# A Comparative Study of Heterogeneous Item Recommendations in Social Systems

Alejandro Bellogín, Iván Cantador, Pablo Castells
Escuela Politécnica Superior, Universidad Autónoma de Madrid, Spain
{alejandro.bellogin, ivan.cantador, pablo.castells}@uam.es

**Abstract**

While recommendation approaches exploiting different input sources have started to proliferate in the literature, an explicit study of the effect of the combination of heterogeneous inputs is still missing. On the other hand, in this context, there are sides to recommendation quality requiring further characterization and methodological research –a gap that is acknowledged in the field. We present a comparative study on the influence that different types of information available in social systems have on item recommendation. Aiming to identify which sources of user interest evidence –tags, social contacts, and user-item interaction data– are more effective to achieve useful recommendations, and in what aspect, we evaluate a number of content-based, collaborative filtering, and social recommenders on three datasets obtained from Delicious, Last.fm, and MovieLens. Aiming to determine whether and how combining such information sources may enhance over individual recommendation approaches, we extend the common accuracy-oriented evaluation practice with various metrics to measure further recommendation quality dimensions, namely coverage, diversity, novelty, overlap, and relative diversity between ranked item recommendations. We report empiric observations showing that exploiting tagging information by content-based recommenders provides high coverage and novelty, and combining social networking and collaborative filtering information by hybrid recommenders results in high accuracy and diversity. This, along with the fact that recommendation lists from the evaluated approaches had low overlap and relative diversity values between them, gives insights that meta-hybrid recommenders combining the above strategies may provide valuable, balanced item suggestions in terms of performance and non-performance metrics.

**Keywords**

Recommender system, Social Web, collaborative tagging, social network, implicit feedback, evaluation.

## 1. Introduction

The environments on which recommender system technologies are commonly deployed have undergone a remarkable evolution in the last few years in terms of scale, richness, and complexity of the available data. Modern recommender applications do not just have a user-item ratings matrix available, but complex user interaction data, rich item profiles, and large-scale (owned, public, or third-party) resources of many different types. This has been paralleled by a no less remarkable progress in the development of effective recommendation algorithms, and an evolution in the understanding of the role of recommendation functionalities in different application domains. The availability and convergence of technologies and resources in social systems –rich item databases, personal user data, user interaction records, user-contributed content, social networks, geospatial information, and so forth– have transformed the context in which the recommendation problem is addressed, multiplying the opportunities for enhanced solutions.

It has been made clear in recent years that a single algorithm is generally insufficient to optimize the effectiveness of recommendations –the Netflix contest[1] is a paradigmatic example of the superiority of hybrid recommenders over stand-alone approaches [28]. Likewise, recommendation based on a single source of input data is generally suboptimal, inasmuch as the multiplicity of available hints for good recommendations are missed. At the same time, the purpose and scenarios for recommendation are diverse, and consequently, a single view of recommendation quality becomes insufficient to assess the value and usefulness of a recommendation approach. Evaluation methodologies and metrics need to be extended for this purpose, which is currently an open area of research and development in the field.

While recommendation approaches exploiting different input sources have started to proliferate in the literature, an explicit study of the effect of the combination of heterogeneous sources is still missing. In fact, there is little reported evidence yet on the comparative effectiveness enabled by different types of

[1] Netflix prize, http://www.netflixprize.com/

input data, when used individually, and the additional gain that can be leveraged from their combined effect. Furthermore, such a study requires an extension of the traditional evaluation dimensions and metrics in order to properly capture the effects of such combinations in their different, relevant angles –an area where a methodological gap has been identified in the field [42]. The recommender systems literature has indeed been strongly focused on recommendation accuracy (to be more precise, on the accuracy in rating value prediction) as the main –generally the only– quality criteria. Aspects such as the coverage, diversity and novelty of recommended items, which may be critical in practice for the target users of recommendation –or the business beneath–, have been barely addressed yet in the literature by a sound body of shared methodologies and metrics [1]. While progress in accuracy optimization seems to have somewhat peaked in the field [28], and is getting circumscribed to small increments, we see considerable room for progress in such, to a large extent, unexplored dimensions. The combination of sources and the extension of evaluation methodologies thus present themselves as two interrelated research goals, where we see in the former a direction for improvement in terms of the latter, and the latter is a necessity to evaluate the former.

Motivated by the above considerations, in this paper, we address the following research questions:

- **RQ1.** *Which available sources of information in social systems are more effective for recommendation?*

  We study this question in terms of several performance metrics borrowed from the Information Retrieval field, for recommendation approaches that exploit different sources of information, namely ratings, tags and social contacts.

- **RQ2.** *Do recommendation approaches exploiting different sources of information in social systems offer heterogeneous item suggestions, from which hybrid strategies may gain additional benefits?*

  We address this question by considering several recommendation quality metrics beyond accuracy, measuring such dimensions as coverage, diversity, novelty and overlap, on the recommendation approaches studied in RQ1.

In order to support this study, we have implemented a set of generic content-based filtering, collaborative filtering, and social recommendation approaches for social systems, and have built three datasets with information of different types obtained from Delicious[2], Last.fm[3] and MovieLens[4]. By using these recommenders and datasets, we conduct a twofold experiment. First, we compare the performance of the recommenders with ranking quality metrics from the Information Retrieval field, namely precision, recall and nDCG. Second, we compare additional characteristics of the recommenders with a number of novel metrics that measure coverage, diversity, novelty and overlap of and between ranked lists of recommended items.

The reminder of the paper is organised as follows. Section 2 describes relevant works related to our study. Section 3 presents the evaluated content-based, collaborative filtering, and social recommendation approaches. Section 4 explains the experimental setup of the study, describing the utilised datasets, the proposed performance and non-performance metrics, and the followed evaluation protocol. Section 5 discusses the results obtained in the conducted experiment, and finally, Section 6 depicts some conclusions and future research lines.

## 2. Related Work

With the advent of the Social Web, a variety of new recommendation approaches have been proposed in the literature. Most of these approaches are based on the exploitation of social tagging information and explicit friendship relations between users.

In **social tagging** systems, such as Delicious, Flickr and Last.fm, users annotate/tag resources (Web pages, photos, music tracks, etc.) for the purpose of personal multimedia content management, browsing and search. Interestingly, these personalisation functionalities can be extended to collaborative recommendation functionalities when the whole set of annotations [user-tag-resource] (known as *folksonomy*) are taken into account. A user's preferences are described in terms of her tags and tagged resources. Based on such a profile model, similarities with other users can be found, and item recommendations can be produced. Hotho and colleagues [24] present FolkRank, a PageRank-like

[2] Delicious, Social bookmarking, http://delicious.com
[3] Last.fm, Internet radio and music catalogue, http://www.last.fm
[4] MovieLens, Movie recommendations, http://movielens.umn.edu

algorithm applied to the tripartite graph formed by nodes associated to users, tags and items of a folksonomy, and weighted edges related to co-occurrences between users and tags, items and tags, and users and items. Other approaches like those proposed by Niwa and colleagues [36], and Shepitsen and colleagues [43], attempt to cluster the tag space, aiming to minimise information redundancy and contextualise item recommendations. Zanardi and Capra [49] investigate an alternative approach that provides item recommendations in a content-based collaborative filtering fashion. In this paper, we evaluate a number of tag-based recommendation approaches [13] that are adaptations of TF-IDF [3] and BM25 [44] Information Retrieval models, and are inspired on previous works on folksonomy-based personalised Web search presented by Noll and Meinel [37], and Xu and colleagues [48].

Apart from social tagging, Social Web systems usually provide **social networking** functionalities. In these systems, users explicitly state friendship[5] relations with other users. The use of this explicit social information has recently started to receive attention in the Recommender Systems field [21], and is currently an active open research direction. Thus, for instance, Ben-Shimon and colleagues [10] present a collaborative filtering strategy that estimates the rating of an item for a user based on the item ratings provided by the user's friends. He and Chu [22], on the other hand, exploit the user's friends' ratings in a probabilistic recommendation model.

As suggested by Bonhard and Sasse [11], we believe that recommender systems can be enhanced by combining relevant information that can be drawn from social network analysis, such as explicit networks of trust, with the matching capabilities of content-based and collaborative filtering recommendation strategies. In this line, the final goal of our research is to investigate effective hybrid recommendation strategies that adaptively merge and exploit the heterogeneous information available in social systems.

**Hybrid recommendation** approaches that combine different sources of social information, especially social tags and contacts, have already been proposed. Konstas and colleagues [27] investigate the application of a Random Walk based algorithm on graphs where the user, tag and item spaces are intra- and inter-linked. Musial [35] studies recommendation methods enhanced with social features of the networks and their members. Pilászky and Tikk [39] compare how effective content-based methods are in predicting ratings on new movies, using movie-metadata, against collaborative filtering and other simple rating-based predictors. Sen and colleagues [40] present an empiric comparison of a large number of recommenders that estimate item ratings by exploiting user tags, ratings and click-through data. Gemmell and colleagues [19] build a hybrid recommender which linearly combines the predictions given by three models, namely, the resource-based popularity, user-based popularity, and an item-based collaborative filtering algorithm. Finally, Seth and Zhang [41] propose a Bayesian model-based recommender that leverages content and social data.

Along with this research on hybrid social recommendation approaches, to our knowledge, there are no rigorous studies yet about how and to which degree each of the available sources of information in social systems is valuable for effective item recommendations. We address this issue here with a broad perspective, not restricting our empirical study to an evaluation of recommenders in terms of performance metrics such as precision and recall only, but also considering a further variety of metrics that aim to capture non-performance measures of recommendation usefulness, such as coverage, diversity, novelty and overlap of recommendations.

## 3. Evaluated Recommenders

Adomavicius and Tuzhilin [2] formulate the recommendation problem as follows. Let $\mathcal{U}$ be a set of users, and let $\mathcal{I}$ be a set of items. Let $g: \mathcal{U} \times \mathcal{I} \rightarrow \mathrm{R}$, where R is a totally ordered set, be a utility function such that $g(u,i)$ measures the gain of usefulness of item $i$ to user $u$. Then, for each user $u \in \mathcal{U}$, we aim at choosing items $i^{\max,\mathrm{u}} \in \mathcal{I}$, unknown to the user, which maximise the utility function $g$:

$$\forall u \in \mathcal{U}\,,\ i^{\max,u} = \arg\max_{i \in \mathcal{I}} g(u,i)$$

Depending on the exploited source of information, and the way in which the utility function $g$ is estimated for different users, the following two main types of recommender systems are commonly distinguished: 1) *content-based recommender systems*, in which a user is recommended items similar to those he preferred in the past, and, 2) *collaborative filtering systems*, in which a user is recommended items that people with similar tastes and preferences liked in the past. We extend this classification by considering *social recommender systems*, i.e., systems in which a user is recommended items that (explicit) friends liked in the past, as a case related but significantly different to collaborative filtering.

With the above formulation, in the next subsections, we present the content-based, collaborative filtering, and social recommenders for Social Web systems used in the empirical study presented herein.

[5] There are social networking sites that utilise other types of social relations, like e.g. fans, followers and professional colleagues

## 3.1 Content-based Recommenders

Many Social Web systems enable users to create or upload content (items), annotate it with freely chosen words (tags), and share it with other users. The whole set of tags constitutes an unstructured collaborative classification scheme that is commonly known as *folksonomy*. More formally, a folksonomy $\mathcal{F}$ can be defined as a tuple $\mathcal{F} = \{\mathcal{U}, \mathcal{T}, \mathcal{I}, \mathcal{A}\}$, where $\mathcal{T}$ is the set of tags that comprise the vocabulary expressed by the folksonomy, $\mathcal{U}$ and $\mathcal{I}$ are respectively the set of users and the set of items that annotate and are annotated with the tags of $\mathcal{T}$, and $\mathcal{A} = \{(u,t,i)\} \in \mathcal{U} \times \mathcal{T} \times \mathcal{I}$ is the set of assignments (annotations) of each tag $t$ to an item $i$ by a user $u$.

