Proceedings of the 2nd ACM RecSys'10 Workshop on

Recommender Systems and the Social Web

EDITORS

Werner Geyer, Jill Freyne, Bamshad Mobasher,
Sarabjot Singh Anand, Casey Dugan

September 26, 2010

Barcelona, Spain

WORKSHOP CHAIRS

Werner Geyer, IBM T.J. Watson Research, Cambridge
Jill Freyne, CSIRO, TasICT Centre, Australia
Bamshad Mobasher, DePaul University, USA
Sarabjot Singh Anand, University of Warwick, UK
Casey Dugan, IBM T.J. Watson Research, Cambridge

PROGRAM COMMITTEE

Shlomo Berkovsky, CSIRO, TasICT Centre, Australia
Peter Brusilovsky, University of Pittsburgh, USA
Robin Burke, De Paul University, USA
Elizabeth M. Daly, IBM T.J. Watson Research, Cambridge
Jon Dron, Athabasca University, Canada
Rosta Farzan, Carnegie Mellon University, USA
Ido Guy, IBM Research Haifa, Israel
Max Harper, University of Minnesota, USA
Andreas Hotho, University of Karlsruhe, Germany
Aaditeshwar Seth, IIT Delhi, India
Jürgen Vogel, SAP Research, Switzerland

FOREWORD

The exponential growth of the Social Web both poses challenges, and presents opportunities for Recommender System research. The Social Web has turned information consumers into active contributors who generate large volumes of rapidly changing online data. Recommender Systems strive to identify relevant content for users at the right time and in the right context but achieving this goal has become more difficult, in part due to the volume and nature of information contributed through the Social Web.

The emergence of the Social Web marked a change in Web users' attitude to online privacy and sharing. Social media systems encourage users to implicitly and explicitly provide large volumes of information which previously they would have been reluctant to share. This information includes personal details such as location, age, and interests, friendship networks, bookmarks and tags, opinion and preferences which can be captured explicitly or more often by monitoring user interaction with the systems (e.g. commenting, friending, rating,tagging etc).

These new sources of knowledge can be leveraged by Recommender Systems to improve existing techniques and develop new strategies which focus on social recommendation. In turn recommender technologies can play a huge part in fuelling the success of the Social Web phenomenon by reducing the information overload problem facing social media users.

The goal of this one day workshop was to bring together researchers and practitioners to explore, discuss, and understand challenges and new opportunities for Recommender Systems and the Social Web. The workshop was encouraging submissions in the following areas:

- Case studies and novel fielded social recommender applications
- Economy of community-based systems: Using recommenders to encourage users to contribute and sustain participation
- Social network and folksonomy development: Recommending friends, tags, bookmarks, blogs, music, communities etc.
- Recommender system mash-ups, Web 2.0 user interfaces, rich media recommender systems
- Recommender applications involving users and groups directly in the recommendation process
- Exploiting folksonomies, social network information, interaction user context and communities or groups for recommendations
- Trust and reputation aware social recommendations
- Semantic Web recommender systems, use of ontologies and microformats
- Empirical evaluation of social recommender techniques, success and failure measures
- Social recommender systems in the enterprise

The workshop consisted both of technical sessions, in which selected participants presented their results or ongoing research, as well as informal breakout sessions on more focused topics.

Papers discussing various aspects of recommender system in the Social Web were submitted and 11 papers were selected for presentation and discussion in the workshop through a formal reviewing process.

The Workshop Chairs

September 2010

CONTENTS

Credibility-aware Web-based Social Network Recommender: Follow the Leader

Jebrin AL-SHARAWNEH
Faculty of Engineering and Information Technology
University of Technology, Sydney, Australia
jebrin@it.uts.edu.au

Mary-Anne WILLIAMS
Faculty of Engineering and Information Technology
University of Technology, Sydney, Australia
Mary-Anne.Williams@uts.edu.au

ABSTRACT
In web-based social networks social trust relationships between users indicate the similarity of their needs and opinions. Trust in this sense can be used to make recommendations on the web because trust information enables the clustering of users based on their credibility which is aggregation of expertise and trustworthiness. In this paper, we propose a new approach to making recommendations based on leaders' credibility in the "Follow the Leader" model as Top-N recommenders by incorporating social network information into user-based collaborative filtering.

To demonstrate the feasibility and effectiveness of "Follow the Leader" as a new approach to making recommendations, first we developed a Social Network Analysis Studio (SNAS) that captures real data from the Epinions dataset, next we used it to verify the proposed model. The empirical results incorporated in this paper, demonstrate that our approach is a significantly innovative approach to making effective CF based recommendations especially for cold start users.

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous

H.3.3 [**Information Systems**]: Information Storage and Retrieval-*Information Filtering*;

General Terms
Algorithms, Experimentation, Simulation, Performance.

Keywords
Recommender systems, web-based social networks, social trust, credibility, user model, user behavior, Follow the Leader, clustering, Top-N.

1. INTRODUCTION
Recommender Systems (RS) aim to provide users with recommendations about items that people with similar tastes and preferences have liked in the past. Collaborative Filtering (CF) is the dominant technique [18] for recommender systems; it relies on the opinions expressed by other users.

Most collaborative filtering based recommender systems build a neighborhood of likeminded customers who appear to have similar preferences [22]. The neighborhood formation scheme (clustering) is in fact the model-building or learning process for a recommender system algorithm. The main purpose of neighborhood formation is to produce recommendations that can be of two types: predict an opinion-score of a product for that user, or recommend to the user Top-N products not already purchased [22] [5] and that the user will like the most. Schafer, Frankowski et al. (2007) [23] introduce two different types of nearest neighbor CF algorithms: user-based algorithms to generate predictions based on similarities between users, and item-based algorithms to generate predictions based on similarities between items.

Previous research [15] has shown that people tend to like items their friends like and are attracted to the activities of others in their social circle. User designated friends in social networks, therefore, can be reliable sources of recommendations [11]. Sinha and Swearingen (2001) compared the quality of recommendations made by recommender systems and by users' friends. They found that users preferred recommendations from friends to recommendations made by recommender systems such as Amazon.com [16].

In daily life, when people seek advice from peers, they consider their past interaction history to locate the right peer, or if advice is received, they utilize these past interactions to judge the advice quality [11]. Furthermore, users would prefer to receive recommendations from people they trust. Hence, recommendations are made based on the ratings given by users who are either directly trusted by the current user or indirectly trusted by another trusting user through trust propagation mechanism [25].

One drawback of CF is that it is unable to distinguish neighbors as friends or strangers with similar taste [16]; while it utilizes neighbors to generate recommendations, it is currently unable to reflect how people seek information using their friends in social networks.

In the early days of the Internet, identifying the close friends of a user was difficult [16]. Now, social networking Web sites such as Facebook and MySpace, make gathering social network information easy, allowing one to integrate social network information and CF when generating recommendations. Therefore, the main objective of our paper is to utilize social network information in making recommendations based on

credibility of users drawn from their trustworthiness and expertise.

2. RELATED WORKS

Social networks have been used in collaborative filtering, recently [6] where trust is used as a basis for forming clusters. Our work is the first that uses a formal "Follow the Leader" model based on web-based social networks and user credibility to generate a cluster of most trustworthy and experienced users in the social network called leaders and then to model how leaders act as advisers to other users.

2.1 Web-Based Social Networks and Trust

Web-based social networks (WBSNs) are online communities (people, organizations or other social entities) [24] connected by a set of social relationships, such as friendship, co-working or information exchange in varied contexts e.g., entertainment, religion, dating, or business.

Over the last few years, interest in social networking websites such as MySpace and Facebook have increased considerably [11]. In WBSNs users are able to express how much they trust other users. Trust provides users with information about the people they share content with and accept content from. Since most participants do not know each other in real life and have no prior direct interactions with each other on social networking sites, the trust inference mechanism is becoming a critical issue when participants want to establish a new trust relation or measure trust values between connected users [17]. The idea here is not to search for similar users as CF does but to search for trustworthy users by exploiting trust propagation over the trust network [18].

In fact, users prefer to receive recommendations from people they trust more. Hence, recommendations are made based on the ratings given by users who are either directly trusted by the current user or indirectly trusted by another trusting user through trust propagation mechanism [25]

Users are allowed to state how much they consider every other user trustworthy, in the context of RSs, it relates to how much they consider the ratings provided by a certain user as valuable and relevant [18]. This additional information (trust statements) can be organized in a trust network and trust metrics can be used to predict the trustworthiness of other users as well (for example, friends of friends).

2.2 Trust-Based Collaborative Filtering

A connection between user similarity and trust was established in [26]. Using experiments they demonstrate that there exists a significant correlation between the trust expressed by the users and their similarity based on the recommendations they made in the system; the more similar two people are, the greater the trust between them.

Similarity has proved to be a key element for neighbor selection in order to provide accurate recommendations. Neighbor trustworthiness and expertise have also been researched as relevant complementary criteria to select the best possible collaborative advice [14].

Similarity can be interpreted in several ways such as similarity in interests or ratings or opinions. Recently [7] explored the relationship between trust and profile similarity. They show through surveys and analysis of data in existing systems that when users express trust, they are capturing many facets of similarity with other users. In a system that has a trust component users make direct statements about people they trust, these statements generate a social network.

Empirical results show that using measures of trust in social networks can improve the quality of recommendations. O'Donovan and Smyth [20] performed an analysis of how trust impacts the accuracy of recommender systems. Using the MovieLens dataset they create trust-values by estimating how accurately a person predicted the preferences of another. Those trust values were then used in connection with a traditional collaborative filtering algorithm [4], and an evaluation showed significant improvement in the accuracy of the recommendations.

Hundreds of millions of people are members of social networks online and many of those networks contain trust data [6]. With access to this information, trust has the potential to improve the way recommendations are made. Trust of strangers can be established based on trust propagation [3]. This term signifies all mechanisms that contribute to the dynamic extension of a trust relation. A common kind of trust propagation is similarity propagation, whereby a new trust relationship can be established between two sufficiently similar users.

2.3 Follow the Leader

As pointed out by social psychology theory [17], the role of a person in a specific domain has significant influences on trust evaluation if the person recommends a person or an object. Thus, the role of the user should also be taken into account in making recommendations.

Follow the leader in dynamic social networks [10], is a model of opinion formation with dynamic confidence in agent-mediated social networks where the profiling of agents as leaders or followers is possible. An opinion leader is specified as a highly self-confident agent with strong opinions. An opinion follower is attracted to those agents in which it has more confidence. The "Follow the Leader"[1] model provides a formal probabilistic approach and we have provided a URL to a paper that describes it in detail for those who are not familiar with it.

Goldbaum's [10] model identifies three types of consumers that seek input from outside experts:

1. The consumer has a fixed set of preferences but imperfect information concerning the available product options. The expert offers information or advice that helps the consumer buy the product that maximizes her exogenous utility.
2. The consumer possesses some innate preferences over the available products, but can be influenced by the opinion of others (peers) and experts. The "expert" may be someone possessing both better information than the general public about the consumer options (as in case 1), but also someone who can provide advice that is consistent with the underlying preferences of a group of consumers.
3. The consumer has no innate preferences. The consumer's tastes are fully fashioned by the influence of peers and experts. In this case, the expert shapes opinion, but need not have any special advantage in evaluating the options.

[1] www.business.uts.edu.au/finance/research/wpapers/wp155.pdf

The expert in all previous cases can be considered as an experienced user who spends time and effort in analyzing product features, and finally making her decision in using the product, and giving high quality feedback on that product. Furthermore, she is in a position to strongly recommend the product to her friends if they have similar needs.

According to [10], in a social network a member is either a leader or a follower who adopted another leader's opinion or recommendation to use a product. Subsequently this member adopts whatever her best friend adopted, otherwise the member has no active friends and consequently it acts as an independent (leader).

Ramirez-Cano and Pitt [21], define the relationship between two agents (users) as a confidence function, such that: "an agent (i) increases its confidence in another agent (j) based on how well (j's) opinion meets the criteria specified in i's mind-set. A mind-set represents the set of beliefs, attitudes, assumptions and tendencies that predetermine the way an agent evaluates a received opinion". Subsequently we conclude that user trust is the determinant relationship between two friends.

3. MOTIVATIONS and CONTRIBUTIONS

Using trust in social networks provides a promising approach to make recommendations to other users based on trust propagation in finding a friend or a friend of a friend with similar interests. However, the quality of recommendations can be improved further by incorporating a users' expertise because trusted expert advice will lead to better recommendations.

We believe seeking advice from a trustworthy expert is more feasible than seeking advice from a normal trustworthy user. This leads us to the question: How to find trustworthy experts in the WBSN?

Our key contribution in this paper is threefold:

- A user model based on its credibility that captures trust relationships between users.
- A clustering approach based on the "Follow the Leader" model and user credibility.
- An effective means to generate high quality user-based collaborative recommendations based on user credibility and following the leaders.

The rest of the paper is organized as follows: Section 4 presents our technical framework. In the following sections we present the evaluation of the proposed model. Finally, we conclude by summarizing our findings and our future plans for further work.

4. TECHNICAL FRAMEWORK

In this section we define the following concepts to be used in our model.

In WBSN, let (U) be a set of users, $\boldsymbol{U} = \{\boldsymbol{u_1}, \ldots, \boldsymbol{u_N}\}$, interacting in a set of contexts $\boldsymbol{C} = \{\boldsymbol{c_1}, \ldots, \boldsymbol{c_L}\}$, such as categories in EPINIONS. In each context \ category there is a set of items (K), such that: $\boldsymbol{I} = \{\boldsymbol{i_1}, \ldots, \boldsymbol{i_K}\}, where\ I \in C$.

Each user $(u \in U)$ rates a set of items M denoted by: $R_u^I = \{R_u^1, \ldots, R_u^i \ldots, R_u^M\}, where\ M \leq K$, and (R_u^i) is the rating value of user *(u)* for item *(i)*. The rating value can be any real number, but most often ratings are integers, e.g. in the range [1, 5].

In a trust-aware system, there is also a trust network among users. We define (T_u^v) to be the direct trust between user *(u)* and user *(v)*, trust value is a real number in [0, 1]: (0) means no trust and (1) means full trust between users or a value on a scale in the range [0, 5]. Binary trust networks are the most common trust networks such as Amazon and eBay.

In this model, we are concerned with the following characteristics of trust: [9], [12]

1. Trust is context based, this means if A trusts B to repair her car this means A would not trust B to provide her with medical advice.
2. Trust is directed from source to target means if A trusts B to repair her car this doesn't mean B should trust A in the same context.
3. Trust is transitive: if A trusts B and B trusts C in the same context or area of expertise, then there is some degree of trust between A and C can be inferred.
4. Trustworthiness is dynamic: since trust can increase and decrease with further experiences (interactions or observation). Trustworthiness can also decay with time.

Thus, we can model a WBSN in a specific context as directed graph over which trust can propagate:

$\boldsymbol{G} = \langle \boldsymbol{U}, \boldsymbol{T} \mid U\ is\ set\ of\ Nodes, T\ is\ set\ of\ edges \rangle$

4.1 Credibility Based Clustering - Follow the Leader

The "Follow the Leader" model [10], provides us with insights to cluster users based on their roles in the WBSN i.e. either leaders or followers. Enriching the "Follow the Leader" model with trust, gives us the potential to analyze WBSN based on user's credibility. Figure-1 shows the basis of our approach. A credibility measure of users reflects their trustworthiness and expertise provides us with means to cluster users in a specific context. Some users can be classified as leaders others can be classified as followers according to their trustworthiness and expertise level.

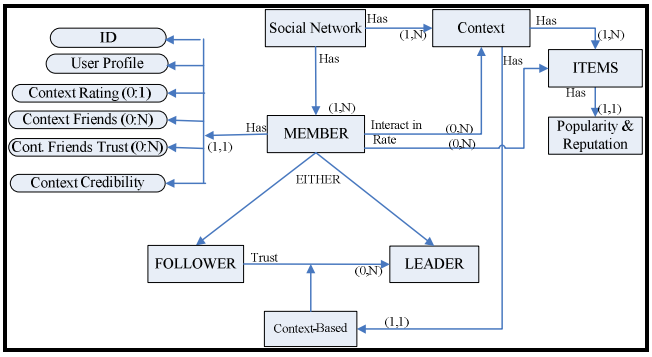


Figure-1: WBSN trust-aware Follow the Leader model

4.1.1 User Credibility

Credibility is a synonym of believability [1]. Credibility of an agent can be measured by its trustworthiness, expertise, and dynamism [13]. The majority of researchers identify two key components of credibility: trustworthiness and expertise. In evaluating user credibility we address these two components as follows:

Trustworthiness

Barney and Hansen (1994), [2] explicitly differentiate between trust and trustworthiness pointing out "while trust is an attribute of a *relationship* between exchange partners, trustworthiness is an *attribute* of individual exchange partners". Therefore, a trustworthy person is someone in whom we can place our trust without any risk of being disappointed or betrayed" [12].

In trust-aware WBSN for a specific context, users act as trustor or trustee. Trustor users provide trust scores for other users (trustee) based on their confidence level that trustee provides reliable ratings for items in a specific context. Consequently trustee's gain reputation for her trustworthiness, hence the more trustors who either directly or indirectly trust a user, the more her credibility increases, formally we represent this relation as follows:

$$\mathrm{Cr}(T_T^u) = f(T_v^u, N_D^u, N_I^u, N_{DMax}, N_{IMax}) \quad (1)$$

Where (T_v^u) is *scaled* trust score of user (v) assigns to user (u) as defined previously, (N_D^u) is the number of direct friends who trust (u), (N_I^u) is the number of indirect friends of (u) i.e. friends of friends. In order to give the user with max number of followers more weight than others, we consider (N_{DMax}) as a reference point, consequently (N_{IMax}) refers to the maximum number of indirect friends in the set.

Scaled trust score is defined as follows: $T_v^u = \frac{T_v^u(Given)}{T_{range}}$, where Trust score range = 5.

Credibility from direct followers trust:

Direct followers / friends of a user provide trust scores specifying how much they trust the user. The aggregation of trust scores is a measure of user' trustworthiness, consequently it is a measure of her credibility. Formally, we denote credibility from direct followers trust as: $\mathrm{Cr}(T_D^u)$, and defined it as:

$$\mathrm{Cr}(T_D^u) = \frac{1}{N_{DMax}} * \sum_{v=1}^{N_D} T_v^u \quad (2)$$

The impact of (N_D^u) appears on the aggregation of trust values from direct followers (friends). As can be seen credibility from direct followers is normalized, hence if the most trustworthy user receives a trust score of 5 from all friends then: $\mathrm{Cr}(T_D^u) = 1$, if that user has the maximum number of direct followers.

Credibility from Indirect Followers Trust

Using the trust transitivity feature of trust [8], friends of friends who trust their nearest friend, and also trust their friends next friend, with a trust score of the product of the two scores, formally $\mathrm{Cr}(T_I^u)$ is defined as follows:

$$\mathrm{Cr}(T_I^u) = \frac{1}{N_{IMax}} * \sum_{v=1}^{N_D} \{T_v^u * \sum_{v=1}^{N_I} T_v^u\} \quad (3)$$

Where (N_{IMax}) refers to maximum number of indirect followers. (T_v^u) in the first part refers to direct trust of direct followers to the target user, while (T_v^u) in the second part refers to indirect followers trust. Trust aggregation for each direct follower over indirect followers, allows us to aggregate the trust associated with all friends of this follower. For example if F12 trusts F1 with score (4) and F1 trusts U with score (3) on a scale of maximum (5), then F12 trusts U with score = (4 * 3) / (5 * 5) = 0.48 in normalized measure or 2.4 in scaled measure of 5.

We believe trust from lower levels of indirect followers have a small impact on trustworthiness of upper level members and associated recursive computation cost increase the complexity of the model. So, we consider the first level of indirect followers only.

Example: we provide the following example based on figure-2 to explain and illustrate the above concepts:

1. Since maximum trust score range is 5, to scale trust values to (1) we divide each *trust value* by (5), hence trust values (3, 4, 3) from direct followers (F1, F2, F3) when scaled turn into (0.6, 0.8, 0.6).
2. If user (u) with the maximum number of direct followers and indirect followers in this context then: $N_{DMax} = 3$, and $N_{IMax=5}$.
3. Credibility from Direct Followers computed as: $\mathrm{Cr}(T_D^u) = \frac{1}{3} * (0.6 + 0.8 + 0.6) = 0.667$
4. Credibility from Indirect Followers computed as: $\mathrm{Cr}(T_I^u) = \frac{1}{5} * \big(0.6 * (1 + 0.8) + 0.8 * (0.4 + 0.6 + 0.6)\big) = 0.472$

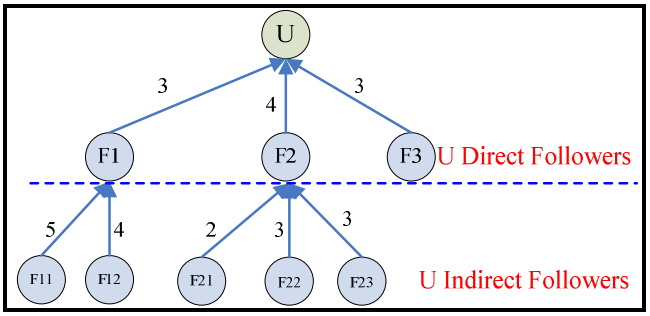


Figure-2: Direct and Indirect Followers

Expertise

Expertise, a key dimension of credibility; it is defined using expertise as the degree of a user's competency to provide an accurate ratings and exhibit high activity [14]. The expertise dimension of credibility captures the perceived knowledge and skills of the user in a given context.

If item (i) received ratings from N users, each provide (R_u^i) for that item then the average of ratings of item (i) is given by:

$$R_{avg}^i = \frac{1}{N} * \sum_{u=1}^{N} R_u^i \quad (4)$$

In order to avoid current user rating influence on (R_{avg}^i) and for small number of raters e.g. (N < 10), we can exclude current user rating from (R_{avg}^i) calculations, this yields:

$$R_{avg1}^i = \frac{1}{N-1} * \{\sum_{u=1}^{N} R_u^i\} - R_u^i \quad (4.a)$$

Thus user trustworthiness in providing rating (R_u^i) for item (i) is measured by comparing the provided rating (R_u^i) with (R_{avg}^i), the smaller the difference between the two ratings, the higher is the user trustworthiness. So, trustworthiness of one rating is given in the following formula: difference between two ratings is given as:

$$\mathrm{Cr}(R_u^i) = 1 - \frac{|R_u^i - R_{avg}^i|}{\mathrm{R_{Max}^i}} \quad (5)$$

Where $\mathrm{R_{Max}^i}$ is the maximum rating scale.

Although other researchers such as [20] use other approaches to penalize ratings too far from reference rating (R_{avg}^i), we

believe this formula works for binary rating and scaled ratings as well, and we show this empirically.

If user (u) provides ratings for (M) items, then accumulated credibility from rating M items is given as:

$$\mathrm{Cr}(R_u) = \frac{1}{M} * \sum_{i=1}^{M} \mathrm{Cr}(R_u^i) = \frac{1}{M} * \sum_{i=1}^{M} \left\{1 - \frac{|R_u^i - R_{avg}^i|}{R_{Max}^i}\right\} \quad (6)$$

In order to differentiate between users who provide more ratings for different items, user credibility increases if the number of ratings increases. To consider this factor, we model user contribution as a weight factor for user ratings credibility; hence users who contribute more than others are rewarded more. So, if we consider the user with the maximum number of ratings over all items (K) as a reference point (N_{Max}^K), then the user contribution weight = (number of ratings of the user / maximum number of ratings among all users), formally given by:

$$R_u^w = \frac{M}{N_{Max}^K} \quad (7)$$

Using equation (7) in (6), yields the user credibility from rating component,

$$\mathrm{Cr}(R_u) = \frac{1}{N_{Max}^K} * \sum_{i=1}^{M} \left\{1 - \frac{|R_u^i - R_{avg}^i|}{R_{Max}^i}\right\} \quad (8)$$

It is clear from the above equation that if the user did not make any rating contributions, then their reward from this component equals zero. The more credible contributions she makes, the more reward she receives.

4.1.2 Computing user Credibility

By aggregating credibility components: trustworthiness (credibility from direct followers and indirect followers) and expertise, we compute user credibility as:

$$\mathrm{Cr}(u) = \alpha * \mathrm{Cr}(R_u) + \beta * \mathrm{Cr}(T_D^u) + \gamma \mathrm{Cr}(T_I^u) \quad (9)$$

Where $\alpha + \beta + \gamma = 1$, and α, β, γ are system tuning parameters represent the importance of each factor. In our experiments we use the values (5/9,3/9,1/9) respectively.

4.2 Clustering users based on credibility

In this model we define a *credibility threshold* from which we identify leaders and followers as follows:

$$\begin{cases} \text{if } \mathrm{Cr}(u) \geq \mathrm{Cr}_{\mathrm{Threshold}}, \text{ then user is Leader} \\ \text{if } \mathrm{Cr}(u) < \mathrm{Cr}_{\mathrm{Threshold}}, \text{ then user is Follower} \end{cases} \quad (10)$$

$\mathrm{Cr}_{\mathrm{Threshold}}$ refers to the *Credibility Threshold,* which is a system parameter that identifies users based on their credibility. It is computed based on leaders' percentage in the WBSNW to be promoted from the target set. In our experiments we used leaders' percentage 10% of the members of the data set.

Leaders usually have the knowledge and power to provide trustworthy advice in recommending the most trustworthy items for other users.

To build a Follow the Leader hierarchy, we use credibility to identify user's roles i.e. (leaders, followers, independents). Followers usually follow the most credible friend in a given context. Using the confidence relation, if a follower finds her credibility more than the credibility of all her friends, then the user acts as either leader if she is qualified as leader, or as an independent.

Due to the dynamism of the network created by its natural evolution some leaders may lose their credibility over time if they behave dishonestly or if they stop making contributions, or their trustworthiness drops.

4.3 Using leaders as potential Top-N recommenders

In order to make recommendations, our approach relies on leaders as the Top-N credible and trustworthy users in the context of providing recommendations. The number of leaders is determined by the credibility threshold.

We compute the predicted rating by user (a) for unknown items (i) using the following formula replacing similarity weight in [18] to credibility weight:

$$Pred(a,i) = R_a^{Avg} + \frac{\sum_{u=1}^{K} \mathrm{Cr}(u) * (R_u^i - R_u^{Avg})}{\sum_{u=1}^{K} \mathrm{Cr}(u)} \quad (11)$$

Where K is number of leaders who rated item (i), (R_u^i) represents the rating for item (i) provided by a leader (u), (R_u^{Avg}) represents leader ratings average.

When the target user (a) does not have ratings or her ratings < 10, i.e. cold start user, then we use the following formula replacing trust weight in [8] by credibility weight:

$$R_{Pred}^i = \frac{\sum_{u=1}^{K} \mathrm{Cr}(U) * R_u^i}{\sum_{u=1}^{K} \mathrm{Cr}(U)} \quad (12)$$

5. EXPERIMENTS AND RESULTS

To demonstrate the feasibility and effectiveness of the "Follow the Leader" as a new approach for making recommendations, first we developed a Social Network Analysis Studio (SNAS) inspired by Goldbaum, (2008) [10], which can be used to simulate network evolution and to build follow the leader models. In order to test and evaluate our model we captured real data from the Epinions dataset, and used it to verify the proposed model. In this section, we discuss the datasets, experimental design, and the results that demonstrate that the idea of "Follow the Leader" as means to build credibility can improve the accuracy of recommendations.

5.1 The Epinions Dataset

For our experiments, we used the Epinions[2] dataset to validate the applicability of our approach, because Epinions provides us with required information; trust relationships between users and corresponding trust ratings between individuals and Ratings of items by the members of the social network.

We used the version of the Epinions data set which has 49,290 users who rated a total of 139,738 different items at least once, writing 664,824 reviews with 487,181 issued trust statements.

To represent a context; our procedure is as follows: (1) we selected 25 items (randomly) from Epinions dataset. (2) We extracted the item ratings from "Ratings data" for all users who rated any of the selected items. (3) All associated trust statements from the "Trust data" were selected. (4) From the (25) items we accumulated the ratings history for each user; items rated and corresponding ratings. (5) Finally we removed self trust statements and duplicated trust statements from the extracted trust data set.

[2] http://www.trustlet.org/wiki/Downloaded_Epinions_dataset

Table-1 shows (9) data sets, we used for initial model verification. Then for final verification we used the (9) items selected and corresponding items ratings history, we generated three datasets;

1. DataSet-1: includes all users who rated any of the items (106, 515, 698, 18560)
2. DataSet-2: includes all users who rated any of the items (619, 1081, 2164, 3010, 6147)
3. DataSet-3: includes all users who rated any of the (9) items in table-1, i.e. includes all users in the data set.

In order to enrich users with more items ratings history, users' ratings history of (16) additional items were included. Corresponding ratings for the items (363, 390, 391, 393, 615, 651, 659, 660, 700, 734, 1083, 1109, 2810, 3017, 6114, 6131) were used in the user rating history who rated them; hence we can consider our datasets refer to any item in the 25 items, which we consider as a context in our approach.

Summary of the three datasets is shown in Table-2:

Table 1. Selected Items Summary

Item ID	Rating No	Rating Avg.	Trust Records	SNW Users	LDRS No	FLWRS NO	LDRS-AVG Items Rated	FLWR AVG Items Rated	LDRS AVG Rating\ITM	FLWR AVG Rating\ITM
106	217	4.44	127	75	9	66	4.4	2.1	3.89	3.32
515	199	3.46	189	81	9	72	4.7	2.1	3.56	3.31
619	140	4.29	259	97	10	87	7.7	3.3	5.00	3.63
698	80	4.21	59	43	4	39	9.8	2.6	3.50	3.77
1081	299	3.43	403	142	18	124	7.6	3.2	3.22	3.10
2164	202	4.78	650	158	17	141	6.2	3.1	4.76	4.19
3010	225	3.84	545	164	19	145	7.4	3.2	2.95	3.08
6147	210	3.42	567	117	13	104	5.8	2.2	3.31	3.10
18560	100	2.14	44	30	3	27	7.0	2.7	2.33	1.52
TOTAL	1672		2843	907	102	805				

Table 2. Datasets Analysis Summary

DATASET	Rating No	Rating Avg.	Trust Records	SNW Users	LDRS No	FLWRS NO	LDRS-AVG Items Rated	FLWR AVG Items Rated	LDRS AVG Rating\ITM	FLWR AVG Rating\ITM
DATASET-1	596	3.70	419	229	26	203	5.9	2.2	3.50	3.18
DATASET-2	1076	3.88	2424	634	74	560	6.7	2.8	3.97	3.45
DATASET-3	1672	3.81	2883	830	94	736	6.6	2.6	4.02	3.42
TOTAL	3344		5726	1693	194	1499				

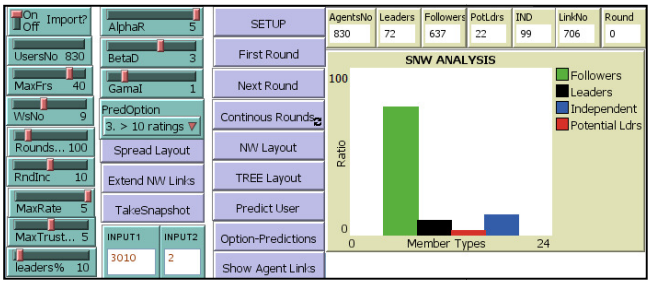


Figure-3 Social Network Analysis Studio (SNAS) – User interface

5.2 Social Network Analysis Studio (SNAS)

We have developed a Social Network Analysis tool utilizing the NetLogo platform [19]. The user interface is shown in figure 3 (for DATASET-3). Although NetLogo was used as simulation tool, it has the facility to capture real data. We use it to analyze and evaluate the validity of our approach.

Figure 4 shows the "Follow the Leader" Model for social network in item 2164. The black and red color nodes refer to leaders and potential leaders, green nodes refer to followers and the blue nodes refer to independents. Potential leaders are normal followers who have a confidence in themselves more than the confidence in any of their friends; their credibility is below the credibility threshold and they usually have adequate number of followers.

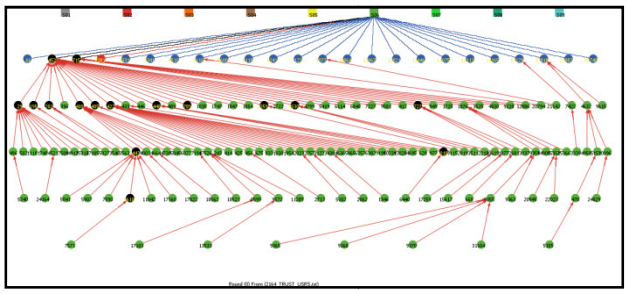


Figure-4 Follow the Leader Model for social network in item 2164

5.3 Experimental Setup

To test the hypothesis that using the "Follow the Leader" approach is an applicable approach to improve the accuracy of recommendations, we use ***leave-one-out*** as our testing strategy. Leave-one-out is an approach that can be used on a known dataset and involves hiding one rating and then trying to predict it with a certain algorithm. The predicted rating is then compared with the actual rating and the difference in absolute value is the prediction error. The procedure is repeated for all

the ratings and an average of all the errors is computed as the Mean Absolute Error (MAE) given the following formula:

$$MAE(1) = \frac{\sum_{i=1}^{N} |R_u^{Actual} - R_u^{Predicted}|}{N} \quad (13)$$

For consistency, we measure MAE(2) with respect to average item rating by all users who rated it, MAE(2) is given the following formula:

$$MAE(2) = \frac{\sum_{i=1}^{N} |R_i^{Average} - R_u^{Predicted}|}{N} \quad (14)$$

We follow the following procedure for each dataset:

1. Generate "Follow the Leader" model for each dataset.

2. Select users to predict item ratings for them, based on the following test options, number of ratings indicate the expertise level of the customer. It is not necessary that it reflects their credibility.

Option-1: for users who rated 4 items or less, these usually are followers in the model and cold start users.

Option-2: for users who rated 5 items and up to 8 items,

Option-3: for users who rated 9 items or more, those usually are leaders in the model for the selected datasets.

3. Generate a predictive rating for each user for a specific item based on the leave-one-out strategy.

5.4 Prediction Algorithms

We use the following algorithms to verify and benchmark our approach prediction accuracy:

1. **CF-Credibility (CF-1)**: based on algorithm – equation (11) outlined in section 4.3.
2. **Neighbors TRUST (CF-2):** based on Golbeck and Hendler (2006) [8] formula, outlined in section 3.1 of the FilmTrust. This algorithm is used as a benchmark to our CF-Credibility (CF-1). "Follow the Leader" model provides the means to identify the nearest neighbors easily as shown in figure 4.
3. **CF-LDRs Similarity(CF-3)**: this is the conventional CF prediction algorithm outlined in Motivation section – formula (1) of [18], with consideration that the similarity is based on leaders in our data set. This algorithm is used as benchmark to our CF-Credibility.

CF-1C and CF-2C refer to cold start users case.

Table-3: Experimental Results of Prediction Algorithms – Cold Start

			MAE-1		MAE-2		
DATASET	Testing Option	No. Test Cases	Cold Start Credibility CF-1C	Cold Start TRUST CF-2C	Cold Start Credibility CF-1C	Cold Start TRUST CF-2C	Coverage (1)
DataSet-1	OPTION-1	200	0.990	0.980	0.291	0.753	59.50%
DataSet-2	Cold Start	488	0.877	0.878	0.276	0.696	76.02%
DataSet-3	< 5 ratings	669	0.934	0.903	0.301	0.709	72.50%
Over-All	Average	452	0.922	0.905	0.291	0.711	71.85%

Table-4: Experimental Results of Prediction Algorithms – Experienced Users

			MAE-1			MAE-2			
DATASET	Testing Option	No. Test Cases	CF-1 Credibility	CF-2 Trust	CF-3 LDrs Similarity	CF-1 Credibility	CF-2 Trust	CF-3 LDrs Similarity	Coverage (1)
DataSet-1		24	1.028	0.900	1.354	0.689	0.748	1.129	41.67%
DataSet-2	OPTION-2	132	0.981	0.940	1.178	0.640	0.650	0.941	56.82%
DataSet-3	5-8 ratings	145	0.959	0.944	1.197	0.605	0.654	0.962	55.86%
DataSet-1		5	0.957	N\A:Leaders	1.525	0.211	N/A	0.694	0.00%
DataSet-2	OPTION-3	14	0.914	0.991	1.160	0.333	0.572	0.606	64.29%
DataSet-3	> 8 ratings	16	0.870	1.092	1.383	0.298	0.701	0.782	62.50%
Over-All	Average	56	0.967	0.948	1.213	0.593	0.658	0.938	55.89%

(1) Coverage here refers to the ratio of returned valid predictions over all predictions for CF-2. This value is (100%) for the CF-1 and CF-3.

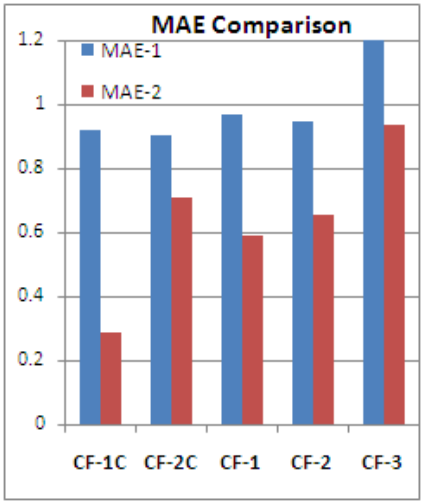


Figure 5. MAE Average for all predictions over all Datasets

6. RESULTS ANALYSIS and DISCUSSION

The results presented in Table-3, Table-4 and Figure 5 show that "Follow the Leader" is a highly effective approach to identify Top-N recommenders in a social network. We have shown that leaders can provide high quality recommendations all the time:

1. Leaders can provide recommendations with (100%) coverage in credibility based prediction algorithms, while the Neighbors Trust algorithm (CF-2, CF-2C) does not, because users in Neighbors Trust do not have sufficient friends to reliably apply the trust algorithm.
2. Cold Start Credibility (CF-1C) algorithm outperforms Neighbor Trust in two dimensions: first the coverage of Cold Start Credibility is (100%) while for Neighbor Trust the average coverage is (72%). Second the quality of recommendations (MAE-2) of Cold Start Credibility is (0.291) while for Neighbor Trust (MAE-2) is (0.711). This result emphasis that normal users (followers) possess less credibility than leaders.
3. CF-Credibility (CF-1) outperforms CF-LDRs Similarity (CF-3) with (25%); this difference emphasis that the credibility based approach provides more accurate results even when using the same leaders to measure the similarity with them.
4. In OPTION-3: for users who rated 9 items or more, we observed that the Neighbor Trust algorithm provided poorer prediction in dataset-1, and predictions are too far from average algorithm performance to be of value because the users in OPTION-3 are almost leaders. Leaders usually do not act as trustors; they are trusted by

other users. In other words leaders are not necessarily good judgers of credibility or at developing trust.

5. In OPTION-3: for users who rated 9 items or more, CF-1 (credibility based) outperform other approaches because leaders have enough ratings to compute the average.
6. Increasing the number of users in the social network produce more credible results; it scales well. This behavior is more obvious in option-2 and option-1 for CF-Credibility algorithm. The reason is that when increasing the size of the network, the number of genuine leaders also tends to increase.

7. CONCLUSIONS

We have evaluated our proposed framework through our Social Network Analysis Studio (SNAS) in a specific context extracted from the widely used Epinions dataset. The results presented in this paper show that our credibility based clustering derived from the "Follow the Leader" model provides a highly effective approach to identify Top-N recommenders, who are leaders in the context with the highest trustworthiness and expertise among all users.

We proved the feasibility of our proposed framework in providing accurate predictions by benchmarking it against the leading algorithms CF-Similarity based [18] and with Social TRUST [8]. The results of the experiments included in this paper show the applicability and performance of the proposed credibility assessment based on "Follow the Leader" Model shows significant measurable enhancements over other approaches.

We have shown that using the "Follow the Leader" model is an effective approach to cluster users based on their credibility which gives high performance predictions. In addition, trust relations can be extracted easily from the model, and user credibility is a valuable parameter in calculating items reputation. Our future plan is to use user credibility to model items credibility which is an important factor in identifying Top-N items in the context.

8. AKNOWLEDGMENTS

Financial assistance was received from (QCIS:UTS) Center for Quantum Computation and Intelligent Systems.

9. REFERENCES

1. Andrade, F., Neves, J., Novais, P., Machado, J. and Abelha, A. Legal security and credibility in agent based virtual enterprises. *Collaborative Networks and Their Breeding Environments*. 503-512.
2. Aquevegue, C. and Ravasi, D., Corporate reputation, affect, and trustworthiness: an explanation for the reputation-performance relationship. in, (2006).
3. Borzymek, P., Sydow, M. and Wierzbicki, A., Enriching Trust Prediction Model in Social Network with User Rating Similarity. in, (2009), IEEE Computer Society, 40-47.
4. Breese, J.S., Heckerman, D. and Kadie, C., Empirical analysis of predictive algorithms for collaborative filtering. in, (1998), 43–52.
5. Deshpande, M. and Karypis, G. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, *22* (1). 177.
6. DuBois, T., Golbeck, J., Kleint, J. and Srinivasan, A. Improving Recommendation Accuracy by Clustering Social Networks with Trust. *Recommender Systems & the Social Web*.
7. Golbeck, J. Trust and nuanced profile similarity in online social networks. *ACM Transactions on the Web (TWEB)*, *3* (4). 12.
8. Golbeck, J. and Hendler, J., Filmtrust: Movie recommendations using trust in web-based social networks. in, (2006), Citeseer, 43-44.
9. Golbeck, J.A. Computing and applying trust in web-based social networks.
10. Goldbaum, D. Follow the Leader: Simulations on a Dynamic Social Network. *UTS Finance and Economics Working Paper No 155,* http://www.business.uts.edu.au/finance/research/wpapers/wp155.pdf.
11. Gürsel, A. and Sen, S., Producing timely recommendations from social networks through targeted search. in, (2009), International Foundation for Autonomous Agents and Multiagent Systems, 805-812.
12. Jaquet-Chiffelle, D.O. D17. 4: Trust and Identification in the Light of Virtual.
13. Kouzes, J.M. and Posner, B.Z. *Credibility: How Leaders Gain and Lose It, Why People Demand It, Revised Edition*. San Francisco, CA: Jossey-Bass, 2003.
14. Kwon, K., Cho, J. and Park, Y. Multidimensional credibility model for neighbor selection in collaborative recommendation. *Expert Systems with Applications*, *36* (3). 7114-7122.
15. Lerman, K. Social networks and social information filtering on digg. *Arxiv preprint cs/0612046*.
16. Liu, F. and Lee, H.J. Use of social network information to enhance collaborative filtering performance. *Expert Systems with Applications*.
17. Liu, G., Wang, Y. and Orgun, M. Trust Inference in Complex Trust-oriented Social Networks.
18. Massa, P. and Avesani, P., Trust-aware recommender systems. in, (2007), ACM, 24.
19. NetLogo NetLogo Home Page. On line at: http://ccl.northwestern.edu/netlogo/.
20. O'Donovan, J. and Smyth, B., Trust in recommender systems. in, (2005), ACM New York, NY, USA, 167-174.
21. Ramirez-Cano, D. and Pitt, J., Follow the Leader: Profiling Agents in an Opinion Formation Model of Dynamic Confidence and Individual Mind-Sets. in, (2006), IEEE Computer Society Washington, DC, USA, 660-667.
22. Sarwar, B.M., Karypis, G., Konstan, J. and Riedl, J., Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. in, (2002), Citeseer, 158–167.
23. Schafer, J., Frankowski, D., Herlocker, J. and Sen, S. Collaborative filtering recommender systems. *The Adaptive Web*. 291-324.
24. Shekarpour, S. and Katebi, S.D. Modeling and Evaluation of Trust with an Extension In Semantic Web. *Web Semantics: Science, Services and Agents on the World Wide Web*.
25. Zarghami, A., Fazeli, S., Dokoohaki, N. and Matskin, M., Social Trust-Aware Recommendation System: A T-Index Approach. in, (2009), IEEE Computer Society, 85-90.
26. Ziegler, C.-N. and Golbeck, J. Investigating interactions of trust and interest similarity. *Decis. Support Syst. 43(2)*. 460–475.

