A Scalable Tag-Based Recommender System for New Users of the Social Web

Valentina Zanardi and Licia Capra

Dept. of Computer Science
University College London
Gower Street, London, WC1E 6BT, UK
{V.Zanardi|L.Capra}@cs.ucl.ac.uk

Abstract. Folksonomies have become a powerful tool to describe, discover, search, and navigate online resources (e.g., pictures, videos, blogs) on the Social Web. Unlike taxonomies and ontologies, which overimpose a hierarchical categorisation of content, folksonomies empower end users, by enabling them to freely create and choose the categories (in this case, tags) that best describe a piece of information. However, the freedom afforded to users comes at a cost: as tags are informally defined and ungoverned, the retrieval of information becomes more challenging. In this paper, we propose *Clustered Social Ranking* (CSR), a novel search and recommendation technique specifically developed to support new users of Web 2.0 websites finding content of interest. The observation underpinning CSR is that the vast majority of content on Web 2.0 websites is created by a small proportion of users (*leaders*), while the others (*followers*) mainly browse such content. CSR first identifies who the leaders are; it then clusters them into communities with shared interests, based on their tagging activity. Users' queries (be them searches or recommendations) are then directed to the community of leaders who can best answer them. Our evaluation, conducted on the CiteULike dataset, demonstrates that CSR achieves an accuracy that is comparable to the best state-of-the-art techniques, but at a much smaller computational cost, thus affording it better scalability in these fast growing settings.

Keywords: cold-start problem, scalability, recommender systems, social tagging, clustering

1 Introduction

The rise of Web 2.0 has transformed users from passive consumers to active producers of content. This has exponentially increased the amount of information that is available to users, from videos on sites like YouTube and MySpace, to pictures on Flickr, music on Last.fm, blogs on Blogger, and so on. This content is no longer categorised according to pre-defined taxonomies (or ontologies). Rather, a new trend called social (or folksonomic) tagging has emerged, and quickly become the most popular way to describe content within Web 2.0 websites. Unlike taxonomies, which overimpose a hierarchical categorisation of

content, folksonomies empower end users by enabling them to freely create and choose the tags that best describe a piece of information (a picture, a blog entry, a video clip, etc.). However, this freedom comes at a cost: since tags are informally defined, continually changing, and ungoverned, *finding* content of interest has become a main challenge, because of the number of synonyms, homonyms, polysemy, as well as the inevitable heterogeneity of users and the noise they introduce.

In order to assist users finding content of their own interest within this information abundance, new techniques, inspired by traditional recommender systems, have been developed: users' profiles are built, collecting information about their tastes/interests; these profiles are then processed to predict what resources they will like. While high accuracy can be afforded for users whose preferences are well known in the system (e.g., users who have rated/tagged a lot of content) [9], very little has been done so far for new users. However, the so-called *cold start* problem is dominant in Web 2.0 websites, where a large number of new users joins the system daily; furthermore, their tastes are more difficult to learn than in traditional recommender systems, as they are not simply expressed as numerical ratings over consumed content, but as freely chosen sets of tags associated to it. In order to retain these users, a recommender system must be capable of recommending content, even when very little information is available about a user's interests.

In this paper, we propose *Clustered Social Ranking* (CSR), a novel search and recommendation technique specifically developed to support new users of Web 2.0 websites finding content of interest. The observation underpinning CSR is that the vast majority of content on Web 2.0 websites is created by a small proportion of users, while the others mainly browse such content. We call the former *leaders*, and the latter *followers*. CSR first identifies who the leaders are; it then clusters them into communities with shared interests, based on their tagging activity. Users' queries (be them searches or recommendations) are then directed to the community of leaders who can best answer them. We have evaluated Clustered Social Ranking on the Web 2.0 CiteULike[1] website; our results demonstrate that CSR achieves an accuracy that is comparable to the best state-of-the-art recommender system techniques, but with a computational cost that is by orders of magnitude smaller, thus affording it better scalability in these fast growing settings.

The remainder of the paper is structured as follows: in Section 2 we review the state-of-the-art in recommender systems for the social web, highlighting their limitations in terms of cold-start problem. In Section 3 we present Clustered Social Ranking; Section 4 presents our experimental setup, in terms of dataset used, computed metrics, and benchmarks, while Section 5 analyses the obtained results. Finally, Section 6 concludes the paper.

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

2 Related Work

Research has been very active in the area of folksonomic searches and recommendations, splurged by the huge popularity of social tagging websites, such as Delicious, Flickr, Digg, Reddit, and the like. In these scenarios, users' tastes and interests are not expressed as numerical ratings, but rather as freely chosen tags associated to content. As a large study of social tagging conducted on the popular Delicious bookmarking system illustrated, folksonomies are so large and dynamic that traditional web search techniques are no longer affordable ([8]). Novel techniques, helping users to find relevant content in these settings, are thus called for.