A folksonomy represents an implicit classification that serves various purposes, such as for resource organisation, promotions, sharing with friends, with the public, etc. Studies have shown, however, that tags are generally chosen by users to reflect their interests. Golder and Huberman [20] analysed tags on Delicious, and found that (1) the overwhelming majority of tags identify the topics of the tagged resources, and (2) almost all tags are added for personal use, rather than for the benefit of the community. These findings lend support to the idea of using tags to derive precise user profiles and resource descriptions. In Last.fm, the set of most popular tags[6] is clearly associated to music genres, showing that social tags really describe user music interests and track music styles.

In this section, we present a number of content-based filtering (CBF) recommendation approaches that exploit tagging information available in Social Web systems. These approaches, preliminary evaluated in [13], are based on user and item profiles defined in terms of lists (vectors) of weighted tags, and compute similarities between such vectors to provide personal recommendations.

We define the profile of user $u$ as a vector $\mathbf{u} = (u_1, \ldots, u_L)$, where $u_t$ is a weight (real number) that measures the "informativeness" of tag $t$ to characterise contents annotated by $u$. Similarly, we define the profile of item $i$ as a vector $\mathbf{i} = (i_1, \ldots, i_L)$, where $i_t$ is a weight that measures the relevance of tag $t$ to describe $i$. There exist different schemes to weight the components of tag-based user and items profiles. Some of them are based on the information available in individual profiles, while others draw information from the whole folksonomy. The simplest approach for assigning a weight to a particular tag in a user or item profile is by counting the number of times such tag has been used by the user or the number of times the tag has been used by the community to annotate the item. Thus, our first profile model for user $u$ consists of a vector $\mathbf{u} = (u_1, \ldots, u_L)$, where

$$u_t = tf_u(t),$$

$tf_u(t)$ being the tag frequency, i.e., the number of times user $u$ has annotated items with tag $t$. Similarly, the profile of item $i$ is defined as a vector $\mathbf{i} = (i_1, \ldots, i_L)$, where

$$i_t = tf_i(t),$$

$tf_i(t)$ being the number of times item $i$ has been annotated with tag $t$.

In an information retrieval environment, common keywords that appear in many documents of a collection are not informative, and are generally not helpful to distinguish relevant documents for a given query. To take this into account, the TF-IDF weighting scheme is usually applied to the document profiles [3]. We adopt that principle, and adapt it to social tagging systems, proposing a second profile model, defined as:

$$u_t = tfiuf_u(t) = tf_u(t) \cdot iuf(t),$$
$$i_t = tfiif_i(t) = tf_i(t) \cdot iif(t)$$

where $iuf(t)$ and $iif(t)$ are inverse frequency factors that penalise tags that frequently appear (and thus are not informative) in tag-based user and item profiles respectively. More specifically, $iuf(t) = \log(M/n_u(t)), n_u(t) = |\{u \in \mathcal{U} | u_t > 0\}|$, and $iif(t) = \log(N/n_i(t)), n_i(t) = |\{i \in \mathcal{I} | i_t > 0\}|, M$ and $N$ being the number of users and items respectively. Note that we incorporate both user and item tag distribution global importance factors, $iuf$ and $iif$, following the vector space model principle that as more rare a tag is, the more important it is for describing either a user's interests or an item's content.

As an alternative to TF-IDF, the Okapi BM25 weighting scheme follows a probabilistic approach to assign a document with a ranking score given a query [44]. We propose an adaptation of such model by assigning each tag with a score (weight) given a certain user or item. Our third profile model has the following expressions:

[6] Last.fm most popular tags, http://www.lastfm.es/charts/toptags

$$u_t = bm25_u(t) = \frac{tf_u(t)\cdot(k_1+1)}{tf_u(t)+k_1\left(1-b+b\cdot{|u|}/{avg(|u|)}\right)}\cdot iuf(t),$$

$$i_t = bm25_i(t) = \frac{tf_i(t)\cdot(k_1+1)}{tf_i(t)+k_1\left(1-b+b\cdot{|i|}/{avg(|i|)}\right)}\cdot iif(t),$$

where $b$ and $k_1$ are set to the standard values 0.75 and 2, respectively.

#### *3.1.1 TF-based Recommender*

To compute the preference of a user for an item, Noll and Meinel [37] propose a personalised similarity measure based on the user's tag frequencies. The model utilises the user's usage of tags appearing in the item profile, but does not take into account their weights in such profile. We have introduced a slight variation in the above formula with respect to its original definition, namely a normalisation factor that scales the utility function to values in the range [0, 1], without altering the user's item ranking:

$$g(u,i) = tf(u) = \frac{\sum_{t:i_t>0} tf_u(t)}{\max_{v\in\mathcal{U},t\in\mathcal{T}}\left(tf_v(t)\right)}$$

#### *3.1.2 BM25-based Recommender*

Analogously to the similarity based on tag frequencies described in Section 3.1.1, but using a BM25 weighting scheme, we propose a similarity function that only takes into account the weights of the user profile. This recommendation model is defined as follows:

$$g(u,i) = bm25(u) = \sum_{(t|i_t>0)} bm25_u(t)$$

#### *3.1.3 TF-IDF Cosine-based Recommender*

Xu and colleagues [48] use the cosine similarity measure to compute the similarity between user and item profiles. As profile component weighting scheme, they use TF-IDF[7]. We adapt their approach with the proposed tag-based profile models as follows:

$$g(u,i) = cos_{tf\text{-}idf}(\mathbf{u},\mathbf{i}) = \frac{\sum_t tf_u(t)\cdot iuf(t)\cdot tf_i(t)\cdot iif(t)}{\sqrt{\sum_t\left(tf_u(t)\cdot iuf(t)\right)^2}\cdot\sqrt{\sum_t\left(tf_i(t)\cdot iif(t)\right)^2}}$$

where the numerator is the dot product of the tf-iuf and tf-idf vectors associated to the user and item, respectively. The denominator is the user and item profile length normalisation factors, calculated as the magnitude value of those vectors.

#### *3.1.4 BM25 Cosine-based Recommender*

Xu and colleagues [48] also investigate the cosine similarity measure with a BM25 weighting scheme. They use that model on personalised Web Search. We adapt and define it for social tagging as follows:

$$g(u,i) = cos_{bm25}(\mathbf{u},\mathbf{i}) = \frac{\sum_t\left(bm25_u(t)\cdot bm25_i(t)\right)}{\sqrt{\sum_t\left(bm25_u(t)\right)^2}\cdot\sqrt{\sum_t\left(bm25_i(t)\right)^2}}$$

### 3.2 Collaborative Filtering Recommenders

Collaborative filtering (CF) techniques match people with similar preferences, or items with similar choice patterns by users, in order to make recommendations. Unlike CBF methods, CF systems aim to predict the utility of items for a particular user according to the items previously evaluated by other users.

In general, CF is based on explicit numeric ratings, that is, the real utility of an item for a particular user is represented by the rating given by that user to the item. There are systems, however, where no explicit ratings are available, but where user interests can be inferred from implicit feedback information. In order to provide item recommendations in such systems, two plausible options exist: use recommenders that directly exploit implicit data [18][25][38][47], or transform implicit data into explicit ratings to apply standard CF algorithms [3][10][15][30].

[7] Xu et al. do not specify if they take user-based or item-based inverse tag frequencies, or both. We chose to use both, since this configuration gave the best performance values.

As mentioned before and explained in Section 4, we have conducted experiments with three datasets, obtained from Delicious, Last.fm and MovieLens systems. In Last.fm, there are no explicit ratings, but user activity data logs in the form (*user*, *item*, *freq*), where *item* is a music track listened by *user*, and *freq* represents the number of times *item* was listened by *user*. Aiming to transform these tuples into numeric ratings, we follow the approach presented by Baltrunas and Amatriain [5], which is based on Celma's studies [15]. This approach consists of taking into account the number of times each user has listened to an artist (or track), in such a way that the artists (tracks) located in the 80-100% interquintile range of the user's listening distribution receive a rating of 5 (in a five point scale), the next interquintile range is mapped to a rating of 4, and so on.

This technique was originally developed for taking into account the distributions of the artists in a user profile. We have adapted it to use it with music tracks, whose distributions are less skewed due to the larger sparsity at track level than at artist level. Therefore, in our adaptation, we have modified the model to deal with sparse profiles. In the future, we want to investigate alternative techniques in order to improve the performance of the generated predictions. We also plan to analyse artist-level transformations followed by different strategies for inferring track ratings, in contrast to what is done in [15], whose aim is at recommending artists. Besides, we also want to explore other systems with explicit ratings (such as Flixster[8]), which would not require the use of this type of approaches.

In Delicious, there are no explicit ratings, nor frequency of item consumption, since each URL can only be bookmarked once by a particular user. Therefore, in this case, we consider a binary transformation of the data, which is a typical technique with implicit data [3], by assigning the same rating for all the items present in the user profile. Finally, in MovieLens, users explicitly assign 1-5 scale ratings to items, so we do not perform any transformation on the user profiles.

In the following subsections, we briefly describe the CF algorithms evaluated in our experiments.

### *3.2.1 User-based CF Recommender*

User-based CF techniques compare the target user's choices with those of other users to identify a group of "similar-minded" people (usually called *neighbours*). Once this group has been identified, those items chosen or highly rated by the group are recommended to the target user. More specifically, the utility gain function $g(u,i)$ is estimated as follows:

$$g(u,i) = \mathrm{C} \sum_{v \in N[u,k]} sim(u,v) \times rat(v,i)$$

where C is a normalisation factor, $rat(v,i)$ is the rating given by user $v$ to item $i$, and $N[u,k]$ denotes the set (with size $k$) of neighbours of $u$. Similarity between users can be calculated by using different metrics: Pearson and Spearman's correlations, cosine-based distance, among others [2]. In this paper, we use Pearson's correlation, which is defined as:

$$sim(u,v) = \frac{\sum_i \big(rat(u,i) - \overline{rat}(u)\big)\big(rat(v,i) - \overline{rat}(v)\big)}{\sqrt{\sum_i \big(rat(u,i) - \overline{rat}(u)\big)^2}\sqrt{\sum_i \big(rat(v,i) - \overline{rat}(v)\big)^2}}$$

where $\overline{rat}(u)$ is the average of the ratings provided by user $u$.

### *3.2.2 Item-based CF Recommender*

Like user-based approaches, item-based CF techniques recognise patterns. However, instead of identifying patterns of similarity between user choices, they recognise patterns of similarity between the items themselves. In general terms, item-based CF looks at each item on the target user's list of chosen/rated items, and finds other items that seem to be "similar" to that item. The item similarity is usually defined in terms of correlations of ratings between users [2]. More formally, the utility gain function $g(u,i)$ is estimated as follows:

$$g(u,i) = \mathrm{C} \sum_{j \in \mathcal{I}_u} sim(i,j) \times rat(u,j)$$

where $\mathcal{I}_u$ is the set of items rated by user $u$. In this paper, we use Pearson's correlation to calculate item similarities.

[8] Flixster, Social movie recommendations, http://www.flixster.com

## 3.3 Social Recommenders

The third and last information source exploited in this paper is the social information, such as contacts and interactions between users. In the literature, there are recommenders that explicitly deal with social information; we implemented and included some of them in our study. Complementarily, we also consider some algorithms which are based on explicit trust relations, since social contacts (i.e., friendship) can be seen as a type of trust relation between users.

In order to avoid any possible lack of information from the social side because of sparsity, we are interested in combining this type of recommenders with collaborative ones, and because of that we present some examples of such hybrid recommenders in subsections 3.3.2 and 3.3.5. In particular, these algorithms combine information from ratings (similarity functions or neighbours) and social contacts. We are aware of other algorithms, which also fall into the hybrid recommender category, exploiting social information along with tags or ratings, such as FolkRank [24] and Random Walks [27]. We plan to consider this type of recommenders in the future.