Augmenting Online Video Recommendations by Fusing Review Sentiment Classification

Weishi Zhang[1], Guiguang Ding[1], Li Chen[2], Chunping Li[1]

[1]School of Software, Tsinghua University, China

[2]Department of Computer Science, Hong Kong Baptist University, Hongkong

zhang-ws08@mails.tsinghua.edu.cn; {dinggg, cli}@tsinghua.edu.cn; lichen@comp.hkbu.edu.hk

ABSTRACT

In this paper we aim to resolve the recommendation problem in online environments when user rating information is not available. As a matter of fact, in most of current websites especially the video-sharing ones, the traditional pure rating based collaborative filtering recommender methods are not fully qualified due to the sparsity of rating data. We hence propose a new recommender algorithm by fusing an unsupervised sentiment classification approach, by which the missing user-item rating matrix can be generated by decomposing user reviews as given to items. In addition, the rating matrix is further integrated with predicted scores on items' descriptive keywords so as to obtain more accurate results. To test the algorithm's practical values, we have first identified the unsupervised sentiment classification's higher precision and recall by comparing it with supervised approach. Moreover, we conducted both a statistic evaluation method and a user study to show the effectiveness of our recommender system on improving online video recommendations' accuracy.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing. H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - *Information Filtering*

General Terms

Algorithms, Performance, Experimentation

Keywords

Information retrieval, online video recommendation, sentiment analysis

1. INTRODUCTION

Recommender systems that suggest unknown interesting items to users have been widely developed in recent years, among which collaborative filtering (CF) method is one of typical approaches that principally derives recommendations for a user based on the preferences of other users who have similar tastes [5]. In most CF systems, the item ratings in the *user-item rating matrix* are assumed to be obtainable from real-users. However, in reality, many websites especially the existing video-sharing ones, do not provide rating supports, or few users have actually inputted rates (e.g., the sparsity problem [16]). It hence unfortunately limits the applicability of CF algorithms and even other pure rating based systems to generate accurate recommendations.

Therefore, in this paper, we have mainly investigated the role of user reviews in complementing the rating sparsity problem. User reviews, as another form of user inputs to indicate their interests, have actually broadly appeared in the resource-sharing websites especially video-sharing ones such as YouTube [28] and YouKu [27].Thus, it will be meaningful to study their impacts on enhancing the accuracy of video recommendations. More specifically, we have aimed at generating the missing rating data from user reviews through the method of sentiment classification, so that given a piece of text, the latent opinion can be discovered to show different possible sentiment polarities (e.g., positive, neutral, or negative) and hence reflect users' preferences on the corresponding item.

With the aim, we have first in depth studied online video reviews, and found that 1). the review contents include much noise information such as advertisements, hyperlink text etc.; 2) besides textual comments, there are various expression faces (e.g., smiley); 3) the ratio of positive and negative reviews is not 1:1, that is different from the common assumption in related sentiment classification methods [4, 23, 25]. In fact, most of related works on sentiment classification are supervised (i.e., a large number of labeled training data is needed) and developed based on standard review datasets [1, 4, 7, 13], so it is not straightforward to adopt them into the rating generation process based on online reviews.

Thus, we have developed a novel unsupervised sentiment classification approach to address the realistic characteristics of online reviews. It concretely contains two components, word sentiment scoring and facial sentiment scoring, which are to be combined while determining the overall sentiment polarity of a review document. The method does not need the accumulation of training data, and can automatically build the user-item rating matrix to be fused into standard CF algorithms.

The main contributions of our work can be summarized as follows: 1) we propose an unsupervised sentiment classification approach that particularly considers the special features of real online video reviews, and identify its higher accuracy being compared with supervised classification method; 2) we propose a strategy to generate item ratings by employing the sentiment classification results; 3) we further integrate the predication of ratings on items' keywords for enhancing the standard *user-item rating matrix*; (4) we perform two experiments: a statistical simulation and a user study, which both significantly show the effectiveness of our approach in providing video recommendations based on real online data.

The rest of this paper is organized to first give the survey on related work (Section 2) and then the overview of our approach refined into two phrases (Section 3). Then Sections 4 and 5 describe the detail of our algorithm phrases, followed by Section 6 with experimental procedures and results analysis. Finally, we conclude this paper and indicate its future directions.

2. RALATED WORK

2.1 Sentiment Classification

In supervised sentiment classification methods, standard machine learning techniques such as Support Vector Machine (SVM) and Naive Bayes have been usually used [1]. Different factors affecting the machine learning process are investigated. For example, linguistic, statistical and n-gram features were researched in [7]. Selected words and negation phrases were investigated in [13]. However, the performance of supervised approaches normally decreases when training data is insufficient [2, 18].

On the contrary, unsupervised approaches make the assumption that there are certain words people tend to use to express strong sentiment, so that they might suffice to classify the documents. In [23], an unsupervised sentiment classification approach was proposed by calculating the mutual information between each phrase in a document and the selected two seed words, excellent and poor. Fewer seed words imply less domain-dependency. The authors in [25] only assign one word *good* as a seed positive word, and use negation words such as *not* to find initial negative expressions. In [26], even the one word *good* is ignored. Instead, seed words are automatically generated based on a linguistic pattern (called "negated adverbial construction") like *not very good*.

Although it has been stated in [14] that unsupervised approaches would perform worse than supervised approaches, given that the latter can be built on large training sets, the process of building training sets is unnecessarily time-consuming so as to make large-scale applications impracticable to certain degree [17].

2.2 Recommender Systems

Since 1990s, recommender systems have been explored in many product domains, i.e., movies [6], TVs [20], web pages [3] with the objective of recommending items matched to users' profiles [24]. In recent years, much more techniques have been developed in recommender systems in order to derive better performance [8, 10, 22]. However, most of works are limited when user preference data (i.e.., ratings) are hardly obtainable from real sites (e.g., the video-sharing sites). To address this limitation, tags (in form of user-defined keywords) have been utilized as supplementary source to predict user interests [22]. In [22], the authors proposed a generic method that allows tags to be incorporated into standard CF algorithms, by reducing the three-dimensional correlation to three two-dimensional correlations and then applying a fusion method to re-associate these correlations. The authors in [10] have developed a strategy to infer user interests by applying machine learning techniques to learn from both the "official" item descriptions provided by a publisher, and tags that users used to annotate relevant items.

However, to the best of our knowledge, few have considered user reviews and integrated their sentiment analysis results into the generation of recommendations. In [12] the authors have attempted to identify features from reviews to infer ratings, but no detailed description of how the method was implemented. The sentiment analysis approaches in [9, 11] are all supervised and hence need manually annotated training data. Our work exerts to address this limitation by proposing an unsupervised sentiment classification approach and applying the results to generate ratings on both items and items' keywords, so as to be effectively fused into standard CF algorithm.

As for related works on online video recommendations, [24] proposed a technique to calculate multimodal relevance between videos and users' click-through data. In [15], the proposed video recommender can construct a per-user profile as an aggregation of tag clouds of videos viewed by the user, and then suggest videos based on the viewing patterns of similar users who were identified according to a similarity function over the user profiles. However, few have recognized the potential usefulness of user reviews, as largely appearing in the video-sharing sites, to further enhance their systems' performance.

3. OVERVIEW OF OUR APPROACH

Our recommender algorithm is proposed to study the roles of online reviews when being fused into standard recommender algorithm based on the unsupervised sentiment classification method, with the goal of compensating the limitation of rating sparsity problem and augmenting the applicability of recommenders in realistic online environments like the video recommending. The algorithm consists of two phases. Figure 1 shows the proposed online item recommender algorithm. Phase 1 and Phase 2 are separated by a dash line.

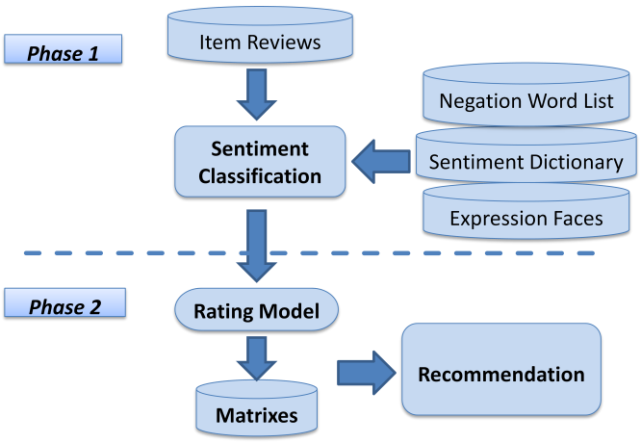


Figure 1: Proposed Online Video Recommender Algorithm.

3.1 Phase 1

In phase 1, an unsupervised sentiment classification approach is designed to automatically label the online item reviews by using three data sets: a sentiment dictionary, an expression face set, and a negation word list. In this approach, the sentiment polarity of sentences is decided mainly based on the sentiment elements (e.g., sentiment words and expression faces) that they contain. After this process, each review r in the reviews set will be assigned a sentiment score denoted by $RS(r)$, and the review will be labeled as positive, negative or neutral, with its $RS(r)>0$, $RS(r)<0$ or $RS(r)=0$ respectively. The details of this approach will be described in Section 4.

3.2 Phase 2

In phase 2, we use the review sentiment classification results from phase 1 to build the *rating model* as input to the user-based CF recommender algorithm to derive personalized recommendations.

First, we build the rating model for users, in which we predict two kinds of ratings, i.e., item ratings and keyword ratings. Then, we build the *user-item rating matrix*. Given a item *i* and a user *u*, the rating of *u* for *i* *r(u,i)* is decided according to the sentiment polarities of all the reviews that the user *u* puts on item *i*. In order to achieve a better performance in recommendations, we also integrate the item keywords into the rating matrix to generate the *user-item-keyword rating matrix*. Finally, we apply the standard user-based CF algorithm. The details of phase 2 will be described in Section 5.

4. PHASE 1 REVIEW SENTIMENT CLASSIFICATION

With a sentiment word set, a negation word list and an expression face set, Phase 1 uses an unsupervised approach to identify the sentiment polarity of reviews. It concretely consists of two steps as follows.

4.1 Initializing Sentiment Element Sets

The sentiment word set, denoted by W_{sen}, includes a list of word items, each of which is assigned with a sentiment score. W_{sen} is initialized by a general sentiment dictionary, which usually includes a lot of positive and negative words. A positive word is assigned with score +1.0, while a negative word is assigned with score -1.0. Monosyllabic words are filtered from W_{sen}, because most of them are too ambiguous to provide reliable sentiment. In addition, since the general sentiment dictionary is applicable to many domains [22, 23], this method has the potential to be domain independent.

The expression face set, denoted by F_{sen}, is the set of the expression faces (e.g., smiley or sad faces) used by the users to express their preferences. Because the faces are widely used by the users in many resource-sharing websites to express their opinions, they play an important role in the task of our item review sentiment classification. First, we manually remove all of the none-sentiment-bearing faces, e.g., [Oh…] and [Well...], from the whole set of crawled expression faces. Then we add the remaining part into F_{sen} and according to the sentiment they express, the faces are divided into two kinds: positive and negative. Each positive face in F_{sen} is assigned with score +1.0, and a negative face is assigned with score -1.0.

For the generation of the negation word list, we manually selected ten most frequently used negation words, such as "不" ('*not*'), "不会" ('*would not*'), "没有" ('*don't have*'), "没" ('*don't have*'), etc. (see the dataset used in the experiment Section 6.1).

4.2 Indentifying Review Sentiment Scores

Through analyzing online reviews, two kinds of reviews have been found in most of view-sharing websites: users expressed their opinions on the items, and/or expressed their opinions on other users' reviews. We call the first kind as *item-oriented reviews* and the second kind as *user-oriented reviews*. Since the sentiment of *user-oriented reviews* is usually not very related to the items directly, we only apply the sentiment classification algorithm on *item-oriented reviews*. There are also some noise reviews including advertisements, hyperlink text, which are not related and removed for the consideration.

Therefore, at first, a pre-processing was conducted to filter out all the *user-oriented* and noise reviews. Given an item *i*, all related reviews of it are denoted by *R(i)*. Each review *r* ($r \in R(i)$) is then divided into clauses by punctuation marks.

Secondly, for each clause, if it contains word items as appearing in W_{sen} (the sentiment word set), each sentiment word item *wi* of a clause is scored by Equation (1), where L_{wi} is the length of the word item, L_{clause} is the length of the clause, S_{wi}^{W} is the word item's score in W_{sen}, and N_{wi} is a negation check coefficient with a default value of 1.0. If the word item is preceded by a negation within the specified zone, N_{wi} is set to -1.0.

$$S_{wi} = \frac{L_{wi}^{2}}{L_{clause}} S_{wi}^{W} N_{wi} \tag{1}$$

Then the sentiment score of a clause *c*, denoted by *CS(c)*, is calculated by $CS(c) = \sum S_{wi}$ for all $wi \in c$. For each review *r*, the *ReviewWordScore* (the sentiment score of a review taking into account of its contained sentiment words), denoted by $RS^{W}(r)$, is subsequently calculated according to Equation (2).

$$RS^{W}(r) = \sum_{c \in r} CS(c) \tag{2}$$

The *ReviewFaceScore* (the sentiment score of a review taking into account of its contained expression faces), denoted by $RS^{F}(r)$, is also calculated according to Equation (3), where S_f is the score of face *f* as appearing in F_{sen}.

$$RS^{F}(r) = \sum_{f \in F_{sen} \cap f \in r} S_f \tag{3}$$

Finally, the sentiment score of the review *r*, denoted by *RS(r)* can be computed using:

$$RS(r) = \alpha RS^{W}(r) + (1-\alpha) RS^{F}(r) \tag{4}$$

where parameter $\alpha \in [0,1]$ determines the weight put on each factor, i.e., the balance between the review's word sentiment score $RS^{W}(r)$ and its facial sentiment score $RS^{F}(r)$.

After this process, each review *r* will be labeled as positive (if $RS(r)>0$), negative ($RS(r)<0$) or neutral ($RS(r)=0$).

5. PHASE 2 ITEM RECOMMENDATION

Because writing reviews is a direct and effective way to show users' preferences on the items in the resource-sharing websites especially the video-sharing ones, we can use the sentimental polarities of reviews as resources for recommendations. On the other hand, the keywords that were typed by contributors (who uploaded the item) as the item's descriptions are also involved in our online video recommender algorithm.

In this phase, we explain how we use the item review sentiment classification results of Phase 1 and item keywords to build the rating model for the input of collaborative filtering recommender algorithm.

5.1 Collaborative Filtering Algorithms

First, we introduce the following notations that we use throughout the rest of the paper.

- U : a set of users.
- I : the set of recommendation items.
- R_{UI}: the *user-item rating matrix* where each value $R_{UI}(u,i)$ correspond to the predicted rating of user u on item i, where $u \in U$ and $i \in I$.
- $V_{UI}(u)$: the rating vector of user u in R_{UI} along a set of items.

R_{UI} theoretically can contain any categorical values. In this paper we consider only binary ratings (1: like, 0: dislike).

Most CF recommender algorithms derive recommendations to a user by using opinions from people who have similar tastes, called neighborhood. Recommendations are generated by considering the ratings of users on items, by computing the pair-wise similarities between the current user and his/her neighbors. One typical method is using the vector cosine similarity. The correlation between user u and v is:

$$S_{UI}(u,v) = \frac{V_{UI}(u) \cdot V_{UI}(v)}{|V_{UI}(u)||V_{UI}(v)|} \quad (5)$$

where u, $v \in U$, and $V_{UI}(u)$and $V_{UI}(v)$ are their rating vectors in R_{UI}.

In this paper, our task is to predict the *top N* interesting items that are unknown to the current user. In user-based CF, to derive the recommendations for a target user u, k most-similar users are selected, which constitute the neighborhood of u, denote by $N(u)$. When predicting the rating of a given user u for an unknown item i, the rating score of i can be computed by:

$$r_{UI}(u,i) = \sum_{v \in N(u)} S_{UI}(u,v) R_{UI}(v,i) \quad (6)$$

We can also predict the rating score of user u for the item i to be a weighted sum of the rates of his/her neighbors [5]:

$$r_{UI}(u,i) = \overline{R_{UI}(u)} + t \sum_{v \in U} w(u,v)(R_{UI}(v,i) - \overline{R_{UI}(v)}) \quad (7)$$

where $\overline{R_{UI}(u)}$ is the mean rating for the user u and the weight $w(u,v)$ reflects the similarity between each user v and the given user u (i.e., the value of $S_{UI}(u,v)$). t is a normalized factor. Then, the *top N* items with the highest $r_{UI}(u,i)$ are selected in the recommendation list for the user u.

A comparison of the two CF methods will be discussed in the section of experiment to see which one performs better in fusing the results of sentiment classification.

5.2 Building the Rating Model

Before applying formula (6) to compute recommendations, we first predict two kinds of ratings of the user u, i.e., item ratings and keyword ratings (see Figure 2). In this model, each user has two rating vectors that are the *user-item vector*, i.e., left part of Figure 2, and *user-keyword vector*, i.e., right part of Figure 2.

The *Like+* parts of the two rating vectors consist of the items and keywords liked by the user u (positive and neutral ones), while the *Dislike or Unknown-* parts consist of the items and keywords disliked or unknown to user u (negative and unknown ones).

After the process of Phase 1, each review r is classified as positive, negative or neutral. In this step, we use the review sentiment classification result, i.e., $RS(r)$ of each review r, and item keywords, to build the rating model.

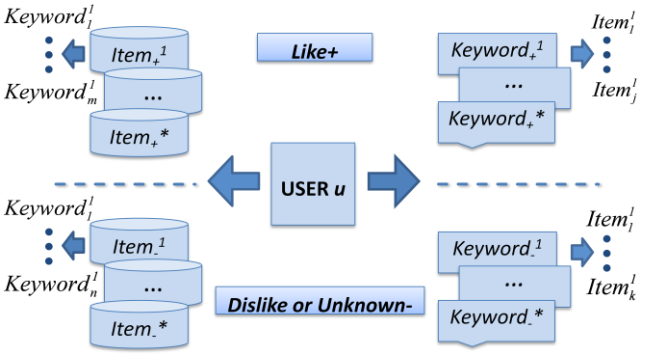


Figure 2: Rating Model of Each User.

5.2.1 Generating Ratings on Items

First, given an item i and a user u, the set of all the reviews that user u puts on item i is denoted as $R(u,i)$. If $|R(u,i)|$ is greater than 0, the sentiment score of item i for user u, denoted by $IS(u,i)$, is computed according to Equation (8). If $|R(u,i)|$ is equal to 0, the value of $IS(u,i)$ is set empty.

$$IS(u,i) = \frac{1}{|R(u,i)|} \sum_{r \in R(u,i)} RS(r) \quad (8)$$

For a user u, we calculate the values of $IS(u)$ for all the items. Then, we build the *user-item vector* in the rating model of u according to the following rules.

- If the value of $IS(u,i)$ is equal or greater than 0 [1], then we add item i into the *Like+* part of the *user-item vector* with the value of 1.
- If the value of $IS(u,i)$ is less than 0 or empty, then we add item i respectively into the *Dislike or Unknown-* parts of the *user-item vector* with the value of 0.

5.2.2 Generating Ratings on Keywords

For each item i, there are n keywords (n>0) describing i, which are provided by the user who uploaded the item on the website. We denote the set of all the keywords of the item i as $K(i)$. If the value of $IS(u,i)$ is not empty, the sentiment score of keyword k for user u, denoted by $KS(u,k)$, is computed according to Equation (9).

$$KS(u,k) = \sum_{i \in I \cap k \in K(i)} \frac{IS(u,i)}{|K(i)|} \quad (9)$$

For a keyword k, if a user u did not put reviews on any of the items that include keyword k, then the value of $KS(u,k)$ is set empty.

[1] According to the habits of the majority online users, if they are interested in the item, they are likely to give a review for it even if the review is neutral ($IS(u,i)$ is equal to 0).

For the user *u*, we calculate the values of *KS(u)* for all keywords. Then, we build the *user-keyword vector* in the rating model of *u* according to the following rules.

- If the value of *KS(u,k)* is equal or greater than 0, then we add keyword *k* into the *Like+* part of the *user-keyword vector* with the value of 1.
- If the value of *KS(u,k)* is less than 0 or empty, then we add keyword *k* respectively into the *Dislike or Unknown-* parts of the *user-keyword vector* with the value of 0.

5.3 Building the Rating Matrix Integrated with Keywords

At this step we build the *user-item rating matrix* R_{UI} as required by CF algorithm (see Section 5.1), and also integrate keywords into the matrix.

First, we build the *user-item rating matrix* R_{UI}. Each $V_{UI}(u)$ (the rating vector of user u) in R_{UI} is set as the values of *user-item vector* of *u* in the rating model.

Then we build the *user-keyword rating matrix*, denoted by R_{UK}. Each $V_{UK}(u)$ (the rating vector of user u) in R_{UK} is set as the values of *user-keyword vector* of *u* in the rating model.

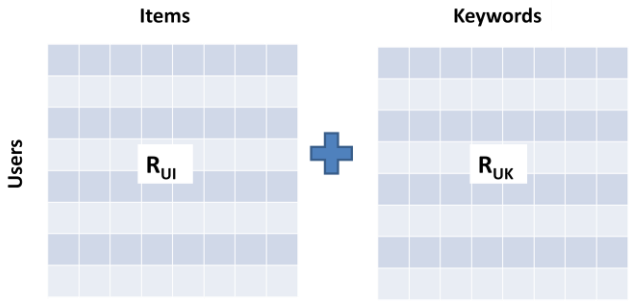


Figure 3: Building R_{UIK}: Integrating R_{UI} with R_{UK}.

Finally, we combine R_{UI} with R_{UK} and get the *user-item-keyword rating matrix*, denoted by R_{UIK} (see Figure 3). Each $V_{UIK}(u)$ (the rating vector of user *u*) in R_{UIK} is the combination of the rating vector $V_{UI}(u)$ and $V_{UK}(u)$, i.e., the first part of $V_{UIK}(u)$ is $V_{UI}(u)$ and second part is $V_{UK}(u)$.

After the integration with keywords, the user-based CF in Section 5.1 is computed and Equation (5) is reformulated as:

$$S_{UIK}(u,v) = \frac{V_{UIK}(u) \cdot V_{UIK}(v)}{|V_{UIK}(u)||V_{UIK}(v)|} \quad (10)$$

where $u, v \in U$, and $V_{UIT}(u)$and $V_{UIT}(v)$ are their rating vectors in R_{UIK}.

Equation (6) is then reformulated as:

$$r_{UIK}(u,i) = \sum_{v \in N(u)} S_{UIK}(u,v) R_{UIK}(v,i) \quad (11)$$

Equation (7) is reformulated as:

$$r_{UIK}(u,i) = \overline{R_{UIK}(u)} + t \sum_{v \in U} w(u,v)(R_{UIK}(v,i) - \overline{R_{UIK}(v)}) \quad (12)$$

The *top N* items with higher $r_{UIK}(u,i)$ are then recommended to the user *u*.

6. EXPERIMENTS

6.1 Data and Tools

The experiments were conducted on the data sets crawled from a popular video-sharing site in China, called YouKu [27], which is a YouTube counterpart in China. We used the video search engine in Youku to crawl the video reviews and keywords. The following nine Chinese queries were used.

> {体育 ti-yu '*sport*', 音乐 yin-yu '*music*', 新闻 xin-wen '*news*', 科技 ke-ji '*science*', 旅游 lv-you '*tourism*', 电影 dian-ying '*movie*' , 原创 yuan-chuang '*originality*' , 汽车 qi-che '*automobile*' , 时尚 shi-shang '*fashion*' }

Finally, we got the data[2] including the reviews and keywords for more than 10,320 videos, each of which had more than 20 reviews and 1 keyword. All the reviews were written in Chinese, while the keywords contained both Chinese and English words.

In Phase 1, a negation word list that contains ten Chinese negations was used:

> {不 bu '*not*', 不会 bu-hui '*would not*', 没有 mei-you '*don't have*', 没 mei '*don't have*', 虽然 sui-ran '*although*', 虽 sui '*although*', 尽管 jin-guan '*although*'', 缺 que '*don't have*', 缺乏 que-fa '*don't have*', 无 wu '*don't have*'}.

For all the experiments, the HowNet Sentiment Dictionary[3] was used as the sentiment dictionary, which contains 4,566 positive words and 4,370 negative words.

6.2 Results of Sentiment Classification

We first tested the accuracy of our sentiment classification method by using a set of data with 1,085 videos and 6,450 users. Each video has at least 100 *video-oriented reviews*, and the total number of reviews is 120,174 in this set. In the experiment we set the value of parameter α in Equation (4) as default 0.3. That is because we considered the facial sentiment score *RSF(r)* was more important than the word sentiment score *RSW(r)* for the sentiment classification. After removing the noise review as mentioned in Section 4.2, we manually labeled the polarities of 1000 reviews that were randomly chosen from the 120,174 reviews. We took the labeled results as the actual polarities of the reviews. As for the reviews that include no users' direct opinions towards the videos, but on the topics extended from the videos, they were labeled as neutral. The numbers of positive, negative and neutral reviews in the labeled set are 553, 339 and 108 respectively, which is the common ratio of most real online reviews [17].

To compare our unsupervised method with supervised classification approaches, a supervised Support Vector Machine (SVM) classifier[4] was used to conduct the sentiment classification with the HowNet Sentiment Dictionary as the feature set using the

[2] http://learn.tsinghua.edu.cn:8080/2006990066/OVRdataset.html

[3] http://www.keenage.com/download/sentiment.rar

[4] WEKA 3.4.11 was used (http://www.cs.waikato.ac.nz/ml/)

famous *tfidf* values. We ran the SVM classifier in 10-fold stratified cross-validation mode.

As the result, Table 1 shows the sentiment classification's precision and recall values from our unsupervised approach (Phase 1) and SVM classifier.

Table 1. Results of the Sentiment Classification.

Method	Review Sentiment	Precision	Recall	F_1
Phase 1	**Positive**	94.38	**94.21**	94.29
	Negative	61.70	80.56	69.88
	Neutral	88.60	80.24	84.21
	Pos+Neu	**97.56**	93.95	**95.72**
SVM	**Positive**	85.83	93.30	89.41
	Negative	58.14	69.44	63.29
	Neutral	82.29	65.78	73.11
	Pos+Neu	91.33	89.83	90.57

From Table 1, we can see that our approach (Phase 1) achieves better results than SVM classifier almost in all the four kinds of reviews (i.e., positive, negative, neutral and positive plus neural) regarding both precision, recall and F_1 measurements. The comparative results hence indicate that our proposed unsupervised approach enables higher accuracy of sentiment classification when being compared to the supervised SVM classifier, that is likely because that our method particularly considers the realistic features of the online item reviews.

On the other hand, from Table 1, we can see that the **Pos+Neu** review set that consists of both positive and negative reviews shows more promising results. Its precision (i.e., the proportion of correctly classified positive and neutral reviews in the set) is 97.56, and the recall (the proportion of correctly classified positive and neutral reviews in actual positive and negative reviews) is 93.95.

In comparison, the classification on negative reviews is less satisfactory by both methods. The reason is probably that most negative opinions are not on the videos' content, but on other topics (e.g., the daily lives of the actors and the social phenomena the videos reflect), and they were labeled as neutral. Also, the sparsity of negative training data led to a low precision of SVM classifier in the negative class.

Thus, the above analysis results indicate that our classification approach is capable of overcoming the challenges of online video reviews' special features and providing reliable results for the building of *user-rating model* in the next phrase of producing video recommendations.

6.3 Results of Recommendations

To compute recommendations, we classified 64,561 positive, 39,576 neutral and 18,037 negative reviews on 1085 videos. The corresponding rating matrix was established for 6,450 users, with generated 61137 video item ratings and 1,958 keyword ratings. We implemented four approaches, i.e., **UI-Ⅰ**, **UIK-Ⅰ**, **UI-Ⅱ** and **UIK-Ⅱ**, to produce the item recommendations (as described in Section 5), and compare with the baseline **Youku** result (in Youku, each video is along with 3 recommended videos mainly based on video popularity).

More specifically, in **UI-Ⅰ** approach, we employed Equation (6) to predict the rating of a given user *u* for an unknown video item *i* and generate the recommendation list. In **UI-Ⅱ** approach, we employed Equation (7) to derive the recommendation list. Compared with Equation (6), Equation (7) complements a normalization process. In **UIK-Ⅰ** approach, Equation (11) was employed that is a reformulation of Equation (6), integrating the keyword rating. It is the same to **UIK-Ⅱ** that employed Equation (12) to fuse the keyword rating by reformulating Equation (7). We set *k* (the number of neighbors for *u*) as 10. The performance of video recommendations was then measured through both statistical evaluation method and user study.

6.3.1 Statistical Experiment Simulation

In the process of statistical experimental simulation, users are often split into training and test sets. The algorithm is trained over the users from the training set and evaluated over the users in the test set [21]. In this paper, we evaluated the accuracy of recommendations using a "cold-start" protocol on the data set. Firstly, we randomly selected 860 (80%) of the items to be training item set, leaving 217 (20%) as testing item set. Then, we selected 500 users with the least item ratings to be test users and evaluated on a test set of $217 \times 500 = 108{,}500$ recommendation lists. The training set consists of 48,647 generated ratings on the 860 training items.

We evaluated the performance of our approach by calculating the precision of a fixed length of recommendation list [8]. We first used the algorithm to derive recommendation items based on the training items related to a test user u. We then defined the per-user precision at *K* recommendation items to be the proportion of the *top K* recommendation items that were actually used by the user in the test set. We averaged the resulting per-user precisions over all 500 test users to get an average precision at *K* recommendation items. Table 2 shows the average precisions respectively at 3 and 10 recommendation items.

Table 2: Average Precisions at 3 and 10 Recommendation Items.

List Set	Precision@3(%)	Precision@10(%)
Youku	2.1	NA
UI-Ⅰ	3.8	3.72
UIK-Ⅰ	4.6	4.46
UI-Ⅱ	3.9	3.77
UIK-Ⅱ	**4.9**	**4.57**

Form Table 2 we can see that the four CF-based fusion approaches obviously outperform **Youku**'s recommendations. The precisions at 3 recommendation set are also slightly better than those at 10 recommendation set respectively by the four algorithms. In particular, **UIK-II** achieves the best result of 4.9 at precision@3(%).

We further conducted a user study to empirically measure the performance of our approach via obtaining users' feedback.

Table 3: Rating Distribution Results for the Nine Recommendation Lists.

	Youku	UI-Ⅰ	UI^{T3}-Ⅰ	UIK-Ⅰ	UIK^{T3}-Ⅰ	UI-Ⅱ	UI^{T3}-Ⅱ	UIK-Ⅱ	UIK^{T3}-Ⅱ
Good (%)	20.1	36.7	40.0	50.1	55.3	35.2	45.1	51.8	**57.5**
Ok (%)	49.4	50.7	46.3	38.0	33.2	**53.4**	43.8	37.6	32.8
Bad (%)	31.5	12.6	13.7	11.9	11.5	11.4	11.1	10.6	**9.7**

6.3.2 User Study

We recruited 80 volunteers participating in the user study. Each volunteer was asked to evaluate the recommendation lists as predicted by the corresponding algorithm for an online user. Each list contained *Top-10* videos sorted by prediction scores. For a user *u*, the volunteer first watched all the videos liked by the user, and then concluded a basic video preference profile for *u*. Then, the volunteer rated the recommendations based on their relevance to the preference of user *u*. The rating was either "Good" (most videos in the list are relevant), "Ok" (some videos are relevant), or "Bad" (most videos are not relevant). Besides evaluating the whole recommender list, the volunteer was also asked to rate the *top 3* videos in each recommendation list because the top-ranked videos are usually more noticeable and relevant (i.e., the results by **UI^{T3}-Ⅰ**, **UIK^{T3}-Ⅰ**, **UI^{T3}-Ⅱ**, and **UIK^{T3}-Ⅱ** in Table 3).

In this experiment, all of the generated 61,137 item ratings based on reviews' sentiment classification results were used as training set in all the four approaches, i.e., **UI-Ⅰ**, **UIK-Ⅰ**, **UI-Ⅱ** and **UIK-Ⅱ**. Each volunteer was shown a series of recommendation lists in a random order (as respectively by the compared methods in Table 3) during each round of evaluation for a user *u*, and was in total required to rate the recommendation lists for at least 50 users. We have 80 volunteers participating in user study, majority voting on 9000 recommendation lists for 1000 online users.

Form Table 3, we can see that the **Youku** lists provides the worst result. The eight lists (as from the four recommender approaches introduced before) all achieve at least total 86% on both Good and Ok ratings. We can also see that the results of **UIK-Ⅰ** and **UIK-Ⅱ** outperform those of **UI-Ⅰ** and **UI-Ⅱ** respectively on Good ratings (50.1% against 36.7%, and 51.8% against 35.2%), which suggests that the integration of keywords can be very useful in further increasing the recommendations' relevance to user preference.

Moreover, in the category of Good ratings, the four **Top3** lists all achieve better results than their corresponding **Top10** lists, among which **UIK^{T3}-Ⅱ** obtains the best result at 90.3% (57.5%+32.8%) on Good and Ok ratings.

The results also clearly show the experimental difference between the two kinds of CF algorithms (e.g., **UI-Ⅰ** vs. **UI-Ⅱ**, and **UIK-Ⅰ** vs. **UIK-Ⅱ**). Almost all **X-Ⅱ** approaches achieve much better performance (such as higher percentages in Good ratings) than their corresponding **X-I** approaches. It hence suggests that the Equation (7) (and (11)) can be better adopted to fuse the sentiment results (and keywords) to produce better recommendations. .

7. CONCLUSION AND FUTURE WORK

In this paper, we proposed an unsupervised sentiment classification approach to meet the special characteristics of the real online video reviews and furthermore developed an online video recommender algorithm that particularly exploited the sentiment classification results to automatically build the rating matrix. The experimental results through both the evaluations on sentiment classification results and recommendations show that our new approach achieves higher performance in augmenting the video recommendations in realistic online environments.

In fact, since our algorithm has no restriction on the type of collaborative-filtering algorithms used, the method can be easily scaled and incorporated into other types of CF recommenders, such as using Trust Inferences in [16] and Boltzman Machines in [8]. Our contribution is indeed primarily to the generation of user-item rating matrix based on user reviews, so as to complement the rating sparsity limitation of current video-sharing sites when they attempt to apply the standard pure rating based CF techniques. Moreover, the rating matrix in our system can in essence contain any categorical values, besides the binary rating (like 1/dislike 0) that were assumed in this paper.

In the future, we will perform more studies to further optimize our algorithm, including the determining of the optimal value for the parameter α in Equation (4) so as to get the best balance between word sentiment score *RSW(r)* and facial sentiment score *RSF(r)* for the sentiment classification. We will be also engaged in further classifying the reviews into more delicate categories in addition to "like" and "dislike/unknown" ones. On the other hand, the similar experimental procedures will be conducted to other review languages (e.g. English) in order to validate our method's cross-language applicability.

8. ACKNOWLEDGMENT

The authors acknowledge the support received from the National Natural Science Foundation of China (Grant No. 60972096) and the National 863 Plans Projects (Grant No. 2009AA01Z410).

REFERENCES

[1] Ethem Alpaydin. 2004. Introduction to Machine Learning. *The MIT Press*, Cambridge.

[2] Anthony Aue and Michael Gamon. 2005. Customizing sentiment classifiers to new domains: a case study. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, Borovets, BG.

[3] M. Balabanovic. 1998. Exploring versus exploiting when learning user models for text recommendation. *User Modeling and User-Adapted Interaction*, 8(4):71 – 102.

[4] John Blitzer, Mark Drezde, and Fernando Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, CZ.

[5] J. S. Breese, D. Heckerman, and C. Kadie. 1998. Empirical analysis of predctive algorithms for collaborative filtering. In *UAI '98*, pages 43–52.

[6] C. Christakou and A. Stafylopatis. 2005. A hybrid movie recommender system based on neural networks. In *Proceedings of the 2005 5th International Conference on Intelligent Systems Design and Applications*, Wroclaw, Poland.

[7] Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the Peanut gallery: opinion extraction and semantic classification of product documents. In *Proceedings of WWW2003*, Budapest, HU.

[8] Asela Gunawardana, Christopher Meek. 2009. A Unified Approach to Building Hybrid Recommender Systems. In *RecSys '09: Proceedings of the 2009 ACM conference on Recommender systems*, pages 117 – 124, New York, USA.

[9] Gayatree Ganu, Noémie Elhadad, Amélie Marian. Beyond the Stars: Improving Rating Predictions using Review Text Content. *Twelfth International Workshop on the Web and Databases*, Providence, Rhode Island, USA, 2009.

[10] Marco de Gemmis, Pasquale Lops, Giovanni Semeraro and Pierpaolo Basile. 2008. Integrating Tags in a Semantic Content-based Recommender. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 163–170, Lausanne, Switzerland.

[11] Niklas Jakob, Stefan Hagen Weber, Mark-Christoph Müller, Iryna Gurevych. Beyond the Stars: Exploiting Free-Text User Reviews to Improve the Accuracy of Movie Recommendations. *TSA'09*, Hong Kong, China, 2009.

[12] C. W. ki Leung, S. C. fai Chan, and F. lai Chung. Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *ECAI-Workshop on Recommender Systems*, 2006, pp. 62–66.

[13] J.C. Na, H. Sui, C. Khoo, S. Chan and Y. Zhou. 2004. Effectiveness of simple linguistic processing in automatic sentiment classification of product reviews. In *I.C. McIlwaine (Ed.), Knowledge Organization and the Global Information Society: Proceedings of the Eighth International ISKO Conference*, pages 49-54, Wurzburg, Germany: Ergon Verlag.

[14] Bo Pang, Lilian Lee, and Shrivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of Conference on Empirical Methods in Natural Language Processing 2002*.

[15] Jonghun Park, Sang-Jin Lee, Sung-Jun Lee, Kwanho Kim, Beom-Suk Chung, Yong-Ki Lee. 2010. An Online Video Recommendation Framework Using View Based Tag Cloud Aggregation. *IEEE MultiMedia, IEEE computer Society Digital Library. IEEE Computer Society*.

[16] M. Papagelis, D. Plexousakis, and T. Kutsuras. 2005. Alleviating the Sparsity Problem of Collaborative Filtering Using Trust Inferences. *Proc. 3rd Int'l. Conf. Trust Management (iTrust 05)*, Springer, pp. 224 – 239.

[17] Likun Qiu, Weishi Zhang, Changjian Hu, Kai Zhao. 2009. SELC: A Self-Supervised Model for Sentiment Classification. In *Proceedings of CIKM'09*, Hong Kong, China.

[18] Jonathon Read. 2005. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL-2005 Student Research Workshop*, Ann Arbor, MI.

[19] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. 2000. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the Second ACM Conference on Electronic Commerce (EC'00)*, pages 285 – 295.

[20] M. V. Setten and M. Veenstra. 2003. Prediction strategies in a TV recommender system – method and experiments. In *Proceedings of International World Wide Web Conference*, Budapest, Hungary.

[21] Guy Shani, Max Chickering, Christopher Meek. 2008. Mining Recommendations From The Web. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 35 – 42, Lausanne, Switzerland.

[22] Karen H. L. TsoSutter, Leandro Balby Marinho and Lars Schmidt-Thieme. 2008. Tag-aware Recommender Systems by Fusion of Collaborative Filtering Algorithms. In *Proceedings of SAC'08*, pages 1995-1999, Fortaleza, Brazil.

[23] Peter Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of documents. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics*, New Brunswick, N.J.

[24] Bo Yang, Tao Mei, Xiansheng Hua, Linjun Yang, Shiqiang Yang and Mingjing Li. 2007. Online Video Recommendation Based on Multimodal Fusion and Relevance Feedback. In *Proceedings of CIVR'07*, pages 73 – 80, Amsterdam, The Netherlands.

[25] Taras Zagibalov and John Carroll. 2008a. Unsupervised Classification of Sentiment and Objectivity in Chinese Text. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pp. 304–311.

[26] Taras Zagibalov and John Carroll. 2008b. Automatic Seed Word Selection for Unsupervised Sentiment Classification of Chinese Test. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pp. 1073-1080.