One stream of research has focused on inferring the semantic relationship between tags, starting from an analysis of how users employ them. For example, [7] and [18] tried to build a navigable hierarchical taxonomy of tags, purely starting from tag usage. In [24], a simple technique to disambiguate tags is proposed, based on an analysis of the relationship between users, tags and resources. In [3], tag co-occurrence is broadly studied, starting from a tri-partite network of users, tags and resources; once again, the aim is to discover semantic relationships between tags, starting from information about how users associate them to resources.

A second stream of research has built upon the inferred relationship between tags to develop recommender system algorithms that assist users finding content of relevance within folksonomies. Some approaches have focused on mixed scenarios, where both numerical ratings and tags are available (e.g., [22,16]). Other approaches have been developed to target pure folksonomic settings, where users' preferences can only be inferred by analyzing their tagging activity. For example, [11] first identifies the best recommenders for a target user, based on what tags they have used in common (and how often); such user then receives as recommendations those items that have been most frequently tagged by them. FolkRank [9] uses a PageRank-like algorithm that employs the traditional random surfer model on the tri-partite graph of users-items-tags, producing very accurate recommendations in well connected networks.

One of the main issues left open by state-of-the-art tag-based recommender systems is the *cold-start problem*: when new users join the system, very little is known about their interests, so that predictions about what items they may like cannot be computed. This problem, well-known also in traditional recommender systems, appears to be aggravated in scenarios where likes and dislikes are not expressed as unambiguous numerical ratings, but rather as freely chosen tags. Some researchers have already moved in this direction: for example, [15] proposes to replace the old concept of users' similarity (which is not computable if users have not rated enough items in common), with a new concept of trust; in so doing, users make explicit who their trusted recommenders are. Such approaches are viable only in scenarios where bootstrapping a user's social network comes at no extra cost; moreover, the underpinning assumption that user's trust is a warranty of user's similarity limits the applicability of such approaches.

In the next section we present Clustered Social Ranking (CSR), our approach to tackle the cold-start problem both effectively and efficiently.

3 Clustered Social Ranking

In order to help new users of Web 2.0 websites find content of interest, we have built a technique that leverages upon the following two observations:

- **Leaders and Followers** - the vast majority of content on Web 2.0 websites is created by a rather small proportion of users (*leaders*), while the others mainly browse such content (*followers*). For example, according to an analysis of the CiteULike social bookmarking website, only 45% of the registered users actively posts items on the website, while the remaining 55% simply browse through other users' libraries [4]; this is confirmed by [25], whose analysis shows that more than 70% of the CiteULike users bookmark less than 10 resources overall.
- **Domains of Interest** - users tend to share interests with a rather small group of other users only. In a study of the CiteULike website [25], it was shown that even the most active users bookmark a rather tiny portion of the whole resource set; moreover, they use a rather small subset of the whole folksonomy, which they share with few other users only. This suggests that users have scoped interests that map to a small proportion of the whole social media content.

Clustered Social Ranking (CSR) exploits these observations as follow: rather than considering, as potential recommenders, the whole set of users within the Web 2.0 website, the much smaller set of leaders is considered (first observation). This is aligned with recent studies of more traditional recommender systems, where it was shown that accurate recommendations could be computed by narrowing the set of potential recommenders to the smaller set of users who have engaged the most with the system [1]. These leaders are clustered in domains of interest, based on their past tag activity (second observation). Whenever a user queries the system (be that a search initiated using an explicit set of tags, or a recommendation generated based on the tags used so far), CSR answers it by first identifying the community (or cluster) that can best answer it; it then relies on state-of-the-art tag-based recommender system techniques, applied within the cluster only, to rank content of interest to the user. We describe how leaders are identified and clustered in Section 3.1, while we illustrate how users' queries are dynamically associated to the best cluster to answer them in Section 3.2. Once a query has been associated to a cluster, we rely upon a previously developed algorithm (Social Ranking [25]) to finally compute an answer.

3.1 Clustering of Leaders

Based on the observation that the vast majority of content in Web 2.0 websites is actually produced by a very small proportion of users, Clustered Social Ranking

begins with the identification of so-called *leaders*, that is, users who have engaged with the system substantially more than average. Furthermore, based on the observation that users bookmark a rather tiny portion of the whole resource set, such leaders are *clustered* into domains of interest (i.e., users tagging the same resources, thus exhibiting similar interests). We decided to group users according to tagged content rather than considering the set of tags they used to avoid ambiguities that synonym or homonym tags may introduce. The rationale behind this *clustering of leaders* is that it should be much easier to find what cluster can best answer a query, among a small set (in the order of tens) of domain-focused clusters, rather than searching among tens of thousands of users who the best recommenders are, most especially so if the target user is new to the system. Leadership is simply defined in terms of activity: users who have tagged more resources than the average user within the system are elected.