### *3.3.1 Friend-based Social Recommender*

Inspired on the approach presented by Liu and Lee [31], we propose a recommender that incorporates social information into the user-based CF model. In this case, it utilises the same formula as the user-based CF technique (Section 3.2.1), but replacing the set of nearest neighbours by the active user's (explicit) friends. That is:

$$N[u,k] = N[u] = \{v \in \mathcal{U}: v \text{ is friend of } u\}$$

With this recommender, we easily incorporate social information into the well-known CF prediction equation, building a straightforward technique that enables a direct interpretation of the suggestions, namely those items recommended by friends.

### *3.3.2 Friend and Neighbour-based Social Recommender*

Our second social recommender also utilises the user-based CF formula, but is based on all the active user's friends, as well as her most similar nearest neighbours, combining them into a new neighbour set:

$$N[u,k] = \{v \in \mathcal{U}: v \text{ is friend of } u\} \cup \{v \in \mathcal{U}: sim(u,v) \geq \rho_u\}$$

where $\rho_u > 0$ is the minimum similarity to be satisfied between the active user and her most similar neighbours. From the formula, it can be seen that this recommender is actually a hybrid one, where collaborative filtering and social information are used.

### *3.3.3 Personal Social Recommender*

The approach of [10] explicitly introduces distances between users in the social graph in the scoring formula:

$$s(u,i) = \sum_{v \in X(u,L)} K^{-d(u,v)} rat(v,i)$$

In this equation, $X(u,L)$ denotes the social tree of user $u$ up to level $L$, and $K$ is an attenuation coefficient of the social network which determines the extent of the effect of $d(u,v)$, that is, the impact of the distance between two users in the social graph (for instance, using Dijkstra's algorithm). For example, when $K = 1$ the impact is constant and the resulting ranking is sorted by the popularity of the items.

We use this recommender to obtain raw scores in order to generate item rankings, since these scores are not in the range of ratings, but lie in the interval [0, 1].

### *3.3.4 Popularity based Recommender*

Recently, in [6], the authors propose a recommender in which the items suggested to a user are the most popular among her set of similar users. The authors use a binary matrix model as input data, and pick those items having the maximum number of 1's amongst the top active user's neighbours, according to some similarity measure (e.g. Pearson's correlation), breaking ties randomly. In this paper, we extend the above algorithm by considering the friends of each user instead of her more similar neighbours. Thus, we propose a friends' popularity recommender that suggests the active user those items more popular among her set of friends. We generate a score by transforming the item position with the following equation, once a ranking has been generated as described above:

$$s(u,i;N) = 1 - \frac{pos(u,i)}{N}$$

where $pos(u,i)$ represents the position of item $i$ in the top-$N$ recommended list for user $u$. We may trim the returned list at some level $N$, or assume $N$ to be exactly the length of the generated recommendation list. Note that, like in the previous recommender, the computed scores cannot be interpreted as ratings.

#### *3.3.5 Trust-based Social Recommender*

An alternative way of introducing social information into a recommender system is by the so called trust-based recommendation approach. Trust-aware recommenders make use of trust networks, in which users express a level of trust in other users [33]. In our approach, since we do not have a real trust network, we have to infer a plausible network from the information we already know about users, i.e., social information. First of all, social contacts among users can only provide positive relations or trust levels. Moreover, they generate constant trust levels, since no distinction is made among a user's contacts. This is why we propose to spread trust in our social network uniformly across each user's contacts. For example, a user with 4 friends would have a level of trust with respect to each one of 0.25, whereas a user with 2 friends would have 0.5.

Once the trust network is defined, a trust metric is required so that trustworthiness of every user can be predicted. We have used two metrics described in [33]: PageRank and MoleTrust. The former is considered as a global metric, since it computes a global reputation value for each user; the latter is considered as a local metric by computing a trust score of a source user on a target user, and is based on a depth-first graph walking algorithm with an adjustable trust propagation horizon.

Finally, we have experimented with two additional mechanisms to incorporate trust metrics into the recommendation models, as proposed in [33]. The first one makes use of the trust metric instead of the similarity metric in the standard user-based CF formula. The second technique, on the other hand, computes the average between Pearson's similarity and trust metric when both values are available; otherwise it uses the only available value, overcoming thus the natural data sparsity.

## 4. Experimental Setup

### 4.1 Datasets

In order to evaluate the presented content-based filtering, collaborative filtering, and social recommendation models, we need datasets rich in collaborative tagging, item rating/consumption, and social networking information. It is difficult to find systems allowing access to all these types of information together. Moreover, depending on the nature of a system, it may be difficult to find users with enough information of each type. Analysing representative social systems, we identified that Last.fm can satisfy our needs, and built a heterogeneous dataset from it. Furthermore, Delicious, despite its lack of explicit rating information, and MovieLens, which has no social network, were also considered as candidates. Consequently we built three datasets[9] from the above systems [14]. In the next subsections, we describe the datasets and the process followed to obtain them. Table 1 shows the main characteristics of each dataset. The numbers in bold correspond to characteristics that are representative of the user and item profiles in the studied systems, in terms of tag assignments, ratings and social contacts. In the next subsections, we shall focus on the type and size of such profiles to provide a comprehensive description of the nature of the systems. In Section 5, we shall analyse how the characteristics influence the values of computed performance and non-performance metrics for the different types of recommendation strategies –content-based, collaborative filtering, and social.

[9] We made the datasets publicly available at the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011, http://ir.ii.uam.es/hetrec2011). The datasets are hosted by the GroupLens research group at University of Minnesota (http://www.grouplens.org).

| | Last.fm | Delicious | MovieLens |
|---|---|---|---|
| Users | 1,892 | 1,867 | 2,113 |
| Items | 17,632 | 69,226 | 10,133 |
| Tags | 11,946 | 53,388 | 13,222 |
| Tags per user (avg.) | 21.92 | 123.74 | 10.09 |
| Tags per item (avg.) | 33.57 | 5.93 | 6.35 |
| Tag assignments | 186,479 | 437,593 | 47,957 |
| Tag assignments per user (avg.) | **98.56** | **234.38** | **22.70** |
| Tag assignments per item (avg.) | **14.89** | **6.32** | **8.12** |
| Tagged items | 16,961 | 69,226 | 5,909 |
| Tagged items per user (avg.) | 37.56 | 56.13 | 13.12 |
| Ratings* | 92,834 | 104,833 | 855,598 |
| Rated items | 17,632 | 69,226 | 10,109 |
| Ratings per user (avg.) | **50.00** | **56.13** | **404.92** |
| Ratings per item (avg.) | **5.26** | **1.51** | **84.64** |
| Friend relations | 25,434 | 15,328 | N/A |
| Friend relations per user (avg.) | **13.44** | **8.24** | **N/A** |

**Table 1**. Description of the built datasets. * In Last.fm, we consider artist listening records (i.e., play counts) as implicit ratings. Similarly, in Delicious, we consider bookmarks as implicit binary ratings. In MovieLens, on the other hand, ratings are explicitly provided by users.

### *4.1.1 Last.fm dataset*

Last.fm is a social music website. As of March of 2009, the site had more than 40 million active users in more than 190 countries[10]. Several authors have analysed or used this system for research purposes; special mention deserves those who have made their datasets public, such as [27] and [15]. To the best of our knowledge, at the time of writing, none of the publicly available Last.fm datasets have all the three sources of user preference information we need.

We built our dataset aiming to obtain a representative set of users, covering all music genres, and forming a dense social network. Thus, we first identified the most popular tags related to the music genres in Last.fm. Then, we used the Last.fm API to get the top music artists tagged with the previous tags. For each artist, we gathered her fans along with their direct friends. Finally, we retrieved all tags and tagged artists of the user profiles. Filtered out those users without listened/tagged artists and friend relations within the obtained social network, the final dataset contains 1.9K users, 17.6K artists (17.0K of them tagged), 186.5K tag assignments (98.6 per user), and 25.4K friend relations (13.4 per user). According to these statistics, we can observe that Last.fm users mostly use the system to listen to music. Comparing Last.fm and Delicious, we show that in the former, users tag less but have more contacts than in the latter. These issues, among others, will help to understand the differences between performance and non-performance values of the studied recommendation approaches in the systems.

### *4.1.2 Delicious dataset*

Delicious is a social bookmarking site for Web pages. As of November of 2008, delicious had 5.3 million users[11]. With over 180 million unique URLs, Delicious can be considered a fairly accurate "people's view" of the Web. This vast amount of user information has been previously successfully exploited to improve Web search, and provide personal recommendations and search results, among others [13][24][37][43][45].

We built our dataset with the same goal in mind as stated for Last.fm dataset: to cover a broad range of document's topics, and obtain a dense social network. We first obtained the most popular tags in Delicious. Then, we downloaded the bookmarks tagged with those tags, and for each bookmark, we obtained the users who tagged it. For this set of users, we downloaded their contacts (*friends* and *fans* as considered by Delicious), and their contacts' contacts. After filtering out relations with users that did not belong to the final user set, we downloaded the top bookmarks and tags of each user, along with the community tags of each bookmark. Finally, we removed those users with less than 20 user contacts in the

[10] http://blog.last.fm/2009/03/24/lastfm-radio-announcement
[11] http://en.wikipedia.org/wiki/Delicious_(website)

user set, updating the social network by removing relations with contacts that no longer belonged to the final user set, and users who had no relations. The previous threshold (20) was established analysing the user contact histogram, so as to avoid the long tail users. As shown in Table 1, the final dataset contains 1.9K users, 69.2K bookmarked Web pages, 437.6K tag assignments, and 15.3K friend relations. On average, each user profile has 56.1 bookmarks, 234.4 tag assignments, and 8.2 friends. From these statistics, as the reader may expect, we can observe that the nature of Delicious is bookmarking/tagging objects, and little interest is given by users to making contacts. In our experiments, we shall analyse how the sparsity of Delicious social network affect the performance of social recommendation approaches, and compare it against that obtained from tag-based recommenders.

#### *4.1.3 MovieLens dataset*

The MovieLens dataset, published by the GroupLens research group at University of Minnesota, is one the most referenced and evaluated repositories in the Recommender Systems community. Its larger public version, called MovieLens10M, consists of approximately 10 Million ratings on a 1-5 rating scale and 95.6K tags applied to 10.7K movies by 71.6K users.

From that repository we created a smallest dataset maintaining only those users with both rating and tagging information. In order to enrich the dataset, we linked its movies with their corresponding Web pages in IMDb[12] and RottenTomatoes[13] systems, and extracted additional information, such as movie directors, cast members, genres, shooting locations, countries, languages, photos, and experts' ratings and scores, among others. We believe this information can be used by researchers and practitioners to investigate further more complex recommendation approaches. Our dataset finally contains 2.1K users, 10.1K movies, 48.0K tag assignments (22.7 per user), and 855.6K ratings (404.9 per user). Based on the large amount of rating information in the dataset, we expect collaborative filtering approaches will get the best performance values. On the contrary, since the number of tag assignments (per user) in MovieLens dataset is much lower than in Last.fm and Delicious datasets, one could expect content-based approaches will perform worse. However, we have to check the above hypothesis empirically because there are other aspects, such as the number of items in the dataset, which may influence the final results drastically.

### 4.2 Evaluation Protocol

In this section, we describe the methodology followed to evaluate the recommendation approaches. For each of the three presented datasets, we randomly split the set of items tagged and consumed (listened, bookmarked, rated) by the users into two subsets. The first subset contained 80% of the items for each user, and was used to build (train) the recommenders. The second subset contained the remaining 20% of the items for each user, and was considered as ground truth data to evaluate (test) the recommenders, which had to predict the relevance of such items for the different users. From the relevance predictions, as we shall detail in Subsections 4.3 and 4.4, we computed various performance and non-performance metrics. In all cases, we performed a 5-fold cross validation procedure to generalize the evaluation results to independent datasets.