[27] http://www.youku.com

[28] http://www.youtube.com

Using Personality Information in Collaborative Filtering for New Users

Rong Hu
Human Computer Interaction Group
Swiss Federal Institute of Technology (EPFL)
CH-1015, Lausanne, Switzerland
rong.hu@epfl.ch

Pearl Pu
Human Computer Interaction Group
Swiss Federal Institute of Technology (EPFL)
CH-1015, Lausanne, Switzerland
pearl.pu@epfl.ch

ABSTRACT
Recommender systems help users more easily and quickly find products that they truly prefer amidst the enormous volume of information available to them. Collaborative filtering (CF) methods, making recommendations based on opinions from "most similar" users, have been widely adopted in various applications. In spite of the overall success of CF systems, they encounter one crucial issue remaining to be solved, namely the cold-start problem. In this paper, we propose a method that combines human personality characteristics into the traditional rating-based similarity computation in the framework of user-based collaborative filtering systems with the motivation to make good recommendations for new users who have rated few items. This technique can be especially useful for recommenders that are embedded in social networks where personality data can be more easily obtained. We first analyze our method in terms of the influence of the parameters such as the number of neighbors and the weight of rating-based similarity. We further compare our method with pure traditional ratings-based similarity in several experimental conditions. Our results show that applying personality information into traditional user-based collaborative filtering systems can efficiently address the new user problem.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *information filtering*; H.1.2 [**Models and Principles**]: User/Machine Systems – *human information processing*

General Terms
Algorithms, Performance, Experimentation

Keywords
Recommender System, User-based Collaborative Filtering, Personality, User Similarity, New User

1. INTRODUCTION
The ubiquity of the web brings an explosive increase of accessible information. Recommender systems have emerged as an intelligent information filtering tool to help users effectively identify information items of interest from such an overwhelming set of choices and provide personalized services [25]. At the same time, recommender systems are considered to be a critical tool for boosting sales in e-commerce websites [2]. Therefore, variations of recommendation algorithms have been widely studied and incorporated in a wide range of online commercial websites [1].

Collaborative filtering (CF) is one of the most successful recommendation technologies. The basic idea behind this method is that it gathers the opinions of other users who share similar interests with a target user (referred to as the "*active user*") and assists this active user to identify items of interest based on these *neighbors*' opinions. These social information filtering approaches automate a process of "word-of-mouth" recommendations [29]. Compared to other recommendation technologies (e.g., content-based filtering), CF provides some prominent advantages to information filtering: (i) the capability to filter items whose content is not easily analyzed by automated processes; (ii) the ability to provide serendipitous recommendations; and (iii) support for social factors by taking into account the interests of like-minded users [10, 17]. Consequently, CF has been becoming popular in both academy and industry fields with great speed.

Despite the overall success of CF systems, they suffer one serious limitation, namely the *cold-start* problem [1]. It includes two major aspects: *new user* and *new item*. Before a recommender system can present a user with reliable recommendations, it should know about this user's preferences/interests, most likely from a sufficient number of behavior records, e.g., ratings or log-archives [11]. Therefore, a new user, having few records in a system, normally cannot get satisfied recommendations. Similar to the new user problem, new items which have not been rated could not be recommended, which is referred to as new item problem. In our study, we will focus on the new user problem. It is a key issue that determinants the initial success of e-retailers, since more accurate recommendations for new users could make these new users stay rather than pushing them to the competitors' sites.

To address the new user problem, most studies present hybrid recommender systems that combine both content information and ratings data [22, 27, 28] to circumvent the problem, where content-based similarity is used for new users or new items. In most currently used systems, demographic information is used as users' attributes to calculate similarity among users. For example, Pazzani [23] uses the gender, age, area code, education, and employment information of users in the restaurant recommendation application.

Recently, many studies have tried to incorporate human personality into recommender systems [6, 12, 13, 16]. Studies show that personalities influence human decision making process and interests [24]. Drawing on the inherent inter-related patterns among users' personalities and their interests/behaviors, personality-based recommenders were developed to provide personalized services. Empirical studies further revealed a significant user acceptance of such recommenders [12]. Additionally, research has suggested that human personality characteristics have the potential ability to lessen the cold start problem associated with commonly adopted collaborative filtering recommender systems [6]. However, few works have empirically verified this hypothesis.

As mentioned in [6], one apprehension of few researchers venturing into the personality-based recommender systems might be due to perceived difficulty in obtaining personality characteristics. However, it can be foreseen that social network sites have the capability of facilitating the personality acquisition processes. The primary functionality of Social Web is to help people socialize or interest with each other throughout the World Wide Web. During the forming of social community, personality is considered as one of the users' identities. On the other hand, personality profiles can be helpful to suggest more connections or enhance the relationships among friends, such as seeking friends with similar personality. Due to the particularity of the social web, people have a strong interest to do personality tests and share such information with their friends. One evidential observation is that there are a range of personality test applications at facebook.com and an amount of users are involved.

The main objective of our work is to investigate the performance of utilizing personality information in user-based CF systems to address the new user problem. We first propose a personality-based similarity measure and a general model which takes both the traditional rating-based similarity and the personality-based similarity into account to find the neighbors of an active user. We compared our method with the pure rating-based CF systems in different cold-start settings. The results positively support the advantage of the personality-based similarity in improving recommendation quality, at least in the case of sparse dataset, for new users.

The remainder of this paper is organized as follows. Section 2 provides a brief overview of some related research work. Section 3 describes the algorithm of traditional user-based CF systems. The section following presents our proposed personality-based similarity in detail. Section 5 describes our experimental work, including experiment design, evaluation metrics, dataset, experimental results and discussion, followed by the section on conclusions.

2. RELATED WORK

In this section, we briefly view some of the research literature related to our work.

2.1 New User Problem

A crucial concern of CF is the new user problem, which refers to difficulties encountered by recommender algorithms when a new user enters into a system. Various approaches have been proposed in previous studies. Most of them leverage hybrid recommendation approaches, which combine content meta-data and ratings to circumvent this problem [1, 7, 22].

Ahn [2] addresses this problem by proposing a new similarity measure called PIP (Proximity-Impact-Popularity) which utilizes domain specific interpretation of user ratings by considering proximity, impact and popularity factors when comparing two ratings. Different from other solutions, this method attempts to make better use of the limited amount of ratings data so as to address the new user problem and improve recommendation performance. However, it cannot address the difficulty of recommending items for completely new users with no ratings.

Some studies follow the idea of associating any new user with a stereotype among a set of predefined ones from learning processes [20]. For example, Pazzani [23] utilizes user's demographic information (e.g., age, gender, education) to identify stereotypes of users that like a certain object. When any new user comes, he/she is associated with one stereotype based on his/her demographic data and gets tailored recommendations. In this work, the demographic information is taken from new users' homepages automatically without extra user effort.

Alternatively, other researchers have suggested that further improvements dealing with cold start in CF systems can be achieved by leveraging user characteristics, specially detailed characteristics [15]. Even though few studies have been done on this topic, existed research has shown its promise [e.g., 15, 18, 20]. Particularly, personality is considered as a consistent behavior pattern and intrapersonal processes originating within the individual [24]. It is relatively stable and predictable. Therefore, it is reasonable to suppose that it is possible to address the cold start problem and possibly improve prediction in current CF systems by incorporating personality characteristics.

2.2 Personality-based Recommender Systems

Recently, personality characteristic information has been integrated into recommendation techniques. Lin and Mcleod [16] proposed a temperament-based filtering model incorporating human factors, especially human temperament, into information recommendation service. They empirically demonstrated that the accuracy and effectiveness of the temperament-based information filtering system surpassed those of pure content-based filtering. Nunes et al. [21] proposed one personality-based recommender by incorporating personality traits into user profiles for the social matching applications. Their system was designed under the scenario of the "Elections for President in France". The system recommends one president candidate to a user by matching the candidates' social reputation profiles (summarized personality profiles from all participates) with the personality profile of the user's ideal president. Their method obtains a high prediction accuracy.

Hu and Pu [13] developed a personality-based music recommender system based on the psychological findings of the correlations between human personality characteristics and musical preferences. For example, individuals who are inventive, have active imaginations, value aesthetic experiences, consider themselves to be intelligent, tolerant of others, and reject conservative ideals tend to enjoy listening to reflective and complex music (e.g., blues, jazz, classical and folk). Their results from an in-depth user study revealed the user acceptance of this personality-based system under the scenarios of making recommendations for active users and their friends. This work mainly emphasizes on user subjective perception of this emerging personality-based music recommender system.

In this paper, we focus on investigating how personality characteristics can be integrated in the commonly used CF framework and how its performances is compared to the traditional user-based CF systems, especially relative to the new user problem.

3. USER-BASED COLLABORATIVE FILTERING

User-based Collaborative Filtering has been studied in-depth during the last decades. It is one of the most successful and widely used recommendation technologies, owing to its compelling simplicity and excellent quality of recommendations. It assumes that if a group of users have similar interests in their previous behaviors, they will express similar interests on other more items in the future. Its basic idea is to find a group of users, who have a history of agreeing with an active user (i.e., they either gave similar ratings or purchased similar items). Once a neighborhood of users is formed, opinions from these neighbors are aggregated to produce recommendations for the active user.

Various algorithms for user-based CF can be grouped into two classes: *memory-based* (or heuristic-based) and *model-based* [1]. Memory-based algorithms essentially are heuristics that make rating predictions based on the entire collection of previously rated items. In contrast to memory-based methods, model-based algorithms use the collection of ratings to learn a *model*, which is then used to make rating predictions. In this paper, we concentrate on memory-based algorithms. In order to simply, the term CF mentioned in the following represents user/model-based CF, except particular specification.

3.1 Similarity Measure

The most important step in CF recommender systems is computing the similarity between users which is used to form a proximity-based neighborhood between a target user and a number of like-minded users. The neighborhood finding process is in fact the model-building or learning process for a recommender system algorithm. Various approaches have been used to compute the similarity $simr(u,v)$ between user u and user v [1, 2, 26]. The letter r means the similarity is calculated based on rating data, distinguishing from the personality-based similarity measure proposed later. The most commonly used similarity calculation method is *Pearson correlation coefficient* [2, 26]. More specifically, the proximity between user u and v is measured as,

$$simr(u,v) = \frac{\sum_{i \epsilon I_u \cap I_v} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \epsilon I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2 \sum_{i \epsilon I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}, \quad (1)$$

where $r_{u,i}$ denotes the rating value which user u gave to item i. $\bar{r}_u$ is the average rating value of user u. I_u is the set of items that user u has rated.

In order to penalize similarity scores that are based on a small number of overlapping items which reflects a lack of confidence, a modified similarity score $simr'(u,v)$ is yielded as follows [22]:

$$simr'(u,v) = \frac{\min(|I_u \cap I_v|, \gamma)}{\gamma} * simr(u,v), \quad (2)$$

where γ controls the required number of overlapping items between two users for similarity calculation. We adopt this modified correlation-based similarity score in our evaluation experiment and set $\gamma = 5$.

3.2 Rating Prediction

To predict the value of unknown rating $r_{u,i}$, CF systems make an aggregation of the ratings from some other (usually, the k most similar) users for item i. More specially, the predicted unknown rating $\tilde{r}_{u,i}$ can be calculated as,

$$\tilde{r}_{u,i} = \text{aggr}_{v \in \Omega_u} r_{v,i}, \quad (3)$$

where Ω_u denotes the set of k users that are the most similar to user u and who have rated item i. Various aggregation strategies are designed and applied based on different applications, such as, averaging the ratings, or using similarities as weights while aggregating. In this paper, we adopt a more general aggregation function,

$$\tilde{r}_{u,i} = \bar{r}_u + \kappa \sum_{v \in \Omega_u} sim(u,v) \times (r_{v,i} - \bar{r}_v), \quad (4)$$

where multiple κ serves as a normalizing factor and is usually selected as $\kappa = 1/\sum_{v \in \Omega_u} |sim(u,v)|$, and $\bar{r}_u$ is the average rating of user u. This aggregation function takes into account the fact that different users may use the rating scale differently. For example, user u thinks a rating 3 in a 5-point rating scale means that this item is ok, while user v might thins a rating 3 represent a negative score. Therefore, the weighted sum uses the deviations from the average ratings instead.

4. PERSONALITY-BASED SIMILARITY

4.1 Personality-based Similarity Measure

As described above, traditional collaborative filtering systems find "neighbors" based on the ratings they gave in common. However, in practice, the item-user matrix *R* is always sparse. That is, the number of ratings already obtained is usually significantly small compared to the number of the ratings that need to be predicted. It is crucial to effectively predict ratings from a small number of examples. If one user has rated some items which few users have been rated, it will be not possible to find a sufficient number of neighbors and to make good recommendations. This situation especially happens when a new user enter into a system with few and no rating records. In order to address this problem, we proposed a new similarity measure method based on users' personality characteristics.

We treat users' personality characteristics as a vector. For user u, his/her personality descriptor $p_u = (p_u^1, p_u^2, \dots, p_u^n)^T$ is a n-dimension vector, and each dimension stands for one characteristic consisting in his/her personality. For example, if users' personalities are measured by Big Five Factor model [8] which describes human personality along major five traits, p_u is a five-dimension vector and each dimension corresponds to one of the five personality traits (details see Section 4.2). Consequently, the personality similarity between two user u and v can be computed as the Pearson correlation coefficient of their personality descriptors.

$$simp(u,v) = \frac{\sum_k (p_u^k - \overline{p_u})(p_v^k - \overline{r_v})}{\sqrt{\sum_k (p_u^k - \overline{p_u})^2 \sum_k (p_v^k - \overline{p_v})^2}}. \quad (5)$$

To consider both rating-based and personality-based similarity measures at the same time, we combine them together and intuitively generate the following model,

$$sim(u,v) = \alpha * simr'(u,v) + (1-\alpha) * simp(u,v), \quad (6)$$

where $simr'(u,v)$ represents the item-based simialrity between user u and v, and $simp(u,v)$ represents the personality-based similarity. α is a weight parameter which controls the percentage

of rating-based similarity contributed into the final similarity measurement.

4.2 Personality Model and Measurement

One of the most widely used and extensively researched personality models within psychology is known as Big Five Factor personality model [5, 8]. This model categorizes human personality traits into five bipolar dimensions:

- *Openness to Experience*: appreciation for art, emotion, adventure, unusual ideas, curiosity, and variety of experience.
- *Conscientiousness*: a tendency to show self-discipline, act dutifully, and aim for achievement; planned rather than spontaneous behavior.
- *Extroversion*: energy, positive emotions, urgency, and the tendency to seek stimulation in the company of others.
- *Agreeableness*: a tendency to be compassionate and cooperative rather than suspicious and antagonistic towards others.
- *Neuroticism*: a tendency to experience unpleasant emotions easily, such as anger, anxiety, depression, or vulnerability.

In literature, several rating instruments have been developed to measure the Big-Five dimensions, from well-established multi-item instruments (e.g., NEO Personality Inventory, or Revised NEO-PI-R [5]) to extremely brief ones (e.g., 5-Item Personality Inventory (FIPI) or 10-Item personality Inventory (TIPI) [8]). Even though the comprehensive instruments have the superior capability of assessing finer facets within each personality dimension, they require more user effort to accomplish it than the brief ones. In the context of online system, users would be unlikely to dwell at the website for a long time to complete a multi-item questionnaire [21]. In our implementation, therefore, TIPI developed by Gosling et al. [8], is utilized. In each Big Five dimension, there are two items which attributes in the two poles of this dimension. Each item consists of two descriptors, separated by a comma, for example, "Extraverted, enthusiastic" in the positive direction of extraversion dimension. Each item was rated on a 7-point scale ranging from 1 (strongly disagree) to 7 (strongly agree). The TIPI personality acquisition process takes about 2-3 minutes to complete.

5. EMPIRICAL ANALYSIS

In this section, we first describe the experiment setup, including experiment design, evaluation metrics and used dataset. Then, we present experimental results and discussions. Our main goal is to explore the possibilities of combining different similarity measures to formulate an efficient recommendation algorithm.

5.1 Experiment Design

In this experiment, we try to compare the predictive accuracy of the three similarity measures on a sparse dataset. The dataset is split into a *training set* and a *test set*. We adopt the all-but-one protocol considering the high sparsity of our dataset [4]. That is, we randomly select one of the rating entries of each user to be the tested item, and use the remaining entries for training. Consequently, as much data as possible from each test user can be used to train the recommenders. It is meaningful when we evaluate the effects of the parameters in our general model.

5.2 Evaluation Metrics

We adopt two kinds of evaluation metrics in our evaluations. One takes into account the ability of accurately predict ratings in the case of individual item-by-item recommendations. The other considers the recommendation filtering task as a two-class classification, like or dislike. In this case, evaluation metrics measures how effectively positive and negative items are classified correctly.

5.2.1 Predictive Accuracy Metric

First, we try to measure how close the predicted ratings generated by various algorithms are to user real ratings. Mean absolute error (MAE) is the most prominent and broadly adopted predictive accuracy metric in information retrieval and recommender community [9]. It is calculated as the average of the differences between predicated ratings $\tilde{r}_i$, and users' real ratings r_i. More specially, MAE is formulated as:

$$MAE = \frac{\sum_{i=1}^{n} |\tilde{r}_i - r_i|}{n},$$

where n is the number of tested items.

Even though it is important to measure the ability of accurately predicting ratings for a recommendation algorithm, we can observe that the majority of users only care about whether a recommended item is interesting or not, rather than exact rating values. In these cases, filtering is considered as a binary classification and the predicted rating errors might not really reflect the filtering performance. For example, if a rating of 3.5 is considered as the cut-off between good and bad, a one-star error that predicts a 4 as 5 and one that predicts a 3 as 4 are totally different. The former makes no difference to users, while the latter erroneously classifies a bad item as a good item. Therefore, we utilize decision-support accuracy metrics as well in our evaluation.

5.2.2 Decision-Support Accuracy Metrics

Decision-support accuracy metrics (also called classification accuracy metric [9]) evaluate a recommender system based on whether it makes correct or incorrect decision outcome, i.e., whether it correctly predicts that an item might be interesting for an active user [7]. To evaluate recommendations generated by different similarity methods, we use two metrics widely used in the information retrieval (IR) community namely, *recall (also called sensitivity)* and *specificity*. However, we slightly modify the definition of them to fit in our all-but-one experiment design. We first defined the rating 3.5 as our cut-off threshold on a 5-point rating scale from 1 to 5. That is, all ratings which are greater than 3.5 are considered as "good" (or "relevant"), others as "bad" (or "irrelevant").

We define the collection of all tested items for user u in l (l = 20, in our experiment) runs of our experiments as *check set* (in order to distinguish from the *test set* used in the evaluation of top-N recommendation list [19, 26]) T_u, which contains all items to be predicted for this user. Furthermore, T_u can be divided into two sets: relevant set REL_u containing all relevant items and irrelevant set $IRREL_u$ containing all irrelevant items. All *hits*, items in the set of REL_u are predicted to be good as well, are include in the *hit set* HIT_u. All *avoids*, items in the set of $IRREL_u$ are predicted to be bad as well, are include in the *avoid set* AV_u.

We define *recall* as the ratio of hit set size to the relevant set size. More specifically, considering the average of all n tested users, it is formularize as:

$$recall = \frac{\sum_u \frac{|HIT_u|}{|REL_u|}}{n},$$

where n is the number of users tested. A recall value of 1.0 indicates that the commendation algorithm was able to retrieval all relevant items, whereas a recall vale of 0.0 indicates that the recommendation algorithm was not able to recommend any of the relevant items.

Along with recall, *specificity* is defined as the proportion of irrelevant items which are correctly identified. More specifically, it is formalized as:

$$specificity = \frac{\sum_u \frac{|AV_u|}{|IRREL_u|}}{n}.$$

A specificity value of 1.0 indicates that the all irrelevant items are successfully eliminated from recommended items, whereas a specificity vale of 0.0 indicates that all irrelevant items are incorrectly classified as good ones and might be recommended to users.

5.3 Dataset

Currently, most available test datasets only contain user rating records (e.g., the MovieLens dataset[1] and the EachMovie dataset[2]) or contents of items (e.g., IMDB[3]). To the best of our knowledge, none of the available datasets contains both users' personality information and their ratings. Therefore, we can only conduct our experiment on a music data set that we have accumulated in our previous study [13]. In that study, users were asked to answer a personality questionnaire based on the Big Five Model and rate the recommended songs. This dataset includes 1,581 songs (1956 – 2009) covering 14 genres in four musical preferences. We only considered users who rated 20 or more songs. Therefore, the reduced data set include 113 users and 646 songs that were rated by at least one of the users. Each user has a 5-dimensional personality descriptor along five traits as well. We compute the *sparsity level* of the dataset as [26],

$$sparsity\ level = \frac{1-\#\text{non entries}}{\#\text{total entries}}.$$

The statistical characteristics of this data set are shown in Table 1.

5.4 Experiment Results

In presenting our evaluation experimental results, we firstly investigate the influences of various parameters in the general model. Then, we compare the performances of rating-based similarity (RBS), personality-based similarity (PBS) and their hybrid (RPBS) in different start-up settings.

5.4.1 Influence of Model Size

The traditional user-based recommendations are computed using a model that utilizes the ratings from k most similar users to predict the unknown items for the active user. To evaluate the sensitivity of the different algorithms on the value of k, we performed an experiment in which we let k take the values from 5 to 110 in increments of 5. Note that these results were obtained using the value of parameter $\alpha = 0.5$ for the hybrid similarity measure. The results are shown in Figure 1.

Table 1. Statistical characteristics of our ratings data set.

Size	#users	113
	#items	646
	#ratings	2479
Sparsity	sparsity level	96.604%
	Ave. #ratings per user	21
	Ave. #ratings per item	3
Rating Distribution	#ratings on the value of 1	254
	#ratings on the value of 2	470
	#ratings on the value of 3	747
	#ratings on the value of 4	585
	#ratings on the value of 5	423

Figure 1(a) shows the MAE results regarding different neighbor sizes. As we can see, MAE results for rating-based CF reaches the minimal value when $k = 40$ and keeps it while the number of neighbors increases. It happens in traditional user-based CF when the item-user matrix R is sparse. Even thought the threshold of the number of neighbor is designed to be high, the actual neighbors can be found in the dataset is few due to the small overlap of ratings. However, the personality-based similarity has not this restraint. The similarity can be calculated only if the personality characteristics vectors of two users are known. Therefore, the MAE values for the other two algorithms can keep decreasing in the tested range until the number of neighbors reaches 90 where the improvement become gently.

Figure 1(b) and (c) respectively shows the recall and specificity results under different neighbor sizes. Along with the results on MAE, the quality for the rating-based CF method increases in the beginning, but remains on one value when the number of neighbors reaches 50 and 40 respectively, while other two methods still increase their recall and specificity until the size increases to the point around 90. Therefore, in our later experiments, we assign the number of neighbors to be 90.

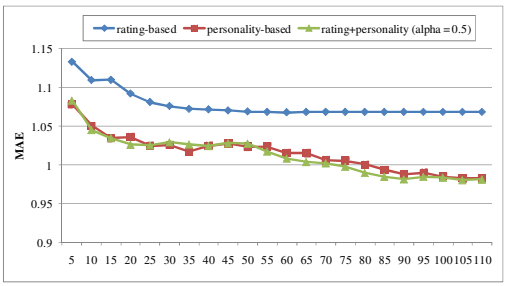
5.4.2 Influence of Weight

One parameter in the combined similarity model, α, is used to control the extent to which the rating-based similarity measurement can contribute. To study the sensitivity of the recommendation algorithm on this parameter, we performed a sequence experiments in which we varied α from 0.0 (pure personality-based CF) to 1.0 (pure rating-based CF) in increments of 0.1. Figure 2 shows the results of MAE, recall and specificity for different values of α. The higher recall and specificity values and lower MAE value indicate better performance. Note that these results were obtained by assigning the number of neighbor $k = 90$.

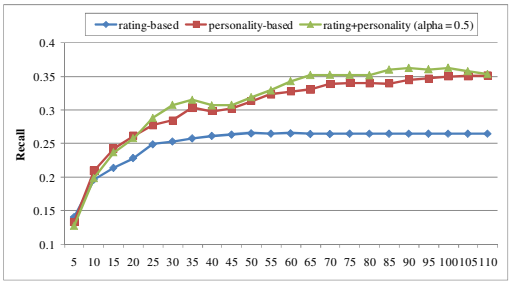
[1] http://www.movielens.org

[2] http://www.research.compaq.com/SRC/eachmovie

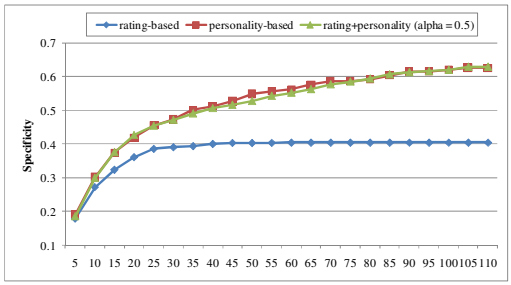
[3] http://www.imdb.com



(a) MAE



(b) Recall



(c) Specificity

Figure 1. The influence of the number of neighbors.

The results in Figure 2 show that the hybrid similarity can achieve the best recommendation quality on MAE, recall and specificity when the value of α is around 0.5. However, as we can see, the performance is not highly sensitive to the changes of α in this setting, as long as α is less than 0.9. It might because we choose a big size of neighbors for displaying best performance for each method. However, the number of actual neighbors in rating-based method cannot reach this value due to the sparsity of the dataset. In this case, personality-based similarity will play a dominate role no matter which value α is chosen to be except the value "1". In our later experiments, we assign the parameter α to 0.5 to investigate the recommendation performance on different start-up settings.

5.4.3 Influence of Training Size

The main objective of our work is to evaluate the performance of personality-based similarity in addressing new user problem. In this section, we will compare the three different similarity measures in different start-up settings with training sets of varying degrees of sparsity. As our previous experiment setup, we randomly choose one item for each one user as the test item, and others are used as training data. Differently, we further randomly select parts of the ratings from the original set with the percentage of μ to form our training sets of different degree of sparsity. We increase μ from 10% to 100% in increments of 10%, so that we have 10 start-up settings for each μ. When sampling, we keep all new training sets have the same distribution in each rating scale (from 1 to 5) as the original set, as did in [3]. We perform 20 runs for each test item. The average results are shown in Figure 3.

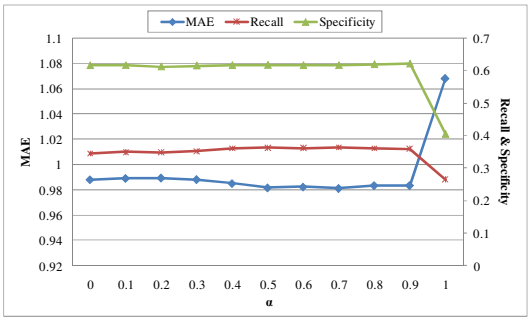
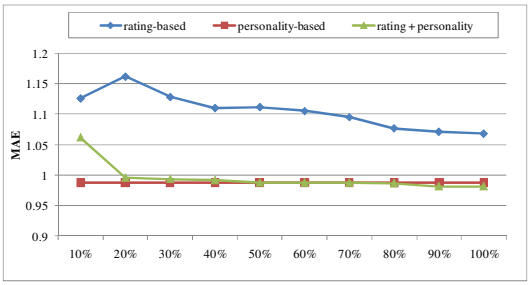


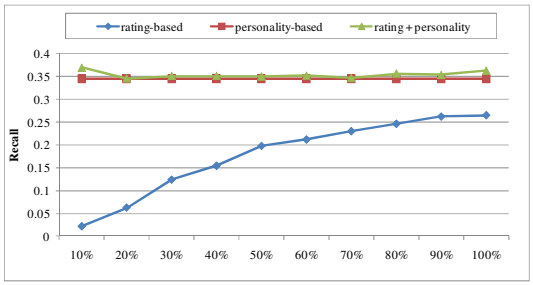
Figure 2. The recommendation quality on different values of parameter α from 0.0 (personality-based) to 1.0 (rating-based).

The result shows superior performance of the personality-based similarity and the hybrid similarity measures over the traditional rating-based one in the situation of sparse user-item matrix, especially when the sizes of training sets are small. To specifically show the improvements, we present the results under the settings of $\mu = 50\%$ and $\mu = 100\%$, considering RBS as the baseline. When the training sets include 50% of all the ratings, there is an improvement of 11% for both PBS and RPBS on MAE. The increases of recall are 74% and 77% for PBS and RPBS respectively, and the increases of specificity are 117% and 111%. When training sets include 100% of the overall ratings (In this setting, there are 20 training ratings on average for each user), we get a 7% decrease on MAE for PBS and an 8% decrease for RPBS. At the same time, there are improvements of 30% and 37% on recall for PBS and RPBS respectively and 52% on specificity for both PBS and RPBS.

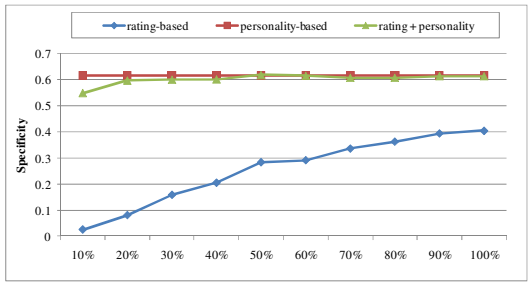
Along with the results shown in the previous sections, the performance of personality-based similarity is somewhat similar to that of the hybrid method. It is because we choose a large size of neighbors and our testing dataset is sparse.



(a) MAE



(b)Recall



(c) Specificity

Figure 3. The comparison of recommendation quality in different start-up settings. X-axis represents of percentage of training set sizes.

5.5 Limitations

Firstly, the experimental data set is relatively small. It only contains 113 users, 646 songs and 2479 ratings. The statistical results might be sensitive to the distribution of the used dataset. More experiments using larger data sets are needed to verify the findings of this work. Secondly, we didn't evaluate the performance of each similarity measure using a dataset whose user-item matrix has a higher density. In that situation, users have more co-rated items so that the rating-based similarity measure could work more effectively. It is interesting and valuable to compare these three similarity measures in such situations so as to find out how personality-based similarity could work there. Finally, we only used a music dataset in our current evaluation experiment. It is still unclear how results would become if these similarity measures are used in other domains. Music is a special product domain, since the relationship between musical preferences and human personalities has been revealed. It is still an open issue whether the personality-based similarity measure can perform as well in other domain as it does in music domain. More works are needed to deal with these issues in the future.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed to incorporate personality-based similarity into user-based CF recommender systems to address the new user problem. We experimentally evaluated the recommendation quality in terms of mean absolute error (MAE), recall and specificity by comparing the personality-based similarity measurement, the traditional rating-based one and their hybrid. Our results showed that both personality-based similarity and the hybrid scheme lead CF recommender systems to generate more accurate recommendations than the traditional rating-based one in a sparse music dataset. In our test setting with 100% training data (on average, 20 ratings per user), in contrast to RBS, PBS has 7% improvement on MAE, 30% improvement on recall and 52% improvement on specificity. With respect to RPBS, there are at least 8% improvement on MAE, 37% on recall and 52% on specificity. In addition, our results from different start-up settings positively support that the personality-based similarity measure can effectively address the new user problem adhere to the traditional user-based CF recommender systems.

However, since the dataset used in our current experiment is relatively small, more evaluation studies in bigger datasets are needed to verify our findings. Additionally, it is meaningful to investigate the performance of the personality-based similarity measure in a dense dataset where users have many co-rated items and to explore the possibility of generalizing to other item domains.

7. ACKNOWLEDGMENTS

We thank the EPFL and the ministry of education of the People's Republic of China for supporting the reported research work.

8. REFERENCES

[1] Adomavicius, G. and Tuzhilin, A. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. Knowledge and Data Eng.*, 17, 6(2005), 734-749.

[2] Ahn, H. J. 2008. A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences* 178:37-51.

[3] Basu, C., Hirsh, H., and Cohen, W. 1998. Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*. C. Rich, and J. Mostow, Eds. AAAI Press 1998.

[4] Breese, J. S., Heckerman, D., and Kadie, C.1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*. G. F. Cooper, and S. Moral, Eds. Morgan-Kaufmann, San Francisco, Calif., 43–52.

[5] Costa, P.T. and McCrae, R.R. 1992. NEO PI-R Professional Manual. In: Psychological Assessment Resources, Odessa, FL.

[6] Dunn, G., Wiersema, J., Ham, J., and Aroyo, L.2009. Evaluating Interface Variants on Personality Acquisition for Recommender Systems. In: *Houben, G.J., McCalla, G., Pianesi, F., Zancanaro, M. (eds.) User Modeling, Adaptation, and Personalization*. LNCS, vol. 5535, pp. 259--270. Springer, Heidelberg.

[7] Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J.1999. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*, J. Hendler, and D. Subramanian, Eds. AAAI Press, Menlo Park, Calif., 439–446.

[8] Gosling, S. D., Rentfrow, P. J., and Swann, Jr. W.B.2003. A very brief measure of the Big-Five personality domains. *Journal of Research in Personality*, 37, pp. 504--528.

[9] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl J. T.2004. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Information Systems*, vol. 22, no. 1, pp. 5-53.

[10] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J.1999. An algorithmic framework for performing collaborative filtering. *In Proc. of SIGIR*.

[11] Hofmann, T.2004. Latent semantic models for collaborative filtering. *ACM Trans. Info. Syst.*, vol 22(1):89-115.

[12] Hu, R. and Pu, P.2009. A comparative user study on rating vs. personality quiz based preference elicitation methods. In: *Proceedings of the 13th international conference on Intelligent User Interfaces*, pp. 367—372.

[13] Hu, R. and Pu, P. 2010. A Study on User Perception of Personality-Based Recommender Systems. In: *P. De Bra, A. Kobsa, and D. Chin (Eds.): UMAP 2010*, LNCS 6075, pp. 291–302.

[14] John, O.P. and Srivastava, S.1999. The Big-Five Trait Taxonomy: History, Measurement, and Theoretical Perspectives. In: Pervin, L., John, O.P. (eds.) Handbook of Personality: Theory and Research, 2nd edn., pp. 102–138. Guilford, New York.

[15] Lam, X.N., Vu, T., Le, T.D. and Duong, A.D. 2008. Addressing Cold-Start Problem in Recommendation Systems. In: ICUIMC 2008, pp. 208–211. ACM Press, New York.

[16] Lin, C. and McLeod, D.2002. Exploiting and Learning Human Temperaments for Customized Information Recommendations. *IMSA*, 218-233.

[17] Linden, G., Smith, B., and York, J.2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, Jan/Feb.:76-80.

[18] Lekakos, G. and Giaglis, G.M. 2006. Improving the Prediction Accuracy of Recommendation Algorithms: Approaches Anchored on Human Factors. Int. with Comp. 18, 410–431.

[19] Karypis, G.2001. Evaluation of Item-Based Top-N Recommendation Algorithms, In *Proceedings of the tenth international conference on Information and knowledge management*, October 05-10, Atlanta, Georgia, USA.

[20] Nguyen, A., Denos, N. and Berrut, C. 2007. Improving New User Recommendations with Rule-Based Induction on Cold User Data. In: RecSys 2007, pp. 121–128. ACM Press, New York.

[21] Nunes, M. A. S. N., Cerri, S. A., and Blanc, N. 2008. Improving recommendations by using Personality Traits. In: *International Conference on Knowledge Management. I-KNOWS08*, Graz-Austria.

[22] Park, S.-T., Pennock, D. M., Madani, O., Good, N., and DeCoste, D.2006. Naive filterbots for robust cold-start recommendations, in: *Proceedings of KDD ' 06*, ACM, Philadelphia, PA, USA

[23] Pazzani, M.1999. A Framework for Collaborative, Content-Based, and Demographic Filtering. *Artificial Intelligence Rev.*, pp. 393-408, Dec.

[24] Rentfrow, P. J. and Gosling, S. D.2003. The do re mi's of everyday life: The Structure and Personality Correlates of Music Preferences. *Journal of Personality and Social Psychology*, 84, 1236—1256.

[25] Resnick, P. and Varian, H. R.1997. Recommender Systems. *Commun. ACM 40*, 56-58.

[26] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J.2000. Analysis of recommendation algorithms for E-commerce. In *Proceedings of the 2nd ACM Conference on Electronic Commerce (EC'00)*. ACM, New York. 285–295.

[27] Salter, J. and Antonopoulos, N. 2006. CinemaScreen recommender agent: combining collaborative and content-based filtering, *IEEE Intelligent Systems* 21, 35 - 41.

[28] Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. 2002. Methods and metrics for cold-start recommendations, in: *Proceedings of SIGIR ' 02*, ACM, Tampere, Finland, pp. 253 - 260.

[29] Shardanand, U. and Maes, P.1995. Social information filtering: Algorithms for automating "word of mouth", in *Proceedings of ACM Conference on Human Factors and Computing Systems*, pp. 210–217, Association of Computing Machinery, New York.

Rating items by rating tags

Fatih Gedikli
Department of Computer Science
44221 Dortmund
Germany
fatih.gedikli@tu-dortmund.de

Dietmar Jannach
Department of Computer Science
44221 Dortmund
Germany
dietmar.jannach@tu-dortmund.de

ABSTRACT

Different proposals have been made in recent years to exploit Social Web tagging data to improve recommender systems. The tagging data was used for example to identify similar users or viewed as additional information about the recommendable items.

In this work we propose to use tags as a means to express which features of an item users particularly like or dislike. Users would therefore not only add tags to an item but also attach a preference or rating to the tag itself, expressing, for example, whether or not they liked a certain actor in a given movie. Since rating data is in general sparse in commercial recommender applications we also present how to infer the user opinion regarding a certain feature (tag) for a given item automatically. In contrast to previous works, we not only infer the user's general preference for a tag but rather determine this preference in the context of a certain item.

An evaluation on the MovieLens data set reveals that our new tag-enhanced recommendation algorithm is slightly more accurate than a recent tag-based recommender even when the explicit tag rating data is 100% sparse, that is, if only derived information can be used.

1. INTRODUCTION

User-contributed tags are today a popular means for users to organize and retrieve items of interest in the participatory Web. Social Tagging plays an increasingly important role both on Social Web platforms such as Delicious[1] and Flickr[2] as well as on large-scale e-commerce sites such as Amazon.com.

Different ways of exploiting these additionally available pieces of information to build more effective recommender systems have been proposed in the last years. For example, tags can be seen as item descriptions and used by a content-based recommender. The set of tags a user attaches to resources also provides valuable information about the user. Thus, the relationship between users and tags can be used to find similar users in neighborhood-based collaborative filtering systems.

The goal of these tag-based approaches is to exploit the existing interactions between users, items and tags to improve the effectiveness of the recommender system, measured in terms of the predictive accuracy or the coverage of the algorithm [4, 5, 6, 18, 21, 23, 26].

In their recent work, Sen et al. [18] explore another way of leveraging tagging information to generate more precise recommendations. The strategy of their so-called "tagommenders" is to automatically infer the user's *preference for individual tags*. In the movie domain, the first task thus consists of determining if and to which extent the user Alice likes movies that are, for example, annotated with the tag "animated". After that, the rating prediction for an item is based on the aggregation of the inferred user preferences for the tags assigned to that item. An analysis of several algorithms and preference inference metrics on a tag-enhanced MovieLens data set showed that more precise recommendations can be made when the user's tag preferences are taken into account.

In the work by Sen et al., the inferred preferences or ratings for tags are "global" in a sense that a tag is either liked or disliked, independent of a specific item. Thus, a particular user Alice either likes movies annotated with the tag *animated* or not. In our work we explore whether *allowing users to give ratings for a tag in the context of an item* can help to further improve the accuracy of recommendations. The intuition behind this idea is that the same tag may have a positive connotation for the user in one context and a negative in another. For example, a user might like *action movies* featuring the actor *Bruce Willis*, but at the same time this user might dislike the performance of Bruce Willis in *romantic movies*.

Our general goal is to explore a new Social Web recommendation approach, in which *users rate items by rating the corresponding tags*. This corresponds to a multi-criteria or multi-dimensional recommendation approach as described in [1] or [2]. In order to analyze the potential value of such multi-criteria ratings and the corresponding richer user models we measure whether we can increase the accuracy of a tag-based recommender by using *inferred* tag ratings only. We aim to demonstrate that rating items by rating tags works on principle in this work.

The paper is organized as follows. In the next two sections, we will outline the overall preference inference and

[1] http://www.del.icio.us
[2] http://www.flickr.com

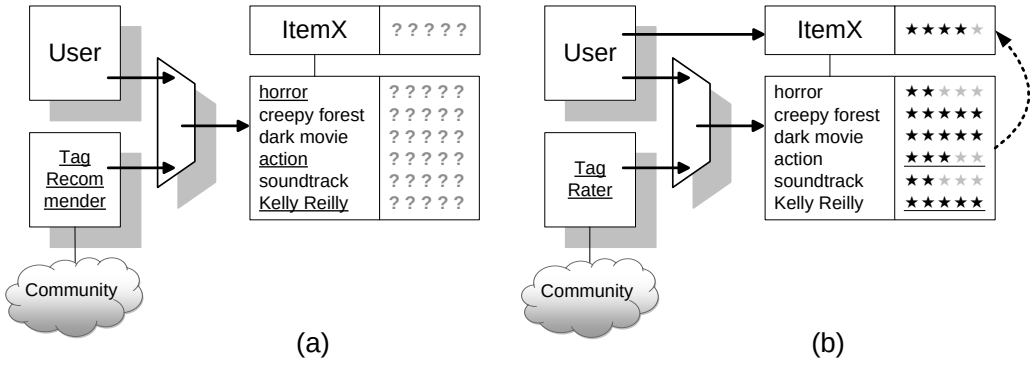


Figure 1: (a) Tagging items (automatically). (b) Rating items by rating tags (automatically) and recommending items based on tag ratings.

recommendation process and give details of our new user- and item-aware method for tag preference prediction and recommendation. In Section 4, the results of a comparative evaluation of the method on the tag-enhanced MovieLens data set are discussed. The paper ends with a discussion of related approaches and an outlook on future work.

2. ILLUSTRATIVE EXAMPLE

We illustrate the basic rationale of our method in the following example. Let us assume *User1* has attached tags to different movies[3] and given overall ratings on a scale from 1 to 5 as shown in Table 1. *User1* particularly likes action movies featuring Bruce Willis and romantic movies featuring Sandra Bullock, but appears to dislike romantic movies starring Bruce Willis.

Movie	Tags	Rating
M1	Bruce Willis, action, ...	5
M2	Bruce Willis, romance, ...	2
M3	Bruce Willis, action, ...	5
M4	Sandra Bullock, romance, drama, ...	5
M5	Bruce Willis, romance, drama, ...	**?**

Table 1: Tags and overall ratings of *User1*.