The literature on clustering algorithms is very rich; we chose to experiment with the Fuzzy *c*-Means algorithm ([2]) because it has the very desirable property of each point having a degree of belonging to a cluster, rather than belonging completely to just one cluster. In our domain, this means that each user can be interested in multiple topics (i.e., belong to multiple clusters). Moreover, Fuzzy *c*-Means has a small computational complexity, which is linear in the number of existing clusters, in the number of items clustered and in the number of iterations performed (the latter being rather small, as we shall demonstrate later). To implement Fuzzy *c*-Means in our target scenario, we modeled each user u_i as a binary vector r_i over resources, where $r_i[j]$ is set equal to 1 if the user u_i has tagged resource p_j. k clusters are initially created, with k chosen following the empirical rule of thumb described in [12], that is, $k \approx (n/2)^{1/2}$, with n being the number of data points (in our case, users) to be clustered; each cluster is represented by a vector (called centroid) c_k, also modeled as a binary vector r_k over resources. The initialisation of such vectors was done by selecting, as centroids, the vectors of k real leaders with non-overlapping resource sets.

After this initialisation phase, Fuzzy *c*-Means performs a series of iterations in which each user is associated to one or more clusters, depending on how well the user is represented by the cluster she is being assigned to. In practice, this degree of belonging is computed as the cosine similarity between the user vector and each centroid vector:

$$sim(u_i, c_k) = cos(r_i, r_k) = \frac{r_i \cdot r_k}{||r_i|| * ||r_k||}$$

After each iteration, these values are normalized and fuzzyfied with a real parameter $m > 1$ so that their sum is 1 for each user; moreover, the centroid of each cluster is updated to be the mean of all users' vectors assigned to it, weighted by their degree of belonging to the cluster. This process is repeated until the algorithm has converged, that is, the change in the degree of belonging between two iterations is no more than a given sensitivity threshold.

Once the clustering of leaders has been completed, we maintain, for each cluster k, the following information:

- *Item Vector*: a vector R_k, where $R_k[j]$ counts how many users within the cluster have tagged resource j.
- *Tag Activity Vector*: a vector TA_k of all distinct tags used by users within k, whereby $TA_k[i]$ counts *how many times* tag t_i has been used to describe resources in R_k.
- *Tag Popularity Vector*: a vector TP_k of all distinct tags used by users within k, whereby $TP_k[i]$ counts *how many distinct users* within k have used tag t_i to bookmark items in R_k.

The above values have all been normalized in a $[0-1]$ range. In the next section, we explain how these vectors are being used to answer users' queries.

3.2 Answering Users' Queries

We use the term *user's query* q to represent both a (proactive) search and a (reactive) recommendation. The former represents the case whereby the user interacting with the system explicitly defines what she is looking for, by means of user-entered tags; the latter represents the case whereby the system recommends items to the user, based on all tags she has used so far. In the following, we do not distinguish between the two cases, and represent a user query q_u as a set of query tags $q_u = \{t_1, \ldots, t_n\}$. In order to answer such query, Clustered Social Ranking performs the following two steps:

(1) Query Association. First, CSR must find what cluster(s) can best answer q_u. To do so, it analysis the user's activity thus far with the system *and* the query tags. If the user has had little interaction with the system (i.e., she has tagged less than l resources, where l is not necessarily the same thresholding value used to define leaders), the query tags drive the association (*tag similarity association*). If the user has been actively engaged with the system instead, CSR further looks into the query tags: if $\{t_1, t_2, \ldots, t_n\}$ have been rarely used by u before (that is, they have been used less than the average tag usage for u), the association is driven once again by query tags; otherwise, it is driven by the user profile (*resource similarity association*). The underpinning idea is that, for users with a long history of interaction with the system, and querying the system within their well defined domain of interest, such history gives more information about what the best cluster is (i.e., who the best recommenders are) to answer a query; however, if the user is not well known to the system (cold-starter), or if indeed she is known, but she is currently looking for content outside her usual domain of interest, then the query tags are more informative of what she is after.

Association of users to cluster is then performed as follow: in the case of *tag similarity association*, we compute the cosine similarity between the query q_u and both the tag activity vector TA_k ($sim_{TA} = cos(q_u, TA_k)$) and the tag popularity vector TP_k ($sim_{TP} = cos(q_u, TP_k)$), for all clusters k. Groups are ranked based on $\max(sim_{TA}, sim_{TP})$, and those with a similarity higher than a given threshold elected to answer the query (in all our experiments, we used a threshold of zero as cosine similarity values in these high-dimensional spaces tend to be very low; we leave the exploration of alternative similarity measures

and thresholds for future work). Note that we use both TA and TP as these vectors provide complimentary information about the cluster: the former indicates how many *different resources* are a potential match for the query; the latter indicates how many *different users* within the cluster speak the same language of the query user (i.e., use the very same tags). In the case of *resource similarity association*, that is, the user is well known and her query tags correspond to her domain of interest, we compute the cosine similarity between her profile r_u (with $r_u[j]$ set equal to 1 if she has tagged resource j) and the item vector R_k (listing what resources have been tagged within cluster k), for all clusters k; as with tag association, groups are ranked based on $cos(r_u, R_k)$, and those with a similarity higher than a given threshold elected to answer the query (once again, in all our experiments, we used a threshold of zero). In both cases, if the similarity is zero towards all clusters, the query is associated to all of them; in practice, this means relying on all leaders to answer the query, regardless of their domain of interest. Note that, as leaders are substantially fewer than users, this is still much lighter than relying on the whole community, as traditional recommender system approaches do. Furthermore, in the experiments reported in the evaluation section, less than 3% of queries required associations to all groups.

