-linear complexity.

Figure 5 shows the performance of all explicit features according to item-top-3. We also include the best implicit performers (num-item-apps, apps-per-movie, and all-implicit) and the random baseline. In general, explicit features significantly outperformed implicit features at the 0.05 level. In fact two individual explicit features (normalized-global-avg, reputation) significantly outperformed the best combination of implicit features (all-implicit). As with implicit features, user normalized features (normalized-global-avg) outperform the non-normalized features (global-avg).

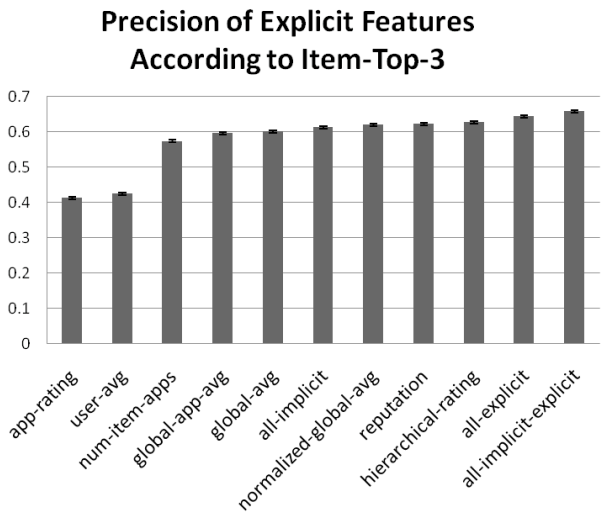

Fig. 5: *The performance of explicit features according to item-top-3. Note that the ordering of results is consistent between metrics. Differences between all neighboring pairs are statistically significant at the 0.05 level. App-rating and user-avg perform poorly due to their low coverage. Three explicit features (normalized-global-avg, reputation, and hierarchical-value) outperformed the best combination of implicit features (all implicit).*

[15]We chose a gamma distribution because its domain (all positive real numbers) covers the possible values of α and β for a beta distribution. We selected a shape parameter of 1.15 and scale parameter of 15.0 by visually tuning the distribution's probability density functions based on our domain intuition. Methods for choosing bayesian hyperparameters has been shown to have little impact over final solutions in other domains [4].

[16]We chose normalized-global-average because it performed best among the features that were not composite.

[17]We experimented with a number of different constants for each of the three components and found most to perform similarly.

[18]As mentioned earlier, we only consider linear kernels for efficiency reasons.

App-rating and user-avg performed poorly due to their low coverage. App-rating only generated a prediction for 0.8% of the survey responses, while user-rating only generated 4.3%. However, app-rating and user-avg did perform well when they generated predictions. 90.0% of the survey results associated with a positive app-rating value were high value tag applications. 80.0% of the survey results associated with a positive user-avg value were high value applications. Thus, the values of these features are important when they exist.

Reputation slightly but significantly outperformed normalized-global-avg. While both features performed similar on tags with a moderate to high number of ratings, reputation outperformed normalized-global-avg on those tags with few ratings. As an example, we calculated the reputation of every tag the moment it was introduced to the system, before it had any ratings. The top five tags were *serial killer, space, superhero, martial arts*, and *dinosaurs* while the bottom five were *dvd-ram, clv, cav, rent*, and *2.5*. If MovieLens had incorporated reputation when tagging features were first introduced, it would have been able to identify *serial killer* as a high value tag and *rent* as a low value tag the moment a user first entered them into the system.

We now address:

RQ3: How well do tag selection algorithms based on explicit behavior perform?

As we mentioned earlier, several of the explicit features (normalized-global-avg, reputation, and hierarchical-value) significantly outperformed the best combination of implicit features (all-implicit). The best single explicit feature, hierarchical-value, outperformed the best combination of explicit features 0.626 to 0.613 for item-top-3. We also found that a tag selection algorithm combining all implicit and explicit features performed even better, with a score of 0.658 for item-top-3. Six features were necessary to achieve these results: hierarchical-value, reputation, user-normalized-global-tag-average, tag-share, user-tag-average and tag-length. We choose these features similarly to how we chose the features for all-implicit. We conclude that tag selection algorithms based on implicit and explicit features outperform those based on just implicit features using offline data. In the next section, we test our offline results with a user study.

USER STUDY

Although our offline analyses provide evidence for the performance of various tag selection algorithms, we had to make simplifying assumptions. For example, although we carefully designed our tag value survey, we cannot be certain that users' feelings for tag value would be similar in our survey and a live online community. Although our metrics exhibit consistency, we cannot be certain of the extent to which they reflect a user's experience. In this section, we explore the performance of our tag selection algorithms by deploying them live on MovieLens and observing user behavior.