More specifically, CBF approaches were built with the whole tag-based profiles of the training items, and with those parts of the users' tag-based profiles formed by tags annotating the training items. These approaches were evaluated with the tag-based profiles of the test items. If an item had no tags, it was not included in the training and test sets. Figure 1 shows the instantiation of the methodology for CBF recommenders and tagged data. CF approaches, on the other hand, were built with those ratings associated to pairs (user, item) in the training set. Correspondingly, training and test models for these approaches only contained consumed items, which were not necessarily tagged. Finally, social approaches were built with all friend relations available in the user profiles, in addition to all the consumed items in the training sets.

[12] Internet Movie Database (IMDb), http://www.imdb.com
[13] Rotten Tomatoes movie reviews, http://www.rottentomatoes.com

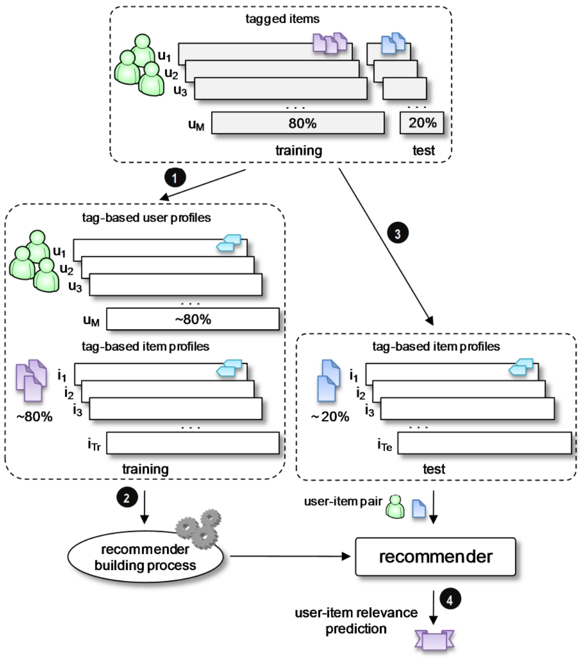

**Figure 1**. Experimental methodology for CBF recommenders and tagging data.

## 4.3 Performance Metrics

In the research literature, prediction accuracy is the most discussed property a recommender system should have. The vast majority of recommenders attempt to predict user tastes or opinions over items (e.g. expressed by numeric ratings), or the probability of usage (e.g. based on purchasing records). In this context, the main assumption is that a system that provides more accurate predictions will be preferred by the user. Thus, many researchers have set out to find algorithms that provide better predictions, and a number of metrics have been proposed to measure the accuracy prediction of such algorithms. Most of these metrics are based on measuring differences between predicted and actual ratings, such as MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error).

Since in social systems like Last.fm and Delicious, users do not explicitly rate items (music artists and Web pages), prediction accuracy metrics, commonly used in the Recommender Systems field, are not suitable to evaluate the algorithms proposed in this paper. For this reason, we shall measure the performance of the recommenders in terms of ranking-based metrics widely used in the Information Retrieval field [8]. Thus, we consider a content retrieval scenario where a system provides the user with a list of *N* recommended items. To evaluate the performance of the system, the selected metrics account for the ratio and position of relevant items in the ranked lists of recommended items. The final performance value is calculated by averaging the performance value over the set of all available users. In our evaluation framework, the set of available items for recommendation is composed by all the items belonging to the test sets (see Section 4.2). We consider as relevant items for the active user those items belonging to her test set; all other items are considered as non-relevant. We describe the utilised performance metrics in the next subsections.

### *4.3.1 Precision*

Precision can be defined as the fraction of recommended items that are relevant [4]:

$$\text{precision} = \frac{|\text{relevant items retrieved}|}{|\text{retrieved items}|}$$

If only the top *N* retrieved items are taken into consideration, the previous ratio is called Precision at *N* or P@*N*. This value can be considered as user-oriented, in the sense that it measures how many relevant documents the user will find in the first results. This characteristic may make P@*N* less stable than other metrics because the total number of available relevant items has a strong influence on the selected value *N* [32]. Because of that, we shall compute and compare P@*N* for different *N* values.

We shall also consider average precision values. The average of P@*N* values at seen relevant items is called Mean Average Precision (MAP) [4]. MAP is a precision metric that emphasises ranking relevant documents higher. Besides, it has shown to have especially good discrimination and stability.

Note that since in our experimental setting, only the items in the user's profile are considered relevant, we cannot count potentially relevant items that the user has not seen, and we therefore get an underestimation of real precision, which is a known limitation of applying Information Retrieval metrics to Recommender Systems [23]. However, as the difference affects all the methods being evaluated, we believe the metric is still consistent for comparative purposes.

### *4.3.2 Recall*

Recall can be defined as the fraction of relevant items that are really recommended [4]:

$$\text{recall} = \frac{|\text{relevant items retrieved}|}{|\text{relevant items}|}$$

If only the top *N* recommended items are taken into consideration, the previous ratio is called Recall at *N* or R@*N* [4].

Again, it has to be noted that the considered set of relevant items is restricted to the items in the users' test sets, which is thus not complete: relevant items unknown to the users are not taken into account. We thus get an overestimation of recall, as we cannot evaluate whether the recommendation approaches are not able to retrieve all relevant items but a representative sample of them.

### *4.3.3 Discounted Cumulative Gain*

Precision and recall do not take into account the usefulness of an item based on its position in a result list. For instance, in the computation of P@10 and R@10, a relevant item at position 1 in the result list is considered as useful as a relevant item at position 10. To address this issue, we shall also compute the Normalised Discounted Cumulative Gain (nDCG) metric [26].

nDCG penalises relevant items appearing lower in a result list. The penalisation is based on a relevance reduction logarithmically proportional to the position of the relevant items. It can also deal with non-binary notions of relevance, which cannot be captured by the previously presented metrics. It is usually calculated only for documents retrieved in the first *N* positions (nDCG@*N*):

$$\text{nDCG}_N = I_N \sum_{n=1}^{N} \frac{2^{rel(i^n)} - 1}{\log_2(1+n)}$$

The value $I_N$ is a normalisation factor for setting nDCG value to 1 when a perfect (ideal) ranking is returned; $rel(i^n)$ represents the relevance score for document $i^n$ (i.e., the item at position $n$ in the result list), which has common values of 10 for highly relevant, 1 for relevant, and 0 for non-relevant items.

## 4.4 Non-performance Metrics

Most recommender systems have been usually evaluated and ranked based on their prediction power, i.e., their ability to accurately predict the user's item choices. However, as pointed out by Shani and Gunawardana [42], it is now widely agreed that accurate predictions are crucial but insufficient to deploy a good recommendation engine. In many applications, people use a recommender system for more than an exact anticipation of their interests. The users may also be interested in obtaining recommendations covering a wide range of their tastes, in rapidly exploring diverse items, or in discovering new items, to name a few of desired properties and functionalities. In this section, we propose a number of metrics to measure different non-performance characteristics of the recommenders: *coverage*, *diversity* and *novelty*. We also present *overlap* metrics to measure similarities (differences) between lists of item recommendations given by distinct recommenders. To better understand these metrics, in the following, we define several factors that will appear in the metric formulations.

Let $R_u$ be the set of items relevant for user $u$, and let $\mathcal{M}$ be the set of recommendation models to be evaluated. We define $L_{m,u}$, the ranked list of recommendations provided to user $u$ by recommendation model $m \in \mathcal{M}$, as:

$$L_{m,u} = \{(u,i,\tau) : i \in \mathcal{I}, \tau > 0\},$$

where $\tau$ is the ranking position of item $i$ in the recommendation list based on the predicted item utility $g_m(u,i)$, having $\tau_{m,u}(i) < \tau_{m,u}(j) \Rightarrow g_m(u,i) \geq g_m(u,j), \forall i,j \in \mathcal{I}$.

We denote by $S_{m,u}$ the set of items that belong to $L_{m,u}$:

$$S_{m,u} = \{i : (u,i,\cdot) \in L_{m,u}\}$$

Finally, we define $S^R_{m,u}$ as the set of those items belonging to $S_{m,u}$ that are relevant for user $u$. That is:

$$S^R_{m,u} = S_{m,u} \cap R_u = \{i: (u,i,\cdot) \in L_{m,u}, i \in R_u\}$$

The previous definitions $S_{m,u}$ and $S^R_{m,u}$ for a given recommendation model $m$ are extended to consider all users with the following expressions:

$$S_m = \bigcup_{u \in \mathcal{U}} S_{m,u},\; S^R_m = \bigcup_{u \in \mathcal{U}} S^R_{m,u}$$

Since some of the non-performance metrics explained below only depend on the top $N$ recommendations provided by each model $m$, we define $\overline{S}_{m,u}$, $\overline{S}^R_{m,u}$, $\overline{S}_m$ and $\overline{S}^R_m$ as, respectively, $S_{m,u}$, $S^R_{m,u}$, $S_m$ and $S^R_m$ on the set $L^N_{m,u}$ of top $N$ recommendations for user $u$, where:

$$L^N_{m,u} = \{(\cdot,\cdot,\tau) \in L_{m,u}, \tau \leq N\}$$

### *4.4.1 Coverage*

Coverage can be defined as the fraction of items for which a recommender $m \in \mathrm{M}$ can provide predictions [23]. Following the proposed notation, it is formulated as follows:

$$cvg(m) = \frac{|S_m|}{|\mathcal{I}|}$$

In this way, the coverage has a value of 1 (maximum) when a recommender is able to return all the different items in the collection. On the other hand, if $m$ is a recommender that always recommends the 10% of the most popular items, its coverage will be 0.1.

Apart from this global coverage, we are also interested in measuring the fraction of relevant items a recommender is able to retrieve. For such purpose, we define coverage of relevant items as follows:

$$cvg^R(m) = \frac{|S^R_m|}{|\bigcup_{u \in \mathcal{U}} R_u|}$$

In this case, the coverage depends on the evaluated items considered for each user. Therefore, a popularity-based recommender will not always obtain an *a priori* known coverage, as in the previous case.

### *4.4.2 Diversity*

A direct way to measure diversity is by computing the average self-information of recommended items. In our context, the diversity for a recommender $m \in \mathrm{M}$ can thus be formulated as follows:

$$div(m) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} div_u(m)$$

$$div_u(m) = H_u(m) = -\sum_{i \in \overline{S}^R_{m,u}} p_{u,i} \cdot \log p_{u,i}$$

The open issue here is how to define the probability $p_{u,i}$ in terms of the diversity offered by item $i$ for user $u$. Different approximations could be proposed. In this paper, we define $p_{u,i}$ in terms of item popularity among the evaluated recommenders. We assume that a recommender $a$ provides diverse recommendations if these are not recommended also by a majority of the other recommenders for the same users. Formally, we set $p_{u,i}$ as follows:

$$p_{u,i} = \frac{\sum_{m \in \mathcal{M}} \delta(m,u,i)}{|\mathcal{M}|},$$

where $\delta(m,u,i) = 1$ iff $i \in \overline{S}^R_{m,u}$, and 0 otherwise. Note that this probability depends on how many recommenders are available for evaluation. Hence, for a particular user, the items with higher probability are those recommended by most of the algorithms. Since the function $-x \log x$ is concave when $x \in [0,1]$, with the above definition we obtain greater diversity values for recommenders providing items neither very popular nor very unpopular. In fact, this function is not symmetric, in a way that penalises more heavily the popular items than it rewards the less well-known ones, which suits well our definition of diversity.

It is important to note that alternative definitions of diversity exist in the literature [2][46][50], and have to be investigated in the future.