(2) Resource Discovery and Ranking. Once the cluster(s) of suitable recommenders have been identified, Social Ranking (SR) [25] is used *within the cluster(s)* to discover and rank resources. In brief, Social Ranking works in two steps: when user u submits a query (be that a search or a recommendation) $q_u = \{t_1, t_2, \ldots, t_n\}$ to discover content that can be described by query tags $\{t_1, t_2, \ldots, t_n\}$, the set of query tags q_u is expanded so to include all $t_i \mid t_i \in q_u$ (for which $sim(t_i, t_i) = 1$), plus all tags $t_{n+1}, \ldots, t_{n+m}$ that are deemed similar to the query tags (for which $0 < sim(t_i, t_j) \leq 1$, with $i \in [1, n]$ and $j \in [n+1, n+m]$). Given tags t_i and t_j, tag similarity is computed as the cosine similarity of the tag vectors w_i and w_j, where $w_i[p]$ counts the number of times that tag t_i was associated to item p. After query expansion, all resources that have been tagged with at least one tag from the extended query set are retrieved. Their ranking depends on a combination of: the relevance of the tags associated to the resource with respect to the query tags (resources tagged with $t_i, i \in [1, n]$ should count more than those tagged with $t_j, j \in [n+1, n+m]$); and, the similarity of the taggers with respect to the querying user u (resources tagged by similar users should be ranked higher, as these users are more likely to share interests with u than others, and thus are in a better position to recommend relevant content). In CSR, rather than considering the similarity between the query user u and each user u_i within the selected cluster(s), we use the similarity computed during association. In so doing, the ranking of resources found *within* a cluster solely depends on the query tags; if the query is associated to more than one cluster, recommendations coming from the closest cluster are ranked higher than those coming from clusters with looser associations instead; to mark the difference further, we magnified the value of the query association by raising it to the power of a positive constant $\alpha > 1$. The rationale for this ranking process is the following: if the query user is a cold starter, or if she is known to the system

but with main interest in a different domain, computing user/user similarity would give meaningless values (in the former case) or misleading values (in the latter); in such case, only the query tags hold meaningful information for the ranking, and we further use the *tag similarity association* to better discriminate between resources coming from different clusters. If the user is well known to the system and she is seeking recommendations within her domain of interest, then user/user similarity should provide the same information given by the *resource similarity association*; we thus prefer the latter as it is cheaper to compute (one similarity computation per cluster instead of one per user within the cluster). The ranking of an item p found within cluster k would thus be computed as:

$$R(p) = \left(\sum_{t_j} sim(t_j, q^*_{u,k}) \right) * (sim_{ASSOC} + 1)^{\alpha}$$

where sim_{ASSOC} is the similarity computed during association between the query user u and the cluster k, and $q^*_{u,k}$ is the set of query tags expanded within k (i.e., using the tag similarity matrix of cluster k).

4 Simulation Setup

Having described the functioning of Clustered Social Ranking, we now describe how we have evaluated it. We begin with a presentation of the simulation setup: we define the metrics we have computed (Section 4.1), illustrate the dataset we have used (Section 4.2), and the benchmarks against which we have compared (Section 4.3). As CSR relies on a number of customisable parameters, we also discuss how these have been set (Section 4.4). We will then analyse the results obtained in Section 5.

4.1 Metrics

To evaluate the effectiveness of our query model, we adopted the standard Precision/Recall metrics computed at different cutting points of the recommendation list. More precisely:

$$Precision = \frac{|relevantContent| \cap |retrievedContent|}{|retrievedContent|}$$

$$Recall = \frac{|relevantContent| \cap |retrievedContent|}{|relevantContent|}$$

The former illustrates how much relevant content is retrieved, out of all content returned to the user; it thus gives a measure of how accurate the approach is. The latter computes how much relevant content is retrieved, out of all relevant content instead; it thus give a measure of coverage. Both metrics have been computed after cutting the final recommendation list at the first 10, 20, 50, 100, 500, 1000 results retrieved.

To evaluate the scalability of our query model, we also analysed the computational complexity it entails, both in terms of online processing (operations computed whenever a query is issued) and offline processing (operations computed in batch mode, when the system is updated, for example, once a week/fortnight/month).