A method that automatically infers global preferences or ratings for tags such as the one described in [18] would probably derive a relatively high value for the tags "Bruce Willis" and "action". At the same time, the tag "romance" would receive a luke-warm rating somewhere between 3 and 4 because the user attached the tag both to a highly-liked and a disliked movie. As a result, the rating prediction for movie $M5$ based on the inferred tag ratings would may be around 4, that is, the system would tend to recommend $M5$.

Now let us assume that we knew more about the individual tags and their importance to *User1* as shown in Table 2 (assuming that we acquired this information directly from the user). In Table 2, we can see, that – perhaps among other reasons – the user did not like $M2$ because of Bruce Willis' appearance in a romantic movie. Since movie $M5$ is quite similar to $M2$ with respect to the attached tags, it is somewhat more intuitive *not* to recommend $M5$, which is exactly the opposite decision as in the example above.

Movie	Tags	Rating
M1	Bruce Willis (5), action (5), ...	5
M2	Bruce Willis (1), romance (2), ...	2
M3	Bruce Willis (5), action (5), ...	5
M4	Sandra Bullock (4), romance (5), ...	5
M5	Bruce Willis **(?)**, romance **(?)**, ...	**?**

Table 2: Tags and detailed ratings of *User1*.

We therefore propose a method that is capable of making recommendations based on more detailed rating data for tags. In addition, we develop a method to infer these detailed tag rating data automatically for cases in which such information is not available or the data are very sparse. In the example above, we would try to approximate the detailed ratings for the items $M1$ to $M5$ as good as possible given only the overall ratings for the movies.

Figure 1 summarizes and visualizes our approach to "rating items by rating tags" for a movie recommendation scenario in which both explicit and inferred tag ratings are exploited. At the core, the usual user-item matrix is extended not only by a set of user-provided tags for the items, but also by tag ratings describing the user's opinion about the item features represented by these tags. Rating items by rating tags consists of two phases. In the first phase, the user assigns one or more tags to the item to be rated. Figure 1 (a) shows the process of assigning tags to items. The user can either create new tags or select existing quality tags in the sense of [16] from the recommendation list of a tag recommender[4]. In the second phase, each individual tag can be given a rating (Figure 1 (b)), that is, the user rates selected tags and assigns an overall rating to the movie. These tag ratings can either be acquired explicitly using some extended recommender system user interface or derived automatically in case no such explicit information is available. In this work we only rely on automatically derived tag ratings as we do not possess real tag rating data yet.

In the next section, we present one possible neighborhood-based metric to derive tag ratings from the overall ratings

[3]These tags can also be derived from the community.

[4]See, for example, the tag recommendation systems of [25] or FolkRank [9].

automatically, see the "Tag Rater" component in Figure 1 (b). Then we propose a metric which is used to derive an overall rating prediction for a not-yet-seen item based on the ratings of its tags (see dotted arrow in Figure 1 (b)).

3. METHOD

In order to infer implicit tag ratings from the data and predict item ratings for a user, we used the following metrics and algorithms.

Similar to [18] and [22], we use a metric $w(m,t)$ that measures the *relevance* of a tag t for an item m. Note that in a setting, where users rate tags, the same tag can be applied many times for the same resource. In our approach, we use the following simple counting metric to determine a tag's relevance, which gives more weight to tags that have been used by users more often to characterize the item[5]:

$$w(m,t) = \frac{\textit{number of times tag } t \textit{ was applied to item } m}{\textit{overall number of tags applied to item } m} \quad (1)$$

In [18], the prediction $\hat{r}_{u,t}$ of the general interest of a user u in the concept represented in a tag t is calculated as follows (method `movie-ratings`):

$$\hat{r}_{u,t} = \frac{\sum_{m \in I_t} w(m,t) * r_{u,m}}{\sum_{m \in I_t} w(m,t)} \quad (2)$$

In this equation, I_t corresponds to the set of all items tagged with t. The explicit overall rating that u has given to movie m is denoted as $r_{u,m}$. The general idea of the method is thus to propagate the overall rating value to the tags of a movie according to their importance.

In our work, however, we are interested in predicting the rating for a tag in the context of the target user u *and* the target item i. Note that the rating prediction in Equation (2) does not depend on the target item i at all. Our tag prediction function, $\hat{r}_{u,i,t}$, for a given user u and an item i is calculated as follows:

$$\hat{r}_{u,i,t} = \frac{\sum_{m \in similarItems(i,I_t,k)} w(m,t) * r_{u,m}}{\sum_{m \in similarItems(i,I_t,k)} w(m,t)} \quad (3)$$

Instead of iterating over all items that received a certain tag as done in [18], we only consider items that are similar to the item at hand, thereby avoiding the averaging effect of "global" calculations. In Equation (3), the calculation of neighboring items is contained in the function $similarItems(i, I_t, k)$, which returns the collection k of the most similar items from I_t. The similarity of items is measured with the adjusted cosine similarity metric. We also ran experiments using the Pearson correlation coefficient as a similarity metric, which however led to poorer results. As another algorithmic variant, we have tried to factor in the item similarity values as additional weights in Equation (3). Again, this did not lead to further performance improvements but rather worsened the results.

When using the user's explicit overall rating $r_{u,m}$ as in Equation (2) or (3), no prediction can be made for the tag rating if user u did not rate any item m tagged with t, i.e., if $I_t \cap ratedItems(u) = \emptyset$. We therefore apply the recursive prediction strategy as described in [24] and first calculate a prediction for $r_{u,m}$ if it is not given. An additional weight value $w_{rpa}(r_{u,m})$ is applied to the recursively predicted value $\hat{r}_{u,m}$ where $w_{rpa}(r_{u,m})$ is defined as follows:

$$w_{rpa}(r_{u,m}) = \begin{cases} 1, & r_{u,m} \text{ is given} \\ \lambda & r_{u,m} \text{ is not given} \end{cases} \quad (4)$$

The combination weight threshold λ is a value between $[0,1]$. The RPA strategy can therefore be incorporated in the tagommender approach of [18] by extending Equation (2) leading to the following form:

$$\hat{r}_{u,t} = \frac{\sum_{m \in I_t} w(m,t) * w_{rpa}(r_{u,m}) * \mathcal{R}(r_{u,m})}{\sum_{m \in I_t} w(m,t) * w_{rpa}(r_{u,m})} \quad (5)$$

The function $\mathcal{R}(r_{u,m})$ either returns $r_{u,m}$ if such a rating exists or an estimated value for $r_{u,m}$ based on the Recursive Prediction Algorithm [24]. Similar to Equation (5), we also applied the RPA strategy to our extended prediction function shown in Equation (3).

The overall rating prediction for an item m for a user u based on the (predicted) tag ratings given the user's average item rating ($\overline{r_u}$) and the user's average tag rating for a given item ($\overline{r_{u,m}}$) is given in Equation (6). Similar to the algorithm `cosine-tag` in [18], we calculate $\hat{r}_{u,m}$ as follows, where T_m is the set of all tags applied to m:

$$\hat{r}_{u,m} = \overline{r_u} + \frac{\sum_{t \in T_m} sim(m,t) * (\hat{r}_{u,m,t} - \overline{r_{u,m}})}{\sum_{t \in T_m} sim(m,t)} \quad (6)$$

The individual tag ratings are weighted according to the adjusted cosine similarity between items and tags, see Equation (7). The similarity metric given in Equation (7) is used to measure the degree of consistency between the item's overall rating received by all users u who rated item m (U_m), and their predicted tag ratings for that item.

$$\text{sim(m,t)} = \frac{\sum_{u \in U_m} (r_{u,m} - \overline{r_u})(\hat{r}_{u,m,t} - \overline{r_{u,m}})}{\sqrt{\sum_{u \in U_m} (r_{u,m} - \overline{r_u})^2} \sqrt{\sum_{u \in U_m} (\hat{r}_{u,m,t} - \overline{r_{u,m}})^2}} \quad (7)$$

4. EVALUATION

In order to measure the predictive accuracy of the presented methods we evaluated our approach on the popular MovieLens data set using a common experimental procedure and well known accuracy metrics. The results of this evaluation are described in this section. The goal of our subsequent evaluation is to analyze if the predictions made by the system are more accurate when we use more detailed tag information even for cases where all tag ratings are automatically derived, i.e., the sparsity level of the explicit tag rating data is 100%.

4.1 Data set and algorithms

4.1.1 Data set

We evaluated our approach to recommendation of items based on tag ratings on the "MovieLens 10M Ratings, 100k Tags" data set[6], which consists of three files: *ratings*, *movies*

[5]Further possible metrics to determine tag relevance are described in [18].

[6]`http://www.grouplens.org/node/73`

and *tags*. The ratings file contains a list of user ratings on a 5-star scale with half-star increments. The movies file contains information about each movie such as the title and the genre, which are, however, not used in our method. The tags file contains the information about which tags have been assigned by the users to the movies. A tag assignment is a triple consisting of one user, one resource (movie) and one tag. No rating information for the tags themselves is available in the original MovieLens database.

To the best of our knowledge, the 10M MovieLens data set is the only publicly available data set which contains both rating and tagging data. It contains 10,000,054 ratings and 95,580 (unrated) tags applied to 10,681 movies by 71,567 users of the online movie recommender service MovieLens.

4.1.2 Tag quality and data pruning

Limited tag quality is one of the major issues when developing and evaluating approaches that operate on the basis of user-contributed tags. In [16], for example, Sen et al. revealed that only 21% of the tags in the MovieLens system had adequate quality to be displayed to the user. Therefore, different approaches to deal with the problem of finding quality tags have been proposed in recent years, see, for example, [7], [16] or [17].

Note that our approach of rating items by rating tags calls for a new quality requirement to tags: tags must be appropriate for ratings. For example, there is no point in attaching a rating to a tag like "bad movie" because the tag already represents a like/dislike statement. It would therefore not be clear how to interpret a rating for such a tag. In our current work and evaluation, we did not take this question into account yet, that is, we did not distinguish between tags that are appropriate for being rated and those which are not. Still, we believe that this is one key question which was not considered before and which should be taken into account in future approaches to extracting rating information for tags automatically.

We applied traditional data pruning measures in order to improve the quality of the existing tag information. In particular, we defined the following two comparably weak requirements for movies and tags to be taken into account in our evaluation. First, we only consider movies that have at least two tags assigned. Second, only those tags are considered that were assigned by at least two users. Note that these constraints are not as strong as the constraints applied in previous works. In [22], for example, the authors require that "a tag has been applied by at least 5 different users and to at least 2 different items". Additionally, content analysis methods were applied in [22] to detect redundant tags, such as *violent* and *violence*, in order to replace them by one representative tag.

Beside the measures taken to improve the tag quality we further applied a random-based subsampling method to avoid problems with memory limitations. The resulting data set used in our experiments finally contained 4,713 movies, 3,979 users, 134,829 ratings and 77,127 tags.

4.1.3 Algorithms

The goal of our analysis is to determine whether recommending items based on user- and item-specific tag ratings can lead to more precise results than previous approaches that only use item-independent tag ratings, even when the tag ratings are inferred automatically.

We compared the following algorithms:

- **TBR-UI** (tag-based recommender with user- and item-specific ratings): it uses the metric in Equation (3) to predict tag ratings and the function in Equation (6) as well as the similarity metric from Equation (7) to predict item ratings.
- **TBR-UI-RPA**: same as above, but with the additional application of the Recursive Prediction Algorithm as described in Section 3.
- **TBR-U** (user-specific ratings only): corresponds to the `movie-ratings` / `cosine-tag` algorithm variant from [18].
- **TBR-U-RPA**: same as above, but with the additional application of the Recursive Prediction Algorithm.
- **Item-Item**: the classical item-to-item baseline recommendation scheme that does not exploit tag information at all. Adjusted cosine is used as a similarity function. Rating predictions are calculated as follows:

$$\hat{r}_{u,m} = \frac{\sum_{i \in ratedItems(u)} sim(m,i) * r_{u,i}}{\sum_{i \in ratedItems(u)} sim(m,i)} \qquad (8)$$

Note that we have used adjusted cosine as a similarity metric for all schemes. Experiments with other similarity metrics such as the Pearson correlation coefficient, however, led to poorer results.

In the user- and item-based schemes, TBR-UI(-RPA), also the parameter k which determines the size of the neighborhood containing the k most similar items from I_t can be varied, see Equation (3). In order to find an optimal value we performed 4-fold cross-validation and varied the parameter. A neighborhood-size of 3 was determined as an optimal choice. Besides this, the combination weight threshold parameter λ in Equation (4) is set as 0.5 as suggested as an optimal value in [24].

4.1.4 Accuracy metrics

In our experiments, we measured the predictive accuracy with the help of different metrics. First, we used the usual *Root Mean Squared Error* (RMSE) metric in order to make our results comparable with the results in literature. Note that we also calculated *Mean Absolute Error* values (MAE), but do not report these numbers here because no significant differences to the RMSE values have been observed. Because of the different criticisms on the RMSE measure, we measured the quality of the recommendations produced by the different algorithms also with the standard information retrieval metrics *precision* and *recall*.

To determine precision and recall, we followed the evaluation procedure proposed in [14] and converted the rating predictions into "like" and "dislike" statements as described in [15], that is, ratings above the user's mean rating are interpreted as "like" statements. In each of the iterations of a four-fold cross-validation procedure, the data set is split into a training set (75% of the data) and a test set (25% of the data). We then determined the set of existing "like" statements (ELS) in the 25% test set and retrieve a top-N recommendation list of length $|ELS|$ with each method based on the data in the training set. The top-N recommendation lists are created based on the prediction score of each

method. The set of predicted like statements returned by a recommender shall be denoted as *Predicted Like Statements (PLS)*, where $|PLS| \leq |ELS|$.

Based on these definitions, *precision* can be defined as $\frac{|PLS \cap ELS|}{|PLS|}$ and measures the number of correct predictions in PLS. *Recall*[7] is measured as $\frac{|PLS \cap ELS|}{|ELS|}$ and describes how many of the existing "like" statements were found by the recommender.

In the evaluation procedure, recommendations and the corresponding precision and recall values were calculated for all users in the data set and then averaged. These averaged precision and recall values are then combined in the usual F-score, where

$$F = 2 * \frac{precision * recall}{precision + recall}$$

4.2 Results and discussion

Table 3 shows the average values for the F-score as well as the individual precision and recall values for the different algorithms in increasing order. We can observe that the previous tag-based `cosine-tag` method (TBR-U) slightly outperforms the traditional item-to-item recommendation scheme, a fact, which was already observed in [18]. The usage of the recursive prediction scheme helps to further improve recommendation accuracy.

Scheme	F-score	Precision	Recall	RMSE
Item-Item	80.26	81.09%	79.45%	1,1363
TBR-U	80.70	81.51%	79.90%	1,1521
TBR-U-RPA	80.77	81.59%	79.98%	1,1353
TBR-UI	81.11	81.92%	80.32%	1,1355
TBR-UI-RPA	81.75	82.56%	80.95%	1,1355

Table 3: Average F-score, precision, recall and RMSE values for different algorithms.

Our new (TBR-UI) method proposed in this paper in turn outperforms both the item-based method as well as the recent `cosine-tag` method from [18]. Again, RPA helps to further improve the results. The overall improvements are about 1.5 points on the F-score compared with the tag-unaware method and about 1 point when compared with the Sen et al.'s method.

As an interesting side effect, note that improvements can be consistently observed both in the precision and the recall values, that is, the new user- and item-specific scheme does not introduce a new tradeoff between these general goals.

Our algorithm also shows very light improvements on the RMSE metric compared with the item-based approach, see Table 3. While these improvements are less significant, the results indicate that our technique does also not lead to a deterioration on this metric.

Overall, our results demonstrate the potential value of a new recommendation approach that is based on the principle of rating items by rating tags because quality improvements could be achieved even in situations when the available tag-rating database is 100% sparse and all tag ratings are automatically inferred. Based on these observations, we expect further quality improvements in settings, in which also explicit tag ratings are available and where the quality of the tags is also higher than in our experiments.

[7] In [14], this metric is called *coverage*.

5. RELATED WORK

In recent years, many researchers have recognized the value of Social Web tagging information for recommender systems and for example use tagging data as an additional source of information to improve the effectiveness of the recommender system, measured in terms of the predictive accuracy or the coverage of the algorithms [5, 11, 12, 18, 21, 23, 26].

The work that is most closely related to our approach is the recent work of Sen et al. [18]. The authors present tag-based recommender algorithms called "tagommenders" which compute a rating prediction for a target item and a target user by exploiting the user's *inferred tag preferences*, that is, the rating prediction for an item is based on the aggregation of the inferred user preferences of its tags. An analysis of several algorithms and preference inference metrics on the tag-enhanced MovieLens data set revealed that more precise recommendations can be made when the user's tag preferences are taken into account. The concept of *tag preference* was also used by Vig et al. [22] and describes "the user's sentiment toward a tag". In other words, a tag preference determines if and to which extent the user likes or dislikes items that have the feature represented by the tag. Consider again, for example, the tag "Bruce Willis" in the movie domain. In this example, tag preference would measure the degree a user likes or dislikes *all* movies in which Bruce Willis appeared. As stated above, in the approach in [18], the inferred tag preference is "global" in the sense that a user either likes or dislikes movies with that actor, independent of a particular movie.

To the best of our knowledge, the concept of tag preference was first introduced by Ji et al. [10]. The authors present a tag preference based recommendation algorithm for a collaborative tagging system, where collaborative tagging means the process of assigning tags to items by many users which is supported by Social Web platforms such as Delicious[8] or Connotea[9]. The authors first compute the target user's *candidate tag set* which consists of all tags for which a high tag preference value was predicted. Afterwards a naive Bayes classifier is used for making recommendations by exploiting the user's candidate tag set. The proposed algorithm was evaluated on a data set collected from the social bookmarking site Delicious. In contrast to the work of Sen et al. [18] the tag preference predictor in [10] does not make use of item ratings at all because the Delicious data set does not support ratings for items (bookmarks) like the tag-enhanced MovieLens data set.

In Vig et al. [22] the authors propose another concept called *tag relevance* which describes "the degree to which a tag describes an item". In the example above, tag relevance would measure how well the tag "Bruce Willis" describes a particular movie. Overall, in previous works *tag preference* was considered a user-specific concept whereas *tag relevance* is considered to be an item-specific concept. In contrast, in our work the proposed concept of a *tag rating* is user- and item-specific which has shown to be a helpful means to capture the user's preferences more precisely and thus

[8] `http://www.del.icio.us`
[9] `http://www.connotea.org`

produce more accurate recommendations.

In [5], the authors exploit tagging data for an existing content-based recommender system in order to increase the overall predictive accuracy of the system. In their approach, the user interests are learned by applying machine learning techniques both on the textual descriptions of items (static data) and on the tagging data (dynamic data). Tags are therefore only considered as an additional source of information used for learning the profile of a particular user. By conducting a user study with 30 users the authors show that a slight improvement in the prediction accuracy of the tag-augmented recommender compared to the pure content-based one can be achieved. Our work rather represents a collaborative filtering with multi-criteria ratings and is thus better capable to exploit the "wisdom of the crowd" to improve the recommendation accuracy.

In recent times, tags were also used for enhancing the performance of traditional collaborative filtering recommender systems. Tag information was incorporated into existing collaborative filtering algorithms in one or the other way for enhancing the quality of recommendations for example in [11, 12, 21, 23] or [26]. Most commonly, tags are considered only as an additional source of information for their proposed methods. In [23], for example, tags are used for building user and item neighborhoods. The underlying idea of this approach is that neighbors that are determined in this way will be better predictors than those which are identified only based on explicit rating data. The evaluation on the tag-enhanced MovieLens data set shows that such an approach outperforms other algorithms based on non-negative matrix factorization and Singular Value Decomposition. In particular, the observed improvements in predictive accuracy were comparably strong for sparse data sets.

In contrast to works in which tags are only used to build better neighborhoods for classical collaborative filtering systems, we introduce a different way of exploiting tags for recommender systems in this work. We propose a new approach, in which *users rate items by rating tags*. This can be seen as being a sort of a multi-criteria or multi-dimensional recommendation approach as described in [1], [2] or [13]. In [1], Adomavicius and Kwon conjecture that multi-criteria ratings will play an important role for the next generation of recommender systems, in particular because multi-criteria ratings can help to handle situations in which users gave the same overall rating but had different reasons for that (which can be observed in the detailed ratings). Besides this, multi-criteria rating information can serve as a valuable source for explaining recommendations. Based on these observations, new user similarity metrics and algorithms were designed in [1] that exploit multi-criteria rating information leading to recommender systems of higher quality. The authors show on a small data set how exploiting multi-criteria ratings can be successfully leveraged to improve recommendation accuracy. Our approach of rating items by rating tags shares the advantages of these multi-criteria recommender systems such as improved accuracy and explanations; however the rating dimensions are not static in our approach and require metrics that are different to those put forward for example in [1]. In this respect, our work is also in line with the ideas of Shirky [19], who was among the first who argued that using predefined (rating) categories leads to different challenges such as the following. First, professional experts are needed who design the rating dimensions; additionally, new rating dimensions may emerge over time that were not covered by the predefined and pre-thought static rating dimensions designed or foreseen by a domain expert. In collaborative tagging systems, the set of rating dimensions is not limited which allows users to pick their particular way of stating their preferences. Of course, this comes at the price of a less homogeneous and more unstructured set of item annotations.

Finally, rating items is an important topic not only in the area of recommender research but also in the Semantic Web community. Revyu[10] [8], the winner of the Semantic Web Challenge of the year 2007, is a reviewing and rating Web site which aims to aggregate review data of items (resources) on the Web. Revyu allows people to rate items by writing reviews and gives users the opportunity to add meta-data to items in the form of Web2.0-style tags. Based on this relatively unstructured information, stronger semantics are later on derived. As stated by the authors, this functionality in itself is partially not particularly novel. The real benefit lies in the massive use of Semantic Web technologies and standards like RDF, SPARQL and the principles of Linked Data [3] in order to expose reviews in a reusable and machine-readable format.

Note that in the Revyu system, tags are merely used for classifying the reviewed items and for automatically extracting additional information. We believe that our work could complement this approach by exploiting the rating information which is implicitly contained in the tags. That way, by deriving individual preferences for the tags provided by a user, a better "understanding" of the free-text reviews could be achieved.

6. SUMMARY AND OUTLOOK

The main new idea of our work is to incorporate item-specific ratings for tags in the recommendation process. Following such an approach, users are able to evaluate an existing item in various dimensions and are thus not limited to the one single overall vote anymore. In contrast to previous attempts toward exploiting multi-dimensional ratings, our work aims to follow a Web 2.0 style approach, in which the rating dimensions are not static or predefined.

The goal of this paper was to propose a first, comparably simple recommendation method that can take item-specific tag ratings into account when generating rating predictions. In addition, we proposed one particular metric to automatically derive user- and item-specific tag ratings from the overall ratings based on item similarities in order to demonstrate that quality improvements can be achieved even when the tag rating data is not explicitly given. The results of the evaluation on the MovieLens data set shows that a measurable accuracy improvement can be achieved.

In our future work we will not only run experiments with other metrics (incorporating, e.g., *default tag ratings* or hybridization strategies) and more sophisticated methods for estimating tag ratings, but also explore further questions related to tag ratings in the Social Web recommendation process.

- **Further experiments and user interfaces.** First, experiments with real tag ratings are planned to measure the corresponding accuracy improvements. A particular question to be answered in that context is that

[10] `http://revyu.com`

of an appropriate user interface (see also [22]) because Web users are currently not acquainted to the interaction pattern "providing ratings for tags". Intuitively, interfaces that allow users to rate tags on a scale from 1 to 5 or allow users to classify tags in two or three categories such as "like", "dislike", or "indifferent" seem appropriate. However, we aim to explore different visualizations to stimulate more precise ratings.

- **Better explanations.** Tags can also be a helpful means to generate explanations for the end user. Explanations for recommendations are one of the current research topics in the recommender systems area because they can significantly influence the way a user perceives the system. In [20], for example, seven possible advantages of an explanation facility are described. In [22], the authors have evaluated explanation interfaces which use tag relevance and tag preference as two key components. Further studies could be conducted to examine the role of tag ratings in helping users understand their recommendations. If tags are both user- *and* item-specific, more personalized and detailed, multi-dimensional explanations can be provided. Based on appropriately designed explanation interfaces, the different aspects of explanations as discussed in [20] (such as transparency, trust, effectiveness and satisfaction) can be analyzed in different user studies. Again, also the question of the appropriate end-user visualization has to be answered.

- **Combination with tag recommenders.** Different techniques to *tag recommendation* have been developed in the last years to stimulate users to use a more consistent set of tags in the annotation process, see, for example, [25]. We expect that the value of item-specific tag ratings is even higher, when the overall set of tags used in the data set is more consistent.

- **New tag quality metrics.** We have stated in Section 4.1.2 that our approach of rating items by rating tags calls for a new quality requirement to tags: tags must be appropriate for ratings. Therefore, in our current work, we aim to develop new tag quality metrics in order to improve the overall performance of recommendation algorithms that are based on tag ratings.

Finally, by making the software used in our experiments publicly available[11], we hope to contribute to the comparability of different algorithms since our study revealed that relevant algorithmic details and parameters are often not reported in sufficient details.

7. REFERENCES

[1] G. Adomavicius and Y. Kwon. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems*, 22(3):48–55, 2007.

[2] G. Adomavicius and A. Tuzhilin. Extending recommender systems: A multidimensional approach. In *Proceedings of the Workshop on Intelligent Techniques for Web Personalization (ITWP'01)*, pages 4–6, Acapulco, Mexico, 2001.

[11] `http://ls13-www.cs.uni-dortmund.de/rec_suite.zip`

[3] T. Berners-Lee. Linked data. `http://www.w3.org/DesignIssues/LinkedData.html`, 2006. Retrieved on July 11, 2010.

[4] T. Bogers and A. van den Bosch. Collaborative and content-based filtering for item recommendation on social bookmarking websites. In *Proceedings of the Workshop on Recommender Systems and the Social Web (RSWEB'09)*, pages 9–16, New York, NY, USA, 2009.

[5] M. de Gemmis, P. Lops, G. Semeraro, and P. Basile. Integrating tags in a semantic content-based recommender. In *Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys'08)*, pages 163–170, Lausanne, Switzerland, 2008.

[6] J. Diederich and T. Iofciu. Finding communities of practice from user profiles based on folksonomies. In *Proceedings of the 1st International Workshop on Building Technology Enhanced Learning Solutions for Communities of Practice (TEL-CoPs'06)*, pages 288–297, Crete, Greece, 2006.

[7] J. Gemmell, M. Ramezani, T. Schimoler, L. Christiansen, and B. Mobasher. The impact of ambiguity and redundancy on tag recommendation in folksonomies. In *Proceedings of the 2009 ACM Conference on Recommender Systems (RecSys'09)*, pages 45–52, New York, NY, USA, 2009.

[8] T. Heath and E. Motta. Revyu.com: A reviewing and rating site for the web of data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC'07 + ASWC'07)*, pages 895–902, Busan, Korea, 2007.

[9] R. Jäschke, L. B. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'07)*, pages 506–514, Warsaw, Poland, 2007.

[10] A.-T. Ji, C. Yeon, H.-N. Kim, and G.-S. Jo. Collaborative tagging in recommender systems. In *Proceedings of the 20th Australian Joint Conference on Artificial Intelligence (AUS-AI'07)*, pages 377–386, Gold Coast, Australia, 2007.

[11] H.-N. Kim, A.-T. Ji, I. Ha, and G.-S. Jo. Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. *Electronic Commerce Research and Applications*, 9(1):73 – 83, 2010.

[12] H. Liang, Y. Xu, Y. Li, and R. Nayak. Collaborative filtering recommender systems based on popular tags. In *Proceedings of the 14th Australasian Document Computing Symposium (ADCS'09)*, University of New South Wales, Sydney, Australia, 2009.

[13] N. Manouselis and C. Costopoulou. Analysis and classification of multi-criteria recommender systems. *World Wide Web*, 10(4):415–441, 2007.

[14] M. Nakagawa and B. Mobasher. A hybrid web personalization model based on site connectivity. In *Proceedings of the Workshop on Web Mining and Web Usage Analysis (WebKDD'03)*, pages 59–70, Washington, DC, USA, 2003.

[15] J. J. Sandvig, B. Mobasher, and R. Burke. Robustness of collaborative recommendation based on association rule mining. In *Proceedings of the 2007 ACM*

Conference on Recommender Systems (RecSys'07), pages 105–112, Minneapolis, MN, USA, 2007.

[16] S. Sen, F. M. Harper, A. LaPitz, and J. Riedl. The quest for quality tags. In *Proceedings of the 2007 International ACM Conference on Supporting Group Work (GROUP'07)*, pages 361–370, Sanibel Island, Florida, USA, 2007.

[17] S. Sen, J. Vig, and J. Riedl. Learning to recognize valuable tags. In *Proceedings of the 13th International Conference on Intelligent User Interfaces (IUI'09)*, pages 87–96, Sanibel Island, Florida, USA, 2009.

[18] S. Sen, J. Vig, and J. Riedl. Tagommenders: Connecting users to items through tags. In *Proceedings of the 18th International World Wide Web Conference (WWW'09)*, pages 671–680, Madrid, Spain, 2009.

[19] C. Shirky. Ontology is overrated. `http://www.shirky.com/writings/ontology_overrated.html`, 2005. Retrieved on July 11, 2010.

[20] N. Tintarev and J. Masthoff. Effective explanations of recommendations: User-centered design. In *Proceedings of the 2007 ACM Conference on Recommender Systems (RecSys'07)*, pages 153–156, New York, NY, USA, 2007.

[21] K. H. L. Tso-Sutter, L. B. Marinho, and L. Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC'08)*, pages 1995–1999 , Fortaleza, Ceara, Brazil, 2008.

[22] J. Vig, S. Sen, and J. Riedl. Tagsplanations: Explaining recommendations using tags. In *Proceedings of the 13th International Conference on Intelligent User Interfaces (IUI'09)*, pages 47–56, Sanibel Island, Florida, USA, 2009.

[23] Z. Wang, Y. Wang, and H. Wu. Tags meet ratings: Improving collaborative filtering with tag-based neighborhood method. In *Proceedings of the Workshop on Social Recommender Systems (SRS'10)*, Hong Kong, China, 2010.

[24] J. Zhang and P. Pu. A recursive prediction algorithm for collaborative filtering recommender systems. In *Proceedings of the 2007 ACM Conference on Recommender Systems (RecSys'07)*, pages 57–64, Minneapolis, MN, USA, 2007.

[25] N. Zhang, Y. Zhang, and J. Tang. A tag recommendation system for folksonomy. In *Proceedings of the 2nd Workshop on Social Web Search and Mining (SWSM'09)*, pages 9–16, New York, NY, USA, 2009.

[26] Y. Zhen, W.-J. Li, and D.-Y. Yeung. Tagicofi: Tag informed collaborative filtering. In *Proceedings of the 2009 ACM Conference on Recommender Systems (RecSys'09)*, pages 69–76, New York, NY, USA, 2009.

Improving Link Analysis for Tag Recommendation in Folksonomies

Maryam Ramezani, Jonathan Gemmell, Thomas Schimoler, Bamshad Mobasher
Center for Web Intelligence
School of Computing, DePaul University
Chicago, Illinois, USA
{ mramezani, jgemmell, tschimo1, mobasher}@cdm.depaul.edu

ABSTRACT

Social tagging applications allow users to annotate online resources, resulting in a complex network of interrelated users, resources and tags often called a Folksonomy. A folksonomy is often represented as a hyper-graph in which each hyper-edge connects a user, resource and tag. This tripartite hyper-graph is often used by data mining applications to provide services for the user such as tag recommenders.One of the most well known approaches is FolkRank which constructs an undirected tripartite graph from the hyper-graph and then applies PageRank.However since FolkRank relies on an undirected graph, it does not accurately represent the f ow of information across the informational channels.In this paper we model a folksonomy as a weighted direct graph. The weights of the edges are def ned by a heuristic that better represents the f ow of information from one node to another.We use the proposed model for tag recommendation and the results show an improvement over FolkRank. We show that even the undirected Adapted PageRank with correct parametrization can do better than FolkRank.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

Tag Recommendation, Folksonomy, Social Tagging

1. INTRODUCTION

Collaborative tagging systems such as Delicious[1], lastFm [2], and Bibsonomy[3] have emerged as powerful applications for Internet users. Tagging systems support users with several benef ts. First they allow users to organize their own data with a level of freedom not possible in traditional taxonomic f ling systems. Secondly they provide users with the means to openly share this information among friends and colleagues. Thirdly they also allow anyone to utilize the collective knowledge of others for discovering new topics, resources or perhaps even new friends. Fourthly, the reuse of tags creates a dynamic user driven approach to formalize semantic relationships.

While Folksonomies have many benef ts, they also present several challenges. Most collaborative tagging applications permit unsupervised tagging; users are free to use any tag they wish to describe a resource. This unsupervised tagging can result in tag redundancy – in which several tags have the same meaning – or tag ambiguity – in which a single tag has many different meanings. Such inconsistencies can confound users as they attempt to utilize the folksonomy. It can be diff cult for users to traverse the sheer volume of data. Moreover, noise in the data can impede the users experience.Data mining applications such as tag recommenders make it easier for the user to navigate the system.

Tag recommendation, the suggestion of tags during the annotation process reduces the user effort. By reducing the effort users are encouraged to tag more frequently, apply more tags to an individual resource, reuse common tags, and perhaps use tags the user had not previously considered. Moreover, user error is reduced by eliminating redundant tags caused by capitalization inconsistencies, punctuation errors, misspellings and other discrepancies. The tag recommender can further promote a core tag vocabulary steering the user toward adopting certain tags while not imposing any strict rules. The tag recommender may even avoid ambiguous tags in favor of tags that offer greater information value. This may aid other users when navigating through the folksonomy to f nd interesting resources related to a tag which is more often used by other users.

In order to develop such recommender applications the f rst step is to create a model of the folksonomy that takes into account the information f ow between users, resources and tags. Hotho et al. suggest an approach for adaptating the PageRank metric for folksonomies to facilitate search and recommendation in folksonomies [4, 10, 9]. However, their representation based on an undirected graph has several shortcomings in that weights that f ow in one direction of an edge will go back along the same edge in the next iteration of PageRank. Therefore, they argue, the resulting rank scores of a node is very similar to its popularity. To overcome this shortcoming, FolkRank was proposed which takes the difference of the rank scores of Adapted PageRank with and without a preference vector. A drawback to this approach is that rank scores can be extremely low or negative, making algorithms that rely on them unpredictable. Moreover an undirected graph does not provide a realistic model of the relations between the objects of the folksonomy because the informational f ow from one object to another object is

[1] http://delicious.com/
[2] http://www.last.fm/
[3] www.bibsonomy.org/

not symmetric.

In this paper we propose a weighted directed graph which can better model the informational channels of a folksonomy. We can then apply PageRank to this model for search and recommendation. Our claim is based on the observation that the user navigation from one object(user, resource, or tag) to another object in a folksonomy is not symmetric and by considering different weights on the edges of each direction we can better model the navigating from one node to the other. Our contributions in this paper are two fold. First, we demonstrate that with appropriate parameterization, Adapted PageRank can outperform FolkRank in tag recommendation. Secondly, our extensive evaluation on three real world datasets reveal that PageRank generates better tag recommendations with a weighted directed graph model of the folksonomy than with a undirected graph model.

The rest of this paper is organized as follows. In Section 2 we review the state of the art in tag recommendation. In section 3 we describe common graph-based algorithms including Adapted PageRank and FolkRank for tag recommendation. We discuss the differences and parameterization of the two algorithms. In section 4 we present our proposed algorithm for creating a weighted directed graph from folksonomy. Our experimental methodology, datasets and results are offered in Section 5. Finally, we conclude the paper with directions for future work.

2. RELATED WORK

As collaborative tagging applications have gained in popularity researchers have started to explore and characterize the tagging phenomenon. In [13] and [7] the authors studied the information dynamics of Delicious, one of the most popular folksonomies. The authors discussed how tags have been used by individual users over time and how tags for an individual resource stabilize over time. Unlike traditional recommender systems which have a two dimensional relation between users and items, tagging systems have a three dimensional relation between users, tags and resources. Recommender systems can be used to recommend each of the dimensions based on one or two of the other dimensions. The authors in [21] apply user-based and item-based collaborative f ltering to recommend resources in a tagging system and use tags as an extension to the user-item matrices. The authors in [16] and [15] use tags as context information to recommend resources.

A comparison of user-based collaborative f ltering and a graph-based recommender based on the Pagerank algorithm to recommend personalized tags is offered in [11]. Association rules are explored in [8] to recommend tags and introduce an entropy-based metric to f nd how predictable a tag is. The title of a resource and the user vocabulary is used in [12] to generate recommendations. The results show that tags retrieved from the user's vocabulary outperform recommendations driven by resource information.

The authors in [23] present general criteria for a good tagging system including high coverage of multiple facets, high popularity and least-effort. They categorize tags to content-based tags, context-based tags, attribute tags, subjective tags and organizational tags and use a probabilistic method to recommend tags. In [2] a classif cation algorithm for tag recommendation is proposed. In [1] a semantic tag recommendation system in the context of a semantic desktop is suggested. Clustering is used by [20] to make real-time tag recommendation. User-def ned tags and co-occurrence are employed by [19] to recommend tags to users on Flickr. The assumption is that the user has already assigned a set of tags to a photo and the recommender uses those tags to recommend more tags. The authors in [6] have completed a similar study and introduce a classif cation algorithms for tag recommendation. Probabilistic models have been used in recommendation in folksonomies in [17] and [22]. Moreover, [17] uses Probabilistic Latent Semantic Analysis for resource discovery and [22] uses single aspect PLSA for tag recommendation.

Increasing interest in improving the effectiveness of tag recommendation attracted many researchers from all over the world to the ECML PKDD Discovery Challenge 2009 [5]which was mainly focused on tag recommendation. Different data mining approaches including content-based techniques, probabilistic models, and factor models have been applied for tag recommendation. The comparison of different approaches [18] shows that graph-based models show the best performance in tag recommendation. The results in [18] show that FolkRank and the tensor factorization models show high performance in tag recommendation while PageRank does not perform as well. Similarly, results in [11] show that FolkRank wins over other algorithms including adapted PageRank.

In this work, we show experimentally that FolkRank is not a sustainable approach, and with correct parameterization PageRank can perform better than FolkRank. In addition, we suggest a weighted directed graph model of the folksonomy that can better model the f ow of information in a folksonomy.

3. GRAPH-BASED TAG RECOMMENDERS

In traditional recommendation algorithms the input is often a user, u, and the output is a set of items, I. The user experience is improved if this set of items is relevant to the user's needs. Tag recommendation in Folksonomies however differs in that the input is both a user, u, and a resource, r. The output remains a set of items, in this case a recommended set of tags, T_r. An algorithm for tag recommendation in Folksonomies therefore requires a means to include both user and resource information in the process so that the recommendation set includes tags that are relevant to the resource and also represent the user's tagging practice. In this section we will f rst describe the data model of the folksonomies and then we describe the details of the popular tag recommendation technique, "FolkRank" and discuss its differences with the Adapted PageRank algorithm, simply called as PageRank in recent publications [18].

3.1 Data Model in Folksonomies

A collaborative tagging system consists of three generic elements: users, resources, and tags. Formally, the model can be described as a four-tuple: $D = \langle U, R, T, A \rangle$, such that there exists a set of users, U; a set of resources, R; a set of tags, T; and a set of annotations, A. Annotations are represented as a set of triples containing a user, tag and resource such that

$$A \subseteq \{\langle u, r, t \rangle : u \in U, r \in R, t \in T\} \quad (1)$$

This model is often viewed as a hypergraph $G = (V, E)$, where $V = U \cup R \cup T$ is the set of nodes and $E = \{\{u, r, t\} | \langle u, r, t \rangle \in A\}$ is the set of hyperedges. To simplify analysis, we can reduce the hypergraph into three bipartite graphs with regular edges. The graphs model aggregate associations between users and resources (UR), users and tags (UT), and tags and resources (TR) [14, 24]. The relation between resources and tags can be formulated as a two-dimensional projection, RT, such that each entry, $RT(r, t)$, is the weight associated with the resource, r, and the tag, t [14, 10]. This weight may be binary, merely showing that one or more users have applied that tag to the resource, or it may be f ner grained using the number of users that have applied that tag to the resource:

$$RT(r, t) = |u \in U : (u, t, r) \in A| \quad (2)$$

Such a measure is equivalent to *term frequency* or *tf* common in

Information Retrieval. Similar two-dimensional projections can be constructed for UT in which the weights correspond to users and tags where

$$UT(u,t) = |r \in R : (u,t,r) \in A| \quad (3)$$

and UR in which the weights correspond to users and resource where

$$UR(u,r) = |t \in T : (u,t,r) \in A| \quad (4)$$

These three projections taken together produce a weighted undirected tripartite graph with a set of nodes: $V = U \cup R \cup T$ and edges RT, UR, and UT.

3.2 FolkRank Vs. Adapted PageRank

In [10] the authors proposed an adaptation of PageRank to the folksonomy data structure using the undirected graph described in section 3.1. If we regard the adjacency matrix of this graph, A, (normalized to be column-stochastic) and a preference vector, p, then we compute the Pagerank vector, w, in the usual manner:

$$\omega = (1-\gamma)A\omega + \gamma p \quad (5)$$

The value of $\gamma \in [0,1]$ controls the inf uence of the preference vector. The preference vector allows the algorithm to be trained on a specif c region of the graph. In order to generate tag recommendations the preference vector is biased towards the query user and resource [11]. These elements are given a substantial weight in the preference vector where all other elements have uniformly small weights. Similar to [11] each user, tag and resource gets a preference weight of 1 while the target user and resource get a preference weight of $1+|U|$ and $1+|R|$, resp. According to Hotho et al. [10] and [9] the FolkRank vector is taken as a difference between two computations of Pagerank: one with a preference vector and one without the preference vector. More formally, if we consider ω_0 to be the f xed point from equation 5 without preference vector and ω_1 as the f xed point with preference vector p,then the FolkRank vector is def nes as

$$\omega = \omega_1 - \omega_0 \quad (6)$$

Because a generic but popular item will receive a similar Pagerank score in both models, its FolkRank will be reduced to 0 (or less, if its new score is less than the original). The stated justif cation for FolkRank [10, 11] is that the ω vector resulting from FolkRank provides valuable results on a large-scale real-world dataset, while ω_1 alone provides an unstructured mix of topic-relevant elements with elements having high edge degree. However, the performance of the FolkRank algorithm is dependent on the tuning of γ, the amount weight given to the preference vector. It has been determined experimentally that for the Delicious dataset, for example, $\gamma = .3$ provides the best results for FolkRank. However, the meaning of γ in the context of FolkRank has been somewhat obfuscated – it is not clear why this one value is best, and values above or below this one give sub-optimal models. In a standard PageRank algorithm, the higher the value of γ, the more emphasis is applied to the preference vector, meaning that the results can be predicted be more topic-relevant. We cannot make this same prediction by adjusting the parameter for FolkRank; it is not clear how a change in γ changes the model itself. We will show through experiment that increasing the value of γ in FolkRank does not improve its performance and indeed PageRank can be optimized to a greater extent than FolkRank.