### *4.4.3 Relative diversity*

We can also measure diversity differences between two recommendation models $m_1, m_2 \in \mathcal{M}$ by computing the relative entropy between their probability distributions:

$$div(m_1, m_2) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} div_u(m_1, m_2)$$

$$div_u(m_1, m_2) = H_u(m_1 || m_2) = \sum\nolimits_{i \in \overline{S}^R_{m_1,u} \cap \overline{S}^R_{m_2,u}} p_{m_1,u,i} \cdot \log \frac{p_{m_1,u,i}}{p_{m_2,u,i}}$$

The interpretation of the relative entropy $H(X||Y)$ is the *number of extra bits required when using distribution $Y$ instead of $X$*. This metric makes the hypothesis that a user has already been recommended with some items using algorithm $m_1$, and attempts to capture how diverse is the list given by $m_2$, once the user has already seen the previous one. In this way, the relative diversity would measure the distance between both probability distributions, and assuming that the first one is already observed.

Additional measures from information theory could be used, such as the joint entropy or the mutual information. However, since we are interested on asymmetric measures, relative entropy (or Kullback-Leibler divergence) seems an appropriate candidate. Note that, in contrast with the standard definition of relative entropy, the summation here is computed on the intersection space, thus, each probability distribution does not need to sum up to 1 there, and, consequently, the quantity calculated in this way could be negative, as we will obtain in the experiments shown in Section 5.4. This result is proved using the Jense's inequality [17].

Again, different approaches can be considered to define the probabilities $p_{m,u,i}$. In this case, given recommender $m$ and user $u$, we assume a uniform distribution of items. That is:

$$p_{m,u,i} = \frac{1}{\left|\overline{S}^R_{m,u}\right|}$$

Using this simplified estimation for the probabilities, we are actually comparing how many relevant items have been presented to the user by each recommender, and summing it as many times as the number of common items in the two lists. Therefore, a possible alternative for computing these probabilities would be to take into account not only the item relevance, but also the item ranking position in the previous and current recommended lists, for example.

### *4.4.4 Novelty*

Novelty can be defined in a twofold manner. On the one hand, it can be defined as the capability of a recommender system to suggest a user with relevant items that have (usually content-based) characteristics not shared by items previously declared as relevant by the user. On the other hand, it can be defined in a more global way in terms of popularity among users [49], that is, as the capability of a recommender system to suggest a user with relevant but non popular items, i.e., items not liked or known by a wide number of users. We follow here the second perspective and define novelty as follows:

$$nov(m) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} nov_u(m),$$

$$nov_u(m) = H_u(m) = -\sum\nolimits_{i \in \overline{S}^R_{m,u}} p_{u,i} \cdot \log p_{u,i},$$

where

$$p_{u,i} = \frac{|\{v \in \mathcal{U} \setminus u : i \in R_v\}|}{|\mathcal{U} \setminus u|}$$

This formula takes into account the proportion of users who are interested in each of the items retrieved by the recommender $m$ that are relevant for user $u$. As we said previously (see Section 4.4.2), the proposed diversity function is concave and asymmetric, and reaches its maximum when the items in set $\overline{S}^R_{m,u}$ have probabilities near to a uniform distribution. If we assume that other users' relevant items are more useful than any random item when presenting them to the user, it makes sense to estimate the probability in this way. It is pretty clear that, in such an extreme situation, every item presented to the user will be novel for her.

As a first approximation to the probability estimation, we make the following simplification:

$$p_{u,i} \sim p_i = \frac{|\{v \in \mathcal{U}: i \in R_v\}|}{|\mathcal{U}|}$$

That is, we remove the dependency on the user, and the items only relevant for user $u$ are no longer assigned a zero probability, consequently, they are considered novel when recommended by the model $m$.

The resulting formulation of novelty has connections to related work. It is equivalent to the average item self-information used in [50], but in our case we compute the expected self-information rather than the average, i.e., we weight the average by a non-necessarily uniform probability of items. On the other hand, with the probability estimation of $p_{u,i}$ proposed above, self-information is equivalent to the so called inverse user frequency (IUF), and thus $nov_u(m)$ is the expected IUF of the items recommended to $u$ by $m$. Further alternative formulations and definitions, as those proposed in the literature [16][34][46], have to be investigated in the future.

### *4.4.5 Overlap*

Aiming to measure the proportion of recommended items that are provided by two algorithms, we propose two overlap metrics. Both metrics are defined for the recommended items that are relevant for the users, and are limited to the top $N$ results in each list.

#### ***Jaccard based overlap***

The simplest approach to measure the overlap between two lists of items is by computing their intersection. Taking into account the cardinality of the sets of relevant items retrieved by the recommendation algorithms $m_1, m_2 \in \mathcal{M}$, the intersection based overlap can be normalised by using the well-known Jaccard similarity coefficient:

$$ove\_jacc(m_1, m_2) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} ove\_jacc_u(m_1, m_2)$$

$$ove\_jacc_u(m_1, m_2) = \frac{\left|\overline{S}^R_{m_1,u} \cap \overline{S}^R_{m_2,u}\right|}{\left|\overline{S}^R_{m_1,u} \cup \overline{S}^R_{m_2,u}\right|}$$

The overlap computed in this way can also be interpreted as a similarity measure between the lists retrieved by the recommender, and, ultimately, as a measure of how similar the algorithms behave.

#### ***Ranking based overlap***

The previous overlap metric does not take into account the ranking position of relevant documents. Thus, for example, the lists of relevant items $L^R_{m_1,u} = \{i_1, i_2, i_3\}$ and $L^R_{m_2,u} = \{i_1, i_2, i_3\}$ would have the same overlap value than the lists $L^R_{m_1,u} = \{i_1, i_2, i_3\}$ and $L^R_{m_2,u} = \{i_3, i_1, i_2\}$, while the similarity between the given list is higher in the first case. As a rank-sensitive measure of overlap, we propose the following metric[14]:

$$ove\_rank(m_1, m_2) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} ove\_rank_u(m_1, m_2)$$

$$ove\_rank_u(m_1, m_2) = \frac{1}{N} \sum_{i \in \overline{S}^R_{m_1,u} \cap \overline{S}^R_{m_2,u}} \left(1 - \frac{\left|\tau_{m_1,u}(i) - \tau_{m_2,u}(i)\right|}{N - 1}\right)$$

In the example presented above, $ove\_jacc_u(m_1, m_2) = 1$ in both cases, but $ove\_rank_u(m_1, m_2) = 1$ in the first one (when $m_1$ and $m_2$ return the same list) and $ove\_rank_u(m_1, m_2) = 1/3$ in the second one.

Note that other more sophisticated ranking overlap metrics can be tested, e.g. those proposed recently by Kumar and Vassilvitskii [29].

[14] This formula is similar to that of Spearman's ρ, which is used to calculate correlations between two ranked variables. In this situation, we do not think it is appropriate to talk about correlations between variables, and this is why we do not make use of the well-known coefficient.

## 5. Results and Discussion

In this section, we present and analyse the performance and non-performance values obtained with the proposed CBF, CF and social recommenders on the different datasets. For CF and social formulas, we set the user neighbourhood sizes to 15. We conducted experiments with other sizes, obtaining irregular results with smaller neighbourhoods, and similar results with larger ones. In the following, we denote as *cb-tf* the TF-based recommender described in Section 3.1.1, as *cb-bm25* the BM25-based described in 3.1.2; *cb-cosine-tfidf* denotes the one described in 3.1.3, and *cb-cosine-bm25* the recommender described in 3.1.4. With respect to the CF algorithms, *cf-user* and *cf-item* correspond to the user- and item-based approaches described in Sections 3.2.1 and 3.2.2, respectively. Finally, social recommenders are denoted as follows: *social-friends* corresponds to the Friend-based Social recommender described in Section 3.3.1, *personal-social* is the model described in Section 3.3.3 (in the experiments, we set $L = 6$ and $K = 2$), and *friends-popularity* denotes the Popularity based Recommender explained in Section 3.3.4. Trust-based recommenders, described in Section 3.3.5, have been evaluated using two different metrics, obtaining the recommenders noted as *trust-local* (when the MoleTrust metric is used), and *trust-global* (when the PageRank metric is used). As already explained, we have also experimented with combinations of social and collaborative filtering algorithms. These recommenders are denoted as *social-friends-cf*, *trust-local-cf*, and *trust-global-cf*.

### 5.1 Recommendation Performance

Table 2 shows the performance values obtained by the recommenders. For the Last.fm and Delicious datasets, in which users do belong to a social network with explicit relations between them, the best performing approach was the *personal-social strategy*, which adapts the well-known CF formula by weighting the similarity between the user's and her neighbours' rating-based profiles with the users' distances in the social graph. These results thus provide empiric evidence that **combining collaborative filtering and social networking information obtains highly performing (ranking-based) recommendations**. Very interestingly, the *social-friends* strategy, which recommends items liked by explicit friends, obtains acceptable precision and recall values. As concluded by Konstas and colleagues [27], in Last.fm, recommendations generated from the users' social networks represent a good alternative to rating-based methods. Merging this strategy with CF, nonetheless, did not improve the results obtained with the approaches separately.

On the MovieLens dataset, which does not have a social network, we only evaluated CBF and CF approaches. The obtained results confirm previous findings with smallest datasets [7], in which CBF outperforms CF in terms of ranking-based performance metrics. Note that most of CF strategies are designed to provide accurate rating predictions by minimizing accuracy errors such as MAE and RMSE. For Last.fm and Delicious datasets, we also show that in general CBF outperforms CF approaches taking into account both precision/recall and nDCG metrics, especially when using *cosine-based similarities*, as previously observed in [13]. These results give an experimental indication that in social systems, **tag-based approaches provide more precise (top ranked) item lists than collaborative filtering approaches**. Analysing the characteristics of the Last.fm and Delicious datasets, we see that their rating densities are $2.7 \cdot 10^{-3}$ and $6.4 \cdot 10^{-4}$ respectively, while in MovieLens, the rating density is around $4.6 \cdot 10^{-3}$. These differences can be considered as the reason of obtaining worse performance results with CF approaches on the former datasets. As mentioned in Section 4.1, our datasets were built in such a way that all music genres in Last.fm, and most popular topics (tags) in Delicious, were covered by the evaluated items, which makes it harder to get rating correlations between the user profiles that we gathered. Besides that, we note that the output of CF algorithms suffers from frequent ties in the top positions, which introduces a degree of randomness for some users at different cut-off positions.