4.2 Dataset

We have conducted experiments using *CiteULike*, a social bookmarking website that aims to promote the sharing of scientific references. CiteULike enables scientists to organize their libraries with freely chosen tags which produce a folksonomy of academic interests. CiteULike runs a daily process which produces a snapshot summary of what articles have been posted by whom and with what tags up to that day. We downloaded one such archive in November 2009, containing bookmarks made between November 2004 to November 2009. We preprocessed the dataset to remove all non-alphabetical and non-numerical tags, following the same methodology proposed by the organisers of the ECML PKDD Discovery Challenge 2009[2]. The so-pruned archive contained 41,246 users, who had tagged 1,254,406 papers overall, using 210,385 distinct tags. To further remove excessive noise in the data, we used the p-Core preprocessing strategy, using the very same approach described in [5]. Based on this technique, both users, resources and tags are iteratively removed from the dataset, in order to produce a smaller but denser subset that guarantees each user, resource and tag to occur in at least p posts/bookmarks. We set $p = 5$; the final dataset contains 2,557 users, 7,480 papers, 3,153 tags, and 59,820 bookmarks.

During the experiments, we ordered the bookmarks according to the original date in which they were published, and we then performed a *temporal* split, so that the first 90% bookmarks have been used for training purposes, while the most recent 10% have been used for testing. We chose a temporal split, rather than a random one, to mimic the actual deployment of a social tagging website. On the training set, Clustered Social Ranking has been executed to pre-compute clusters and the associated information (see Section 3); each test bookmark has then been used as a query: the user who registered the bookmark is treated as the query user, and the tags associated to the bookmark as query tags. This information is given in input to CSR and a list of recommendations thus produced. The previously described metrics (i.e., precision and recall) have then been measured, considering as *relevant* the one resource to which the test bookmark refers to. Note that, because there is only one relevant resource we are after in the recommendation list (whose length has been cut from 10 to 1000), the measured precision is always very small; indeed, what is important is not the absolute precision value, but rather the precision that CSR entails relative to our benchmarks, described next.

[2] http://www.kde.cs.uni-kassel.de/ws/dc09/

4.3 Benchmarks

We have compared the precision/recall that Clustered Social Ranking achieves by comparing them with those of two benchmarks:

1. FolkRank *(FR)* - We have implemented the algorithm proposed by [9], which models the system as a weighted tri-partite graph where nodes refer to users, tags and resources. FR uses a random surfer strategy to recommend resources to users, following the idea that a resource that has been tagged with important tags and by important users becomes important itself. FolkRank is a state-of-the-art algorithm in tag-based recommender systems, whose accuracy has proved to be very high in dense Web 2.0 datasets.
2. Social Ranking *(SR)* - We have been comparing Clustered Social Ranking with the Social Ranking algorithm [25], where tag expansion is conducted by leveraging information from the whole community, and considering as potential recommenders any user of the system. Unlike FR, SR has shown high accuracy in sparse datasets; however, its computational overhead is non negligible in large folksonomies.

4.4 Parameters Tuning

Implementing Clustered Social Ranking requires the setting of a number of parameters. The first parameter refers to the thresholding value used to distinguish leaders from followers. In an actual deployment, this parameter would be set by dynamically studying the average bookmarking activity of users in the system, and by selecting a value above the average so to elect as leaders the smallest set of users who collectively tagged (almost) all resources in the system. In our datasets, we have used two thresholds: the first elects as leaders those users who have tagged more than 10 resources in the training set (shortly called UM10); the second selects as leaders those users tagging more than 30 items instead (shortly called UM30). Table 1 reports how many users are elected as leaders, and how many resources they have collectively bookmarked, with respect to the original dataset. Note that, when using a threshold of 30, less than 20% of users are elected as leaders, while only loosing less than 3% of bookmarked resources. This confirms the fact that a small portion of users is responsible for the vast majority of tagged content in the system; we thus expect that, by restricting our attention to this small set of users, coverage (i.e., recall) should not be hindered.

Dataset	**Num. of Users**	**Num. of Resources**	**Num. of Tags**
CiteULike	2484	7310	3137
CiteULike UM10	1189 (47%)	7291 (99%)	3056 (97%)
CiteULike UM30	432 (17%)	7116 (97%)	2811 (89%)

Table 1. Clusters Features

The second parameter affecting CSR is the number k of clusters. We have adopted the empirical rule of thumb described in [12], whereby $k \approx (n/2)^{1/2}$, with n being the number of data points (in our case, users) to be clustered. When considering leaders those who tagged more than 10 resources (CiteULike UM10), we used 26 clusters, to which we converged after 12 iterations; when clustering leaders who tagged more than 30 resources (CiteULike UM30), we used 14 clusters, to which we converged after just 5 iterations.

We set the remaining parameters required by CSR as follow: query expansion was limited to a maximum of $5 * m$ tags, with m being the number of query tags; upon query association, the minimum number of bookmarks l required for a user not to be considered in the cold start region was set to 10; finally, the α exponent used to mark differences between recommendations coming from clusters of different relevance was set to 5.