Based on the finite size of the MovieLens user population, we chose to deploy three tag selections algorithms for three months. We include num-item-apps to compare our algorithms to the most popular algorithm used by real tagging systems. We include implicit-only as a selection algorithm designers of existing tagging systems can adopt without adding any user interface elements. We include all-implicit-explicit for designers of systems who are willing to add thumb-based tag moderation. We do not include all-explicit because we assume that system designers would use the implicit signals at their disposal. We evaluate each tag selection algorithm by comparing the number of thumbs up and down received for tags displayed by the algorithm.

Methodology

We deployed each tag selection algorithm live on MovieLens, taking care to dynamically update each feature as users generated new data. For example, when a user gave a thumb rating to a tag we updated the explicit features, and when a user performed a tag search we updated num-searches.

We chose an experimental setup based on three requirements:

- *Control for users.* As we have seen, a few power users can significantly skew results. We avoided user-specific differences by controlling for users.
- *Control for movies.* Individual movies' characteristics such as popularity may affect tag selection algorithms. We avoided movie-specific differences by controlling for movies.
- *Ensure ordering consistency for a particular user and movie.* We want a particular user to see the same ordering for a particular movie, regardless of when they visit the movie. We accomplish this by deterministically selecting experimental conditions based on a user and movie.

Based on these requirements, we decided to show each user a different algorithm for each movie. Our approach ensured that each user saw roughly the same number of examples from each experimental condition, each user consistently saw the same algorithm for a particular movie, and different users saw different conditions for the same movie. We chose a tag selection algorithm for a user id and movie id using a pseudo-random hashing function[19] that deterministically assigned a particular user id and movie id to an experimental group.

We ran the experiment for three months and two days starting in April 2008 (we explain the significance of the two days in the next section). During this time 5,695 users logged into MovieLens, and 592 users (10.4% of active users) applied 18,271 thumb ratings to tag applications. Two users rated 1,000 or more tags, while 366 users rated five or fewer. The num-item-apps experimental condition received 6,448 ratings, all-implicit received 6,190 ratings, and all-implicit-explicit received 5,633 ratings.

We use *thumbs up precision*, the fraction of thumbs ratings that are thumbs up, as our measure of a tag selection algorithm's effectiveness. We chose this metric based on its ease of interpretability, its similarity to our offline metrics, and its reflection of actual tagging system behavior.

False Start

Two days after launching the survey, we discovered an unexpected trend. The thumbs up precision was 26% for num-item-apps, 17% for all-implicit, and 14% for all-implicit-explicit. These results were ordered precisely opposite to what our offline analyses predicted. We carefully re-examined our algorithms, but found no errors.

[19]We used John Von Neumann's middle square method for generating random numbers. We chose this method because we need the numbers to be deterministic given user and item ids, and not dependent on ordering.

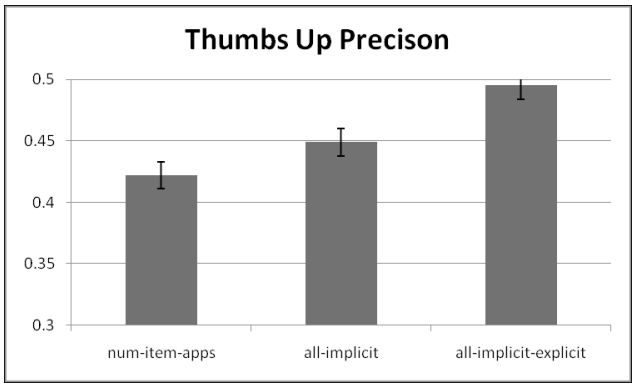

Fig. 6: *The fraction of thumb ratings received by tags displayed in each experimental group. These results validate our offline results. Tag selection algorithms based on implicit and explicit data outperform those based on just implicit data. Algorithms based on implicit data outperform the industry standard. Differences are statistically significant ($p \leq 0.05$).*

The cause of the backwards results lay in the user interface itself. When we introduced tagging in MovieLens, we displayed the number of users who applied the tag to the item alongside each tag. This design is similar to those seen on sites such as Amazon and LibraryThing. Num-item-apps, one of our experimental conditions, also returned the number of users who had applied a particular tag to an item. Thus, tags ordered by num-item-apps were consistent with the number displayed alongside the tags. We hypothesize that users interpreted the number shown alongside each tag as a signal of quality, and rated tags applied more times more highly. As a result, num-item-apps performed best. The implicit-only algorithm only slightly reordered the results, and performed second best. The implicit and explicit algorithm had the greatest divergence from num-item-apps, and performed worst. After two days, we realized the source of the backwards results and removed the numbers displayed alongside each tag. [20]

User Study

After correcting the user interface, the thumb ratings from the user study validated our offline results. 42.0% of ratings applied to tags displayed by num-item-apps received a thumbs up rating compared to 44.1% for all-implicit and 49.1% for all-implicit-explicit. All pairwise differences are statistically significantly ($p \leq 0.05$, n = 6277, 5996, 5433).