|  | Last.fm |  |  |  |  |  | Delicious |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | MAP | P@10 | P@20 | R@10 | R@20 | nDCG | MAP | P@10 | P@20 | R@10 | R@20 | nDCG |
| cb-tf | 0.015 | 0.018 | 0.016 | 0.014 | 0.023 | 0.170 | 0.003 | 0.003 | 0.003 | 0.004 | 0.007 | 0.054 |
| cb-bm25 | 0.006 | 0.007 | 0.006 | 0.005 | 0.009 | 0.097 | 0.006 | 0.008 | 0.006 | 0.008 | 0.012 | 0.048 |
| cb-cosine-tfidf | *0.034* | *0.050* | *0.041* | *0.035* | *0.057* | *0.224* | *0.017* | *0.023* | *0.017* | *0.023* | *0.033* | *0.101* |
| cb-cosine-bm25 | 0.020 | 0.032 | 0.026 | 0.022 | 0.035 | 0.144 | 0.011 | 0.015 | 0.011 | 0.015 | 0.021 | 0.062 |
| cf-user | *0.008* | *0.009* | *0.009* | *0.006* | *0.013* | 0.095 | *0.010* | *0.008* | *0.007* | *0.014* | *0.022* | 0.062 |
| cf-item | 0.005 | 0.004 | 0.004 | 0.003 | 0.005 | *0.112* | 0.005 | 0.004 | 0.004 | 0.007 | 0.012 | *0.074* |
| social-friends | 0.028 | 0.043 | 0.041 | 0.034 | 0.065 | 0.150 | 0.014 | 0.023 | 0.019 | 0.028 | 0.042 | 0.047 |
| personal-social | **0.070** | **0.085** | **0.064** | **0.074** | **0.110** | **0.279** | **0.038** | **0.054** | **0.038** | **0.061** | **0.086** | **0.147** |
| friends-popularity | 0.015 | 0.021 | 0.020 | 0.016 | 0.031 | 0.137 | 0.009 | 0.011 | 0.010 | 0.015 | 0.027 | 0.055 |
| trust-global | 0.006 | 0.007 | 0.007 | 0.005 | 0.010 | 0.074 | 0.003 | 0.003 | 0.003 | 0.003 | 0.005 | 0.035 |
| trust-local | 0.012 | 0.016 | 0.015 | 0.012 | 0.024 | 0.070 | 0.005 | 0.006 | 0.006 | 0.008 | 0.015 | 0.024 |
| social-friends-cf | *0.014* | *0.014* | *0.015* | *0.010* | *0.023* | *0.122* | *0.008* | *0.008* | *0.008* | *0.012* | *0.021* | 0.045 |
| trust-global-cf | 0.007 | 0.009 | 0.009 | 0.006 | 0.013 | 0.093 | 0.005 | 0.005 | 0.004 | 0.005 | 0.009 | *0.051* |
| trust-local-cf | 0.003 | 0.009 | 0.006 | 0.006 | 0.008 | 0.014 | 0.005 | 0.003 | 0.003 | 0.010 | 0.018 | 0.015 |

|  | MovieLens |  |  |  |  |  |
|---|---|---|---|---|---|---|
|  | MAP | P@10 | P@20 | R@10 | R@20 | nDCG |
| cb-tf | **0.027** | **0.095** | **0.074** | **0.017** | **0.025** | **0.382** |
| cb-bm25 | 0.013 | 0.040 | 0.033 | 0.007 | 0.011 | 0.224 |
| cb-cosine-tfidf | 0.020 | 0.052 | 0.045 | 0.009 | 0.014 | 0.336 |
| cb-cosine-bm25 | 0.013 | 0.032 | 0.028 | 0.006 | 0.009 | 0.205 |
| cf-user | *0.011* | *0.037* | *0.039* | *0.005* | *0.010* | *0.104* |
| cf-item | 0.001 | 0.001 | 0.001 | 0.000 | 0.000 | 0.014 |

**Table 2**. Obtained performance values.

Based on the conclusions drawn so far, we can provide an answer to **RQ1**, in the context of the evaluated datasets –Last.fm, Delicious and MovieLens. Social tagging is a source of information that can easily be exploited to provide precise item recommendation ranking lists. Additionally, when explicit social networks are available, incorporating characteristics of the social graphs into the computation of user neighbourhoods in collaborative filtering significantly improves the ranking-based recommendation performance.

## 5.2 Recommendation Coverage, Diversity and Novelty

Table 3 shows coverage values of the recommenders when the lists to be compared include only relevant items ($cvg^R$) or every item ($cvg$). In accordance with the obtained precision and recall results, **CBF approaches have coverage values higher than CF and social approaches**.

There is an increase of coverage when social recommenders are combined with CF approaches on both the Last.fm and the Delicious datasets; in particular for *social-friends-cf* and *trust-global-cf* with respect to their social counterparts. Despite this common property, there is an interesting difference between Last.fm and Delicious results. In Delicious, item-based CF can hardly return any item, not necessarily relevant, in their recommendation list, perhaps because of its multi-domain nature.

Other interesting, more expectable, result is the fact of having very high coverage values from CF approaches on MovieLens with respect to their values on Last.fm and Delicious. Similarly to the ranking-based performance metrics analysed in Section 5.1, the coverage of CF recommendations depends on the sparsity of the used rating data. Since the MovieLens dataset has a denser rating matrix than Last.fm and Delicious, it is reasonable that CF performs better in terms of coverage. Furthermore, note that in MovieLens, the coverage values on all (not only the relevant) items of CBF and CF are very close to each other.

| **Coverage** | Last.fm | | Delicious | | MovieLens | |
|---|---|---|---|---|---|---|
| | cvg | $\text{cvg}^R$ | cvg | $\text{cvg}^R$ | cvg | $\text{cvg}^R$ |
| cb-tf | **0.483** | **0.706** | **0.272** | 0.188 | **0.846** | 0.995 |
| cb-bm25 | **0.483** | 0.668 | **0.272** | 0.256 | **0.846** | **0.996** |
| cb-cosine-tfidf | **0.483** | 0.701 | **0.272** | **0.285** | 0.671 | 0.786 |
| cb-cosine-bm25 | **0.483** | 0.682 | **0.272** | 0.282 | 0.671 | 0.786 |
| cf-user | *0.345* | 0.180 | *0.192* | 0.110 | 0.722 | 0.255 |
| cf-item | 0.206 | *0.310* | 0.072 | *0.140* | *0.803* | *0.323* |
| social-friends | 0.330 | 0.211 | 0.148 | 0.042 | | |
| personal-social | 0.183 | 0.191 | 0.185 | 0.147 | | |
| friends-popularity | *0.351* | *0.249* | *0.196* | *0.077* | | |
| trust-global | 0.083 | 0.080 | 0.035 | 0.034 | | |
| trust-local | 0.285 | 0.185 | *0.196* | 0.074 | | |
| social-friends-cf | *0.346* | *0.233* | 0.191 | 0.095 | | |
| trust-global-cf | 0.343 | 0.170 | *0.192* | *0.101* | | |
| trust-local-cf | 0.158 | 0.011 | 0.139 | 0.013 | | |

**Table 3**. Coverage values taking into account relevant and non-relevant items.

Tables 4 and 5 show the obtained diversity and novelty values at different result list lengths. Since these metrics are derived from the self-information magnitude, their values are higher when more items are considered in their computation. This effect could be avoided by introducing a normalisation factor in the metric definition. However, we maintain the non-normalized values since they are suitable for our comparative analysis purposes.

Table 4 shows that in Last.fm and Delicious, the *user-based CF strategy* obtained the highest diversity values, followed by social recommenders, such as the *social-friends strategy*. Tag-based approaches retrieved less diverse item lists, in concordance with the well-known content over-specialisation limitation of CBF techniques [2]. In MovieLens, on the contrary, the CBF approaches retrieved the most diverse result lists. However, they were followed very closely by the *user-based CF strategy*. The high diversity values of tag-based recommenders on the MovieLens dataset could be related to the fact that these recommenders have a very high coverage on such repository, retrieving a large percentage of available items, thus tending to capture the diversity of the whole data. Despite this particular case, and according to the obtained results, we may conclude that, in general, **user-based CF offers highly diverse item recommendations**. Social recommenders such as *social-friends* and *trust-local strategies* and their hybridisation with CF (*social-friends-cf*, *trust-local-cf*) provide high diversity on their results as well.

| **Diversity** | Last.fm | | | Delicious | | | MovieLens | | |
|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 50 | 10 | 20 | 50 | 10 | 20 | 50 |
| cb-tf | 1.563 | 3.267 | 8.866 | 1.178 | 2.442 | 6.493 | 3.042 | 6.381 | 16.708 |
| cb-bm25 | 1.501 | 3.121 | 8.360 | 1.349 | 2.795 | 7.370 | 3.017 | 6.309 | 16.556 |
| cb-cosine-tfidf | *1.650* | *3.477* | *9.523* | 1.515 | 3.145 | 8.326 | **3.272** | **6.693** | **17.095** |
| cb-cosine-bm25 | 1.620 | 3.398 | 9.227 | *1.521* | *3.160* | *8.346* | 3.236 | 6.631 | 17.006 |
| cf-user | **2.732** | **5.532** | *14.054* | **3.041** | **6.068** | **14.735** | *3.037* | *6.087* | *15.450* |
| cf-item | 1.238 | 2.512 | 6.608 | 1.287 | 2.689 | 7.129 | 2.075 | 4.149 | 10.386 |
| social-friends | 2.615 | 5.391 | 13.287 | 2.598 | 4.944 | 9.467 | | | |
| personal-social | 1.798 | 4.015 | 11.650 | 2.221 | 4.859 | 12.825 | | | |
| friends-popularity | 1.860 | 4.040 | 11.428 | 2.056 | 4.391 | 11.468 | | | |
| trust-global | 1.635 | 3.306 | 8.154 | 1.925 | 3.757 | 8.761 | | | |
| trust-local | *2.677* | *5.448* | *14.306* | *2.731* | *5.490* | *13.438* | | | |
| social-friends-cf | *2.669* | *5.448* | **14.297** | *2.761* | *5.525* | *13.531* | | | |
| trust-global-cf | 2.176 | 4.432 | 11.524 | 2.421 | 4.836 | 11.850 | | | |
| trust-local-cf | 2.206 | 4.466 | 9.724 | 2.573 | 5.111 | 10.723 | | | |

**Table 4**. Diversity values for different result list sizes (length $\overline{S}_{m,u}$) and non-relevant items.

Table 5 shows the novelty values obtained on the different datasets. Compared to the conclusions derived from the diversity analysis, the recommenders seem to behave in the opposite direction when analysing the novelty of retrieved items. Whereas in Last.fm and Delicious, CBF approaches provide more novel item suggestions (confirming previously reported observations [16]), in MovieLens, CF strategies are the ones that offer the highest novelty in the generated item lists. In Last.fm and Delicious, apart from CBF approaches, the *cf-item* strategy and hybrid *social-friends-cf* and *trust-global-cf* strategies, obtained very high novelty values.

Furthermore, due to their content over-specialisation, tag-based approaches retrieved less diverse items. Because of that, we may expect that these approaches should also offer less novelty, but this does not seem to be the case in our experiments. There is an aspect that may help clarify this apparent contradiction. The utilised notion of novelty, which is given in Section 4.4.4 and is based on the popularity of the recommended items, aims at measuring the capability of a recommender to suggest a user with relevant but non popular items (i.e., not liked or known by a wide number of users). The items returned by CBF approaches for the active user are quite overspecialised to her tag-based profile, and thus they are not necessarily relevant for many of the users in our limited datasets, especially Last.fm and Delicious.

Additionally, these results show that the proposed novelty and diversity metrics are, in fact, capturing two different, although related, concepts. In general, it is assumed that systems promoting novel results would tend to generate diverse rankings for the users [46]. In our experiments, we have observed that CBF approaches obtain higher novelty values (and lower diversity scores, as discussed above); however, this may be a consequence of their higher coverage, which would allow them to retrieve more long-tail (novel) items, although very similar between them (overspecialisation effect). Thus, if we also take into account the coverage of each method, social approaches (such as *trust-local*) obtain high novelty scores while maintaining a decent coverage. At the same time, these recommenders also return diverse recommendations, which is in agreement with the expected relation between novel and diverse approaches reported in the literature.