5 Results

We now present the results of our evaluation, focusing on effectiveness first (Section 5.1). As our approach is particularly geared towards new users, we present results subdivided in two groups: queries performed by active users, that is, those who have bookmarked at least 10 resources in the training set ($UM10$); and queries performed by new users, who have bookmarked less than 10 resources in the training set ($UL10$). In both cases, we discarded from the test set all queries for which the searched content does not belong to the training set, since none of the implemented algorithms would be able to answer such queries successfully. Of the remaining 4,575 test bookmarks (i.e., our queries), 3,156 were done by active users ($UM10$) and 1,419 by non active ones ($UL10$).

5.1 Evaluation of Effectiveness: Precision and Recall

As shown in Figure 1, Clustered Social Ranking (CSRUM10 and CSRUM30), Social Ranking and FolkRank all achieve very similar precision and recall for active users (with (C)SR being slightly better than FR).

We now turn our attention to new users instead. As Figure 2 illustrates, these users are much more difficult to predict, and even an advanced query engine like FolkRank is not capable of computing valuable recommendations, as too little information is available about these users. However, CSR is capable of exploiting the little information available in the query and about leaders to produce a precision and recall which are comparable to those of SR, and 28% and 40% respectively better than those provided by FR (for recommendation lists of 50 elements). We can thus conclude that, when considering active users, both SR, CSR and FR have very similar performance, both in terms of precision and recall. When considering cold-start users instead, SR and CSR are the most effective techniques, with a neat gain over FR. As we shall discuss next, the computational cost that CSR entails is sensibly lower than both FR and SR, thus making it the most suitable approach overall in scenarios where both effectiveness and efficiency equally matter.

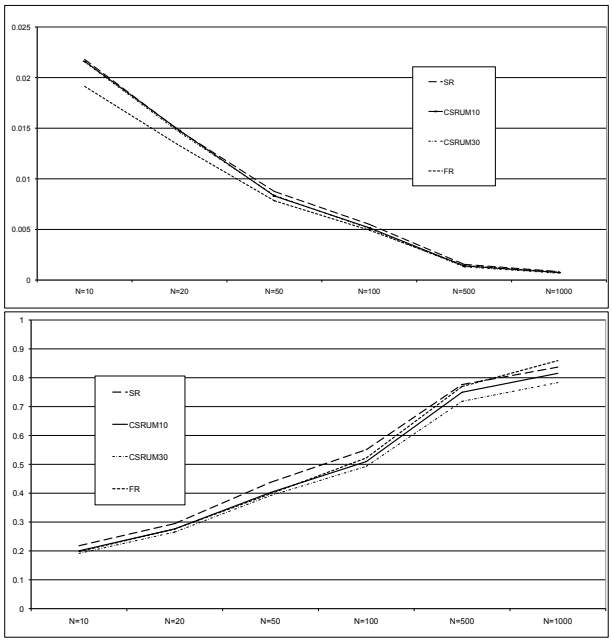

Fig. 1. Precision/Recall for Active Users on CiteULike

5.2 Evaluation of Efficiency: Computational Complexity Analysis

In this section, we analyse the computational complexity that FR, SR and CSR entail, in order to demonstrate that CSR is the most lightweight approach, with a computational cost which is by orders of magnitude lower than that entailed by the other techniques.

In quantifying the computational cost of such approaches, we distinguish between *offline* cost, that is, the cost entailed to pre-compute all the data structures the algorithms rely on (for example, the matrix of users' similarities). Typically, recommender systems recompute these values periodically (e.g., weekly, fortnightly, monthly); there is a tension between accuracy and scalability: the more often the update process is run, the more accurate the recommendations computed, but also the higher the computational cost entailed. For each approach, we will also quantify the *online* computational cost of executing each query. Table 2 reports the computational complexity of each approach.

FolkRank requires no offline pre-computation; rather, it maintains the tripartite graph of users, resources and tags live. Whenever a new query arrives, FR traverses such graph using i iterations (typically 30-35), computes a score for all resources, and derives a recommendation list based on such scores. If we indicate with Y the number of arches in this graph, the online cost entailed by FolkRank is thus $O(i * Y)$.

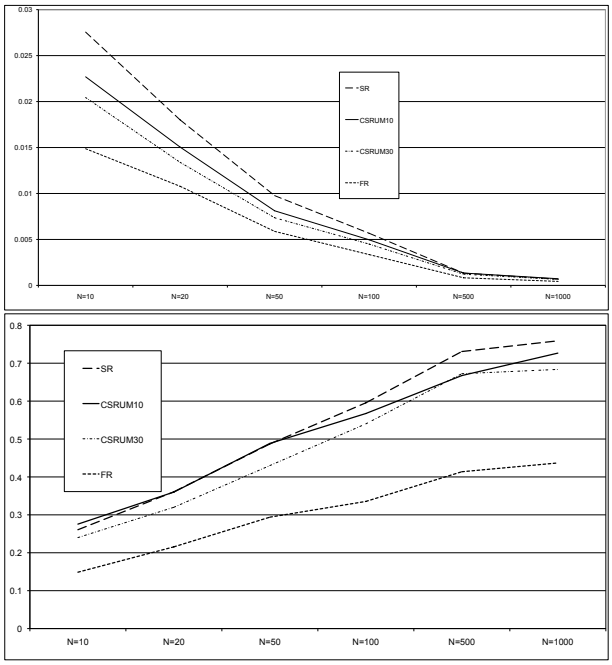

Fig. 2. Precision/Recall for New Users on CiteULike

Approach	**Offline**	**Online ~ (per query)**	**OfflineCUL**	**OnlineCUL (all queries)**
FR	-	$O(i * Y)$	-	23,000 Million
SR	$O\left(\frac{U*(U-1)}{2} + \frac{T*(T-1)}{2}\right)$	$O(R * T)$	8 Million	10 Million
CSR	$O\left(i * k * u + k * \frac{t*(t-1)}{2}\right)$	$O(k * r * t)$	1.5 Million	4 Million