One key question is how the performance gap between implicit and explicit changes with the volume of tagging activity. Are sites with substantially more tagging activity likely to see different results than those on MovieLens? Our last research question explores this relationship:

RQ4: How does the performance of each class of tag selection algorithms change as tag density increases?

It seems plausible that tagging sites with an abundance of tagging activity, and therefore an abundance of implicit data, would be able to rely solely on implicit tag selection algorithms. To study the effects of increased activity on tag selection algorithm performance, we grouped our tag selection algorithm results by the number of tags applied to a movie. Movies with more tags overall simulate sites with more tagging activity. Movies with less tags overall simulate sites with less tagging activity.

[20] Although we don't include these two days in our analyses in the results for the following section, doing so would have little effect on them. The false start results only account for 5% of all thumb ratings during the experimental time period.

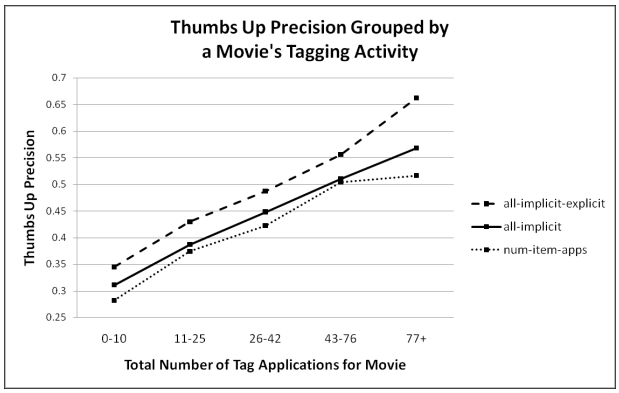

Fig. 7: *Thumbs up precision for different tag selection algorithms, grouped by the total number of tag applications for the movie. We used equal-count binning. The relative performance of the implicit and explicit tag selection algorithms seems to increase with tagging activity. The differences for the last group (77+) are statistically significant. This indicates that explicit tag moderation is valuable even for sites with a great deal of tagging activity.*

Figure 7 shows the thumbs up precision of each class of tag selection algorithm grouped by the number of tag applications per movie. We used equal-count binning to choose the tag application ranges for the X axis. 95% confidence intervals range from 0.021 to 0.026. We generally find that performance increases with tagging activity. The implicit and explicit algorithm outperforms the implicit only algorithm (all differences are significant at the .05 level except for 26-42 apps). The implicit-only algorithm outperforms the industry standard algorithm (only the 77+ bin is significant).

The performance gap between the implicit and explicit algorithm and the other two algorithms seems to increase as tagging activity increases. This indicates that even sites with moderately high tagging activity (79-200 tag applications per item[21]) can achieve significant improvements by deploying thumb-based tag moderation. The performance of the algorithms beyond 200 tags per item is inconclusive.

DISCUSSION

We begin by summarizing our two main contributions for designers. First, based on our exploration of metrics we recommend that system designers use *item-top-n* when evaluating tag selection algorithms. Second, we have demonstrated in both offline and online analyses that tag selection algorithms combining a variety of implicit and explicit signals of tag value perform best. We encourage designers to include explicit mechanisms for users to provide feedback about a tag's value.

Our results indicate that even sites with relatively high tagging activity may benefit from using all-implicit-explicit algorithm. Although we hypothesize that the all-implicit-explicit algorithm continues to add benefit for items with more than 200 tag applications, more research is needed to

[21] 10 items in our dataset had more than 200 tag applications

be certain. Regardless, active sites should benefit from using the algorithm for less popular items. Even Delicious, one of the most popular tagging sites, has fewer than 200 tag applications for most items[22]

Our analyses focused on systems with personal tag applications. Although some of the features we explore require individual tag applications (apps-per-movie, tagshare) many others apply to sites with shared applications. To study the performance of our techniques in shared systems, we simulated a shared system by only retaining the first tag application of a tag for an item. The item-top-3 value for all-implicit-explicit, our best performer only dropped by 0.003 to 0.655. These results provide evidence that machine learning-based tag selection algorithms can be adapted for sites with shared applications.

Although our user study highlighted statistically significant differences between algorithms, we wonder about how users perceive these differences. In the user study all-implicit beat num-item-apps by about 3%. This difference translates to one extra good tag per 33 tags displayed. All-implicit-explicit beats num-item-apps by 8%, translating to an extra good tag per 12 tags displayed. While we suspect that this larger improvement is noticeable, more research is necessary.