4. DIRECTED GRAPH MODEL

The problem with the weighting schema described in 3.1 is that the weight of the undirected edges does not accurately ref ect the f ow of information across the folksonomy. We extend this graph model to consider the expectation of hypothetical user navigating from one node to another. Consider r representing a non-popular resource and t representing a popular tag associated to r in a folksonomy. In [10], the weight between r and t is def ned by the number of users who associate them together represented with RT(r,t) above. Our weighting schema suggests that the weight from r to t should be higher than the weight from t to r since t is a popular tag and a user is more likely to navigate from a non-popular resource to a popular tag than navigating from a popular tag to a non-popular resource. We calculate the popularity of t as the sum of all users who have used the tag t. Next, the weight from r to t is def ned as multiplicative factor of popularity and $RT(r,t)$.Similarly, the popularity of r will be the sum of number of users who have annotated resource r in their prof le. Equations 7 and 8 show the formal def nitions of edge weights from r to t and from t to r respectively.

$$weight(r \rightarrow t) = RT(r,t) \times \sum_{r' \in R} RT(r',t) \quad (7)$$

$$weight(t \rightarrow r) = RT(r,t) \times \sum_{t' \in T} RT(r,t') \quad (8)$$

The weight of an edge between tag t and user u is def ned as follows. Popularity of tag t is the sum of all resources associated with t and the popularity of user u is the sum of all resources tagged by the user u. Weights of UR edges are def ned similarly. Following equations show the formal def nitions for the edge weights

$$weight(u \rightarrow t) = UT(u,t) \times \sum_{u' \in U} UT(u',t) \quad (9)$$

$$weight(t \rightarrow u) = UT(u,t) \times \sum_{t' \in T} UT(u,t') \quad (10)$$

$$weight(u \rightarrow r) = UR(u,r) \times \sum_{u' \in U} UR(u',r) \quad (11)$$

$$weight(r \rightarrow u) = UR(u,r) \times \sum_{r' \in R} UR(u,r') \quad (12)$$

In the following section we use the suggested graph model for tag recommendation and compare the results with FolkRank.

5. EXPERIMENTAL EVALUATION

Here we describe the methods used to gather and pre-process data for the experiments and provide details of our datasets. We discuss our testing methodology and brief y explain the common metrics recall and precision and then detail the results of our experiments.

5.1 Datasets

We have chosen three datasets for our experiments: Delicious, Bibsonomy and Citeulike. In order to reduce noise and focus on the denser portion of the dataset a P-core was taken such that each user, resource and tag appear in at least p posts as in [3, 11]. A post is def ned as as a user, resource and all tags applied by that user to that resource Delicious is a popular Web site in which users annotate URLs. On 10/19/2008, 198 of the most popular tags were taken

Folksonomy	Delicious	Citeulike	Bibsonomy
Users	7,665	2,051	402
Resources	15,612	5,376	2,014
Tags	5,746	3,343	1,755
Posts	720,788	42,278	15,760
Annotations	2,762,235	105,873	53,554

Table 1: Datasets

from the user interface. For each of these tags the 2,000 most recent annotations including the contributors of the annotations were collected. This resulted in 99,864 distinct usernames. For each user the social network was explored recursively resulting in a total of 524,790 usernames. From 10/20/2008 to 12/15/2008 the complete prof les of all users were collected. Each user prof le consisted of a collection of posts including the resource, tags and date of the original bookmark. The top 100 most prolif c users were visually inspected; twelve were removed from the data because their post count was many orders of magnitude larger than other users and were suspected to be Web-bots. Due to memory and time constraints, 10% of the user prof les was randomly selected. A p-core of 20 was taken from this dataset for experiments.

The Bibsonomy dataset was gathered on 1/1/2009 encompassing the entire system. This data set has been made available online by the system administrators and it was used for the ECML/PKDD discovery challenge 2009. They have pre-processed the data to remove anomalies. A 5-core was taken to reduce noise and increase density.

Citeulike is used by researchers to manage and discover scholarly references. The dataset is available to download. On 2/17/2009 the most recent snapshot was taken. The data contains anonymous user ids and posts for each user including resources, the date and time of the posting and the tags applied to the resource. A P-core of 5 was calculated.Table 1 summarizes the base statistics for each dataset.

The average number of tags in a Delicious post is 3.82, and the average number of posts per user is 94.04. For Citeulike, the averages are 2.50 tags per post and 20.61 posts per user; for Bibsonomy, 3.39 and 39.2. We therefore note that Delicious has by far the most tagging data per user, followed by Bibsonomy, and Citeulike last. In terms of posts per resource, we again see that Delicious has far and away the highest average with 46.1; Citeulike and Bibsonomy are nearly identical in this respect (7.86 vs. 7.82, respectively). An important distinction between the datasets is their focus. Users in Delicious are able to tag any URL available on the Web. As such an individual's interests are often varied encompassing many topics. In Citeulike, however, researchers tag scholarly publications and their tagging is often focused in their area of expertise. Due to its dual-nature, we expect Bibsonomy users to display a somewhat mixed approach.

We have adopted the test methodology as described in [11]. In this approach, called *LeavePostOut*, a single post is randomly removed from each user's prof le. The training set is then comprised of all of the remaining data, while the test set contains one test case per user. Each test case consists of a user, u, a resource, r, and all the tags the user has applied to that resource. These tags, T_h, are analogous to the holdout set commonly used in Information Retrieval. The tag recommendation algorithms accept the user-resource pair and return an ordered set of recommended tags, T_r. From the holdout set and recommendation set utility metrics were calculated. For each evaluation metric the average value is calculated across all test cases.

For evaluation we adopt the common recall and precision measures as is common in Information Retrieval. Recall is a measure of completeness and is def ned as:

$$r = |T_h \cap T_r|/|T_h| \tag{13}$$

Where T_h is the hold-out set tags which represent the correct answer and T_r is the set of recommended tags. Recall measures the percentage of items in the recommendation set that appear in the holdout set. Precision measures the exactness of the recommendation algorithm and is def ned as:

$$p = |T_h \cap T_r|/|T_r| \tag{14}$$

which measures the percentage of items in the holdout set that appear in the recommendation set.

5.2 Experimental Results

Here we present our experimental results with tuning of value of γ and changing the number of recommended tags. We have recorded the values of precision and recall with changing the number of recommended tags from 1 to 10 and also changing the value of γ between .05 to 1. The Adapted PageRank results are shown with abbreviation of "APR" in the graphs. Figures 1 through 6 show the effect of changing the value of γ on precision and recall for the three data sets. The values are recorded for a recommendation set of 10 tags. Increasing the value of γ indicates a greater emphasis on the preference vector and our results show that increasing γ has a consistently positive effect on tag recommendation using the Adapted PageRank algorithm. Even though the results for $\gamma = .99$ still show an improvement, setting the value of $\gamma = 1$ will drastically reduce the recommendation precision to nearly zero. This is expected, as $\gamma = 1$ means that there is no learning involved and the f nal resulting vector is equal to the preference vector.

Our results show that for FolkRank, on the other hand, by increasing the value of γ, the performance decreases. To explain this we should note that FolkRank is basically the difference of the resulting weights of the adapted PageRank with and without preference vector. As the value of γ increases the result is more and more focused on the preference vector which results in higher weights for the nodes that are connected to the target nodes. The differential approach in FolkRank is supposed to compute a topic-specif c ranking of the elements in a folksonomy. However, the fact that the weights from the Pagerank without preference (ω_0) are subtracted from the one with preference vector (ω_1) creates negative values in the resulting vector for any node which has a higher value in ω_1. This is useful for small values of γ to remove the high weighted popular nodes, however, as the value of γ increases ω_1 is more and more representative of the actual ranks considering the specif c resource and tag. By subtracting ω_0 from ω_1 we end up with an ad-hoc unrealistic weight vector, since we might omit many high weighted nodes in ω_1 only because they have occurred with a higher weight in ω_0.

Figure 7 shows the comparison of different techniques for different values of precision and recall in the Citeulike dataset. We keep the value of γ constant in this experiment and set it to the value that results in best performance for each technique. From f gures 3 and 4 we can observe that this value is .3 for FolkRank and .99 for Adapted PageRank. In f gure 7 the values of precision and recall are found by recommending 1 to 10 tags. Note that the most right point of this graph is the one which is presented in f gures 3 and 4. Although in this f gure it seems that all points converge to the same value, we can see in f gures 3 and 4 that there is a big difference among them. In this f gure we can observe that the adapted PageR-

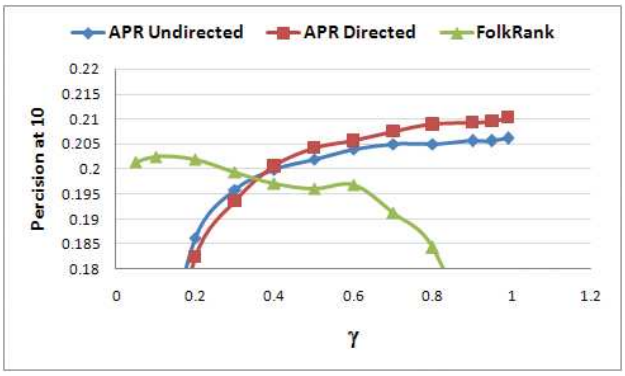


Figure 1: The effect of changing γ on precision for a recommendation set of 10 tags in Bibsonomy data set

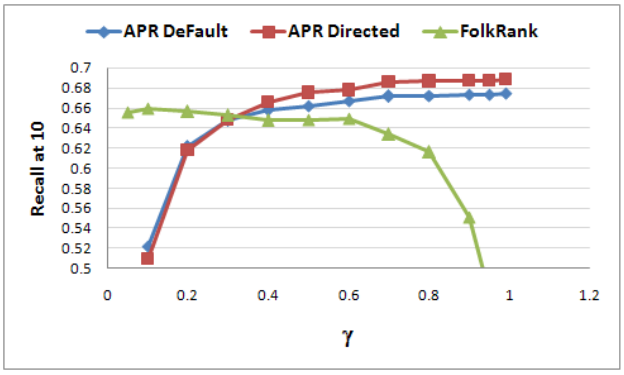


Figure 2: The effect of changing γ on recall for a recommendation set of 10 tags in Bibsonomy data set

ank using the directed graph outperforms the undirected version and the FolkRank for all values of precision and recall.

The results from different data sets show that increasing the value of γ results in increasing the recommendation accuracy of the PageRank algorithm and it has the opposite effect for the FolkRank. We can see that performance of FolkRank extremely drops when the value of γ increases. The comparison of directed and undirected graphs show that directed graph produces better or similar results across different datasets. In Bibsonomy and Citulike the results of the directed graph are considerably better for certain values of γ. However, in Delicious there is not a signif cant difference between the the directed and undirected approaches.

We performed signif cance test to determine if the differences between observed values from each approach in different data sets are signif cant. We used pair sample t-test and compared the mean of precision and recall resulted from the constant γ which produces the best results for each technique. The results from the signif - cance test show that the differences between the values from each approach is signif cantly different from the other ones for Bibsonomy and Citeulike datasets with $p < .001$. However, in Delicious dataset the differences are not signif cant which means that we can not reject the null hypothesis that the two approaches (directed and undirected graph models) produce similar results. This difference is likely because of existence of much broader concepts in Delicious and deserves further investigation. The signif cance tests show that in all datasets the Adapted PageRank signif cantly outperforms FolkRank.

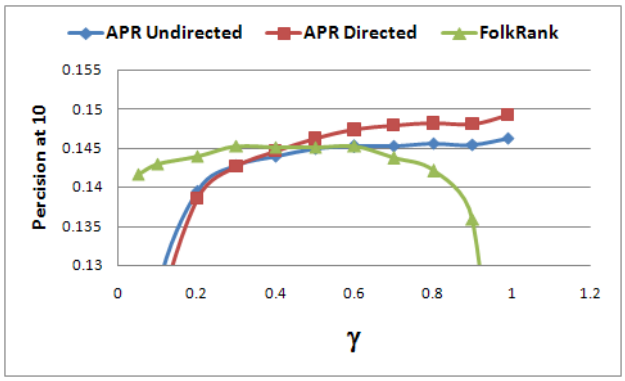


Figure 3: The effect of changing γ on precision for a recommendation set of 10 tags in Citeulike data set

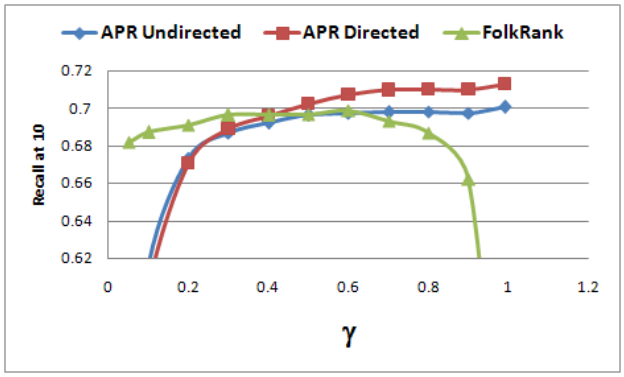


Figure 4: The effect of changing γ on recall for a recommendation set of 10 tags in Citeulike data set

6. CONCLUSION

In this paper we suggested to model the folksonomy as a weighted directed graph which can capture the informational channels of a folksonomy. We then applied PageRank to this model for tag recommendation. Our extensive evaluation on three real world datasets reveal that with appropriate parameterization, Adapted PageRank can outperform FolkRank in tag recommendation. In addition, we have shown that with modeling the folksonomy as a directed weighted graph, we can get additional improvement. For future work, we plan to suggest other weighting schemas which can better model the probability of navigation in the folksonomy and apply our different models for search and resource recommendation.

7. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation Cyber Trust program under Grant IIS-0430303 and DePaul University Summer Research Grants 2009 and a grant from the Department of Education, Graduate Assistance in the Area of National Need, P200A070536.

8. REFERENCES

[1] B. Adrian, L. Sauermann, and T. Roth-Berghofer, "Contag: A semantic tag recommendation system," in *Proceedings of I-Semantics' 07*, T. Pellegrini and S. Schaffert, Eds. JUCS, 2007, pp. pp. 297–304. [Online]. Available: http://www.dfki.uni-kl.de/ sauermann/papers/horak+2007a.pdf

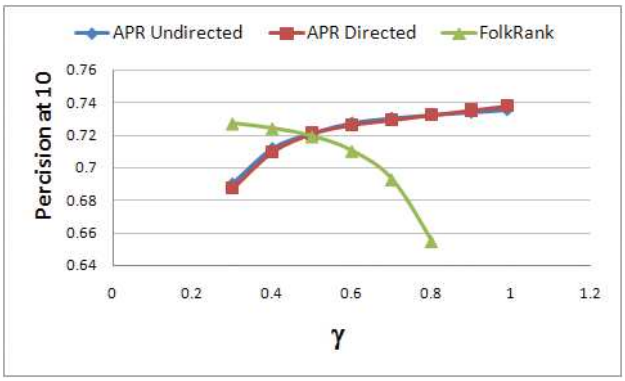


Figure 5: The effect of changing γ on precision for a recommendation set of 10 tags in Delicious data set

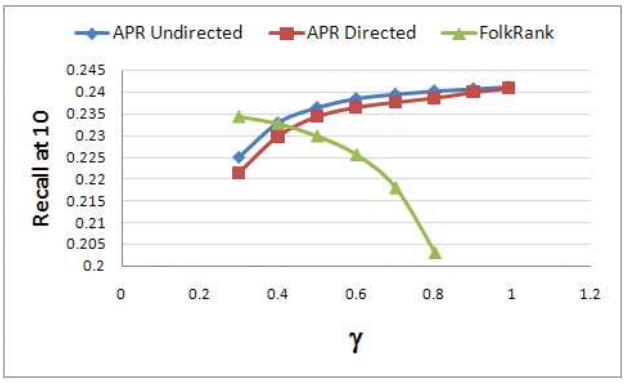


Figure 6: The effect of changing γ on recall for a recommendation set of 10 tags in Delicious data set

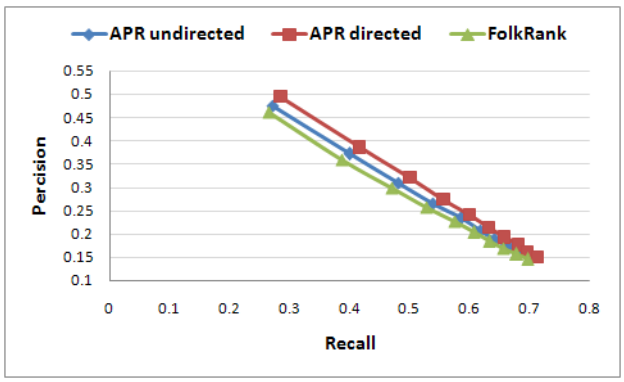


Figure 7: Comparison of undirected APR,directed APR for $\gamma = .99$ and FolkRank for $\gamma = .3$ on Citeulike data set

FGIR 2006, 2006. [Online]. Available: http://www.kde.cs.uni-kassel.de/stumme/papers/2006/hotho2006folkrank.pdf

[10] ——, "Information retrieval in folksonomies: Search and ranking," in *The Semantic Web: Research and Applications*, 2006, pp. 411–426.

[11] R. Jaschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme, "Tag Recommendations in Folksonomies," *LECTURE NOTES IN COMPUTER SCIENCE*, vol. 4702, p. 506, 2007.

[12] M. Lipczak, "Tag recommendation for folksonomies oriented towards individual users," in *Proceedings of the ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.

[13] G. Macgregor and E. McCulloch, "Collaborative tagging as a knowledge organisation and resource discovery tool," *Library Review*, vol. 55, no. 5, pp. 291–300, 2006.

[14] P. Mika, "Ontologies are us: A unif ed model of social networks and semantics," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 1, pp. 5–15, 2007.

[15] R. Y. Nakamoto, S. Nakajima, J. Miyazaki, S. Uemura, and H. Kato, "Investigation of the effectiveness of tag-based contextual collaborative f ltering in website recommendation," in *Advances in Communication Systems and Electrical Engineering*. Springerlink, 2008, pp. 309–318.

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

[17] A. Plangprasopchok and K. Lerman, "Exploiting social annotation for automatic resource discovery," *CoRR*, vol. abs/0704.1675, 2007.

[18] S. Rendle and L. Schmidt-Thieme, "Pairwise interaction tensor factorization for personalized tag recommendation," in *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*. New York, NY, USA: ACM, 2010, pp. 81–90.

[19] B. Sigurbjörnsson and R. van Zwol, "Flickr tag

recommendation based on collective knowledge," pp. 327–336, 2008.

[20] Y. Song, Z. Zhuang, H. Li, Q. Zhao, J. Li, W.-C. Lee, and C. L. Giles, "Real-time automatic tag recommendation," in *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 2008, pp. 515–522.

[21] K. H. L. Tso-Sutter, L. B. Marinho, and L. Schmidt-Thieme, "Tag-aware recommender systems by fusion of collaborative f ltering algorithms," in *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2008, pp. 1995–1999.

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

[23] Z. Xu, Y. Fu, J. Mao, and D. Su, "Towards the semantic web: Collaborative tag suggestions," *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland, May*, 2006.

[24] C. M. A. Yeung, N. Gibbins, and N. Shadbolt, "Mutual contextualization in tripartite graphs of folksonomies," in *Lecture Notes in Computer Science, The semantic Web*. Springer, 2008, pp. 966–970.

Towards Understanding the Challenges Facing Effective Trust-Aware Recommendation

Yue Shi, Martha Larson, Alan Hanjalic
Multimedia Information Retrieval Lab
Delft University of Technology, Delft, The Netherlands
{y.shi, m.a.larson, a.hanjalic}@tudelft.nl

ABSTRACT
We introduce a method for generating semi-synthetic social data collections, which we use to study trust-aware recommendation. Specifically, we examine the effects of social graph degree distribution on user-based collaborative filtering that substitutes trusted users for conventional neighbors. Our semi-synthetic data collections are created via a naïve pruning process that maps a user-item matrix onto various social graphs with the degree distributions of real-world Web-based social systems. Our goal is to extend our understanding of the challenges facing effective trust-aware recommendation beyond the current possibilities, which are limited by data set availability. The improvement offered by trust-aware recommendation is shown to have substantial dependence on the degree distribution of the social graph.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *Information Filtering*

General Terms
Algorithms, Performance, Experimentation

Keywords
Recommender systems, social trust networks, trust-aware recommendation, collaborative filtering, Kronecker graph

1. INTRODUCTION
Trust-aware recommendation, i.e., approaches that make use of the information in a social graph, has recently been receiving increased research attention [6][8][12][16][17][18][21][25]. One aim of this research is to exploit the potential benefits that social trust information can bring to conventional collaborative filtering (CF). User-based CF is a highly successful technique based on the concept that an item should be recommended to a user if users with similar preferences like it [1][10]. Under conventional user-based CF approaches, the similarity of preference between users is derived from the user-item matrix, and a similarity measure is used to calculate the match between two users' profiles. The closest matches form the user's neighborhood and the profiles of users in the neighborhood provide the basis for recommendation. The large number of user-user comparisons necessary to determine an appropriate neighborhood is the major source of computation expense for conventional CF and also makes it unsuitable for distributed settings. Social trust information has the potential to address these issues [23]. If trusted users replace the neighbors used by conventional user-based CF, reliable recommendation is possible without the need to compute similarity neighborhoods.

The goal of the research presented in this paper is to improve our understanding of how particular properties of social data collections impact the benefit that social trust information contributes to CF performance. In particular, we investigated how the degree distribution of a social graph impacts user-based CF that substitutes trusted users for conventional neighborhoods. We consider social trust to include any connection that one user explicitly establishes with another, regardless of the user's motivation for creating the social connection. Defined in this way, social trust relationships include "friends" on YouTube, "people" on Delicious, and "trusted reviewers" on Epinions. We define a social data collection to be a set of user profiles containing item ratings or comparable item information (i.e., a user-item matrix) together with social trust information, consisting of the trust set of each user. All other users to which a given user has explicitly established links form that user's trust set. Taken together, the trust sets of the users in a collection define a user graph (i.e., a social graph), in which the nodes are the users and the edges are the trust connections. The degree of a node is the number of members in the corresponding user's trust set.

The starting point of our investigation is recent work in social network modeling, which has revealed that social networks as they exist in Web-based systems are characterized by certain properties [4]. In particular, the degree distributions of nodes follow a power-law with a characteristic slope. We must necessarily assume that in real-world social data systems these characteristic distributions hold. In this paper we address the following question: *How does the degree distribution known to characterize real-world Web-based social systems impact trust-aware recommendation?* Our investigation consists of experiments on a series of semi-synthetic social data collections that are generated by mapping existing user-item matrices to social graphs with the degree distributions of known Web-based social systems. By using semi-synthetic graphs, we are able to experiment with multiple social data collections. In contrast, current studies of trust-aware recommendation limit themselves to a single data set. In most cases this set is derived from Epinions.com, e.g., [12][16][17][18][21], although a few exceptions exist, e.g., [6][11]. Our goal is to arrive at a better understanding of the potentials and limitations of trust-aware recommendation. Our study makes use of currently available public resources, but extends our present understanding of trust-aware recommendation by moving beyond the analysis of a single data set. Our work makes two key contributions: first, a method for generating semi-simulated social data collections that exploits recent advances in social network modeling [13] and, second, our finding that the extremely long tail that characterizes degree distributions of real-world Web-based social systems has serious consequences for trust-aware recommendation approaches.

In the next section, we cover related work. Then we present our method for generating semi-synthetic social data collections and the results of our experimental analysis of three such data collections. The last section presents a discussion and conclusion.

2. RELATED WORK

2.1 Social Trust and Recommendation

2.1.1 The Nature of Social Trust

The concept of trust is used differently in different contexts. Within computer science, trust can be a reflection of the quality of a peer in a peer-to-peer system or the reliability of an information source (cf. [7]). For the purpose of recommendation, trust is understood to be a relationship between users and the type of friends to whom they would turn in real life for recommendations [16]. A full understanding of how users assign trust relationships in social collections is not required in order for social trust information to aid recommendation. Minimally, social trust must only serve to capture shared tastes between users. As pointed out in [8], trust-based recommender systems generally assume a relationship between trust and similarity for this purpose. For instance, in the work that has been devoted to empirically investigating the correlation between social trust and interest similarity, a correlation was found for books [23] and then extended to books and film [24]. In [23], support was found for the hypothesis about a correlation between trust and user similarity when the trust network is closely associated with a particular application. In this work, we assume that connections between users in the user graph reflect a match in user taste as least as well as the user-user similarity calculations used in conventional user-based collaborative filtering.

2.1.2 Trust-aware recommendation

Various techniques have been proposed to improve CF by integration of social trust information. Closest to the work presented in this paper are approaches that involve replacing the neighbor set on which predictions are based in conventional user-based CF with a trust set of users found via the social network. In [17], an algorithm called MoleTrust carries out a depth-first walk of the trust graph. When the propagation horizon is set to one (i.e., only immediate neighbors are used) MoleTrust outperforms CF for rating prediction. Increasing the propagation horizon is shown to increase the coverage, but at the cost of prediction accuracy. Trust information becomes noisier and less useful as trust propagation moves beyond the first degree, i.e., draws on information outside of the users trust set. FilmTrust [6] uses a trust-flow-based method called TidalTrust and excels in cases where a user has an opinion on an item that diverges from average. In [11], social annotation and the social network are integrated in an approach that uses a random walk with restarts. TrustWalker presents a random walk model to combine traditional item-based collaborative recommendation and trust-based recommendation [12], which achieves promising improvement with respect to accuracy and coverage.

Another empirical study on trust-aware recommendation investigates the influence on recommendation performance in terms of both an individual social friend and a community-like ally in the social network [25], which also indicates the efficiency of recommendation based on socially selected users compared to traditional user-based CF. In [21], the question of whom users should trust is examined with particular attention to the role of key users. Ma et al. [16] have proposed a matrix factorization framework to fuse the user-item rating matrix and user social trust network. The framework makes use of the social network as a constraint for learning user latent features rather than basing the factorization solely on the user-item rating matrix.

Different approaches to social trust in recommendation aim to exploit different advantages. As mentioned above, our work is aimed at reducing computational complexity and making conventional CF more suitable for use in a decentralized system. Here, we mention some additional aspects. In view of the fact that CF suffers from data sparseness, which leads to noise since the similarity measure must be frequently calculated for users with little profile overlap, [18] aims to use social trust to mitigate this effect. In [8], the point is made that trust might be able to capture something beyond overall similarity between user profiles. A trust network could be exploited to extend the coverage of conventional CF [17]. Since users pick their own trustees, trust-based approaches could be more robust to attack than conventional CF [17]. Finally, trust-based recommendation systems may have the advantage over conventional systems because of greater transparency of the source of the recommendation [3]. Our work could possibly aid the development of trust-aware recommendation approaches that fulfill other potentials of social trust, a topic we leave for future investigation.

2.2 Social Network Modeling

Real-world social networks have been shown to be characterized by a specific set of properties including power-law degree distributions [4][20], small diameter [19][22], shrinking diameter and densification power law [15]. Those properties are used to guide the modeling of social networks, which has resulted in a number of network models, e.g., preferential attachment model [2][5] and the small-world model [22]. The recently introduced Kronecker graph model has been demonstrated to be capable of capturing multiple real social network properties simultaneously [13][14]. We choose this model to synthesize the social graphs that we use to create our semi-synthetic social data collections.

3. EXPERIMENTAL FRAMEWORK

3.1 Semi-synthetic social data collections

We generated semi-synthetic data collections using a naïve pruning process that maps an existing collection of user profiles (i.e., a user-item matrix) onto a social graph that has been generated such that its degree distribution corresponds to that of a real-world Web-based social system.

3.1.1 Kronecker graph model

In order to synthesize social graphs with real-world properties, we adopt the stochastic Kronecker graph (KG) model [13]. If we assume that P_1 is a $N_1 \times N_1$ Kronecker initiator matrix containing values $\theta_{ij} \in P_1$ denoting the probability of the existence of an edge, then, a social network with number of nodes N_1^k, denoted as P_k, can be synthesized by the kth Kronecker product of P_1, as shown below:

$$P_k = P_1^{[k]} = \underbrace{P_1 \otimes P_1 \otimes \ldots \otimes P_1}_{k \text{ times}} \quad (1)$$

Finally, a graph can be obtained by sampling an instance from the probability distribution defined by P_k. We adopt the 2×2 initiator matrices empirically obtained in [13] in order to simulate social networks, i.e., Epinions, Delicious and Flickr. Notice that although the KG model is designed to simulate a large range of properties of social networks, here, we only investigate degree distributions. We use a representation of the synthetic graph that consists of a list of nodes ordered by degree from large to small.

3.1.2 Idealized trust sets

Our naïve mapping from an existing user user-item matrix to the synthetic social graph builds on the basic assumption, mentioned above, that the trust set of a user reflects that user's interests at least as well as the set of neighbors computed via similarity under

conventional user-based CF. Building on this assumption, we create an idealized trust set for each user consisting of all users more similar than a threshold, θ. User-user similarity is calculated using the Pearson correlation [10], conventional user-based CF:

$$sim(u,v) = \frac{\sum_{i \in C_{uv}} (r_{ui} - \overline{r_u})(r_{vi} - \overline{r_v})}{\sqrt{\sum_{i \in C_{uv}} (r_{ui} - \overline{r_u})^2 \sum_{i \in C_{uv}} (r_{vi} - \overline{r_v})^2}} \quad (2)$$

Here, u and v are two users represented by standard profile vectors consisting of the ratings that they have assigned to items in the collection. C_{uv} denotes the intersection of items rated by both user u and v and r_{ui} denotes the user u's rating on item i. The average rating value of the user u across all co-rated items in C_{uv} is denoted by $\overline{r_u}$.

3.1.3 Social pruning

Our semi-synthetic social data collection is then generated by a process of social pruning that results in a *socially pruned user graph* (SPUG) is a user graph in which connections have been removed in order to make its degree distribution respect that of the KG model, which represents a real-world social system. The mapping process between users in the idealized user graph and nodes in the synthesized social graph is straightforward. The nodes in the synthetic social network have been ranked by order of degree (i.e., number of connections). Also, the users in the data collection (i.e., the original user-item matrix) have been ranked by the number of neighbors. We then associate the most highly connected node with user with the most neighbors and move down the list, pairing users and nodes in order. In each case, we prune the users from the idealized trust set so that it matches the degree of the associated node in the synthetic social network. A judicious choice of θ ensures that in a majority of cases, the degree distribution of the idealized user graph can be forced into the degree distribution of the target Web-based social system only by removing links. The pruning process is facilitated by our choice of mapping the most highly connected users to most highly connected nodes. In theory, any mapping between users and nodes would be adequate for our purposes. In the very rare case that a link must be added, we do so by adding a random connection. The resulting SPUG follows exactly the same degree distribution as the network that the KG model simulates, which means that the semi-synthetic social data collection shares the property of degree distribution with a real-word Web-based social system.

3.2 Top-N recommendation

We experiment on the task of Top-N recommendation, i.e., produce a list of items for the user ranked in order of user preference.

3.2.1 User-based CF with Trust Sets

For the purpose of ranking items we use the Trust Set Average Rating (TrustSetAvRat). The TrustSetAvRat rating score for user u on item q is calculated as:

$$r_{uq} = \frac{\sum_{v \in T(u)} t_{uv} r_{vq}}{\sum_{v \in T(u)} t_{uv} I_{vq}} \quad (3)$$

where *T(u)* denotes the trust set of user u and t_{uv} is a weight indicting how much user u trusts v. TrustSetAvRat is equivalent to trust-based recommendation in [6] and MT1 in [17]. Here, we only use binary trust, i.e., t_{uv}=1 if u trusts v, 0 otherwise. I_{vq} is an indication function where I_{vq}=1 if user v rated item q, 0 otherwise.

3.2.2 Experimental baselines

As a naïve baseline, we use ItemAvRat, the collection wide average rating assigned to an item. Under this approach, for every user the recommended list contains the Top-N items collection-wide, ranked in order of overall popularity. As a second, more sophisticated baseline, we use the idealized trust sets calculated using user-user similarity in Eq. 2. This baseline, which we designate IdTrustSet, is comparable to conventional user-based CF. As can be observed in Eq. 3., IdTrustSet lacks the factor weighting the contributions of the individual users by their similarities with the target user used in conventional user-based CF. Since we are interested in a comparison with the extremely long tails of characteristics of the degree distributions of real-world social graphs, we choose θ such that the degree distribution of the resulting idealized trust sets approaches a power law distribution.

3.3 Data sets

In our experiments, we used the publicly available Jester (JS) data collection (rating scale -10–10) [9]. Our approach constitutes an innovative new use of these data. Using the latest JS data set (which contains a total of 64K users and 150 items), we create two subsets: JS1, which contains 1024 (2^{10}) users and JS2, which contains 4096 (2^{12}) users. We select only users that have rated at least 20 of the 150 items. Note that since the node count of a graph synthesized by the KG model is always a power of two, our choice of set size ensures an exact match during pruning.

3.4 Experimental Protocol

We adopt the widely used 5-fold cross-validation protocol in our experiments. In each fold we use 80% ratings of each user for training and remaining 20% ratings for testing. For Top-N recommendation, we take items with rating equal or higher than 7.5 to be relevant items. The training set is used to generate the predictions on all other items, which then can be compared with ground truth in the test set for evaluation. We report results using the Mean Reciprocal Rank (MRR), the inverse position of the topmost relevant document in the Top-N recommendation list.

4. EVALUATION

We create socially pruned user graphs by pruning the JS1 and JS2 data sets to respect the degree distributions of the synthetic graphs using the process described in Section 3.1.3. We create three semi-synthetic data sets with degree distributions corresponding to those of Epinions, Delicious and Flickr. Our implementation of Kronecker graph modeling is based on the publicly available software SNAP (http://snap.stanford.edu/snap/index.html).

The data sets have a dramatically large number of users with very small trust sets (< 10), differing dramatically from idealized trust sets. In Table 1, CF performance in the case of socially pruned user graphs (indicated as SPUG) can be seen to be markedly lower than in the case of either IdTrustSet (i.e., the baseline comparable to conventional user-based collaborative filtering) or ItemAvRat, the naïve baseline.

Table 1. MRR performance comparison between the baselines and JS1 and JS2 data sets, the socially pruned data sets

	ItemAvRat	IdTrustSet (~ user-based CF)	Epinions SPUG	Delicious SPUG	Flickr SPUG
JS1	0.279	0.345	0.130	0.142	0.126
JS2	0.180	0.263	0.144	0.161	0.146

We conclude that the very long tail observed in real-world Web-based systems has serious consequences for trust-aware recommendation. When the user-graph is forced into the degree distributions known to exist in the wild, CF performance is not longer able to approach the simple baseline.

5. DISCUSSION AND CONCLUSIONS

This paper has made a novel contribution to the current understanding of the challenges facing the effective use of social trust information for collaborative filtering. Our goal has been to transcend some of the limitations affecting current research on trust-aware recommendation and to shed light on a specific area of dependency between a property of social data collections (degree distribution of the social graph) and the performance of trust-aware recommendation. Our contributions are twofold. First, we proposed a method for producing semi-synthetic social data collections from existing data collections (i.e., user-item matrices). Second, our experiments provide evidence that the very long tail characterizing the degree distributions of real-world social graphs can have a devastating impact on the performance of CF.

Our results suggest that if a Web-based community is intended to support trust-aware recommendation, it should be planned in such a way that it will foster the emergence of patterns of social relationships among users that are advantageous for trust-aware recommendation. Suitable incentives should be provided to users so that user graphs are encouraged to develop, contrary to their natural tendencies, to have a degree distribution with a relatively shorter long tail. Note that we do not recommend forcing users into certain behavior, but rather an approach that would simply encourage users who set up few connections to set up more by making clear to them the potential benefits. Our exploratory experiments with idealized trust sets suggested that a long-tail distribution does not necessarily stand in the way of effective trust-aware recommendation, rather it is the *very* long tail that is damaging. Recall that our technique for generating semi-synthetic social data collections built on the assumption that in the real-world users' trust sets reflect user interest at least as well as the similarity neighborhoods used in conventional user-based collaborative filtering. The picture brightens, if, in a particular social data collection, users connect to each in a way that results in trust sets providing a better basis for recommendation than similarity neighborhoods. However, in a social data collection in which users choose the "wrong friends", the negative impact of very long-tail degree distributions could be exacerbated to the point of making trust-aware recommendation useless. Web-based communities are well advised to provide incentives to users to chose the "right friends."

Our future research will involve developing techniques to synthesize graphs that model second-degree relationships among users in order to test recommendation approaches that exploit trust propagation.

6. ACKNOWLEDGEMENTS

The research leading to these results was carried out within the PetaMedia Network of Excellence and has received funding from the European Commission's 7th Framework Program under grant agreement n° 216444.

7. REFERENCES

[1] Adomavicius G., and Tuzhilin, A., 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE TKDE, 17, 6, 734-749.

[2] Barabasi A. L., and Albert R., 1999. Emergence of scaling in random networks. Science, 286, 509-512.

[3] Bonhard, P., Sasse, M., July 2006. 'Knowing me, knowing you' — using profiles and social networking to improve recommender systems. BT Technology Journal 24, 3, 84-98.

[4] Clauset, A., Shalizi, C. R., and Newman M., 2009. Power-law distributions in empirical data. SIAM Review, 51, 661-703.

[5] Flaxman A. D., Frieze A. M., and Vera J., 2007. A geometric preferential attachment model of networks II. WAW ′07, 41-55.

[6] Golbeck, J., 2006. Generating predictive movie recommendations from trust in social networks. iTrust ′06, 93-104.

[7] Golbeck, J. 2006. Trust on the world wide web: a survey. *Found. Trends Web Sci.* 1, 2, 131-197.

[8] Golbeck, J., 2009. Trust and nuanced profile similarity in online social networks. ACM Trans. Web, 3, 4, 1-33.

[9] Goldberg, K., Roeder, T., Gupta, D., Perkins, C., July 2001. Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval 4, 2, 133-151.

[10] Herlocker, J., Konstan, J., Borchers, A., and Riedl, J., 1999. An algorithmic framework for performing collaborative filtering. SIGIR ′99, 230-237.

[11] Konstas, I., Stathopoulos, V., and Jose J. M., 2009. On social network and collaborative recommendation. SIGIR ′09, 195-202.

[12] Jamali, M., and Ester, M., 2009. TrustWalker: a random walk model for combining trust-based and item-based recommendation. KDD ′09, 397-406.

[13] Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., and Ghahramani, Z., 2010. Kronecker Graphs: An approach to modeling networks. JMLR, 11(Feb):985-1042.

[14] Leskovec, J., and Faloutsos, C., 2007. Scalable Modeling of Real Graphs using Kronecker Multiplication. ICML ′07, 497-504.

[15] Leskovec, J., Kleinberg, J., and Faloutsos, C., 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. KDD ′05, 177-187.

[16] Ma, H., King, I., and Lyu, M. R. 2009. Learning to recommend with social trust ensemble. SIGIR '09, 203-210.

[17] Massa, P. and Avesani, P., 2007. Trust-aware recommender systems. RecSys ′07, 17–24.

[18] Massa, P., and Bhattacharjee, B., Using trust in recommender systems: An experimental analysis. iTrust ′04, 221-235.

[19] Milgram, S., 1967. The small-world problem. Psychology Today, 2, 60-67.

[20] Newman, M., 2005. Power laws, Pareto distributions and Zipf's law. Contemporary Physics, 46, 5, 323-351.

[21] Victor, P., Cornelis, C., De Cock, M., and Teredesai, A. M. 2008. Key figure impact in trust-enhanced recommender systems. *AI Commun.* 21, 2-3, 127-143

[22] Watts, D. J., and Strogatz, S. H., 1998. Collective dynamics of 'small-world' networks. Nature, 393, 440-442.

[23] Ziegler, C.-N., Lausen, G., 2004. Analyzing correlation between trust and user similarity in online communities. *Trust Management*, 251-265.

[24] Ziegler, C. and Golbeck, J. 2007. Investigating interactions of trust and interest similarity. *Decis. Support Syst.* 43, 2, 460-475.

[25] Zheng, R., Wilkinson, D., and Provost, F., 2008. Social network collaborative filtering. Working paper CeDER-8-08, Center for Digital Economy Research, New York University.

Toward a Design Space for the Recommendation of Communities

Sven Buschbeck
German Research Center for Artificial Intelligence

Anthony Jameson
German Research Center for Artificial Intelligence

ABSTRACT

Recent years have seen an explosion in the number of on-line communities; a person looking for a community for some purpose may have hundreds to choose from. Helping users make such choices therefore constitutes an interesting application area for recommender systems, which has so far been explored only sporadically. To facilitate further exploration of this area, we discuss the following questions: What goals can a user be pursuing when looking for a community? What sorts of information about communities can be available that could be used by a recommender system? The answers to the second question are based on a systematic examination of a sample of 15 communities on three web-based platforms.

1 INTRODUCTION

One reason for the vitality of the recommender systems field is the great variety of types of entities that can be recommended. Each new type of entity may have a different set of relevant properties, and people may have different reasons for being interested in that type of entity. Therefore, new combinations of recommendation techniques and interfaces may be required.

A case in point is the class of online communities, whose importance has been growing rapidly in recent years. Many people now spend a considerable proportion of their time acting as a member of one online community or the other; and they may have more superficial contact with a larger number of other online communities. Consequently, helping people to find the right online community for a given purpose is a promising application area for the recommender systems community.

This short paper aims to strengthen the existing foundation for studying the recommendation of communities by briefly summarizing previous work on this topic; considering the various goals that people may have when looking for an online community; and surveying the various properties of online communities that may be relevant to recommendation. The discussion of this latter point is fleshed out with an analysis of a sample of web-based communities.