Nonetheless, to draw well founded conclusions about the novelty of item recommendations from different recommenders, we believe experiments on larger datasets –probably less biased by their building processes– have to be conducted. We also believe that other definitions of novelty, based on the capability of suggesting a user with relevant items having characteristics not shared by previously declared relevant items, could be used. Hence, novelty would be more closely related to the notion of diversity within each user's preferences (sometimes referred to as unexpectedness [1]). In addition to that, novelty metrics may be defined in terms of the life time of users' preferences and items. For such purpose, time-aware evaluation methodologies, which would split the training and test sets according to different time periods, may be taken into consideration [12].

| **Novelty** | Last.fm | | | Delicious | | | MovieLens | | |
|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 50 | 10 | 20 | 50 | 10 | 20 | 50 |
| cb-tf | 3.617 | 7.241 | 18.120 | 2.073 | 4.202 | 10.889 | 1.436 | 2.621 | 5.731 |
| cb-bm25 | **3.642** | **7.285** | **18.219** | **3.479** | **6.990** | **17.559** | 1.209 | 2.289 | 5.218 |
| cb-cosine-tfidf | 3.486 | 6.983 | 17.505 | 3.459 | 6.959 | 17.494 | *1.469* | *2.981* | *7.572* |
| cb-cosine-bm25 | 3.464 | 6.942 | 17.405 | 3.434 | 6.912 | 17.403 | 1.432 | 2.926 | 7.507 |
| cf-user | 3.003 | 6.083 | 13.767 | 2.853 | 5.580 | 11.894 | **3.082** | **6.010** | **14.371** |
| cf-item | *3.287* | *6.587* | *16.396* | *2.999* | *5.935* | *14.547* | 2.908 | 5.641 | 13.757 |
| social-friends | 2.238 | 4.113 | 9.074 | 0.933 | 1.568 | 2.636 | | | |
| personal-social | 1.797 | 3.158 | 6.227 | 2.168 | 4.062 | 9.251 | | | |
| friends-popularity | *2.933* | *5.777* | *12.948* | *2.477* | *4.470* | 8.405 | | | |
| trust-global | 2.402 | 4.940 | 12.430 | 2.176 | 4.359 | *11.347* | | | |
| trust-local | 2.756 | 5.171 | 12.035 | 2.206 | 3.616 | 7.329 | | | |
| social-friends-cf | *3.119* | 5.946 | 13.006 | 2.574 | 4.583 | 9.153 | | | |
| trust-global-cf | 2.949 | *6.037* | *13.720* | *2.801* | *5.744* | *12.619* | | | |
| trust-local-cf | 0.867 | 1.571 | 2.764 | 1.181 | 1.965 | 3.659 | | | |

**Table 5**. Novelty values for different result list sizes (length of $\overline{S}_{m,u}$) and non-relevant items.

## 5.3 Recommendation Overlap

Tables 6 and 7 show, respectively, the Jaccard and ranking-based overlap values between each pair of recommenders. In general, the recommendation lists of **CBF approaches overlap significantly between them**, **and do not overlap with other types of recommenders**. In Last.fm and Delicious, there is also significant overlapping between item lists from some social recommenders, and, of course, between some hybrid approaches and their CF and social counterparts. The recommended item lists of user- and item-based CF approaches do not overlap in all the cases.

| **Jaccard overlap** Last.fm | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item | social-friends | personal-social | friends-popularity | trust-global | trust-local | social-friends-cf | trust-global-cf | trust-local-cf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cb-tf | | 0.419 | 0.335 | 0.242 | 0.011 | 0.011 | 0.016 | 0.011 | 0.014 | 0.008 | 0.013 | 0.014 | 0.010 | 0.001 |
| cb-bm25 | | | 0.200 | 0.206 | 0.013 | 0.012 | 0.012 | 0.014 | 0.013 | 0.011 | 0.011 | 0.013 | 0.012 | 0.001 |
| cb-cosine-tfidf | | | | **0.532** | 0.016 | 0.015 | 0.028 | 0.014 | 0.024 | 0.011 | 0.023 | 0.024 | 0.015 | 0.001 |
| cb-cosine-bm25 | | | | | 0.016 | 0.014 | 0.024 | 0.013 | 0.022 | 0.012 | 0.021 | 0.022 | 0.015 | 0.001 |
| cf-user | | | | | | 0.022 | 0.047 | 0.134 | 0.059 | 0.060 | 0.049 | **0.405** | **0.344** | 0.003 |
| cf-item | | | | | | | 0.012 | 0.011 | 0.018 | 0.019 | 0.013 | 0.018 | 0.020 | 0.002 |
| social-friends | | | | | | | | 0.114 | **0.442** | 0.041 | **0.399** | 0.284 | 0.050 | 0.008 |
| personal-social | | | | | | | | | 0.129 | 0.134 | 0.127 | 0.129 | 0.133 | 0.005 |
| friends-popularity | | | | | | | | | | 0.051 | **0.395** | 0.103 | 0.056 | 0.027 |
| trust-global | | | | | | | | | | | 0.062 | 0.056 | 0.089 | 0.002 |
| trust-local | | | | | | | | | | | | 0.144 | 0.064 | 0.072 |
| social-friends-cf | | | | | | | | | | | | | 0.158 | 0.003 |
| trust-global-cf | | | | | | | | | | | | | | 0.038 |
| trust-local-cf | | | | | | | | | | | | | | |

**Table 6a**. Jaccard based overlap for N=100 on Last.fm dataset.

| **Jaccard overlap** Delicious | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item | social-friends | personal-social | friends-popularity | trust-global | trust-local | social-friends-cf | trust-global-cf | trust-local-cf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cb-tf | | 0.088 | 0.122 | 0.103 | 0.005 | 0.002 | 0.001 | 0.019 | 0.010 | 0.007 | 0.011 | 0.004 | 0.008 | 0.010 |
| cb-bm25 | | | **0.219** | **0.248** | 0.006 | 0.001 | 0.001 | 0.009 | 0.007 | 0.005 | 0.008 | 0.005 | 0.007 | 0.008 |
| cb-cosine-tfidf | | | | **0.417** | 0.009 | 0.002 | 0.002 | 0.014 | 0.010 | 0.006 | 0.011 | 0.008 | 0.009 | 0.011 |
| cb-cosine-bm25 | | | | | 0.009 | 0.002 | 0.002 | 0.014 | 0.010 | 0.006 | 0.012 | 0.008 | 0.009 | 0.011 |
| cf-user | | | | | | 0.046 | 0.033 | 0.129 | 0.049 | 0.068 | 0.058 | **0.451** | **0.353** | 0.008 |
| cf-item | | | | | | | 0.009 | 0.055 | 0.032 | 0.036 | 0.031 | 0.033 | 0.040 | 0.009 |
| social-friends | | | | | | | | 0.130 | 0.332 | 0.021 | 0.306 | 0.292 | 0.033 | 0.028 |
| personal-social | | | | | | | | | 0.295 | 0.126 | 0.329 | 0.101 | 0.124 | 0.094 |
| friends-popularity | | | | | | | | | | 0.041 | **0.708** | 0.065 | 0.042 | 0.176 |
| trust-global | | | | | | | | | | | 0.049 | 0.048 | **0.215** | 0.006 |
| trust-local | | | | | | | | | | | | 0.074 | 0.051 | 0.189 |
| social-friends-cf | | | | | | | | | | | | | 0.164 | 0.009 |
| trust-global-cf | | | | | | | | | | | | | | 0.028 |
| trust-local-cf | | | | | | | | | | | | | | |

**Table 6b**. Jaccard based overlap for N=100 on Delicious dataset.

| **Jaccard overlap** MovieLens | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item |
|---|---|---|---|---|---|---|
| cb-tf | | **0.853** | **0.607** | 0.596 | 0.021 | 0.001 |
| cb-bm25 | | | 0.589 | 0.587 | 0.019 | 0.001 |
| cb-cosine-tfidf | | | | **0.956** | 0.015 | 0.001 |
| cb-cosine-bm25 | | | | | 0.015 | 0.001 |
| cf-user | | | | | | 0.001 |
| cf-item | | | | | | |

**Table 6c**. Jaccard based overlap for N=100 on MovieLens dataset.

| **Ranking overlap** Last.fm | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item | social-friends | personal-social | friends-popularity | trust-global | trust-local | social-friends-cf | trust-global-cf | trust-local-cf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cb-tf | | **0.429** | 0.340 | 0.252 | 0.014 | 0.014 | 0.018 | 0.013 | 0.016 | 0.011 | 0.016 | 0.018 | 0.013 | 0.001 |
| cb-bm25 | | | 0.204 | 0.210 | 0.016 | 0.014 | 0.013 | 0.016 | 0.015 | 0.014 | 0.014 | 0.016 | 0.015 | 0.001 |
| cb-cosine-tfidf | | | | **0.546** | 0.021 | 0.018 | 0.031 | 0.018 | 0.027 | 0.014 | 0.028 | 0.030 | 0.019 | 0.001 |
| cb-cosine-bm25 | | | | | 0.021 | 0.017 | 0.028 | 0.016 | 0.025 | 0.015 | 0.026 | 0.028 | 0.019 | 0.001 |
| cf-user | | | | | | 0.028 | 0.051 | 0.147 | 0.067 | 0.076 | 0.058 | **0.454** | **0.409** | 0.002 |
| cf-item | | | | | | | 0.013 | 0.013 | 0.023 | 0.023 | 0.015 | 0.023 | 0.026 | 0.002 |
| social-friends | | | | | | | | 0.111 | **0.300** | 0.047 | **0.353** | **0.293** | 0.057 | 0.004 |
| personal-social | | | | | | | | | 0.134 | 0.154 | 0.126 | 0.146 | 0.149 | 0.003 |
| friends-popularity | | | | | | | | | | 0.059 | 0.274 | 0.116 | 0.066 | 0.013 |
| trust-global | | | | | | | | | | | 0.076 | 0.071 | 0.114 | 0.002 |
| trust-local | | | | | | | | | | | | 0.170 | 0.078 | 0.067 |
| social-friends-cf | | | | | | | | | | | | | 0.198 | 0.002 |
| trust-global-cf | | | | | | | | | | | | | | 0.042 |
| trust-local-cf | | | | | | | | | | | | | | |

**Table 7a**. Ranking-based overlap for N=100 on Last.fm dataset.

| **Ranking overlap** Delicious | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item | social-friends | personal-social | friends-popularity | trust-global | trust-local | social-friends-cf | trust-global-cf | trust-local-cf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cb-tf | | 0.093 | 0.134 | 0.111 | 0.004 | 0.002 | 0.001 | 0.024 | 0.010 | 0.009 | 0.011 | 0.004 | 0.010 | 0.010 |
| cb-bm25 | | | **0.237** | **0.262** | 0.006 | 0.001 | 0.001 | 0.011 | 0.007 | 0.007 | 0.008 | 0.005 | 0.009 | 0.008 |
| cb-cosine-tfidf | | | | **0.440** | 0.009 | 0.001 | 0.001 | 0.017 | 0.010 | 0.008 | 0.012 | 0.008 | 0.012 | 0.011 |
| cb-cosine-bm25 | | | | | 0.009 | 0.001 | 0.001 | 0.018 | 0.010 | 0.008 | 0.012 | 0.008 | 0.012 | 0.011 |
| cf-user | | | | | | 0.052 | 0.021 | 0.141 | 0.054 | 0.091 | 0.066 | **0.450** | 0.413 | 0.006 |
| cf-item | | | | | | | 0.007 | 0.061 | 0.036 | 0.045 | 0.033 | 0.038 | 0.048 | 0.008 |
| social-friends | | | | | | | | 0.089 | 0.191 | 0.020 | **0.198** | 0.179 | 0.029 | 0.011 |
| personal-social | | | | | | | | | **0.231** | 0.147 | **0.263** | 0.112 | 0.140 | 0.057 |
| friends-popularity | | | | | | | | | | 0.050 | **0.427** | 0.067 | 0.050 | 0.086 |
| trust-global | | | | | | | | | | | 0.061 | 0.063 | 0.257 | 0.007 |
| trust-local | | | | | | | | | | | | 0.083 | 0.062 | 0.118 |
| social-friends-cf | | | | | | | | | | | | | **0.199** | 0.005 |
| trust-global-cf | | | | | | | | | | | | | | 0.030 |
| trust-local-cf | | | | | | | | | | | | | | |

**Table 7b**. Ranking-based overlap for N=100 on Delicious dataset.

| **Ranking overlap** MovieLens | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item |
|---|---|---|---|---|---|---|
| cb-tf | | **0.819** | **0.617** | 0.946 | 0.026 | 0.002 |
| cb-bm25 | | | 0.595 | 0.594 | 0.024 | 0.002 |
| cb-cosine-tfidf | | | | **0.946** | 0.019 | 0.001 |
| cb-cosine-bm25 | | | | | 0.019 | 0.001 |
| cf-user | | | | | | 0.001 |
| cf-item | | | | | | |

**Table 7c**. Ranking-based overlap for N=100 on MovieLens dataset.

Very interestingly, and differently to preliminary conclusions we derived in [7], we observe that the above results apply to all datasets for both Jaccard and ranking-based overlap metrics. The Jaccard metric measures the number of items two recommenders have in common in their recommendation lists. The ranking-based metric, on the other hand, is similar to the Spearman correlation, and measures how differently two recommenders rank the items they have in common in their recommendation lists. Thus, the fact that both Jaccard and ranking-based overlap values are high means that two recommenders not only share many items, but also provide quite similar item rankings. As shown in the tables, in our experiments, this is especially notorious in the *cb-cosine-tfidf* and *cb-cosine-bm25 strategies*.