A few features that had a small (but significant) impact on performance may still be noticeable to users. User-average, for example, was quite accurate when it was available. Moreover, it provides immediate feedback in response to a user's rating for tag. For these reasons, we believe that designers should incorporate user-specific features such as user-average. Based on these results, we believe that users may benefit from more complex personalized algorithms, such as algorithms that leverage patterns in tag value among subgroups of users.

During the "false start" in our user study, we found that a tag's context influences users' perceptions of the tag's quality. Tags applied more often were deemed higher quality. The interpretability of num-item-apps standard makes it easy for designers to add context consistent with its ranking of tags (the number of times a tag has been applied). It seems more difficult to add explanations for composite features such as all-implicit-explicit. One option designers might consider is to display a score along side each tag. When users click on a score, systems could display an explanation of the individual features contributing to the score.

ACKNOWLEDGMENTS

The authors thank F. Maxwell Harper, Andrew Sheppard, Dan Frankowski, and the rest of GroupLens for their feedback on the ideas presented in this paper. We also thank the MovieLens community for their ratings, feedback and suggestions. This paper is funded in part by National Science Foundation grants IS 03-24851 and IIS 05-34420.

[22]We randomly selected 500 recently tagged URLs and found that 233 (53%) had less than 200 tag applications.

REFERENCES

1. M. Ames and M. Naaman. Why we tag: motivations for annotation in mobile and online media. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 971–980, New York, NY, USA, 2007. ACM.
2. A. Arnt and S. Zilberstein. Learning to perform moderation in online forums. *Web Intelligence, 2003. WI 2003. Proceedings. IEEE/WIC International Conference on*, pages 637–641, 2003.
3. M. Balabanovic and Y. Shoham. Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.
4. J. Besag and P. J. Green. Spatial statistics and bayesian computation. *Journal of the Royal Statistical Society. Series B (Methodological)*, 55(1):25–37, 1993.
5. V. Bouthors and O. Dedieu. Pharos, a collaborative infrastructure for web knowledge sharing. In *ECDL*, pages 215–233, London, UK, 1999. Springer-Verlag.
6. B. P. Carlin and T. A. Louis. Bayes and empirical bayes methods for data analysis. *Statistics and Computing*, 7(2):153–154, 1997.
7. C. Cattuto, V. Loreto, and L. Pietronero. Semiotic dynamics in online social communities. In *The European Physical Journal C (accepted)*. Springer-Verlag, 2006.
8. D. Cosley, D. Frankowski, S. Kiesler, L. Terveen, and J. Riedl. How oversight improves member-maintained communities. In *Proceedings of CHI 2005*, pages 11–20, New York, NY, 2005. ACM Press.
9. D. Cosley, D. Frankowski, L. Terveen, and J. Riedl. Using intelligent task routing and contribution review to help communities build artifacts of lasting value. In *Proceedings of ACM CHI*, Montreal, CA, 2006.
10. A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC, July 2003.
11. N. Glance, D. Arregui, and M. Dardenne. Knowledge pump: Supporting the flow and use of knowledge. In *Information Technology for Knowledge Management*. Springer-Verlag, 1998.
12. S. Golder and B. A. Huberman. The structure of collaborative tagging systems. *Journal of Information Science (accepted)*, 2006.
13. J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, Jan. 2004.
14. T. Joachims. Training linear svms in linear time. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226, New York, NY, USA, 2006. ACM.
15. C. Lampe and P. Resnick. Slash (dot) and Burn: Distributed Moderation in a Large Online Conversation Space. *Proceedings of SIGCHI*, pages 543–550.
16. G. MacGregor and E. McCulloch. Collaborative tagging as a knowledge organisation and resource discovery tool. *Library View (accepted)*, 55(5), 2006.
17. M. M. Markus Weimer, Iryna Gurevych. Automatically assesing the post quality in online discussions on software. In *Proceedings of the Association for Computational Linguistics*, 2007.
18. D. Millen, J. Feinberg, and B. Kerr. Social bookmarking in the enterprise. *ACM Queue*, 3(9):28–35, 2005.
19. S. Sen, F. M. Harper, A. LaPitz, and J. Riedl. The quest for quality tags. In *GROUP '07: Proceedings of the 2007 international ACM conference on Supporting group work*, pages 361–370, New York, NY, USA, 2007. ACM.
20. S. Sen, S. K. Lam, A. M. Rashid, D. Cosley, D. Frankowski, J. Osterhouse, F. M. Harper, and J. Riedl. tagging, communities, vocabulary, evolution. In *Proceedings of the ACM 2006 Conference on CSCW*, Banff, Alberta, Canada, 2006.