The research described in this position paper is being conducted in the context of the 7th Framework EU Integrating Project GLOCAL: Event-based Retrieval of Networked Media under grant agreement 248984.

Because the authors are especially interested in online communities for sharing media, parts of the discussion are especially relevant to these and similar communities.

2 RELATED WORK

We use the term *community* to denote a collection of computer users who have joined an explicitly defined group on a platform (such as Orkut, Flickr, or Facebook) that supports the formation of such groups and provides facilities for the members to exchange information with each other and engage in other joint activities. In some platforms, other terms such as *group* are used instead. Hence, in our terminology a web portal like Flickr is referred to as platform or a network of communities rather than as a community.

Brocco and Groh ([1]) discuss the problem of *team recommendation*, introducing a useful set of general concepts for characterizing both team members and teams. Since their focus is on composing new teams rather than recommending existing ones, and since working teams are different in some ways from online communities in general, the results of this work require some adaptation to be applied to the problem of community recommendation.

The problem of helping users to find interesting communities was addressed directly by the TOUCHGRAPH LIVEJOURNAL BROWSER (see, e.g., [6]), which uses information visualization techniques to show a user what communities of interest his or her friends belong to. Although this sort of visualization does not constitute a recommendation, it could be used as part of a recommender system for communities.

Similarly, Diedrich et al. focused in [5] on creating user profiles on the basis of the tagging behavior of each user. Although this approach does not deal directly with the recommendation of communities to users, it does provide a promising way of classifying users to whom a given community might be recommended.

Information about the communities that friends belong to, along with other information, was used for real recommendation of communities in the system SONARS ([2]).

Spertus et al. ([7]) empirically explored methods for recommending communities to a given user on the basis of their similarity to a community that the user belongs to. They compared several similarity metrics that consider only the set of members of each community, treating each commu-

nity as a *bag of users*. Chen et al. ([4]) went a step further by treating each community in their algorithms as both a bag of users and a *bag of words* (e.g., a semantic description of the community). As we will discuss below, there is a great deal of other information about communities that a recommender could in principle take into account, but focusing on bags of users and/or bags of words has advantages in terms of efficiency and scalability, and it remains to be seen what can be gained by taking other types of information into account.

Research by Chen et al. [3] yields some relevant insights even though it concerns the recommendation of individual persons rather than the recommendation of communities. Comparing several person recommendation algorithms, these researchers found that different algorithms and types of information were best suited to different goals and contexts (e.g., looking for people that the user already knows vs. looking for new friends).

Our brief analysis in this paper argues that this idea is likely to apply to community recommendation as well: We will point to the diverse goals that people can have when looking for communities and to the diverse types of information about communities that are commonly available. Our aim is to motivate research on the relevance of different types of information for recommendation to users with different goals.

3 POSSIBLE USER GOALS

There are many reasons why a user might be interested in finding a community, some less obvious than others. Even though a recommender system may not be able to find out much about the current user's goals, it is important for designers to be aware of this range of goals. Since an exhaustive analysis of users' possible goals would be beyond the scope of this paper, we present several examples to illustrate the range of possibilities.

Note in particular that a user may be able to benefit from the existence of a given community without joining that community or even interacting with its members. Regardless of their primary reason for existence, communities can be exploited as information resources.

Goal: Find Information or Media: A user may want to get information or media relating to a specific topic. For example, a journalist may be searching for photos of erupting volcanoes while writing a story about the recent eruption in Iceland. If there is an active community of experts on that topic, it might be a good option to join that community or at least try to get access to the collection of media maintained by that community. Note that in some cases it may be easier to find a community of people interested in a given topic than to find a specific piece of information or a particular medium related to that topic. So even if the community is not in itself of interest to the user, it can serve as a stepping stone to the desired information or media.

Goal: Establish Real-World Contacts: A user may want to organize some kind of event, such as a local soccer match for children, and need to acquire helpers. Hence she is searching for people with specifically required knowledge, skills and experience from her region willing to support her. A community of people from the same geographical region who are interested in similar topics may be a good place for her to find the required persons.

Goal: Get On-Line Help and Feedback: A user may be working on a specific task and be looking for help and/or feedback. As with the previous goal, a relevant online community can be a useful resource; but the requirements placed on such a community are partly different.

Goal: Offer Information or Media: Another goal could be to offer information or media to other users. For example, suppose that a user is a member of an open source software project. Now that a new major release is coming up, she would like to inform other potentially interested users about the new features and invite them to download and test the new piece of software. Especially helpful to the user will be large communities interested in this topic. Additionally, the user may prefer communities with highly active members who are likely to respond promptly.

4 ANALYSIS OF TYPICALLY AVAILABLE INFORMATION

Finding a community that satisfies a particular goal requires taking into account relevant information about that community—or having a recommender system take it into account. Since communities are complex entities, they have many qualitatively different attributes (as has been discussed in some of the related work summarized in section 2). But which of these attributes might be known to a recommender system, and what goals might they be relevant to?

As an initial empirical basis for answers to these questions, we analyzed a sample of communities on three widely used web platforms: Orkut (which supports social networks in general, without restriction to a particular population or type of content), Flickr (which supports communities that exchange visual media), and LinkedIn (which supports business-related communities). These three platforms were chosen because each is typical of a particular type of social network: generic, media-related, and business-related social networks, respectively. For each platform, 5 communities of 1000 to 40000 users (14000 users on the average) from 5 popular topic areas (technology, soccer, news, travel, and nature) were selected for examination.

Table 2 discusses the information that could be found about a community's individual members, while Table 1 looks at the attributes of the communities themselves. The tables also include remarks about how the various types of information about communities might be useful for the recommendation of communities to users with particular goals.

Question Considered		Detailed Results			Overall Results	
Question Category	*Question*	Orkut	Flickr	LinkedIn	To what extent was this information found in our sample of communities on three platforms?	How could this information be useful in the recommendation of communities to users with particular goals?
Demographic Variables	What is their age distribution?	Unknown	Unknown	Unknown	Unknown	Users searching for social contact might be interested in a specific age range.
	Do we know the gender distribution?	Yes	Maybe (Optional user profile field)	No	Mandatory field in user profiles on Orkut, optional on Flickr, non-existing on LinkedIn.	If a user is interested in finding contacts to meet in person or to talk about gender-specific topics, he or she might want to filter by gender
	What languages do the users speak?	Mostly English	Mostly English	Mostly English	Mostly available	A user's languages must in general include the community's language in order that the user can make any use of the community (except for some superficial uses)
	Can we get to know the geographical location of the user?	Yes	Maybe	Maybe (Location might be derived from user's occupation)	Mostly available; country has to be entered in Orkut and users of Linkedin tell their location indirectly by naming their current employer, but it is an optional property on Flickr.	If a user's intention is to meet community members personally, the geographical location is of very high importance.
Fuctional variables	Can we get to know something about skills, interests or hobbies?	Maybe (optional list of interests)	Maybe (Only as part of user´s plain text description)	Yes (skills and education)	Always available on business-oriented platforms, less often on social networks	If a user wants to hire / acquire people, skills and eduction are very important to make a decission
Social network variables	Can we get to know something about other connections of this user?	Yes	Yes	Yes		
	Can we access the number of a user's contacts and communities, and further about these?	Yes	Yes	Yes	Always available	If the searcher aims to establish contact, another user's contacts and communities can be an indicator of the how much the other person is interested in establishing new contacts as well

Table 1. *Analysis of recommendation-relevant information about the members of online communities that can be found in a sample of communities on three platforms.*

5 QUESTIONS FOR FUTURE RESEARCH

The ideas and information presented above and in the tables are intended to serve as a starting point for the investigation of issues like the following:

- What recommendation techniques are best suited for matching users who have particular goals with communities of a given type?
  Although it seems unlikely that every combination of user goals and community type will call for a different recommendation algorithm, it also seems unlikely that a single community recommender system can perform effectively in all of the situations mentioned above.
- What particular forms can interaction with a community recommender system take?
  For example, to what extent should the system try to acquire information about the user's goals in looking for a community, and how could it do so?

Though we have taken but a first small step toward answering these questions, we believe that it is important to start with a realistic understanding of the range of user goals and the types of available information.

REFERENCES

1. Michele Brocco and George Groh. Team recommendation in open innovation networks. In *Proceedings of the 2009 ACM Conference on Recommender Systems*, pages 365–368, New York, 2009. ACM.

2. Francesca Carmagnola, Fabiana Vernero, and Pierluigi Grillo. SoNARS: A social networks-based algorithm for social recommender systems. In *Proceedings of UMAP 2009, the 17th International Conference on User Modeling, Adaptation, and Personalization*, Berlin, 2009. Springer.

3. Jilin Chen, Werner Geyer, Casey Dugan, Michael Muller, and Ido Guy. "Make new friends, but keep the old" – Recommending people on social networking sites. In Saul Greenberg, Scott Hudson, Ken Hinckley, Meredith R. Morris, and Dan R. Olsen, editors, *Human Factors in Computing Systems: CHI 2009 Conference Proceedings*, pages 201–210. ACM, New York, 2009.

4. Wen-Yen Chen, Dong Zhang, and Edward Y. Chang. Combinational collaborative filtering for personalized community recommendation. In *Proceedings of KDD 2008*, New York, 2008. ACM.

5. Jrg Diederich and Tereza Iofciu. Finding communities of practice from user profiles based on folksonomies. In *First International Workshop on Building Technology-Enhanced Learning Solutions for Communities of Practice*, Crete, 2006.

6. Amy Soller. Adaptive support for distributed collaboration. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, pages 573–595. Springer, Berlin, 2007.

7. Ellen Spertus, Mehran Sahami, and Orkut Buyukkokten. Evaluating similarity measures: A large-scale study in the Orkut social network. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 678–684, New York, 2005. ACM.

Question Considered		Platform-Specific Results			Overall Results	
Question Category	*Question*	Orkut	Flickr	LinkedIn	To what extent was this information found in our sample of communities on three platforms?	How could this information be useful in the recommendation of communities to users with particular goals?
Group Longevity	Do we know how long the community has existed?	Yes	Yes	Yes	Always	In combination with the community's activity, it can be an indicator of the quality of organization and motivation of community members.
Organization	Is there someone who is in charge?	Yes	Yes	Yes	The creator becomes the owner of a community	The owner's user profile can be an indicator of the community's intended purpose / topic.
	Can we learn something about the geographical location of the community?	Yes; mostly Asia and South America	In some cases	In some cases	There is a geographical location for each community in Orkut only; in the other networks, the community location could be calculated from the location of the members.	If a user is interested in meeting the members in person, geographical proximity is of great interest. It also has implications concerning other likely properties of members.
Social Performance	What is the number of community members?	13666 on the average	16154 on the average	13316 on the average	Always available	The number of users represents the popularity of the community's topic, but it can also be seen as indicator of the quality of the content and organization. If a user is looking for feedback, the number of community members is important.
Information System	Does the community maintain some collection of objects?	Yes	Yes	Yes	All communities maintain a collection of some kind of objects	The collected items (regardless of their type) are a good source for knowing how active a community is and what it is interested in.
	If Yes, what sort of objects are maintained in the collection?	Events and surveys	Media	News items, job offers	On two of three platforms, communities maintain a collection of events (also called "News")	
	If it is a collection of media items, what kind of media?	n/a	Photos and videos	n/a	Supported types of media are always available in communities with a media collection	Image features can be extracted to learn about the interests of the community.
	What is known about the media?	n/a	Title, description, tags, and device meta data (EXIF)	n/a	A lot of detailed information	Metadata usually contains location, date, and topic in terms of title, description, and tags, thus providing insight into the community's interests.
	How are the media items geographically distributed?	n/a	Mostly Europe, North America, and Central Asia	n/a	Locations can be plotted on a map if geographic data is available.	Relevant if a user is interested in a specific region or if she is interested in a specific topic and wants to be sure that the content is not dominated by a particular local point of view.
	How many items are in the collection?	23 surveys and 0 events on the average	29,3311 on the average	Mostly unknown	Mostly available	On the one hand, a high number of items can be evidence of the community's motivation, but it can also mean that the community is less organized or focused.
	What is the source of the items?	Members of C	Members of C	Members of C	Always available	The number of contributions from a given user can be used as a weight for the corresponding user profile when the system tries to learn about the community's interests.
	Can members comment on objects?	Yes	Yes	Yes	Always available	The number of comments suggests how active a community is and how much the members are interested in social interaction.
Communication System	Can we learn something about the message-based communication between the members of C?	Yes	Yes	Yes	Mostly available; sometimes only to community members.	(See the comments below)
	If Yes, is there a forum?	Yes	Yes	Yes	Always available	
	If Yes, how many posts does the forum contain?	229 on the average	942 on the average	Mostly unknown		The number of posts relative to the number of community members can be evidence of the quality of content and moderation, and focus of members' interests.
	How many replies are there to each post?	7 on the average	34 on the average	Mostly unknown	Indirectly available by counting replies. High variability: Some posts got 100+ replies, while many remained unanswered.	The ratio of number of replies to number of posts can represent the users' willingness to communicate and interact.
	Is there a specific language associated with the community?	Yes	No	No	The majority of communities do not define the spoken language in advance, but English is the language of the vast majority of communities.	User's language must match community in order that the user can make any use of the community.
	Are there communities with mixed languages?	No	No	No	None of these communities contained discussions in different languages.	If a community's language is not given explicitly, it can be derived from the members' communication
	Can users communicate with each other directly?	Yes	Yes	Yes	Means of direct communication are available, but the communications themselves are not accessible from the outside for privacy reasons	Users' current interests could be derived from communication content if it was available.
	If Yes, how?	Mail, instant messaging	Mail	Mail	At least email is implemented on all platforms.	Communication frequency can be an indicator of how communicative a user is.

Table 2. *Analysis of recommendation-relevant information about online communities as such that can be found in a sample of communities on three platforms.*

Collaboration and Reputation in Social Web Search

Kevin McNally, Michael P. O'Mahony, Barry Smyth, Maurice Coyle, Peter Briggs
CLARITY Centre for Sensor Web Technologies
School Of Computer Science & Informatics
University College Dublin
{firstname.lastname}@ucd.ie

ABSTRACT

Recent research has highlighted the inherently collaborative nature of many Web search tasks, even though collaborative searching is not supported by mainstream search engines. In this paper, we examine the activity of early adopters of HeyStaks, a collaborative Web search framework that is designed to complement mainstream search engines such as Google, Bing, and Yahoo. The utility allows users to search as normal, using their favourite search engine, while benefiting from a more collaborative and social search experience. HeyStaks supports searchers by harnessing the experiences of others, in order to enhance organic mainstream result-lists. We review some early evaluation results that speak to the practical benefits of search collaboration in the context of the recently proposed *Reader-to-Leader* social media analysis framework [11]. In addition, we explore the idea of utilising the reputation model introduced by McNally et al. [6] in order to identify the *search leaders* in HeyStaks, i.e. those users who are responsible for driving collaboration in the HeyStaks application.

Categories and Subject Descriptors

H.4.0 [**Information Systems Applications**]: General

General Terms

Algorithms, Experimentation, Security

Keywords

Collaborative Web Search, Social Recommender System, Reputation Model, HeyStaks

1. INTRODUCTION

The so-called *Social Web* paradigm serves to emphasise how the culture of the Internet has evolved far beyond simple information transfer, and how it is becoming an increasingly social and collaborative environment. The Internet today is a place where individuals learn about the views and opinions of others, a place where they can express their own opinions or rate content and services. They can even comment on the opinions of others, and otherwise share their views and observations about the world in which we live. And they can do this all within an eco-system of implicit and explicit communities, which harness groups, large and small, of like-minded users. Today billions of people participate in these types of social activities whether through large destination sites such as Facebook, MySpace, Twitter, Wikipedia, Flickr, or Amazon, or via the millions of blogs and discussion boards that cover every conceivable topic.

This level of information sharing and social activity has provided the raw material for a new form of online collaboration helping millions of users to make better decisions during their everyday lives. Amazon's user reviews have proved to be a vital source of trusted product information, for example, to help millions of users to make better purchases, whether through Amazon itself or elsewhere. Similarly, Twitter is now an important way for people to discover interesting Web pages with their friends and followers. Indeed the potential of the Internet as an open collaboration platform is exemplified by a new generation of so-called *social tools* that allow groups of users to collaborate on a wide range of common tasks, from document creation and editing (e.g. Google Docs, Write.ly, etc.), messaging (e.g. Twitter, Yammer etc.), data modeling and visualization (e.g. ManyEyes), knowledge sharing and conversation (e.g. Wikis and blogs) to the grand vision of Google Wave as a new platform for collaborative communication.

In all of this there is one aspect of Internet life that has yet to get the truly *collaborative* treatment, and that is Web search. After email, Web search is perhaps the most used Internet service as the leading search engines deal with literally billions of queries every day. However, until relatively recently Web search has largely been viewed as a solitary service in which individual users interact, in isolation, with their search engine of choice. All this is set to change, however, as researchers have begun to question the solitary nature of Web search, proposing a more collaborative search model in which groups of users can cooperate to search more effectively [3, 12, 13, 14, 15, 18]. Moreover recent work by [7] highlights the inherently collaborative nature of more general purpose Web search. Indeed, despite the absence of explicit collaboration features from mainstream search engines, there is clear evidence that users implicitly engage in many different forms of collaboration as they search, although, as reported by [7], these collabora-

tion "work-arounds" (email, instant messaging etc.) are often frustrating and inefficient. Naturally, this has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines.

In this paper we consider the need for, and potential of, a more collaborative vision of Web search, one in which the appropriate experiences of relevant users can be harnessed in pursuit of a better, more productive, search experience. We consider the behaviour of users of *HeyStaks* (www.heystaks.com) which has been designed to provide just such a collaboration facility as an additional layer on top of mainstream search engines. Specifically, we examine the behaviour of a group of early adopters of the system, focusing on any collaboration they have taken part in. We consider user behaviour in terms of the Reader-to-Leader framework proposed in [11], which distinguishes between the users of a social system as a function of their level of interaction within that system. In addition, we show how a model of user reputation can be applied to discover search leaders, i.e. those users who drive search collaboration by contributing high quality search knowledge to their community.

2. A REVIEW OF COLLABORATIVE INFORMATION RETRIEVAL

Collaborative information retrieval research takes a fresh look at information retrieval and Web search, which highlights the potential for collaboration between searchers during extended search tasks. Recent work by [7] highlights the inherently collaborative nature of general purpose Web search. For example, during a survey of just over 200 respondents, clear evidence for collaborative search behaviour emerged. More than 90% of respondents indicated that they frequently engaged in collaboration at the level of the *search process*. For example, 87% of respondents exhibited "back-seat searching" behaviours, where they watched over the shoulder of the searcher to suggest alternative queries. A further 30% of respondents engaged in search coordination activities, by using instant messaging to coordinate searches. Furthermore, 96% of users exhibited collaboration at the level of *search products*, that is, the results of searches. For example, 86% of respondents shared the results they had found during searches with others by email. Almost 50% of respondents telephoned colleagues directly to share Web search results, while others prepared summary documents and/or Web pages in order to share results with others.

Thus, despite the absence of explicit collaboration features from mainstream search engines there is clear evidence that users implicitly engage in many different forms of collaboration as they search, although, as reported by [7], these collaboration "work-arounds" are often frustrating and inefficient. Naturally, this has motivated researchers to consider how different types of collaboration might be supported by future editions of search engines. The resulting approaches to *collaborative information retrieval* can be usefully distinguished along two important dimensions, namely *time* and *place*. In terms of the former, collaborative search systems can be designed to support *synchronous* or *asynchronous* collaborative search. And in terms of the latter, systems can be designed to support either *co-located* or *remote* forms of collaborative search.

Co-located systems offer a collaborative search experience for multiple searchers at a single location, often a single PC (e.g. [1]) or, more recently, by taking advantage of computing devices that are more naturally collaborative, such as table-top computing environments (e.g. [17]). In contrast, remote approaches allow searchers to perform their searches at different locations across multiple devices; see e.g. [8, 9, 21]. While co-located systems enjoy the obvious benefit of an increased faculty for direct collaboration that is enabled by the face-to-face nature of co-located search, remote services offer a greater opportunity for collaborative search.

Synchronous approaches are often characterised by systems that broadcast a "call to search" in which specific participants are requested to engage in a well-defined search task for a well defined period of time; see e.g. [16]. In contrast, asynchronous approaches are characterised by less well-defined, ad-hoc search tasks and provide for a more open-ended approach to collaboration in which different searchers contribute to an evolving search session over an extended period of time; see e.g. [8, 19]. In this paper we will focus on a community-based approach to collaborative Web search in which the *asynchronous* search experiences of communities of like-minded *remote* searchers are harnessed to provide an improved search experience that is more responsive to the learned preferences of a community of searchers.

In designing HeyStaks our primary goal is to provide social Web search enhancements, while at the same time allowing searchers to continue to use their favourite search engine. As such, a key component of the HeyStaks architecture is a browser toolbar that permits tight integration with search engines such as Google, allowing searchers to search as normal while providing a more collaborative search experience via targeted recommendations. We now briefly look at the HeyStaks architecture and recommendation engine, as well as how users interact with the system. A more detailed description of the HeyStaks architecture and recommendation engine is given by Smyth et al. [20].

2.1 The HeyStaks System Architecture

HeyStaks adds two important collaboration features to any mainstream search engine. First, it allows users to create *search staks*, as a type of folder for their search experiences at search time. Staks can be shared with others so that their own searches will also be added to the stak. Second, HeyStaks uses staks to generate recommendations that are added to the underlying search results that come from the mainstream search engine. These recommendations are results that stak members have previously found to be relevant for similar queries and help the searcher to discover results that friends or colleagues have found interesting, results that may either be buried deep within Google's default result-list or not present at all.

HeyStaks takes the form of two basic components: a client-side *browser toolbar* and a back-end *server*. The toolbar allows users to create and share staks and provides a range of ancillary services, such as the ability to tag or vote for pages. The toolbar also captures search result click-thrus and manages the integration of HeyStaks recommendations with the default result-list. The back-end server manages the individual stak indexes (indexing individual pages against query/tag terms and positive/negative votes), the stak database (stak titles, members, descriptions, status, etc.), the HeyStaks social networking service and, of course, the recommendation engine.

HeyStaks users can create staks to search within, auto-

matically storing results as they search. If others join the stak these results can be shared, and the new stak members can input their own shareable search knowledge. This can be particularly useful when a user wishes to harness the knowledge of a community bound together by a particular topic. Members of a community searching within a stak can input search results simply by selecting them, or by performing HeyStaks specific actions: Adding keywords to a page viewable by their community (*tagging*), voting on the page (*vote-up* if they like the page, or *vote-down* if they don't), or sharing pages directly with other HeyStaks users. These results are subsequently recommended to any member of the stak, if deemed relevant by the HeyStaks recommendation engine, appearing as an augmentation to a search engine's query result list,. This recommendation engine is discussed in the following section.

2.2 The HeyStaks Recomendation Engine

In HeyStaks each search stak (S) serves as a profile of the search activities of the stak members. Each stak is made up of a set of result pages ($S = \{p_1, ..., p_k\}$) and each page is anonymously associated with a number of implicit and explicit interest indicators, including the total number of times a result has been selected (sel), the query terms ($q_1, ..., q_n$) that led to its selection, the number of times a result has been tagged (tag), the terms used to tag it ($t_1, ..., t_m$), the votes it has received (v^+, v^-), and the number of people it has been shared with ($share$) as indicated by Eq. 1.

$$p_i^S = \{q_1, ..., q_n, t_1, ..., t_m, v^+, v^-, sel, tag, share\} \quad (1)$$

In this way, each page is associated with a set of *term data* (query terms and/or tag terms) and a set of *usage data* (the selection, tag, share, and voting count). The term data is represented as a Lucene (*lucene.apache.org*) index, with each page indexed under its associated query and tag terms, and provides the basis for retrieving and ranking *promotion candidates*. The usage data provides an additional source of evidence that can be used to filter results and to generate a final set of recommendations. At search time, recommendations are produced in a number of stages: first, relevant results are retrieved and ranked from the stak index; next, these promotion candidates are filtered based on the usage evidence to eliminate noisy recommendations; and, finally, the remaining results are added to the Google result-list according to a set of *recommendation rules*.

Retrieval & Ranking. Briefly, there are two types of promotion candidates: *primary promotions* are results that come from the active stak S_t; whereas *secondary promotions* come from other staks in the searcher's stak-list. To generate these promotion candidates, the HeyStaks server uses the current query q_t as a probe into each stak index, S_i, to identify a set of relevant stak pages $P(S_i, q_t)$. Each candidate page, p, is scored using a *TF*IDF*-based retrieval function as per Equation 2, which serves as the basis for an initial recommendation ranking.

$$score(q_t, p) = \sum_{t \epsilon q_t} tf(t \epsilon p) \bullet idf(t)^2 \quad (2)$$

Evidence-Based Filtering. Staks are inevitably noisy, in the sense that they will frequently contain pages that are not on topic. As a result, the retrieval and ranking stage may select pages that are not strictly relevant to the current query context. To avoid making spurious recommendations HeyStaks employs an *evidence filter*, which uses a variety of threshold models to evaluate the relevance of a particular result, in terms of its usage evidence; tagging evidence is considered more important than voting, which in turn is more important than implicit selection evidence. Further, pages that have received a high proportion of negative votes will be eliminated.

Recommendation Rules. After evidence pruning we are left with revised primary and secondary promotions and the final task is to add these *qualified recommendations* to the Google result-list. HeyStaks uses a number of different recommendation rules to determine how and where a promotion should be added. Once again, space restrictions prevent a detailed account of this component but, for example, the top 3 primary promotions are always added to the top of the Google result-list and labelled using the HeyStaks promotion icons. If a remaining primary promotion is also in the default Google result-list then this is labeled in place. If there are still remaining primary promotions then these are added to the secondary promotion list, which is sorted according to HeyStaks relevance values. These recommendations are then added to the Google result-list as an optional, expandable list of recommendations.

In summary, HeyStaks is designed to help users to collaborate during Web search tasks and, importantly, it succeeds in integrating collaborative recommendation techniques with mainstream search engines. In the next section, we turn our attention to an examination of the usage of the system and consider, for example, the types of users that are active within the system, what kind of search staks are created and the nature and extent of the collaboration that exists in the search activities that are performed by users in these staks.

3. EVALUATING SEARCH COLLABORATION IN HEYSTAKS

So far we have introduced HeyStaks as a social search utility that introduces a collaboration layer on top of mainstream search, one in which users are able to organise and share their search experiences in order to help each other to search more efficiently. Here, we describe a recent evaluation based on the current HeyStaks Beta system with a view to answering some important questions about how users are actually using the service:

- Do HeyStaks users actually create search staks and, if so, how many on average?
- When users create staks, are they *private* or *public* staks? The answer to this question speaks to the openness (or otherwise) of HeyStaks users and their willingness to share their search experiences with others.
- Do users actively share the staks that they create? And do users tend to join staks created by others?
- Are HeyStaks' users benefiting from search collaboration? Is there evidence that users are selecting promoted results that come from their staks?
- Is there any evidence that some users are better searchers than others? Do some users tend to be more successful

when it comes to contributing valuable search knowledge to staks, search knowledge that others tend to benefit from during subsequent searches?

3.1 Modelling User Behaviour

For the purpose of this evaluation it is useful to consider HeyStaks users and their activities in the light of recent studies of user behaviour and online services, including social media [2, 10, 4, 5, 11]. The literature contains a number of useful frameworks, for example, that describe how the behaviour of online users changes as users become more or less engaged in a particular online or social media service. For example, Porter's *Funnel Model* distinguishes between four different types of user behaviour including, *interested*, *first-time use*, *regular use*, *passionate use*, and the rapid fall-off in participation that tends to occur at each stage; see [10] and [4] for a related model. Most recently, Preece and Shneiderman [11] have proposed their *Reader-to-Leader* framework that distinguishes between casual social media participation and more active engagement by proposing four classes of users:

1. *Readers* are users who consume the social media created by others, by and large without actively contributing themselves. For example, large numbers of users visit discussion boards, read blogs, and refer to Wikipedia without posting content or commenting themselves.

2. *Contributors* are users who do make a meaningful contribution to the evolving social media usually by rating or commenting on content that others have created. Contributors are defined by a tendency to follow the lead of other more active users. For example, in Wikipedia a contributor is someone who tends to edit existing pages rather than create new pages of content.

3. *Collaborators* are users who engage in more deliberate activities that serve to elicit some form of communal response, often in the form of an identifiable episode of collaboration between at least two users. These collaborations can be light-weight and short-lived (e.g. two users engaging in conversation via a thread of comments on a particular blog-post) or they can be longer-lasting engagements with a wider community of users (e.g. connecting with large numbers of people on social network websites like Last.fm and Facebook).

4. *Leaders* are the synthesizers of the social media space. They tend to be those users who contribute the most to a particular service, provide the most comments, the most ratings or the most blog posts, but they typically also go further by creating the conditions for new discussions to develop. The popular bloggers are leaders as they catalyze a community response to their views and opinions on a particular topic or theme, for example.

Of course these user categories do not define crisp, mutually exclusive subsets of users and, as discussed in [11], in many cases users will switch between being contributors and collaborators or between being readers and contributors, for example. Nevertheless this framework is a useful starting point to understand the different types of user engagement that can exist in a social service such as HeyStaks.

3.2 From Readers to Leaders in HeyStaks

In our evaluation we consider the usage data associated with 299 active HeyStaks users who have joined the Beta service during 2009. While detailed demographic data is unavailable for these users, the Beta invitations were largely circulated among our local researchers and their friends and so it is likely that many of these users are college students or recent graduates. In total, the data set covers 99,097 *activity records* covering all aspects of HeyStaks activity (stak creation, sharing, joining, search, result selection, result tagging and voting etc.).

Figure 1: The HeyStaks user-base.

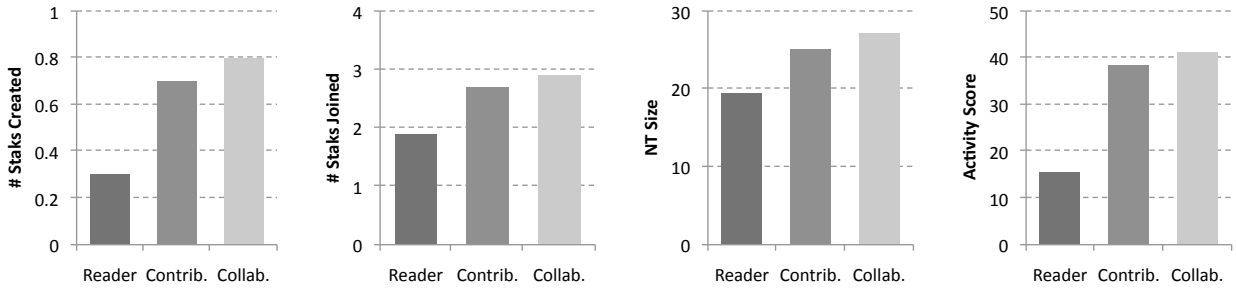
We apply the *Reader-to-Leader* framework as described above to analyse users and their activities in HeyStaks. Moreover, we introduce an additional superset class of users which we label *All Users*; this is necessary given the Beta nature of the HeyStaks deployment. A description of this user class and the other framework classes in the context of the HeyStaks domain is as follows:

- *All Users*: This is our group of 299 test users, each of whom have engaged in some minimal level of HeyStaks activity. Specifically, this is activity beyond creation of a *My Searches* stak and joining one other stak. Importantly, many of these users will not have engaged in any social activity, choosing not to share their staks or join staks created by others.

- *Readers*: An important criterion in HeyStaks is whether or not users actively engage in the sharing of search experience and in this context *readers* are defined to be those users that have displayed at least some form of sharing behaviour. Specifically we define an *active-shared stak* to be a stak that is shared with at least 2 users, containing at least 10 pages. Then a user is considered to be a *reader* if they are a member of at least one active-shared stak.

- *Contributors*: These are users who have added content to an active-shared stak. Typically, they will have selected or tagged or voted on a new result or page which then gets added to the stak and is available for future promotion/recommendation.

- *Collaborators*: In HeyStaks, the selection of a *recommended result* is the basic unit of search collaboration. Users who contribute new content to a stak, which eventually gets recommended to another stak member, and which is ultimately selected (or tagged or shared)



(a) Staks created. (b) Staks joined. (c) Network size. (d) Activity scores.

Figure 2: Average number of staks created and joined, network size and activity score across readers, collaborators and contributors.

by this other stak member, are considered to be *collaborators*.

- *Leaders*: In general, leaders are those users who are responsible for driving search collaboration within the HeyStaks application. Thus, leaders will be drawn from the set of collaborators and we examine to what extent each of these users contribute to the collaborations that take place in the system. We explore leadership in detail in Section 5 where, for example, we introduce a model of *user reputation* which we believe to be a useful indicator of search leadership.

In the next section, we analyse activity in HeyStaks by examining user behaviour as a function of class membership and the various types of staks that users have created and joined. We examine the level of activity that users from different classes are engaged in and consider the benefit accruing to users as a result of stak membership through the introduction of *stak reuse coefficients*, which we discuss in Section 4. Finally, in Section 5 we discuss leadership in the context of HeyStaks and propose a reputation model to identify search leaders in the user community.

4. EVALUATION AND ANALYSIS

The primary goal of this paper is to analyse the behaviour of those users who are the chief collaborators within HeyStaks - our *search leaders*. We begin by analysing the activity of our 299 early adopters of the HeyStaks system, at both the user and stak-level, and categorise these users according to the Reader-to-Leader model.

4.1 Analysis of User Behaviour

Figure 1 presents the breakdown of the 299 users in terms of the main reader-to-leader user categories. The various classes of users (readers, leaders etc.) are not disjoint, i.e. leaders are a subset of collaborators, collaborators are a subset of contributors etc. Currently about 70% of the 299 users are yet to engage in the social-side of HeyStaks search: they have not shared or joined staks and so have not yet enjoyed any type of collaboration benefit. In contrast, 90 users are classified as readers. In turn, about 40% of readers (37 users) are contributors: as a result of their search actions, new content has been added to shared staks as a precursor to collaboration. And about 70% of these contributors (24 users) are classified as collaborators; that is, they have played a role in the recommendation of content that was subsequently selected by some other user.

Users have partaken in a total of 99,097 activities across all staks. The vast majority of these actions, over 95%, are result selections, a figure reflecting the fact that selection is the most natural type of search activity. An important issue to consider is the extent to which users engage in the sort of activities that ultimately facilitate search collaboration. Do they create and join staks, for example? How many other users are they connected with (their *network size*)? All other things being equal, if a user creates and joins many staks then they are more likely to be connected with other users and will thus benefit from a larger collaboration network to act as a source of recommendations.

Figures 2(a) to 2(d) show the mean user values of key engagement metrics across *active-shared* staks, in terms of staks created and shared, network size and activity levels for the main user groups (readers, contributors and collaborators). Clearly as users transition from reader to contributor to collaborator we see a consistent increase in engagement level across all indicators. For example, collaborators on average create (0.8) and join (2.9) more staks than either contributors (0.7 and 2.7 respectively) and readers (0.3 and 1.9 respectively). Collaborators tend to have the largest number of users with which they can share search results: for example, network size is large across this user group, 27.1 on average, compared to an average network size of 19.4 for readers. Increasing engagement across user class is reflected in Figure 2(d), showing mean total number of selects, tags, votes and shares across each user category, with collaborators registering the highest average score.

4.2 Analysis of Active Staks

We now turn our attention to analysing stak membership across the various classes of users. In HeyStaks, there are two types of stak, i.e. either private or public, and each of these stak types can be shared or unshared. We look at various activity metrics within these different stak types to gain greater insight into user behaviour.

Figure 3 shows mean scores with respect to number of members, distinct results and activities within private and public instances of unshared and shared staks. Users tend to input a greater amount of search results into private rather than public staks. This may seem to indicate that people have a greater wish to retain their own results rather than share with others. However, it is the *private-shared* staks

which show the greatest amount of activity, with more distinct pages (71.2) and activities (85.6) on average than the other stak types. This, coupled with the fact that public-shared staks are the more active public staks, indicates the desire of users' to share their search results with members of the HeyStaks community.

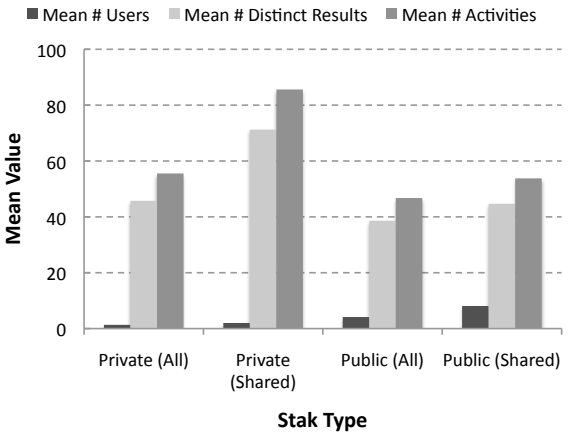


Figure 3: Summary statistics for active staks.

The above analysis shows the levels of activity within the various stak types across all user classes; we now examine the type of staks that the different classes of users tend to create and join. Results are presented in Figure 4 for readers, contributors and collaborators. It is clear from the figure that all users are members of more public staks than private staks. Moreover, of the 3.3 public staks of which contributors and collaborators are members, some 2.9 (87%) of these on average are shared. In contrast, of the 1.3 private staks that these users belong to, less than 20% are shared. Thus, these findings show that the stak type most favoured by all classes of users is the shared public stak, indicating that users have a strong preference to belong to staks where they can benefit from the search activity of other users.

4.3 Stak Reuse Coefficients

In the previous section, various statistics relating to stak activity were discussed. A key objective of HeyStaks is that users who have created or joined staks benefit from the previous search activity that they themselves and/or other members have performed in the context of these staks. Thus, an important measure of stak utility is to consider the number of times that stak members have selected promoted results relative to the number of organic result selections made in the stak.

Accordingly, we define the *reuse coefficient* for each stak as follows. Let n_p and n_o be the numbers of promoted and organic result selections made in stak S_i, respectively; thus the reuse coefficient, C_{S_i}, for the stak is given by:

$$C_{S_i} = \frac{n_p}{n_o} \quad (3)$$

With this approach, the effectiveness of staks in assisting users to locate search results can be readily assessed. For example, a reuse coefficient of 0 indicates that no promoted

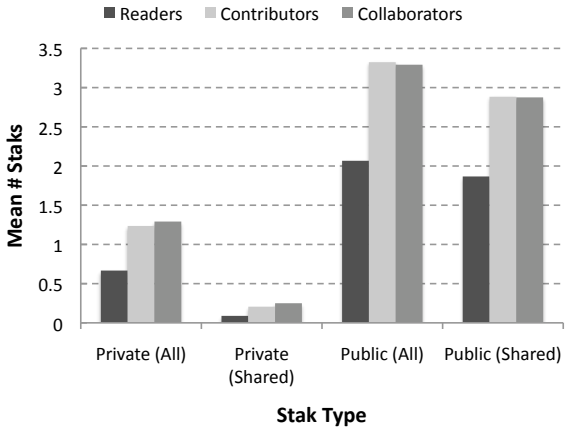


Figure 4: Mean number of staks that readers, contributors and collaborators are members of versus stak type.

results were selected in a stak, while a coefficient of 1 indicates that as many promoted results were selected as organic results. In general, higher reuse coefficients indicate that users benefit to a greater degree from the previous searches performed by themselves or by other stak members.

Figure 5 shows the mean stak reuse coefficients versus stak type for readers, contributors and collaborators. It is clear from the results that reuse coefficients are significantly greater for public staks. In the case of collaborators, for example, a mean reuse coefficient of 0.88 applies for public staks compared to only 0.07 for private staks. Similar differences are seen for the other user classes. Thus, the benefit to users of public stak membership is more than an order of magnitude greater than private stak membership, with users being 10 times more likely to benefit from useful result promotions in public staks.

Further, while contributors gain greater benefits in terms of reuse than readers, there is an additional benefit seen for collaborators. For example, in public staks, the reuse coefficients are 0.88, 0.65 and 0.25 for collaborators, contributors and readers respectively. These results clearly indicate the advantages of increased engagement with the system, with collaborators selecting on average over 1.3 times the number of promoted results than contributors and more than 3 times the number of promoted results than readers.

5. TOWARDS SEARCH LEADERS

Thus far, we have examined various summary statistics and reuse coefficients for the reader, contributor and collaborator classes of users, where each class is defined by its level of engagement with the system. We now turn our attention to the search *leaders* in the community. There are many ways in which leaders could be identified and defined; for example, those users who have created and joined the most staks or those who have added the most results to staks. In general, however, a high degree of activity does not guarantee that users play a *productive* role in the system. For example, a particular user may add many results to staks but few of these may be selected when promoted to other users, implying that these results are not considered to be

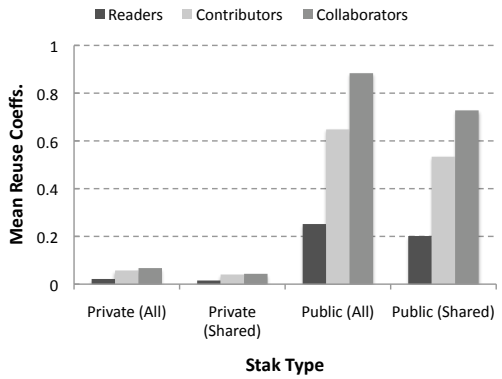


Figure 5: Mean stak reuse coefficients versus stak type for readers, contributors and collaborators.

Input: Set $\mathcal{A}$ of user activity tuples $\langle u, p, t, S, type\rangle$, set S of all staks, array $\mathcal{R}$ of user reputation scores
Output: Updated array $\mathcal{R}$ of user reputation scores

```
1.  UserReputation(A, S, R)
2.  begin
3.      foreach activity a ∈ A
4.          if a.type = promotion
5.              u_c ← a.u
6.              p_c ← a.p
7.              t_c ← a.t
8.              S_c ← staks(S, u_c)
9.              A ← {a' ∈ A : a'.p = p_c & a'.S ∈ S_c & a'.t < t_c}
10.             U ← {a'.u : a' ∈ A} − {u_c}
11.
12.             foreach u ∈ U
13.                 R[u] ← R[u] + 1/|U|
14.             end
15.         end
16.     end
17. end
```

Figure 6: User reputation algorithm.

useful or relevant by other users of the system. Thus, in the next section, we explore leadership in the context of *user reputation*, in which those users that contribute the most from a search collaboration perspective can be identified.