Table 2. Computational Complexity of FR, SR and CSR

Social Ranking requires the offline computation of two matrices: one storing users' similarity, and another storing tags' similarity. These matrices are symmetric, thus its offline cost is $O(U * (U-1)/2 + T * (T-1)/2)$, with U being the number of users in the system and T the number of tags. The online cost depends instead on the number of resources and tags which have to be taken into account in order to answer each user query; in the worst case, all resources R have been tagged with all tags T in the system, thus $O(R * T)$.

Clustered Social Ranking requires two offline computations: the execution of the Fuzzy c-Means Algorithm to cluster leaders, and the computation of the tags' similarity matrices for each cluster. The former is linear in the number of users to cluster u, the number of iterations i required to converge, and the

number of clusters k [10]. The latter is a symmetric matrix, so if we indicate with t the number of tags used within each cluster, the offline cost of CSR is $O(i * k * u + k * t * (t-1)/2)$. The online computational complexity can be estimated as $O(k * r * t)$, in the worst case where the query is associated to all clusters k, within which all resources r have been tagged, with all tags t.

These theoretical limits are upper bounds on the actual complexity of each approach. For example, queries in SR/CSR never require all R/r resources and T/t tags to be answered, nor are they associated to all k clusters. To give a flavour of the actual cost entailed by each approach in a real deployment, we have computed their offline and online cost, when answering all $4,575$ queries from the CiteULike UM30 dataset. The results are reported in the last two columns of Table 2. As shown, the online cost of FolkRank is prohibitive in real deployments, despite the fact that the actual size of the tri-partite graph Y is much smaller (i.e., sparser) than a full one. FR's main disadvantage is that it requires a complete computation of the Page Rank vector for each query, making it unsuitable to work with data from large Folksonomies (as also confirmed by [5]).

Both CSR and SR amortize the cost of pre-computing tags' and users' similarities, affording a much smaller computational cost overall (offline + online); note that, for example, the offline processing of SR could be repeated once every other query in the test set, and still be computationally cheaper than FR (in practice, several thousands queries are normally answered within a single offline update). We now take a closer look at CSR versus SR. The online cost of CSR is half that of SR. More importantly, the offline cost is an order of magnitude smaller; this is because, in practice, $u << U$ (leaders are much fewer than all users), so the cost of clustering them is much smaller than computing users' similarity. For example, in CiteULike UM30, there are only 432 leaders, as opposed to 2,484 users overall: the cost of clustering leaders is only 30K computations (with $k = 14$ clusters and $i = 5$ iterations to converge), while the cost of quantifying users' similarity is 3M. Moreover, each cluster has a much smaller number of tags, so that, even if a separate tags' similarity matrix has to be computed for each cluster, their overall cost (1.5M computations) is much smaller than that entailed by the single matrix maintained by SR (5M computations). The neat reduction in the offline cost of CSR also means that such data structures can be re-computed more often that those used by SR, thus being able to achieve higher accuracy without compromising scalability. Note that frequent updates are of paramount importance in rapidly growing settings and especially for new users, where one update more can make the difference between knowing a little about the user's preferences (her first few bookmarks) or nothing at all.

6 Conclusion

In this paper, we have presented Clustered Social Ranking, a novel search and recommendation technique specifically developed to support new users of Web 2.0 websites finding content of interest. CSR exploits the fact that the vast ma-

jority of content on Web 2.0 websites is created by a small proportion of users, while the others mainly browse such content. CSR first identifies who the leaders are, clusters them into communities with shared interests, and subsequently answers users' queries (be them searches or recommendations) by directing them towards the community of leaders who can best answer them. Our evaluation conducted on the CiteULike website demonstrates that CSR achieves high accuracy, while entailing a low computational cost, thus making it the most suitable solution in these fast growing settings.

Acknowledgement. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under the Grant Agreement n. 234239. The authors are solely responsible for it and it does not represent the opinion of the Community. The Community is not responsible for any use that might be made of information contained therein.