5.1 From Collaboration to Reputation

In relation to the Reader-to-Leader framework, the leaders in the community will be drawn from the collaborator class of users, given that these users have the highest level of engagement with the system. Crucially, collaborators can be said to be productive users in the sense that they are responsible for *collaboration events*, i.e. they have added results to staks which have been promoted to and selected by other users of the system.

Our proposed user reputation algorithm [6] is given in Figure 6. For simplicity, the algorithm shown is one suitable for offline execution, but note that the algorithm can be readily modified such that user reputation scores are updated in real time when new activities are performed by users.

The algorithm takes as input a temporally ordered set of user activities $\mathcal{A}$ which are retrieved from the HeyStaks database. Each entry $a \in \mathcal{A}$ is a tuple $\langle u, p, t, S, type\rangle$, where $a.u$ is the user who performed the activity, $a.p$ is the associated result page, $a.t$ is the time when the activity occurred, $a.S$ is the active stak at the time of the activity and $a.type$ indicates whether or not the activity relates to a HeyStaks promotion. In addition, the set of all staks $\mathcal{S}$ and the current (previously calculated) set of user reputation scores $\mathcal{R}$ are provided as a starting point.

Briefly, the algorithm operates as follows. For each promotion activity $a \in \mathcal{A}$ (line 3), the set of staks S_c that the current user u_c is a member of is retrieved (line 8). Then, the set of prior activities relating to the current page p_c, in any of the staks in S_c, is determined (line 9) and the users who performed these activities are identified (line 10). Finally, a unit of reputation is distributed equally among these users and added to their existing reputation score (lines 12–14). This process continues until all activities are processed and the array $\mathcal{R}$, which contains each user's updated reputation scores, is returned. Thus, early producers of results, i.e. those who are among the first to add results to staks, benefit the most in terms of reputation when these results are subsequently selected by other users.

5.2 Results

The results of applying the reputation model to the 24 collaborators are shown in Figure 7. The trend is long-tailed, with 6 users achieving a reputation score of greater than 5, and with two users achieving a score of more than 10. In general, this is the kind of trend that is to be expected from a user reputation perspective, where a small subset of users contributes the most in terms of driving search collaboration, and the remaining users contributing some, but significantly less, search knowledge to the community.

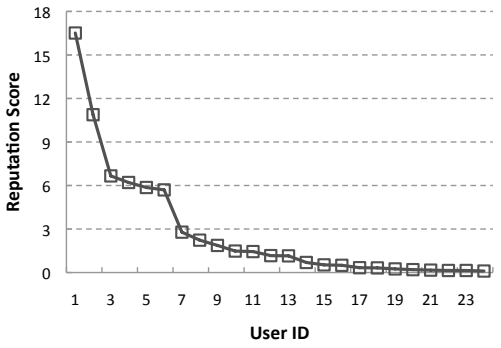


Figure 7: Reputation scores for 24 collaborators.

Given the trend observed in Figure 7, we can reasonably define search leaders as those users at the head of the reputation curve, with the knee-point acting as the cutoff point. Thus, given the set of HeyStaks users and corresponding activities analysed in this paper, we can identify 6 search leaders from the set of 24 collaborators. Note that, with this approach, leadership is a function of usage data and those users that are identified as search leaders at particular points in time can change as other (or new) users increase their level of activity within the system.

There are a number of improvements that could be made to the reputation model as described here. The current model does not reward those users who add results to staks

that are subsequently promoted to and selected by many distinct users in the system. Further, additional reputation could be awarded when promoted results are selected from public shared staks, given that private shared staks are typically limited in membership to small 'cliques' of users. In future work, as the deployment of HeyStaks continues to expand and further usage data becomes available, we will explore the benefits of developing and applying more advanced user reputation models to identify search leaders within the system, and to deal with any attempts to game the system in order to manipulate reputation scores.

6. CONCLUSIONS

Although there is much evidence that many search tasks are inherently collaborative, mainstream search engines do not explicitly support collaboration during search. The main contribution of this paper is to analyse the activities of users of HeyStaks, a novel social search utility. In this regard, we have considered users in the context of the *Reader-to-Leader* social activity framework as proposed in [11]. We have examined the various stak types that users tend to create (e.g. public versus private staks), the degree to which staks are shared between users, and the benefits of stak membership through the introduction of stak reuse coefficients.

Our findings indicate that, for all classes of users, membership of shared public staks is by far the most popular choice, in which users are able to benefit from the previous search activity of other users. In addition, the results show that the greatest benefit in terms of promoted result selections applies to collaborators and leaders, the most engaged of the classes of users examined. These findings are promising from the point of view that the system rewards users with more benefit (i.e. useful result recommendations) as they increase their level of activity within the system.

In addition, we have described a user reputation model that is designed to identify the search leaders in the system; those users who contribute the most from a result collaboration perspective. While our model provides a more sophisticated approach to detecting leaders over more simple collaboration count approaches, we note that HeyStaks is in Beta deployment and hence our analysis is therefore performed on relatively small amounts of user activity data. In future, as HeyStaks gains traction in the wider community, further opportunity will exist for the analysis of user activity, stak creation, result sharing, and the development of more advanced reputation models to more fully understand the nature and benefits to users of the collaborative Web search approach as adopted in the HeyStaks application.

7. ACKNOWLEDGMENTS

This work is supported by Science Foundation Ireland under grant 07/CE/I1147.

8. REFERENCES

[1] S. Amershi and M. R. Morris. Cosearch: a system for co-located collaborative web search. In *CHI*, pages 1647–1656, 2008.

[2] B. M. Evans and E. H. Chi. An elaborated model of social search. *Information Processing and Management*, 2009.

[3] D. Horowitz and S. Kamvar. The Anatomy of a Large-Scale Social Search Engine. In *WWW*, 2010.

[4] A. J. Kim. *Community Building on the Web: Secret Strategies for Successful Online Communities*. Peachpit Press, 2000.

[5] C. Li and J. Bernoff. *Groundswell: Winning in a World Transformed by Social Technologies*. Harvard Business Review, 2008.

[6] K. McNally, M. P. O'Mahony, B. Smyth, M. Coyle, and P. Briggs. Towards a reputation-based model of social web search. In *IUI*, pages 179–188, 2010.

[7] M. R. Morris. A survey of collaborative web search practices. In *CHI*, pages 1657–1660, 2008.

[8] M. R. Morris and E. Horvitz. S^3: Storable, shareable search. In *INTERACT (1)*, pages 120–123, 2007.

[9] M. R. Morris and E. Horvitz. Searchtogether: an interface for collaborative web search. In *UIST*, pages 3–12, 2007.

[10] J. Porter. *Designing for the Social Web*. New Riders, 2008.

[11] J. Preece and B. Shneiderman. The reader to leader framework: Motivating technology-mediated social participation. *AIS Trans. on Human-Computer Interaction*, 1(1):13–32, 2009.

[12] M. C. Reddy and P. Dourish. A finger on the pulse: temporal rhythms and information seeking in medical work. In *CSCW*, pages 344–353, 2002.

[13] M. C. Reddy, P. Dourish, and W. Pratt. Coordinating heterogeneous work: Information and representation in medical care. In *ECSCW*, pages 239–258, 2001.

[14] M. C. Reddy and B. J. Jansen. A model for understanding collaborative information behavior in context: A study of two healthcare teams. *Inf. Process. Manage.*, 44(1):256–273, 2008.

[15] M. C. Reddy and P. R. Spence. Collaborative information seeking: A field study of a multidisciplinary patient care team. *Inf. Process. Manage.*, 44(1):242–255, 2008.

[16] A. F. Smeaton, C. Foley, D. Byrne, and G. J. F. Jones. ibingo mobile collaborative search. In *CIVR*, pages 547–548, 2008.

[17] A. F. Smeaton, H. Lee, C. Foley, and S. McGivney. Collaborative video searching on a tabletop. *Multimedia Syst.*, 12(4-5):375–391, 2007.

[18] B. Smyth. A community-based approach to personalizing web search. *IEEE Computer*, 40(8):42–50, 2007.

[19] B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, and O. Boydell. Exploiting query repetition and regularity in an adaptive community-based web search engine. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, 14(5):383–423, 2004.

[20] B. Smyth, P. Briggs, M. Coyle, and M. P. O'Mahony. A case-based perspective on social web search. In *ICCBR*, pages 494–508, 2009.

[21] B. Smyth, P. Briggs, M. Coyle, and M. P. O'Mahony. Google? shared! a case-study in social search. In *UMAP*, number 283-294. Springer-Verlag, June 2009.

Niche Trend Search for Recommender System based on Knowledgeable Blogger Group

Takumi Shihoya
University of Hyogo
Japan
nd10d015@stshse.u-hyogo.ac.jp

Shinsuke Nakajima
Kyoto sangyo University
Japan
nakajima@cse.kyoto-su.ac.jp

Kazutoshi Sumiya
University of Hyogo
Japan
sumiya@shse.u-hyogo.ac.jp

Yoichi Inagaki
Kizasi Company,Inc
Japan
inagaki@kizasi.jp

ABSTRACT

We propose a niche trend search method for recommender systems based on knowledgeable blogger ranking. User preferences change since they are greatly influenced by trends and fashion. Therefore, it is important to detect and provide trend information focused on individual user preferences and to recommend items related to the detected trends. We have analyzed blog content in order to detect niche trend information.We describe here the value of identifying niche trends and explain how to perform niche trend searches based on knowledgeable bloggers.

Keywords

Niche Trend, Knowledgeable Blogger Ranking, Recommender system

1. INTRODUCTION

Recently, the amount and diversity of content on the Web have been increasing very rapidly. In this context, recommender systems play an important role in providing users with suitable content from enormous content sets. However, most conventional recommender systems only perform simple recommendations based on user preferences or recommendations obtained by collaborative filtering based on similar user experiences such as buying histories. However, user preferences are not constant since they are greatly influenced by changing trends and fashions in the world. Therefore, it is important to detect and provide trend information that is focused on each user 's preferences and to recommend items related to the detected trends. In particular, in these days of the long tail phenomenon, we believe that detecting niche trends carries weight, although it is not easy to do. Thus, we propose a niche trend search method for recommender systems based on knowledgeable blogger ranking.

We have already proposed the knowledgeable blogger ranking system [1]. The system calculates knowledge scores for bloggers and ranks blog entries based on the bloggers' knowledge level. The knowledge level of bloggers is evaluated based on their use of domain-specific words in their past blog entries. A blogger is given multiple scores with respect to various topic areas. In this method, blog entries written by knowledgeable bloggers receive higher rankings than those written by general bloggers. Thus, the knowledgeable bloggers for a specific topic can be regarded as specialists in that topic. We then try to detect niche trends by analyzing blog entries posted by highly knowledgeable bloggers.

The most important problem to be solved in this study is how to detect niche trend information. We attempt to analyze blog content to detect such trends. Blog content refers to various kinds of publicly available media content that is produced by end-users. This content is increasing exponentially on the Web. Furthermore, blog content conveys the direct opinions of the author and is free from media bias; that is, it is different from mass media content such as that carried on TV, radio, and in newspapers. Therefore, blogs have become very important for reputation analysis in business area. Accordingly, it is reasonable to detect user trends from blog content.

1.1 Why niche trends?

Our goal in this study was to search for trends in users 'topics of interest and to search for blog articles that described that topic and that would be suitable for a recommender system. In this study, we distinguished between "major trends" and "niche trends." "Major trends" are known to more people than "niche trends," whereas "niche trends" are known to smaller groups who have focused on just a few topics. Therefore, a "niche trend" has a higher rarity value than a "major trend."

For example, if someone is interested in dieting, in the case where a system recommends to this user some trend information about dieting, a major trend would not have a high rarity value because it is possible that the person already knows about it. On the other hand, a niche trend about di-

eting that is known only to specialists (knowledgeable bloggers) would have a high rarity value for the user. Thus, it is very useful to detect niche trends before many people discover them.

1.2 Definition of "Trend"

In this study, "trend" is defined as something that is considered to be a hot, or popular, topic in a certain field and that exists for a limited time only.
For example, in Japan, "natto" attracted a lot of attention as an effective diet food in 2006, and "bananas" also attracted attention as an effective diet food in 2007. In this case, both "natto" and "banana" would be trend keywords in the field of dieting.
In our research, we adopted an increase in the rate of each keyword appearance in blog articles in order to detect trend keywords.

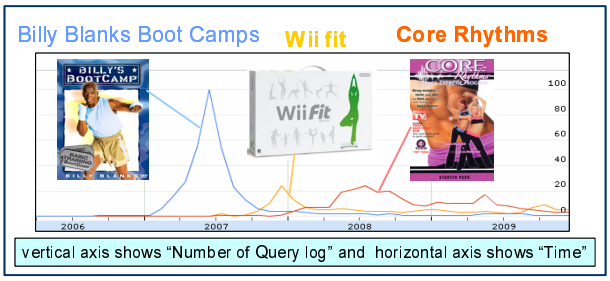


Figure 1: Example of changing public trends.

Figure 1. shows an example of trends that change according to users' interest in the topic concerning dieting and weight loss. This is from Google insight [8], and it shows the fluctuations in the number of queries to the Google search engine. Our system measures the number of each keyword appearance in blog articles in each time period. Figure 1 indicates a number of keyword appearances for "Billy Blanks Boot Camps," "Wii fit," and "Core Rhythms." In this way, we can understand how popular these keywords were and when these trends were hot.

If we analyze entire blog articles in a database, we can only detect major trends. However, we concentrated on detecting niche (deep) trends by analyzing focused blog articles.

1.3 Related work

Much research has been done to develop methods to extract trend information [2], [3], [4], [5]. Ishikawa et al.[2] proposed a method to visualize trends extracted from time series documents such as news articles. Their method makes it possible to extract major topics in specific time periods and to visualize the trend pattern over a long term. However, one problem is that their system extracts not only important trends but also trends having less influence on a user ' s decision. Toda et al. [3] proposed a method of extracting topics from blog articles and then developed a system that can detect topic-transitions in each topic category. Although these systems succeed in extracting trend information, they do not detect niche trend information. On the contrary, our proposed method can extract niche (focused) trends that have rarity values.

Various research has been done on developing techniques to extract topics from consumer generated media (CGM). For example, [6] and [7]. Sekiguchi et al.[6] proposed a method of discovering common topics for people who share similar interests. By extracting topic keywords, then can find, i.e. discover, certain communities that have the same interests. However, these studies do not take knowledgeable blogger groups into consideration.

2. KNOWLEDGEABLE BLOGGER RANKING FOR SPECIFIC TOPIC

2.1 About Knowledgeable Blogger

We previously proposed a knowledgeable blogger ranking method based on bloggers' knowledge level [1]. Our method assumes that a person who has extensive knowledge about a certain topic is more credible than a person with less knowledge or with a generic level of knowledge. Figure 2 depicts the relationship between a knowledgeable person and credible information. By analyzing bloggers' past entries and ranking bloggers based on their knowledge level, we can provide more credible blog entries to the end user.

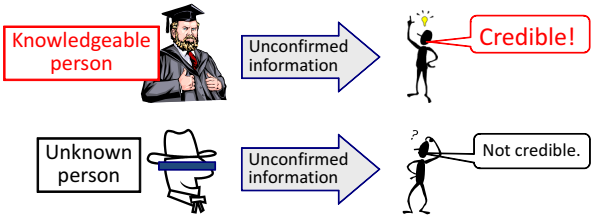


Figure 2: Relationship between knowledgeable people and credible information.

The system first extracts topic areas and creates a term dictionary that provides domain-specific words for each of the topic areas. The system then evaluates a blogger's knowledge level based on the blogger's usage frequency of these domain-specific words. Generally, a blogger's knowledge level varies with respect to different topic areas. A person may be an expert in one field, while they may know little to nothing about another field.

2.2 Identifying KBs for each KBG

Generally, bloggers have specific interests, and they post blog entries related to specific topic areas. We call a blogger who is familiar with a topic area a "Knowledgeable Blogger (KB)" for this topic area. A set of knowledgeable bloggers for a topic area is called a "Knowledgeable Bloggers' Group (KBG)" for that topic area. We first extract some keywords representing the topic areas discussed daily in blogs. Each keyword becomes the title of the topic area and also represents the name of the KBG familiar with the topic area. We then extract frequently used words for each topic area, and create a dictionary, which summarizes the topic areas and their domain-specific words.

For the purpose of finding a relevant group of bloggers for each topic area, we next assign bloggers to the relevant topic areas. It is possible for a blogger to be assigned to more than one KBG. We regard a blogger who continues writing blog

entries related to a topic area as a KB of this KBG. We list several conditions in order to determine whether a blogger is assigned to a KBG.

2.3 Calculating KBs' knowledge scores for the KBG

The above section describes how KBs are selected for each topic. Then, we calculate the KBs' knowledge scores, which indicate how knowledgeable a KB is about a particular topic.

Basically, scores are calculated based on how often as well as how in-depth a blogger writes blog entries related to a certain topic. If a blogger uses the domain-specific words of a topic extensively, high knowledge scores are attached to that blogger.

We first calculate $score_g(e)$, the score of an individual entry, e, with respect to the topic, g, as follows:

$$score_g(e) = \sum_{j=1}^{n} \alpha_j \cdot \beta_j \cdot \gamma_j \quad (1)$$

where $n = 400$ is the number of domain-specific words, $\alpha_j = \frac{n-j}{n}$ is the weight of word j, which decreases as j increases, β_j is the co-occurrence frequency of word j, and γ_j is a binary value that indicates whether entry e contains word j or not.

Once we have the individual entries' scores, we next calculate $score_g(b)$, the score of a blogger, b, with respect to the topic, g:

$$score_g(b) = \frac{l}{n} \cdot \frac{log(m)}{m} \cdot \sum_{i=1}^{m} score_g(e_i) \quad (2)$$

where e_i is an entry written by blogger b, m is the number of entries that blogger b has posted within a given period, $n = 400$ is the number of domain-specific words, and l is the number of domain-specific words that occurred in all of the entries written by blogger b. Here, $\frac{l}{n}$ indicates the coverage ratio of the domain-specific words that blogger b used. Additionally, $\frac{log(m)}{m}$ reduces the effect of a blogger who frequently posts a large number of entries, but most of which are unrelated. Thus, bloggers who write few entries but who obtain high usage of the domain-specific words in each entry, have higher knowledge scores.

3. NICHE TREND SEARCH BASED ON KNOWLEDGEABLE BLOGGERS

We propose a niche trend search method based on knowledgeable blogger ranking for specific topics. Our proposed method is organized in three steps.

Step 1. Finding a KBG related to user preference

Step 2. Detecting trend words by analyzing blog content of the KBG

Step 3. Determining whether trend words are part of a major trend or a niche trend

3.1 Finding KBG related to user preference

With the proposed method, we extract user preferences using her/his Web browsing histories. First, the system investigates the term frequency of browsed Web pages and extracts a vector space model based on term frequency-inverse

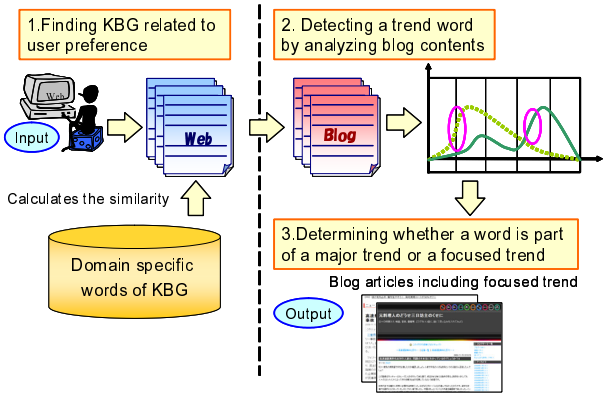


Figure 3: Basic concept of trend search.

document frequency (TF-IDF). Then, it calculates the similarity between the vector model and the vector of domain-specific words of each KBG. KBGs that have a high similarity score can represent the area of a user's topic of interest.

3.2 Detecting a trend word by analyzing blog contents of the KBG

In this section, we explain how to detect trend words. The system tries to detect trend words by analyzing increasing rates of term frequency of trend word candidates. First, the system performs a morphological analysis for blog articles published from a KBG related to user interest, and extracts nouns. Then, the number of word appearances are measured for every one-week period. The system measures the rate of increase of the number of appearances. If the rate of increase is over the threshold, then the word is regarded as a trend word. The system repeats the same procedure for all words appearing in the target blog articles.

$$W_{ave.} = \frac{1}{n} \sum_{i=1}^{n} W_{Pi} \quad (3)$$

$$I_i = \frac{W_{Pi} - W_{Pi-1}}{W_{ave.}} \quad (4)$$

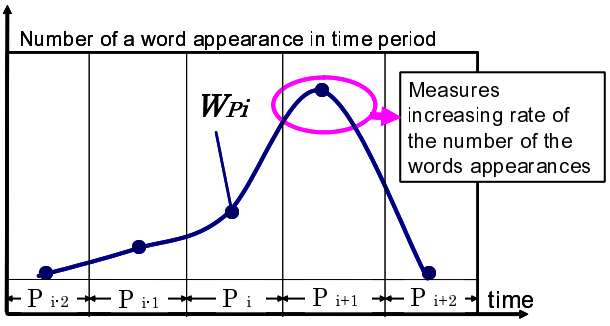


Figure 4: How a trend word is extracted.

Here, $W_{ave.}$ corresponds to an average number of word appearances for one time period, P_i corresponds to the i th

time period, W_{Pi} corresponds to the number of word appearances in time period P_i, and I_i corresponds to the rate of increase of the number of word appearances.

When I_i becomes larger than a threshold, the target word is regarded as a trend word.

3.3 Determining whether a word is part of a major trend or a niche trend

In this section, we describe how to determine whether a word is part of a major trend or a niche trend. Major trend words may appear in blog articles belonging to diverse KBGs. Niche trend words may appear in blog articles belonging to only one KBG (or a small number of KBGs.) We apply the TF-IDF method in order to represent the niche degree of trend words. We use trend word frequency, referred to as TF, and inverse KBG frequency, referred to as IGF. Thus, we use TF-IGF as shown below:

$$iGF = log\left(\frac{N}{gf}\right) + 1 \quad (5)$$

$$FD_t = tf \cdot iGF \quad (6)$$

Here, iGF corresponds to inverse KBG frequency, N corresponds to the number of all blog articles in the target time period, and gf corresponds to the frequency of KBG blog articles that contain the target trend word.

The value FD_t corresponds to the focused degree of target trend word t, and tf corresponds to the target trend word frequency in target KBGs related to user preference.

Therefore, we believe that a trend word with high FD_t brings high rarity value to the target user. Our proposed system can search for trend words with a high rarity value by using user preference or searching for keywords (topics) that the user is interested in.

Figure 5 illustrates the case of a user who wants to obtain trend information related to dieting. The diet based on eating bananas is well known to most Japanese people as an effective method for losing weight. In contrast, the diet based on eating molokhia occurred at around the same time the banana diet became popular, but it was only familiar to people who were especially focused on dieting. Thus, it has rarity value for people who are going on a diet. In this way, the system can detect niche trend words by focusing on target content in blog articles published by knowledgeable blogger groups.

4. CONCLUSION

In this paper, we proposed a niche trend search method for a recommender system based on knowledgeable blogger ranking. Through the results of this study, we clarified the following:
(1) The value of detecting niche trends.
(2) How to detect trend words by analyzing blog content of KBGs.
(3) How to determine whether a word is part of a major trend or a niche trend.

Our next step is to implement our proposed method and evaluate it based on a prototype system.

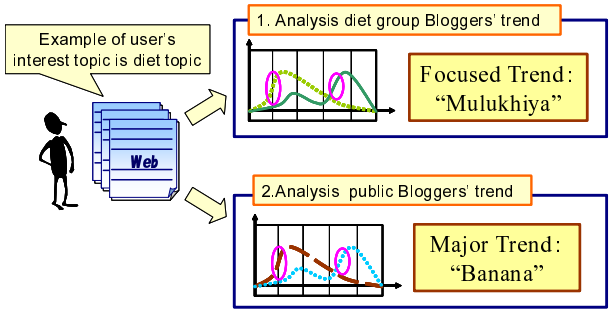


Figure 5: Example of trend search.

Acknowledgments

This research is being supported by the National Institute of Information and Communications Technology Japan and MEXT (Grant-in-Aid for Young Scientists (B) #20700089).

5. REFERENCES

[1] Shinsuke NAKAJIMA, Jianwei Zhang, Yoichi INAGAKI, Tomoaki KUSANO and Reyn Nakamoto. Blog Ranking Based on Blogger's Knowledge Level for Providing Credible Information, Proc. of the 10th International Conference on Web Information Systems Engineering. WISE2009, pp.227-234:

[2] Yoshiharu Ishikawa and Mikine Hasegawa. T-Scroll: Visualizing Trends in a Time-Series of Documents for Interactive User Exploration Research and Advanced Technology for Digital Libraries P235-P246:

[3] Tomoko TODA, Naoki FUKUTA and Hiroshi ISHIKAWA. Extraction of topic transition pattern of each category based on clustering Blog article DEWS 2007 A8-3:

[4] Takashi MENJO and Masatoshi YOSHIKAWA. Trend Prediction in Social Bookmark Service Using Time Series of Bookmarks DEWS2008 B9-5:

[5] Keisuke DAIRIKI, Ikki OHMUKAI and Hideaki TAKEDA. Information Recommendation based on Innovators ' Activity for Social Bookmarking Service The 22nd Annual Conference of the Japanese Society for Artificial Intelligence, 2008 2D3-3:

[6] Yuichiro SEKIGUCHI, Harumi KAWASHIMA, Hidenori OKUDA, Masahiro OKU. Topic Detection from Blog Documents Using Users' Interests Proceedings of the 7th International Conference on Mobile Data Management (MDM'06) 0-7695-2526-1/06:

[7] Makoto TAKAGI, Yasuma MORI, Keiichi TAMURA, Susumu Kuroki, and Hajime KITAKAMI. Method for Extracting Frequent Communities from Blog User Spaces IPSJ SIC Technical Report 2007-MPS-65 2007/6/25:

[8] Google Insights for Search. http://www.google.com/insights/search/

Resource Recommendation for Social Tagging: A Multi-Channel Hybrid Approach

Jonathan Gemmell, Thomas Schimoler, Bamshad Mobasher, Robin Burke
Center for Web Intelligence
School of Computing, DePaul University
Chicago, Illinois, USA
{jgemmell, tschimo1, mobasher, rburke}@cdm.depaul.edu

ABSTRACT

Social tagging systems allow users to annotate online resources with arbitrary labels producing rich information spaces. Given the complexity and size of these information spaces, recommender systems are essential in helping users discover new resources, tags or even other users. Building such recommenders has proved a challenge, because the data is noisy, large, and multi-dimensional. There has been active interest in sophisticated models that leverage all three dimensions of a social tagging system (user, resource, tag). In particular, hybrid algorithms that blend recommendation components drawing separately on complementary dimensions have demonstrated excellent results in tag recommendation. We extend these results to the problem of resource recommendation: predicting resources of interest to a user, given only the user's tagging history. We also examine characteristics of the data that may predict the performance of the different components. We call these characteristics *information channels* and offer metrics for their evaluation. Our evaluation on three large real world datasets demonstrate that hybrid recommenders surpass the effectiveness of their constituent components while maintaining their simplicity and efficiency.

Categories and Subject Descriptors

H.2 [**Database Management**]: H.2.8 Database application—*Data mining*; H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval—*Search process*

General Terms

Experimentation, Performance

Keywords

Social Tagging, Information Channels, Hybrid Recommenders

1. INTRODUCTION

Social tagging systems such as Delicious[1] and LastFM[2] allow users to organize and share resources such as bookmarks or songs. Domain-specific applications such as Citeulike[3] and BibSonomy[4] help researchers to organize scholarly publications. Social tagging is quickly becoming ubiquitous in a variety of domains. For example, Amazon[5], YouTube[6] and others now include tagging along with their other services.

Tagging systems are popular in part because they allow users to annotate online resources with any tag they wish, free from any preconceived conceptual hierarchy. The accumulation of many users' opinions about what resources are important and the tags used to describe them produces a vast information space that users can explore. Due to the size and complexity of social tagging systems, recommender systems are a critical tool in assisting users to discover resources, tags or even other users. Unfortunately, the noise and dimensionality of the data make recommendation difficult.

Recent efforts have focused on integrative techniques that combine all three aspects of the data (users, resources, tags) into a single model. For example, in tag recommendation, graph-based models [6] and tensor factorization [10, 11, 16] have been employed. These approaches are computationally intensive and are not readily extensible to resource recommendation. Our own work in tag recommendation has focused on hybrid models [3, 4] that combine several components, each exploiting different aspects of the data.

In this paper, we extend our approach to the problem of resource recommendation: the personalized suggestion of resources for a given user. In contrast to other efforts on recommending resources in social tagging applications, we do not assume any context information that might be provided by a tag or another resource. We are interested in generating a personalized resource list like those typically associated with e-commerce recommendations.

We propose a linear weighted hybrid recommendation framework that combines multiple complementary components into a single integrated model. The relative contributions of the components are learned through random-restart hill climbing. The hybrid performs better than the individual components alone and it maintains the simplicity and computational efficiency of its components.

To help understand our experimental results, we explore the notion of an *information channel*: the power one dimension possesses in predicting or modeling another dimension of the social tagging data. To quantify the strength of these information channels, we develop a family of metrics based on conditional entropy. These metrics reveal marked differences in the characteristics of the datasets, which are reflected in the performance of the recommendation components.

[1] delicious.com
[2] www.last.fm
[3] www.citeulike.org
[4] www.BibSonomy.org
[5] www.amazon.com
[6] www.youtube.com

The rest of the paper is organized as follows. In Section 2 we present related work on recommendation techniques in social tagging applications. We present our linear weighted hybrid scheme in Section 3. Section 4 introduces information channels. Our experimental results and evaluation are offered in Section 5. Finally, we conclude the paper with a discussion of our results.

2. RELATED WORK

One of the first techniques to demonstrate the value of an integrative approach for recommendation in social tagging systems was a graph-based variant [6] of the well-known PageRank algorithm. This approach produces excellent results in tag recommendation. Its computational requirements, on the other hand, make it ill-suited for large-scale deployment. It works best when it can triangulate elements from one dimension (i.e. tags) given elements from the two other dimensions (i.e. a user and a resource). For resource recommendation, however, the input consists solely of a user. This method is therefore unable to effectively exploit its model and produces inferior results.

Tensor factorization is another integrative solution for making recommendations in tagging applications. Tucker decomposition is one such example that factors the three dimensional tagging data into three feature spaces and a core residual tensor [16]. For the problem of predicting a resource given a user and a tag, this method has been shown to be effective. However, it has not been shown to be effective when the input is a user alone. Unlike the graph-based model, online computation of recommendations is highly efficient. However, the offline computation required to build the model is not scalable to the demands of real-world applications.

A pair-wise interaction tensor factorization model has also been proposed. It offers far more reasonable run times in both the construction of the model and the generation of recommendations [10, 11]. It has been used to optimize the ranking of tags given the known user-resource pairs in the data. Tags may then be recommended for a new user-resource pair. As with Tucker decomposition, it is not clear that this technique can be applied to make predictions given a single element of the data such as a user.

Using data from MovieLens, one of the few systems that contains both ratings and tags, hybrid recommenders were used to predict ratings [14]. Efforts in linear weighted hybrid recommenders have explored tag recommendation as well [3, 4]. One approach demonstrated that the graph-based model may be improved by incorporating item-based collaborative filtering. Another effort designed a hybrid for BibSonomy for the PKDD-ECML 2009 challenge. In this paper we extend those efforts, proposing a framework for constructing linear weighted hybrid resource recommenders.

3. RESOURCE RECOMMENDATION

We define resource recommendation to be the problem of making personalized suggestions for a user, based only on the user's tagging history. As in the an e-commerce recommender, we do not recommend resources already tagged by the user.

3.1 Data Model

The foundation of a social annotation system is the annotation: the record of a user labeling a resource with one or more tags. A collection of annotations results in a complex network of interrelated users, resources and tags [8]. A social annotation system can be described as a four-tuple: $F = \langle U, R, T, A \rangle$, where, U is a set of users; R is a set of resources; T is a set of tags; and A is a set of annotations. An annotation contains a user, resource and all tags the user applied to the resource.

A social annotation system can be viewed as a three dimensional matrix, *URT*, in which an entry *URT(u,r,t)* is 1 if u tagged r with t. Aggregate projections of the data can be constructed, reducing the dimensionality but sacrificing information [9]. For example, the relation between resources and tags can be defined as $RT(r, t)$, the number of users that have applied t to r. This notion strongly resembles the "bag-of-words" vector space model [12].

A similar two-dimensional projection can be constructed for UT, in which an entry contains the number of times a user has applied a tag to any resource. Finally, UR is a binary matrix indicating whether or not a user has annotated a resource. An alternative approach would be to define an entry in the matrix as the number of tags a user has applied to a resource. Our previous work and continued experimentation has shown that the binary model for UR produces better results.

Drawing on the RT projection, each resource, r, may be modeled as a vector over the set of tags, where each weight, $w(t_i)$, in each dimension corresponds to the importance of a particular tag, t_i.

$$\vec{r^t} = \langle w(t_1), w(t_2)...w(t_{|T|}) \rangle \tag{1}$$

Similarly, a resource can be modeled as a vector over the set of users where each weight, $w(u_i)$, corresponds to the importance of a particular user, u_i to produce $\vec{r^u}$. Analogous vector models can be constructed for users ($\vec{u^r}$,$\vec{u^t}$) and tags ($\vec{t^u}$,$\vec{t^r}$).

3.2 Component Recommenders

For each of our component resource recommenders, we assume that it accepts a user-resource pair and returns a score, $\phi(u, r)$, describing the relevance of the resource to the user. The resources not previously tagged by the user are sorted by their corresponding scores and the top n resources are returned:

$$S(u) = TOP^n_{r \in R} \phi(u, r) \tag{2}$$

Resources in the user profile are not considered. In the rest of this section we discuss individual recommenders before turning our attention to the linear weighted hybrid recommender.

3.2.1 Popularity Model

Perhaps the simplest recommender is one which merely recommends the most popular resources. We call this approach Pop and define its measure as:

$$\phi(u, r) = \sum_{v \in U} \theta(v, r) \tag{3}$$

where $\theta(v, r)$ is 1 if v has annotated r and 0 otherwise. While Pop is not necessarily an effective recommender, it does serve as a baseline and may benefit the hybrid.

3.2.2 User-Based Collaborative Filtering

User-based collaborative filtering works under the assumption that users who have agreed in the past are likely to agree in the future [5, 15]. A neighborhood of the most similar users is identified through a similarity metric. For any given resource the score sum can then be calculated as:s

$$\phi(u, r) = \sum_{v \in N} \sigma(u, v) \theta(v, r) \tag{4}$$

where N is the k nearest neighbors to u and $\sigma(u, v)$ is the cosine similarity between the users u and v. As before, $\theta(v, r)$ is 1 if v has

annotated r and 0 otherwise. When users are modeled as resources we call this approach KNN_{UR}. When users are modeled as tags we call this technique KNN_{UT}.

3.2.3 Item-Based Collaborative Filtering

Item-based collaborative filtering [2, 13] relies on discovering similarities among resources rather than among users. We may model the resources as a vector over the user space, KNN_{RU}. When relying on tags, the vector contains the frequency with which a resource has been annotated with the tags, KNN_{RT}. Given r we define N as the k nearest resources drawn from the user profile and then define the relevance of r for the user as:

$$\phi(u,r) = \sum_{s \in N} \sigma(r,s)\theta(u,s) \tag{5}$$

If a user has annotated resources similar to r then $\phi(u,r)$ will be high.

3.2.4 Tag Model Similarity

Given that we may define both users and resources as a vector over the tag space, we may directly measure the similarity between the two elements. We call this model *TagSim* and define its measure as:

$$\phi(u,r) = \frac{\sum_{t \in T} RT(r,t) \times UT(u,t)}{\sqrt{\sum_{t \in T} RT(r,t)^2} \times \sqrt{\sum_{t \in T} UT(u,t)^2}} \tag{6}$$

This method works under the assumption that the frequency with which a user employs a tag measures his interest in the topic described by that tag. We assume that the frequency of the tags applied to the resource adequately describe the resource. If these two models are similar we can infer a relationship between the user and resource.

3.3 Linear Weighted Hybrid

We employ a linear weighted hybrid model [1], which aggregates the results of several components in linear combination. The constituent recommenders are freed from the burden of covering all the available dimensions and instead focus on only a few. A successful hybrid creates a synergistic blend of its constituent parts producing results superior to what they could achieve alone.

In order to ensure that the relevance scores for each recommendation approach are on the same scale, we normalize the scores $\phi(u,r) \in \Phi$ in the range 0 to 1 producing Φ'. A hybrid resource recommender will accept a user u and query its component recommenders, $c \in C$, for each resource, r, then combine the results in the linear model:

$$\phi_h(u,r) = \sum_{c \in C} \alpha_c \phi'_c(u,r) \tag{7}$$

where α_c is the strength given to the recommender, c. We require that the sum of the α vector equal 1. The resources are then sorted by their relevance scores and the top n are returned. As additional recommenders are added to the hybrid, its complexity grows. The challenge then becomes how to ascertain the correct α for each component in order to maximize the effectiveness of the hybrid.

We use a hill climbing technique because of its speed and simplicity. The α vector is initialized with random positive numbers constrained such that the sum of the vector equals 1. The vector is then randomly modified and tested to ascertain if it achieves better results. If the result is improved, the change is accepted; otherwise it is rejected. Occasionally a change to the α vector is accepted even when it does not improve the results in order to more fully explore the α space. Modifications continue until the vector stabilizes. In order to ensure that a local maximum has not been discovered, the experiment is repeated 100 times from different starting points.

4. INFORMATION CHANNELS IN SOCIAL TAGGING SYSTEMS

An information channel models the relationship between the underlying dimensions in a tagging system: users, resources and tags. A strong information channel between two dimensions means that information in the first dimension will be useful in building a predictor for the second dimension. For example, a strong information channel between users and tags means that user characteristics will be a good basis on which to predict choice of tags.

Information channels vary in their strength because users employ social tagging systems in a variety of ways. Consider the problem of resource recommendation. The channel from tags to resources produces a descriptive model of the resources based on how users have described them, and the channel from users to resources provides an alternative model, in which resources are characterized by who has tagged them. Other researchers have noted that there are two classes of tagging behavior "categorizing" and "describing." Describing behavior emphasizes the use of tags to objectively label resources in ways likely to be shared among a user population. It follows that a taggging system in which users engage in describing behavior would end up with a strong channel between tags and resources. Alternatively, if the emphasis of the system is on sharing resources with friends, the resource-user channel may be more dominant.

An analysis of information channels in a social tagging system may help us understand the behavior of recommendation algorithms. We propose entropy and conditional entropy for the evaluation of information channels. Entropy measures the amount of uncertainty associated with a dimension, in this case the user, resource or tag dimensions. It relies heavily on probabilities, however the notion of probabilities in social annotation systems can be ambiguous. The probability of resource might be its likelihood to occur in a user profile, a tag profile or in an annotation. We define the probability of a resource r as:

$$p(r) = \frac{\sum_{u \in U} \sum_{t \in T} URT(u,r,t)}{y} \tag{8}$$

where y is defined as the number of non-zero entries in URT. We may then define the entropy as:

$$H(R) = -\sum_{r \in R} p(r) log_y p(r) \tag{9}$$

Entropy calculations often use the log base of 2, 10 or e. In this work we use a base of y. Doing so bounds the maximum entropy to 1. This will not change the relative values within a dataset, but it will permit the comparison of values across datasets.

Conditional entropy measures the uncertainty of a dimension given another dimension. The conditional entropy of the resource space given the tag space is defined as:

$$H(R|T) = -\sum_{r \in R} \sum_{t \in T} p(r,t) log_y \frac{p(r,t)}{p(t)} \tag{10}$$

where $p(r,t)$ is the likelihood of r and t occurring together in URT, or more formally:

	$H(U)$	$H(U\|R)$	$H(U\|T)$	$H(R)$	$H(R\|U)$	$H(R\|T)$	$H(T)$	$H(T\|U)$	$H(T\|R)$
Delicious	0.551	0.257	0.431	0.631	0.338	0.418	0.434	0.315	0.221
Amazon	0.609	0.275	0.348	0.646	0.312	0.297	0.505	0.244	0.156
LastFM	0.608	0.314	0.357	0.623	0.328	0.431	0.436	0.185	0.245

Table 1: The entropy and conditional entropy of users, resources and tags across all six datasets.

$$p(r,t) = \frac{\sum_{u \in U} URT(u,r,t)}{y} \quad (11)$$

The conditional entropy of resources given users, $H(R|U)$ can be similarly calculated. Once $H(R)$, $H(R|T)$ and $H(R|U)$ have been calculated, it is possible to evaluate the information channels. If $H(R|T)$ is roughly equal to $H(R)$, it means that tags are not offering additional information about the resource space; it might then be difficult to predict a resource given a tag. On the other hand, if $H(R|T)$ is much less than $H(R)$ it means that tags may be a good predictor of resources. Comparing the $H(R|T)$ and $H(R|U)$ values may suggest which information channel is most useful.

Analogous definitions can be constructed for the entropy and conditional entropy of the user and tag spaces. It is important to note that $H(R|T)$ is not equal to $H(T|R)$. It may be the case that tags are good predictors of resources, but resources are not good predictors of tags.

5. EXPERIMENTAL RESULTS

In this section we describe the methods used to gather and preprocess our datasets. Following an outline of our methodology, we examine the results for each dataset separately, and finally draw some general conclusions.

5.1 Datasets

The experiments below are conducted using data from three large real-world social tagging systems. On all datasets we generate p-cores [7]. Users, resources and tags are removed from the dataset in order to produce a residual dataset that guarantees each user, resource and tag occur in at least p annotations. We define a post to include a user, a resource, and every tag the user has applied to the resource.

Several reasons exist to construct p-cores. By eliminating infrequent items, the size of the data is dramatically reduced allowing the application of recommendation techniques that would otherwise be computationally impractical. By removing rarely occurring users, resource or tags, noise in the data can be dramatically reduced. Because of their scarcity, these are the very items likely to confound recommenders. Recommendation in the so-called long tail is a valid area of exploration, but it lies outside the scope of this paper.

Delicious is a popular Web site in which users annotate URLs. On 19 October 2008, 198 of the most popular tags were taken from the user interface and the site was recursively explored. From 20 October to 15 December, the complete profiles of 524,790 users were collected. Due to memory and time constraints, 10% of the user profiles was randomly selected, and a 20-core taken for experiments. The dataset is our largest, containing 7,665 users, 15,612 resource and 5,746 tags. It contains 720,788 annotations.

Amazon is one of the world's largest retailers. The site includes a myriad of ways for users to express and discover opinions of the products: ratings, editorial reviews, customer reviews, product details, and customer purchasing habits. Recently, Amazon has added social tagging to this list. Beginning on 1 July 2009 we recursively explored the site to gather 1.5 million user profiles. Many users had extremely small profiles or used idiosyncratic tags. After taking a 20-core of the data it contained 498,217 annotations with 8,802 users, 10,679 resource and 5,559 tags.

LastFM users upload their music profile, create playlists and share their musical tastes online. We selected 100 random users from the system and recursively explored the "friend" network. Only about 20% of the users had annotated a resource. Users have the option to tag songs, artists or albums. A p-core of 20 was drawn from the data. It contains 2,368 users, 2,350 resources, 1,141 tags and 172,177 annotations. The experiments presented in this paper are limited to album annotations, though our experiments on song and artist data reveal similar trends.

5.2 Methodology

For each of the users, their annotations were divided randomly among five folds. Each annotation included the user, a resource and all tags applied by the user to the resource. As opposed to tag recommendation in which it is advantageous to recommend tags from the user profile, in resource recommendation we attempt to discover new resources the user would like.

Four folds were used as training data to build the component recommenders. The fifth was used to train the model parameters and ascertain the optimal weights of the component in the hybrids. The fifth fold was then discarded and we performed four fold cross validation on the remaining folds. The results were averaged over each user, then over the final four folds.

Given a user, the recommenders are evaluated on their ability to recommend resources found in the user's holdout set, R_h, when compared to the recommendation set, R_r. Recall is a common metric for evaluating the utility of recommendation algorithms. It measures the percentage of items in the holdout set that appear in the recommendation set. Recall is a measure of completeness and is defined as: $r = |T_h \cap T_r|/|T_h|$

Precision is another common metric for measuring the usefulness of recommendation algorithms. It measures the percentage of items in the recommendation set that appear in the holdout set. Precision measures the exactness of the recommendation algorithm and is defined as: $p = |T_h \cap T_r|/|T_r|$

The recall and precision will vary depending on the size on the recommendation set. In the following experiments we present the metrics with recommendation sets of size one through ten.

5.3 Experimental Evaluation

The entropy and conditional entropy values are found in Table 1. The entropy of the resource space is similar for all datasets: 0.631, 0.646 and 0.623. However, the conditional entropy varies across datasets. In Delicious and LastFM, users appear to predict resources better than tags, while in Amazon, the tags are marginally better.

The metrics highlight other differences in the data. In LastFM $H(T|U)$ is particularly low. This may be because users employ idiosyncratic tags for their music collection. In contrast, in Amazon, the $H(T|R)$ is lower than $H(T|U)$, perhaps because users are

	Pop	$TagSim$	KNN_{ur}	KNN_{ut}	KNN_{ru}	KNN_{rt}
Delicious	0.004	0.263	0.512	0.069	0.119	0.033
Amazon	0.053	0.254	0.419	0.001	0.131	0.147
LastFM	0.006	0.153	0.410	0.005	0.425	0.001

Table 2: Contribution of the individual components in the hybrids for each of the six data sets.

employing tags commonly given to products such as "blueray" or "dvd."

Detailed analysis of the results for each dataset appear below. However, as shown in Figures 1 through 3, the hybrid outperforms its constituent components. The learned weights of the components of the hybrid are shown in Table 2. Each hybrid draws in different degrees from all the components and combines multiple information channels to achieve superior results.

In all datasets KNN_{ur} achieved the best results among the individual components and is the strongest weighted component is every hybrid. This is not surprising since the resource recommendation task can be viewed as the prediction of user-resource pairs. The user-resource information channel would be particularly relevant.

Many differences among the data sets are also apparent. First, while KNN_{ur} is always the most effective recommender, the remaining recommenders demonstrate varied utility. For example KNN_{ut} does well in Amazon but performs poorly in LastFM. Secondly, the overall effectiveness of the recommenders varies from datasets to Dataset. Amazon permits recall and precision measures of up to 30 or 40 percent. Delicious is a much more difficult target. Thirdly, the relative contributions of the components shown in Table 2 vary greatly among the datasets, revealing that in each dataset certain information channels are dominant over others and require a different emphasis on the components to effectively exploit the data. This evidence suggests a great deal of variability among social tagging systems. The manner in which users interact with the applications produces data with varying strength across the information channels.

5.3.1 Delicious

In Delicious users annotate a diverse collection of web pages. While users may share resources or view other user profiles most often they annotate web pages so that they may revisit them later. Unlike Amazon, where tags are often drawn from familiar classification terms, Delicious users often employ idiosyncratic tags. The diversity of the user profiles and inconsistency of the tags make Delicious a difficult target for resource recommendation.

As shown in Figure 1, KNN_{ur} outperforms all other components. In particular it outperforms KNN_{ut}, suggesting that in this dataset and for resource recommendation, users are better modeled by resources than by tags. We also see the dominance of the user-resource channel over the user-tag channel in the $H(U|R)$ and $H(U|T)$ values. $H(U|R)$ is 0.257 revealing that the entropy of the user space given the resource space is relatively low. On the other hand, $H(U|T)$ is 0.431, much closer to the entropy of the user space alone, 0.551. In this instance, our conditional entropy metrics correctly predicts which user-based collaborative filtering method would prove superior.

The conditional entropy metrics also predict which item-based collaborative filtering method would perform better. $H(R|U)$ is much lower than $H(R|T)$ revealing that users provide more information about the resources than do tags. As expected, KNN_{ru} outperforms KNN_{rt}. $TagSim$ which relies on modeling both users and resource over the tag space also fares poorly, likely because neither users or resource are modeled well by tags. Moreover a user's tag profile can encompass many diverse topics whereas a resource's tag profile will be specific to its own topic.

If we consider the contribution of each component to they hybrid (see Table 2), we see the majority of the weight lies in KNN_{ur}, which would be expected since this was the most successful individual recommender. The next strongest component is $TagSim$. This might be surprising, given that $TagSim$ works poorly on its own. However, the tag information is entirely ignored in KNN_{ur}, so we must see this as an instance where the complementary tag-related channel is providing useful signals.

KNN_{ru} provides some utility as well, which reinforces the user-resource channel, but instead of modeling users as resources as in KNN_{ur}, it models resources as users. KNN_{ut} which was the second-strongest individual component is poorly represented in the hybrid. We know that users are not well-represented by tags. A likely explanation is that once the other user-based collaborative filtering method is included in the hybrid, no additional information is provided by the tag-based representation.

5.3.2 Amazon

Figure 2 displays the experimental results of the Amazon dataset. The hybrid shows improvement over the components, but not as much as in the other datasets. The dataset also appears to be one of the easier targets, allowing more than 25% recall and as much as 45% precision.

As usual KNN_{ur} does very well. For the task of recommending resources in Amazon it appears better to model users as resources rather than by tags. This was predicted by the conditional entropy values $H(U|R)$ and $H(U|T)$ which show resources to be more informative of users than tags.

The $H(R|U)$ and $H(R|T)$ are very close. However KNN_{ru} does much better than KNN_{rt}, revealing that the recommendation task is as important a consideration as the entropy metrics when predicting which components will prove superior. For resource recommendation, the task is to predict the relation between users and resources. In this scenario the user-resource channel is particularly relevant and KNN_{ru} therefore outperforms KNN_{rt}.

Once again $TagSim$ performs poorly when compared to other approaches. This is likely due to the fact the a user may use tags across a diverse selection of resources, but tags applied to a resource are often focused on the resource's content.

The most dominant component in the linear weighted hybrid is once again KNN_{ur}. The next two strongest components are $TagSim$ and KNN_{rt}. These components exploit information channels complementary to those used by KNN_{ur}. Because these components rely on the tag space, we can infer that the tags in Amazon offer greater utility than in Delicious.

This observation is confirmed by an analysis of the conditional entropy metrics as well as the application itself. Even though the $H(U)$ and $H(R)$ values are roughly equal in the two datasets, the $H(U|T)$ and $H(R|T)$ value show that tags are better predictors of both users and resource in Amazon than they are in Delicious. It

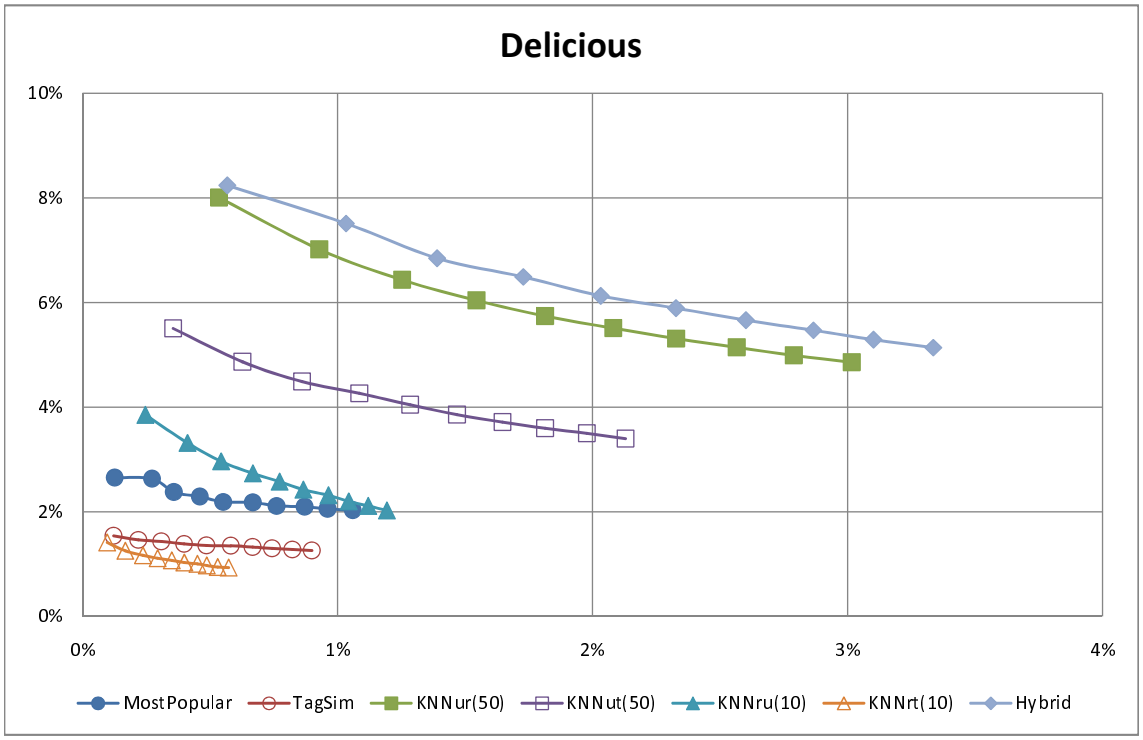


Figure 1: The recall and precision plotted for recommendations sets of size one through ten in Delicious.

may be that the users in Amazon tend to engage in "describing" behavior more often that those in Delicious, using labels such as "sony" or "1080p."

5.3.3 *LastFM*

In LastFM, KNN_{ur} and KNN_{ru} perform well. The results are found in Figure 3. Both these recommenders concentrate on the user-resource relation, an advantageous strategy when trying to match users to resources. Furthermore, $H(U|R)$ is lower than $H(U|T)$ and $H(R|U)$ is lower than $H(R|T)$ once again demonstrating that entropy and conditional entropy might be used to explain why some recommendation techniques prove superior over others.

In contrast to other datasets, there is a wide gulf between these approaches and the remaining techniques. KNN_{ut}, KNN_{rt} and *TagSim* all perform nearly as bad or worse as the non-personalized baseline.

This may be due to the fact that its users do not store their music within the LastFM application. Instead users upload their listening habits through a process called 'scrobbling'. The system records which songs they listened to and how often they have listened to them. Since the song titles are uploaded in batches and the user is not actively engaged in the music when it is added to his profile, the annotation process becomes a burdensome additional task which is often neglected. Rather than using the application for organizing and exploring music through the tag space, users often employ the system to find new music and friends through the resource and user space. Visual examination of the tag space reveals that when the users do annotate albums, the tags are often overly generic, such as "rock," or not descriptive of the resource, such as "album i own."

In terms of resource recommendation, this analysis explains why tags would offer little utility in resource recommendation. Similarly, an integrative recommender utilizing multiple dimensions would not achieve a significant boost in performance over those that focus solely on the user-resource relation.

These conclusions are supported by the entropy and conditional entropy values. The extremely low $H(T|U)$ value (0.185) shows that users are employing idiosyncratic tags. These tags cannot be used to effectively model either the user or the resource to which they are applied. In contrast, Amazon's $H(T|R)$ value (0.156) shows how users are agreeing on which tags should applied to the resources permitting usage of the tag space for modeling resources.

Table 2 reveals that the LastFM hybrid is almost exclusively composed of KNN_{ur} and KNN_{ru}. Once again this reliance on a single information channel confirms the weakness of the tag dimension. $TagSim$ offers some utility to the hybrid but far less than it does in either Delicious or Amazon.

5.3.4 *Summary*

A key finding of this work (and our work with other datasets not included here for reasons of space) is that social tagging systems are very diverse, even though the essential functionality that they provide is very similar from system to system. This diversity may be generated by the nature of the resources being labeled, the nature of the users that a site attracts, the type of task that the system is intended to support, or some combination of these factors. For example, Amazon users annotate resources from the consumer product space and use tags common in this environment, such as "bluray." LastFM users, on the other hand, generally focus on interacting with other users and use tags which help display their tastes. A tag like "albums I own," which is the most popular tag on LastFM, is useless for retrieval of resources, but potentially very helpful in matching users with similar music collections.

The diversity of social annotation applications is manifested in

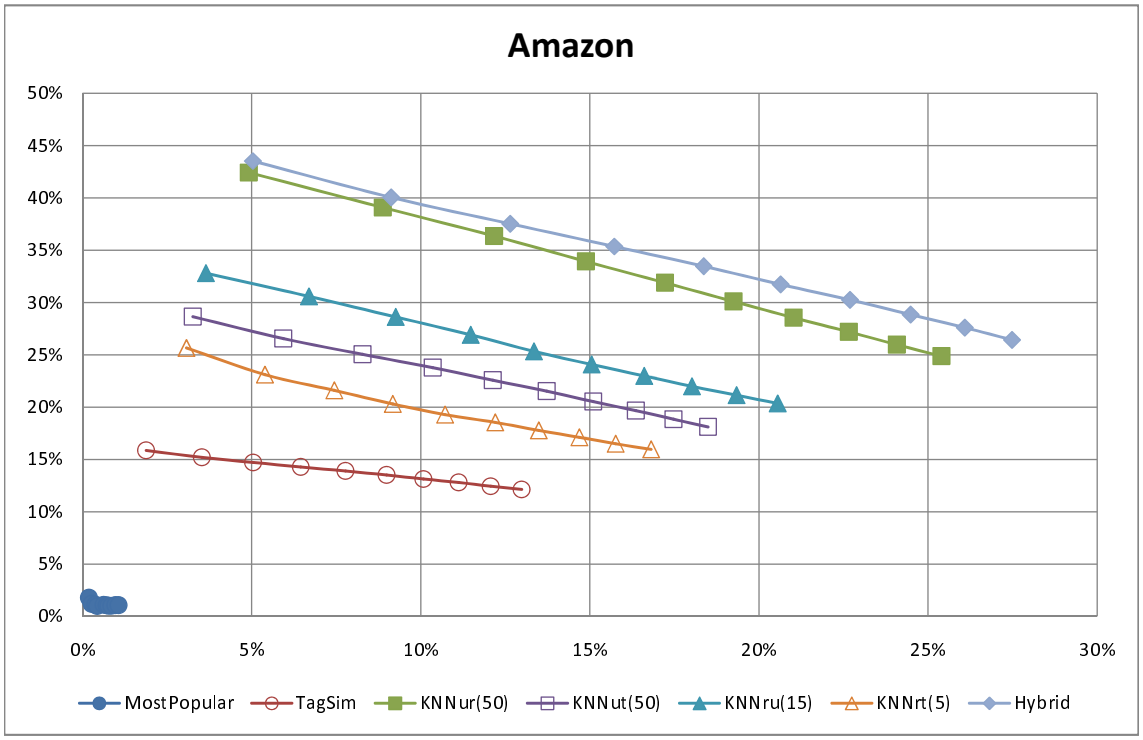


Figure 2: The recall and precision plotted for recommendations sets of size one through ten in Amazon.

the characteristics of the datasets derived from these applications, as demonstrated by the entropy and conditional entropy measures. This is the sense is which we wish to define information channels, as measures of the utility of one dimension in predicting another. The entropy analysis clearly shows that the resource-tag channel is much stronger in Amazon than in LastFM: compare $H(R)$ vs $H(R|T)$ and $H(T)$ vs $H(T|R)$ in Table 1.

The varying characteristics of the information channels in the data lead to differences in the performance of recommendation components that draw from these channels. We see this both in terms of the precision and recall values obtained by each component in isolation and also by the contribution of each component to the weighted hybrid. As shown in Figures 2 and 3, many of the components perform well in Amazon, while some of the tag-focused recommenders perform worse than even the non-personalized baseline in LastFM. Also, the relative weights of the components found in Table 2 also show that Amazon enjoys the contribution of many recommenders whereas LastFM can only find utility with a few.

Our results also demonstrate the need for exploiting multiple information channels. On these and other datasets, the linear weighted hybrid recommender is superior to recommenders operating on any two-dimensional projection of the data, a result which also has been demonstrated in graph-based and matrix factorization solutions. However, our proposed linear-weighted hybrid achieves this integration of information channels in a highly-scalable and updatable solution.

The hybrid also offers extensibility. In this work we focused on recommenders which draw from the three dimensions of users, tags, and resources, but it might be useful to incorporate other recommendation types. For example, a recommender based on recency might favor resources recently added to the user profile over those that have not been used lately. A content-based recommender could make use of product descriptions, web page content or other data associated with resources for recommendation. It may be useful to make use of context in recommending resources: recent queries, other recently-visited resources, etc. The weighted hybrid has the advantage that such components can be easily incorporated.

6. CONCLUSIONS

Our results motivate several conclusions. First, the experiments reveal that not all social tagging systems are equal. The way users interact with a system can dramatically affect the strength of its information channels. Second, metrics such as those based on conditional entropy can characterize the differences between systems, explain why certain recommenders outperform others, and predict which social tagging systems would benefit most from an integrative approach such as a hybrid recommender. Finally, weighted linear hybrids provide a simple and efficient means to aggregate several dimensions of the data into a single framework. These hybrids can be constructed from components that are computationally efficient and easily implemented. Moreover, the hybrid offers a high degree of extensibility. These characteristics are not shared by other integrative approaches such as graph-based models and tensor factorization.

7. ACKNOWLEDGMENTS

This work was supported in part by a grant from the Department of Education, Graduate Assistance in the Area of National Need, P200A070536.

8. REFERENCES

[1] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

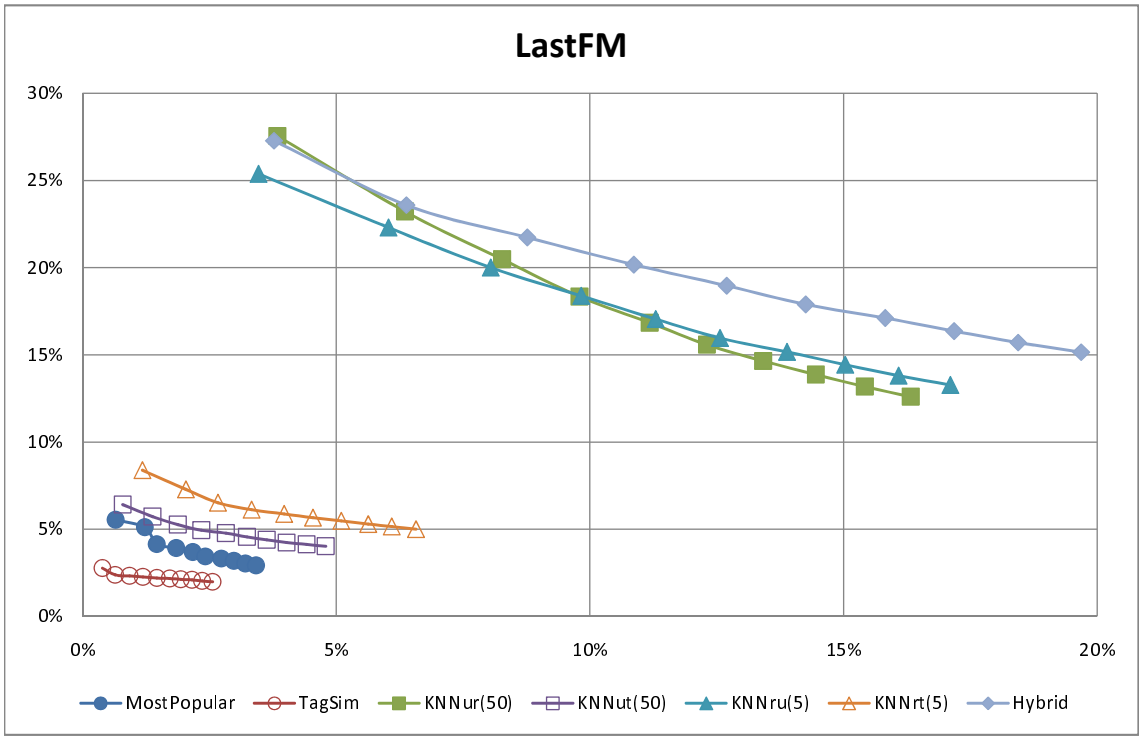


Figure 3: The recall and precision plotted for recommendations sets of size one through ten in LastFM.

[2] M. Deshpande and G. Karypis. Item-Based Top-N Recommendation Algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.

[3] J. Gemmell, M. Ramezani, T. Schimoler, L. Christiansen, and B. Mobasher. A fast effective multi-channeled tag recommender. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases Discovery Challenge*, Bled, Slovenia, 2009.

[4] J. Gemmell, T. Schimoler, B. Mobasher, and R. Burke. Improving folkrank with item-based collaborative filtering. In *Recommender Systems & the Social Web*, New York, New York, 2009.

[5] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In *22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, California, 1999. ACM.

[6] A. Hotho, R. Jaschke, C. Schmitz, and G. Stumme. Information Retrieval in Folksonomies: Search and ranking. *Lecture Notes in Computer Science*, 4011:411–426, 2006.

[7] R. Jaschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag Recommendations in Folksonomies. *Lecture Notes In Computer Science*, 4702:506–513, 2007.

[8] A. Mathes. Folksonomies-Cooperative Classification and Communication Through Shared Metadata. *Computer Mediated Communication, (Doctoral Seminar), Graduate School of Library and Information Science, University of Illinois Urbana-Champaign, December*, 2004.

[9] P. Mika. Ontologies are us: A unified model of social networks and semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1):5–15, 2007.

[10] S. Rendle and L. Schmidt-Thieme. Factor Models for Tag Recommendation in BibSonomy. In *ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, Bled, Slovenia, 2009.

[11] S. Rendle and L. Schmidt-Thieme. Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, New York, New York, 2010.

[12] G. Salton, A. Wong, and C. Yang. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[13] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-Based Collaborative Filtering Recommendation Algorithms. In *10th International Conference on World Wide Web*, Hong Kong, China, 2001.

[14] S. Sen, J. Vig, and J. Riedl. Tagommenders: connecting users to items through tags. In *Proceedings of the 18th international conference on World wide web*, Madrid, Spain, 2009.

[15] U. Shardanand and P. Maes. Social Information Filtering: Algorithms for Automating "Word of Mouth". In *SIGCHI Conference on Human Factors in Computing Systems*, Denver, Colorado, 1995.

[16] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM conference on Recommender systems*. Lausanne, Switzerland, 2008.

Weighted *Content Based* Methods for Recommending Connections in Online Social Networks

Ruth Garcia
Universitat Pompeu de Fabra
Barcelona,Spain
rutholimpia.garcia01@campus.upf.edu

Xavier Amatriain
Telefonica R+D
Barcelona, Spain
xar@tid.es

ABSTRACT

Online Social Networks currently have an important role in the life of millions of active internet users. Cases like Twitter are of special attention since a lot of connections are made between people who never met before and with no need of reciprocation. For this reason it is important to find new ways to provide recommendations that may be of interest for users. Should these recommendations focus on the popularity, on the activity, location, common friends or content? Should recommendations be influenced by egocentric or global network metrics? This research is the first phase of an in-depth study of a large dataset based on Twitter which aims to answer the previous questions. Despite many studies based on global rankings, the authors believe that recommendations should mostly be based on the preferences made by users in their own networks. This stage of the study focuses on the popularity and activity of links as indicators to predict connections.For this end, the authors compute a weight for each of these features, which varies for each user. Each pair tested is accepted if it satisfies a minimum total weight. Results show a slight but important improvement in performance when using two features instead of one, the results gives an insight that if more features are considered more improvements in predictions will be found. The results of this paper can and should be accompanied with more research.

Keywords

followee, follower, features, Activity, Popularity, weight, O.S.N

1. INTRODUCTION

Nowadays, Online Social Networks have an important role in the life of millions of active internet users and an even stronger growth and penetration is expected in the following years. According to Mccann [4], 62.5% of Active Internet Users in the world belonged to at least one O.S.N by March of 2009 compared to the 57% in the previous year. From the first group, 56.4% try to "Find New Friends" in these networks.

In contrast to networks such as Facebook, Orkut and Myspace where privacy, reciprocity and knowing your connections is important, suggesting friends in environments like Twitter goes beyond the "you may know this person" approach since many connections with strangers do not need reciprocation. Furthremore, the majority of the times profiles are public. In fact, KwaK et al.[3], researches concluded that only 22% of all connections on Twitter are reciprocal. By April of 2010, Twitter has 105 million registered users and 37% of active users use Twitter on their phone[1].

For this reason, recommendations in this type of environment could be more broad and complex. This challenge has attracted major attention from several researchers interested in finding efficient and better ways to connect people. For example, there are methods to rank people in twitter such as TunkRank [9] or TwitterRank [10] while others have focused in avoiding spammers by measuring popularity such as the one presented by Avello [2]. Scellato et al. [6] analyzed how the location affects the social structure and Sudheendra et al.[7] proposes algorithms to find the most efficient path between two users that are not connected, among others. In contrast to recommendations of products where global rankings strongly influence people's choices, recommending people in O.S.N is more abstract. A high ranking does not necessarily mean a better recommendation since people get connected for reasons that may be hard to predict. The authors attempt to identify the reasons of connectivity by measuring how five features influence people when deciding to connect with someone. The features are a) popularity, b) activity, c)location, d)friends in common and e)content of tweets. In this research, which is the first stage of a deeper study, the authors study if the two first features are good indicators to predict connections and if they work better together than alone.

In this regard, the main objectives of this paper are: 1) Propose a method to weight popularity and activity for each user 2)Base predictions of future followees according to these measured weights 3)Observe if the two features together perform better in predictions than alone.

This study does not aim to propose a "good" or "better" way to recommend people in Twitter but it rather tries to explore new possibilities on personalized recommendations. To the best of our knowledge this paper is the only one

[1]http://www.blogherald.com/2010/06/28/twitters-meteoric-rise-compared-to-facebook-infographic/twitter-statistics-infographic-911/.

proposing personalized recommendations in twitter based on *content based* weighted features. In the following sections we will explain the approach, the thresholds we used, the weighting algorithms for the two features, the calculation of predictions and the evaluation method. At the end of the paper, we will discuss the results and propose new challenges for future research.

2. APPROACH

In environments similar to Twitter, users have three types of connections: only followers, only followees and reciprocal connections. Followees are people the user choose to follow and from whom they receive tweets in their so called *public timeline*, followers are those users who have chosen to follow the user in question but their tweets will not appear in the user's timeline whereas reciprocal connections are users that are both followers and followees and therefore information is shared in both directions. Basing recommendation on users who are only followers is not a good idea since in the majority of cases users do not have control over who follows them. Spammers or heavy advertisers looking for reciprocations can become their followers.

For this reason, we have carefully studied the role of followees because we believe that the user's preferences are better reflected in them but we do not substract reciprocal connections as Avello did in [2]. It is true that sometimes users follow back others for "politeness" but with time the tendency is to un-follow those heavy advertisers or people that do not provide information of interest. Active users aiming to find interesting information build a network of followees where the majority of members represent people who the users *care* to keep.

Studies of Kwak et al. [3] and Avello[2] have found homopholy in Twitter network showing that users "engaged in a social activity seem to be associated more closely with ones who are similar to them along a certain dimension such as location, age, political view or organization affiliation, compared to ones who are dissimilar." This shows that despite the lack of reciprocity in Twitter there is a tendency for people to get together with people similar in one or more aspects. In this part of the study, we analyze how similar followees are in a specific network considering popularity and activity. *Popularity* is the ratio of followees and followers and *activity* is the number of tweets a user has posted since entenring the O.S.N.

In this phase of the study , weights are calculated for these two aspects. In order to do that, we first determine if that specific feature (popularity and/or activity) is actually relevant for that user. A feature is considered relevant if it is present in a certain percentage of followees. A recommendation is accepted if its calculated weight is bigger or equal than a threshold. In Section 3.2 we propose one metric for computing these weights.In Section 3.3 we explain how to make predictions according to this algorithm. We assume that these weights are indicators of how important that specific feature is for the user. In the following sections Popularity and Activity are referred as features, characteristics, indicators and aspects interchangeably.

We also focus only in the user's egocentric network of followees [2] and therefore in this study we call the recommendations as *content based people recommendations*. Due to our personalized recommendation approach, we omit calculation of global networks, global rankings and the use of collaborative filtering information since we only take information from the users' own networks. The results in Section 5 show the influence of each one of the features analyzed as well as the combination of both in the performance of finding predictions.

[2]egocentric network is used as "all the edges leading into and out of single user" in [5]

3. DATASET

The dataset was taken from the Web site of Munmon de Choudhury [3]. For this part of the study, we focused on two of the three files provided: the user file and the social graph file. The information given is that Tweets were collected between 2006 and 2009, the user file contains details of 456,107 profiles and the social graph file has a sample of 2,476 users with 815,554 followees links. The details of the profiles in the socialgraph are listed in the user file. Due to the evaluation method adopted and explained in Section 4, we considered those users with a total number of followees bigger or equal to 10. This reduces the sample to 2,381 users and 815,188 followees links. We leave for future research the evaluation of the content of tweets, location and friends in common.

3.1 Thresholds

For the calculation of weights we take into account the following thresholds:

- α : Popularity Threshold (followees/followers).
- $\beta 1$-$\beta 2$: Activity Range (Range of a total number of posts).
- δ : Feature Acceptance (%).
- γ : Weight Acceptance (≤ 1).

The popularity threshold (α) is calculated by the division of followees/followers. This ratio is not an accurate indicator of the prestige of a user. A ratio of 2 does not necessarily mean more popularity than a ratio of 1.5 but we have preferred not to use other more complex methods (see Section 1) because we are not calculating levels of prestige for recommendations but rather weather the user cares or not about popularity. We say that if a user's ratio of followees/followers is bigger than or equal to α then that user cares about popularity and therefore recommendations should consider popularity. We assume that any popular person will have a ratio bigger than one. For this reason, we start by setting α=1 but this value can be changed if needed. In Section 5, we analyze the impact of different α starting values. After determining if a user cares or not about popularity, we recalculate α based on the most likely α value the user will prefer. Section 3.3 specifies how this is done. For more information regarding the different methods of calculating prestige refer to [2].

The activity (β) is the total number of tweets a user has posted since the moment of joining the O.S.N. We assume that if a users has more followees with activities within the range $\beta 1$-$\beta 2$ then that user cares about the level of activity when choosing to follow someone. We assume that values

[3]For more information refer to http://www.public.asu.edu/ mdechoud/datasets.html

lower than $\beta 1$ mean inactivity or a new user and values above $\beta 2$ are candidates for spammers or heavy advertisers.We have analyzed the total dataset and observed that the majority of users are within 200-29000 number of posts. Just like the case of popularity, after determining if the activity is relevant for the user we determine the $\beta 1$ value that best suits the user. Section 3.3 also specifies how this is done. We chose not to recalculate $\beta 2$ because it is very unlikely that a user can post more than 29000 posts during 2 years but $\beta 1$ can vary and therefore other values for $\beta 1$ are evaluated in Section 5.

The feature acceptance threshold (δ) is the minimum percentage allowed in the network of followees for a feature to be considered relevant. For example, if δ=0.3 and α=1 , the popularity is relevant if 30% or more of the followees in that network have a popularity ratio bigger or equal than 1. Likewise, the weight acceptance threshold (γ) is the minimum weight a recommendation should have in order to be accepted. For example, if (γ)=0.7 and the total weight of a recommendation, $TW(r)$, is 0.6, then that recommendation is rejected. As it will be seen in Section 3.2, weight will not be bigger than 1. This approach resembles the System of Symeonidis et al. [8] where items with ratings bigger than 3 stars (varies according to user) are considered for calculations, the rest is not.

3.2 Weighting features

The weights are calculated after determining the relevant features for a particular user. The weight of a relevant feature for a user, u, is determined by the number of followees satisfying α, PF, and/or within the range $\beta 1$-$\beta 2$, BF, divided by the total number of followees. In his study we only consider this division for the calculation of weights because it is based only in the *egocentric network of followees*, leaving more complex metrics for future research. If Popularity is relevant for user u then the weight is calculated as :

$$W_p = \frac{\sum_{\forall f \in PF} f}{\sum f} \quad (1)$$

If Activity is relevant for user u, the weight is calculated as :

$$W_a = \frac{\sum_{\forall f \in BF} f}{\sum f} \quad (2)$$

The total Weight of relevant features for u is denominated as $TW(u)$. If only popularity is relevant then $TW(u) = W_p$, if it is only activity then $TW(u) = W_a$, if it is both $TW(u) = W_a + W_p$.

3.3 Filtering Recommendations

After obtaining the W_p and/or W_a for u, we should filter the future followees candidates, we follow the following steps:

1. Recalculate α and/or $\beta 1$
2. Filter Recommendations according to α and/or $\beta 1$
3. Weight Recommendatios
4. Accept or reject Recommendations according to weight

Step 1 is an step forward that aims to obtain more personalized values for α and/or $\beta 1$. For user u, we calculate this value by finding the median in group PF and/or BF. we have analyzed several other methods such as the average and metrics with thresholds but the Master thesis of the first author of this paper shows that the median is a good indicator [1] for this value.

After new values for α and/or $\beta 1$ are obtained, step 2 says that the recommendations, r, should be filtered according to these thresholds. For each one of the relevant features for u, a recommendation, r in the testing set is considered if $popularity_r \geq \alpha$ or in its turn $\beta 1 \leq activity_r \leq \beta 1$.

Step 3 says to weight the recommendation. We assumed that features that are not relevant for u should not be evaluated in r. The total Weight of each relevant feature r is denominated as $TW(r)$. If only popularity is relevant for r then $TW(r) = W_p(u)$, if it is only activity then $TW(r) = W_a(u)$, if it is both $TW(r) = W_a(u) + W_p(u)$. In other words, we weight r according to the parameters of u since we should judge the importance of the feature according to u.

Step 4 says that a recommendation, r, should be in the accepted group, AR, if the total $TW(r)$ of that recommendation is bigger or equal than γ. In this way, we have that:

$$r \in AR, \text{ if } TW(r) \geq \gamma, \quad (3)$$

4. EVALUATION METHOD

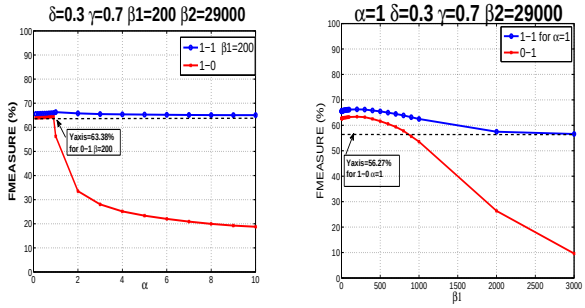
The evaluation method chosen was 10-fold cross-validation. The data has been divided randomly into 10 parts and one part is held out for testing 10 times. The training set is based on the remaining nine parts every time thus executing 10 times the learning procedure. According to Witten et al.[11], extensive tests on numerous datasets have shown that 10 is about the right number of folds to get the best estimates of errors. Nevertheless, we do not repeat the sampling 10 times as it is recommended in [11].

Since every testing set fold has actual connections <u,f> only, we have decided to randomly add fake connections to the testing set. Fake connections were carefully built from the 2,381 users mentioned in Section 3 that are not actually connected. We know that this could cause some unreal assumptions since some of these pairs could be actually connected by now but we consider this to be the most rigorous way of evaluating our algorithms. A recommendation is positive when we predict an existing followee and a recommendation is negative when the followee recommended is a fake connection.

The results show the level of precision, recall and the f-measure of predicting future followees for cases when popularity , activity and both the popularity and activity were considered. If a user u did not receive any recommendations, the recall was 0 and precision is not calculated. We also considered the recommendation percentage, which is the percentage of users in the testing set who received recommendations.

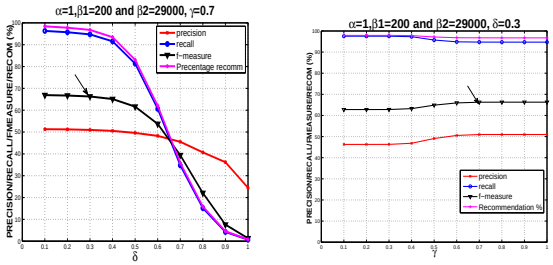
5. RESULTS

Labels with a) 1-0 represent cases where only feature popularity was evaluated, b) 0-1 only activity and c) 1-1 when both features were used. Each point in the x and y axis represent the average of the 10 learning procedures executed for each testing fold. The fake connections and the 10 different testing sets were the same every time.



(a) 1-0 and 1-1 (b) 0-1 and 1-1

Figure 1: FMeasure for a) 1-0, 1-1 with dashed line Yaxis of 0-1 at β1=200 and b) 0-1, 1-1 with dashed line Yaxis of 1-0 at α=1



(a) δ 1-1 (b) γ 1-1

Figure 2: Performance,recall,precision, recommendation percentage for 1-1 cases when a)δ and b) γ change

Improvements with two features instead of one are not very big but we believe they are important and promising. The dashed lines represent the minimum performances that the results of 1-1 cases should achieve. The graphs show that the more the feature being evaluated increases the more the results approach the dashed line.In other words, when two features are considered and a threshold reaches a point where its value is hard to achieve then the less it will be used in the algorithm, leaving the other feature to take the lead. Therefore the results in those situations start to behave as if there was only one feature present stable at the value marked by the dashed line. Figure 1a) shows an improvement of around 2% when α=1 while Figure 1b) have improvements of around4% when β1=200. Not in one moment cases considering only one feature perform better than cases with two features. For this reason, two features are better than one even if it is in a slight proportion. We expect more promising results when other features such as Content and common friends are added in future research.

On the other hand, Figure 2a and 2b show that the threshold γ does not makes a significant difference in the results. This means that we could automatically accept a recommendation if one or two of its features satisfies the new α and/or β1 rather than evaluating them again with γ. Nevertheless, we decided to keep γ because it may become significant when more features are added in future research and because it is a way to control how rigorous recommendations should be (i.e γ=1 when we want for force the algorithm to accept recommendations with the same amount of relevant features than u) .

6. DISCUSSION AND CONCLUSION

This paper proposes a way of recommending based on their followees. Non-reciprocal connections give space to innumerable ways of recommendations and challenges. In this study, two features were analyzed:popularity and activity. The authors have tried to identify and measure the impact of features influencing users to connect to others. A weighting algorithm based only on egocentric networks was proposed and discussed. For each user tested, the weighting algorithm takes into account the frequency of relevant features appearing in their followees profiles. A dataset with more than 800,000 connections was analyzed to test this weighting algorithm. Even with small improvements, we demonstrated that it is better to consider two features than only one when doing predictions because it help us to understand more the preferences of users when they choose connections. These preferences do not always go in hand with global rankings of prestige and activity hence we only analyzed the egocentric network. We demonstrated that weighting the impact of features in a personalized way could be an interesting , novel and personalized way of recommending connections. Although these experiments were carried out with Twitter, the algorithm and concepts can be applied in other environments. We look forward to continue with the analysis of other features such as location, content of tweets and connections in common. We are very enthusiastic about the potential improvements that considering more features could add to our results. We are also eager to compare our results with global ranking algorithms. In other future research, only reciprocal connections should be also studied.

7. REFERENCES

[1] R. Garcia. Hybrid methods for recommending connections in online social networks. Master Thesis, June 2010. UPF.

[2] D. Gayo-Avello. Nepotistic Relationships in Twitter and their Impact on Rank Prestige Algorithms. *Arxiv preprint arXiv:1004.0816*, 2010.

[3] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *Proceedings of the 19th international conference on WWW*, pages 591–600. ACM, 2010.

[4] U. McCann. UM Wave 4-Social Media Tracker: Power to the people. Universal McCann GmbH, Frankfurt am Main.

[5] M. Roth, A. Ben-David, D. Deutscher, G. Flysher, I. Horn, A. Leichtberg, N. Leiser, Y. Matias, and R. Merom. Suggesting friends using the implicit social graph. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 233–242. ACM, 2010.

[6] S. Scellato, C. Mascolo, M. Musolesi, and V. Latora. Distance Matters: Geo-social Metrics for Online Social Networks.

[7] M. S. L. J. H. Sudheendra Hangal, Diana MacLean. All friends are not equal: Using weights in social graphs to improve search. In S. University, editor, *4th ACM Workshop on SONAM*, July 25, 2010.

[8] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Feature-Weighted User Model for Recommender Systems. *User Modeling 2007*, pages 97–106, 2009.

[9] D. Tunkelang. A Twitter Analog to PageRank (Authors' BlogOnline), January 2009.

[10] J. Weng, E. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 261–270. ACM, 2010.

[11] I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.