References

1. Xavier Amatrian, Neal Lathia, Josep M.Pujol, Haewoon Kwak, and Nuria Oliver. The wisdom of the few. In *Proc. of SIGIR*, Boston, Massachusetts, 2009.
2. Bezdek J. C. Pattern recognition with fuzzy objective function. Kluwer Academic Publishers Norwell, MA, USA Press, 1981.
3. Andrea Capocci and Guido Caldarelli. Folksonomies and clustering in the collaborative system citeulike. *Journal of Physics A: Mathematical and Theoretical*, 41(22):224016–224023, 2008.
4. Kevin Emamy and Richard Cameron. Citeulike: A researcher's social bookmarking service. 2007.
5. Jonathan Gemmell, Thomas Schimoler, Maryam Ramezani, and Bamshad Mobasher. Adapting K-Nearest Neighbor for Tag Recommendation in Folksonomies. In *Proc. of 7th Intelligent Techniques for Web Personalization and Recommender Systems Workshop*, volume 528, 2009.
6. J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In *Proc of SIGIR*, pages 230–237, Berkley, CA, USA, 1999. ACM Press.
7. Paul Heymann and Hector Garcia-Molina. Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems. Technical Report 2006-10, Stanford University, April 2006.
8. Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Can social bookmarking improve web search? In *Proc. of the International Conference on Web Search and Web Data Mining*, pages 195–206, New York, NY, USA, 2008. ACM.
9. Andreas Hotho, Robert Jaschke, Christoph Schmitz, and Gerd Stumme. Information retrieval in folksonomies: Search and ranking. In York Sure and John Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *LNAI*, pages 411–426, Heidelberg, June 2006. Springer.
10. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, September 1999.
11. Ae-Ttie Ji, Cheol Yeon, Heung-Nam Kim, and Geun-Sik Jo. Collaborative tagging in recommender systems. In *Proc. of Advances in Artificial Intelligence*, pages 377–386, New York, NY, USA, 2007. ACM.

12. Mardia K.V, Kent J.T, and Bibby J.M. Multivariate analysis. Academic Press, 1979.
13. Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. Addressing cold-start problem in recommendation systems. In *Proc. of the 2nd International Conference on Ubiquitous Information Management and Communication*, pages 208–211, New York, NY, USA, 2008. ACM.
14. Cane Wing-ki Leung, Stephen Chi-fai Chan, and Fu-lai Chung. Applying cross-level association rule mining to cold-start recommendations. In *Proc. of the IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, pages 133–136, Washington, DC, USA, 2007. IEEE Computer Society.
15. Paolo Massa and Paolo Avesani. Trust-aware recommender systems. In *Proc. of Conference on Recommender Systems*, pages 17–24, New York, NY, USA, 2007. ACM.
16. Reyn Nakamoto, Shinsuke Nakajima, Jun Miyazaki, and Shunsuke Uemura. Tag-based Contextual Collaborative Filtering. In *18th IEICE Data Engineering Workshop*, 2007.
17. Manos Papagelis, Ioannis Rousidis, Dimitris Plexousakis, and Elas Theoharopoulos. Incremental collaborative filtering for highly-scalable recommendation algorithms. In *Proc. of the 15th International Symposium on Methodologies of Intelligent Systems*, pages 553–561, LNAI 3488. 2005.
18. Alexandre Passant. Using Ontologies to Strengthen Folksonomies and Enrich Information Retrieval in Weblogs. In *Proc. of International Conference on Weblogs and Social Media*, 2007.
19. Han Peng, Xie Bo, Yang Fan, and Sheng Ruimin. A scalable p2p recommender system based on distributed collaborative filtering. *Expert systems with applications*, 2004.
20. A. Schein, A. Popescul, L. Ungar, and D. Pennock. Methods and metrics for cold-start recommendations. In *Proc. of the 25th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 253–260, 2002.
21. Shilad Sen, Shyong K. Lam, Al M. Rashid, Dan Cosley, Dan Frankowski, Jeremy Osterhouse, Maxwell F. Harper, and John Riedl. Tagging, communities, vocabulary, evolution. In *Proc. of the 20th Conference on Computer Supported Cooperative Work*, pages 181–190, New York, NY, USA, 2006. ACM Press.
22. Karen H. L. Tso-Sutter, Leandro Balby Marinho, and Lars Schmidt-Thieme. Tag-aware Recommender Systems by Fusion of Collaborative Filtering Algorithms. In *Proc. of 23rd Annual Symposium on Applied Computing*, pages 16–20. ACM Press, 2008.
23. Xian Wu, Lei Zhang, and Yong Yu. Exploring social annotations for the semantic web. In *Proc. of the 15th International Conference on World Wide Web*, pages 417–426, New York, NY, USA, 2006. ACM Press.
24. Ching Man Au Yeung, Nicholas Gibbins, and Nigel Shadbolt. Mutual Contextualization in Tripartite Graphs of Folksonomies. In *Proc. of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC), Busan, South Korea*, volume 4825 of *LNCS*, pages 960–964, Berlin, Heidelberg, November 2007. Springer Verlag.
25. Valentina Zanardi and Licia Capra. Social ranking: uncovering relevant content using tag-based recommender systems. In *Proc. of the ACM Conference on Recommender systems*, pages 51–58, New York, NY, USA, 2008.