From the obtained results on recommendation overlapping, and taking into account that whereas CBF approaches offer high coverage and novelty, social-CF hybrids offer high performance and diversity, we may expect that meta-hybrid recommenders combining the above strategies could provide valuable, balanced item suggestions in terms of the above metrics, for different contexts depending on the needed level of personalisation.

## 5.4 Relative Recommendation Diversity

The proposed relative diversity metric aims at capturing the information gain obtained with a recommender in comparison to another. It represents whether or not a recommendation list provides additional information to a previously presented recommendation. As described in Section 4.4.3, the relative diversity $div(m_1, m_2)$ measures how diverse a recommendation list is given by a recommendation model $m_2$, once the user has already seen the list provided by a model $m_1$. This is performed by computing $H(m_1||m_2) = H(X||Y)$ for each user, which can be interpreted as the *number of extra bits required to code samples from $X$ based on distribution $Y$ instead of using the distribution $X$*. Consequently, the larger this measure is, the more different the two distributions are, and thus, the more information is conveyed by the second recommender with respect to the first one.

Table 8 shows the relative diversity values computed for each pair of recommenders on the three datasets. Note that there are negative values in the table. As explained in Section 4.4.3, this is due to the fact that we are not using a probability space, but an intersection of two spaces, and thus, the Jensen's inequality does not hold. Nevertheless, since we are interested in capturing how different the two recommendation lists are, we can ignore the sign of the measure and focus on its absolute value. For completeness, however, the sign is preserved in the table.

| **Relative diversity** Last.fm | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item | social-friends | personal-social | friends-popularity | trust-global | trust-local | social-friends-cf | trust-global-cf | trust-local-cf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cb-tf | | 15.67 | -10.20 | -2.25 | 1.65 | 1.61 | 1.49 | -0.75 | 1.63 | 0.70 | 0.81 | 2.02 | 1.38 | 0.45 |
| cb-bm25 | -4.12 | | -5.15 | -3.87 | 0.77 | 0.63 | -0.01 | -1.00 | 0.58 | 0.60 | -0.16 | 0.48 | 0.78 | 0.18 |
| cb-cosine-tfidf | **32.85** | **20.72** | | **20.93** | 7.09 | 6.25 | 8.10 | 5.69 | 8.22 | 3.58 | 6.14 | 9.34 | 6.58 | 0.97 |
| cb-cosine-bm25 | 15.17 | 16.34 | -3.94 | | 6.20 | 5.14 | 5.75 | -0.12 | 6.55 | 3.22 | 4.47 | 7.25 | 5.59 | 0.90 |
| cf-user | 0.03 | 0.31 | -1.15 | -0.91 | | 1.13 | -2.80 | -14.21 | 0.74 | 4.55 | -3.93 | -3.10 | 8.84 | 0.20 |
| cf-item | -0.30 | -0.02 | -1.33 | -1.09 | 0.42 | | -0.19 | 0.70 | 0.10 | 0.67 | -0.09 | 0.21 | 0.54 | 0.12 |
| social-friends | 2.99 | 3.01 | 1.94 | 2.81 | 16.12 | 2.20 | | 0.41 | **42.25** | 15.05 | 1.71 | **21.35** | 16.60 | 0.53 |
| personal-social | 3.68 | 4.45 | 6.02 | 5.62 | **56.16** | 0.06 | **21.10** | | **32.81** | **59.01** | **20.42** | **37.08** | **56.05** | 0.06 |
| friends-popularity | 1.07 | 1.13 | -0.57 | 0.09 | 1.84 | 7.27 | -7.44 | -3.86 | | 7.54 | -7.99 | 5.61 | 7.33 | 0.73 |
| trust-global | 0.12 | 0.17 | -0.44 | -0.37 | 1.01 | 0.35 | -3.15 | -16.88 | -0.52 | | -4.57 | -2.88 | 1.53 | -0.01 |
| trust-local | 2.98 | 2.79 | 3.18 | 4.59 | **25.48** | 2.05 | 10.90 | 9.12 | 32.00 | 22.51 | | **31.21** | 19.70 | 1.03 |
| social-friends-cf | 1.53 | 1.74 | 0.63 | 1.25 | **28.05** | 1.70 | 9.88 | -8.47 | 15.11 | 13.26 | -1.08 | | **20.44** | 0.18 |
| trust-global-cf | 0.04 | 0.20 | -1.07 | -0.81 | 3.97 | 0.77 | -2.88 | -14.28 | 0.40 | 4.66 | -3.95 | -2.45 | | 0.07 |
| trust-local-cf | -1.13 | -0.33 | -2.38 | -2.21 | -0.08 | 0.30 | -0.61 | 0.07 | -1.45 | 0.48 | -3.13 | 0.04 | 0.53 | |

**Table 8a**. Obtained relative diversity values for N=100 on Last.fm dataset (values $|div(m_1, m_2)| \geq 20$ in bold).

| **Relative diversity** Delicious | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item | social-friends | personal-social | friends-popularity | trust-global | trust-local | social-friends-cf | trust-global-cf | trust-local-cf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cb-tf | | 0.03 | -0.28 | -0.24 | -0.30 | 0.00 | -0.01 | -0.01 | 0.00 | 0.01 | -0.02 | -0.25 | -0.21 | -0.01 |
| cb-bm25 | 0.42 | | -0.95 | -1.38 | -1.43 | 0.03 | 0.03 | 0.10 | 0.08 | 0.03 | 0.11 | -1.25 | -0.61 | 0.07 |
| cb-cosine-tfidf | 1.90 | 4.20 | | 1.31 | -2.16 | 0.09 | 0.09 | 0.46 | 0.20 | 0.08 | 0.31 | -1.82 | -0.77 | 0.23 |
| cb-cosine-bm25 | 1.50 | 5.05 | 4.80 | | -2.25 | 0.11 | 0.12 | 0.60 | 0.29 | 0.10 | 0.39 | -1.92 | -0.75 | 0.31 |
| cf-user | 0.69 | 0.51 | 0.84 | 0.90 | | 2.15 | 0.79 | -2.34 | 1.31 | 2.37 | 1.16 | 6.02 | 6.50 | 0.34 |
| cf-item | 0.17 | 0.09 | 0.14 | 0.17 | 0.01 | | 0.14 | 0.00 | 0.43 | 0.52 | 0.25 | 0.36 | 0.48 | 0.03 |
| social-friends | 0.24 | 0.28 | 0.46 | 0.47 | 0.31 | 0.18 | | -1.87 | 0.63 | 0.53 | -1.59 | 1.77 | 0.93 | 0.14 |
| personal-social | 2.16 | 0.58 | 2.32 | 4.30 | **13.03** | 1.96 | 4.52 | | **12.76** | **13.77** | **12.64** | **10.11** | **13.51** | 0.55 |
| friends-popularity | 0.33 | 0.20 | 0.42 | 0.41 | 0.76 | 0.24 | 2.49 | -3.75 | | 1.00 | -0.69 | 1.51 | 0.97 | 0.33 |
| trust-global | 0.19 | 0.04 | 0.08 | -0.03 | 0.14 | 0.04 | 0.04 | -4.60 | 0.18 | | -0.26 | 0.24 | 0.27 | 0.00 |
| trust-local | 0.62 | 0.40 | 0.84 | 1.01 | 1.79 | 0.64 | 4.27 | -3.07 | 6.47 | 2.16 | | 3.21 | 2.32 | 0.85 |
| social-friends-cf | 0.48 | 0.40 | 0.62 | 0.56 | 1.33 | 1.35 | 0.75 | -1.96 | 0.85 | 1.33 | 0.13 | | 4.21 | 0.12 |
| trust-global-cf | 0.41 | 0.22 | 0.38 | 0.34 | -0.42 | 0.68 | 0.16 | -3.52 | 0.60 | 3.25 | 0.17 | 1.42 | | -0.01 |
| trust-local-cf | 0.16 | 0.17 | 0.17 | 0.22 | -0.23 | 0.33 | 0.24 | -0.09 | 0.41 | 0.09 | -0.75 | 0.12 | 0.78 | |

**Table 8b**. Obtained relative diversity values for N=100 on Delicious dataset (values $|div(m_1, m_2)| \geq 10$ in bold).

| **Relative diversity** MovieLens | cb-tf | cb-bm25 | cb-cosine-tfidf | cb-cosine-bm25 | cf-user | cf-item |
|---|---|---|---|---|---|---|
| cb-tf | | **34.90** | **58.90** | **57.13** | **9.32** | 0.88 |
| cb-bm25 | **-13.38** | | **20.33** | **22.98** | 3.44 | 0.72 |
| cb-cosine-tfidf | **-11.93** | **10.98** | | 5.65 | 0.67 | 0.93 |
| cb-cosine-bm25 | **-10.39** | **9.16** | -1.10 | | 0.18 | 0.96 |
| cf-user | 6.40 | **9.00** | 8.76 | 8.78 | | 0.52 |
| cf-item | -0.09 | -0.07 | -0.09 | -0.09 | -0.05 | |

**Table 8c**. Obtained relative diversity values for N=100 on MovieLens dataset (values $|div(m_1, m_2)| \geq 9$ in bold).

The results show that **CBF recommenders give enough diverse recommendations by themselves so that subsequent recommenders would be redundant from a user-perspective**. This can be observed in the table by looking at the rows corresponding to CBF approaches, and checking the low (absolute) values of these cells. This result is reversed in the MovieLens dataset, where CBF approaches do not provide sufficiently diverse recommendations, and other recommenders could further improve the diversity of the rankings. This result can be attributed to the denser rating matrix available in this dataset, which may imply richer, more diverse relations between users and movies.

Another interesting finding is that the *personal-social* approach does not present enough diverse recommendations when compared with other approaches –mainly CF, popularity-based and global social strategies, and hybrid recommenders–, whose top recommendation lists are very diverse (see Table 4). This is especially noticeable in the Delicious dataset, for which the above recommenders achieved the highest diversity values. Hence, although *personal-social strategy* provides diverse recommendations by itself, its suggestions could be further diversified by combining them with other types of recommendations.

## 5.5 Summary of Results

The presented results lead to the following answer to **RQ2**. The combination of different recommendation input sources and algorithms allows optimizing for different quality dimensions, in a non-exclusive way, by selecting the appropriate combination of alternatives that leverage the desired strengths, in terms of coverage, diversity, novelty and/or overlap. In particular, our observations motivate the combination of CBF and CF recommenders, CBF and social recommenders, or even some CBF, CF, and social recommenders, as meaningful options. For instance, the combination of *tag-based approaches* (with high coverage and novelty) and *social-CF hybrids* (with high performance and diversity) may provide balanced (high overlap) and precise item recommendations. Analogously, a combination of *user-based CF* and the *personal-social* approach enables diverse and non-redundant recommendations, besides the benefits on accuracy and novelty of CF.

The analysis of the results shows that social networks are a key source of information to provide accurate recommendations, in both Last.fm and Delicious systems. In MovieLens, where there is no social networking information, CBF was the most accurate strategy. Figure 2 shows a summary of the performance results, where only the best performing algorithm of each group (among content-based filtering, collaborative filtering, social, and hybrid recommenders) is considered. In the figure, we observe that apart from the social recommender, the CBF approach obtained very good performance in Last.fm and Delicious.

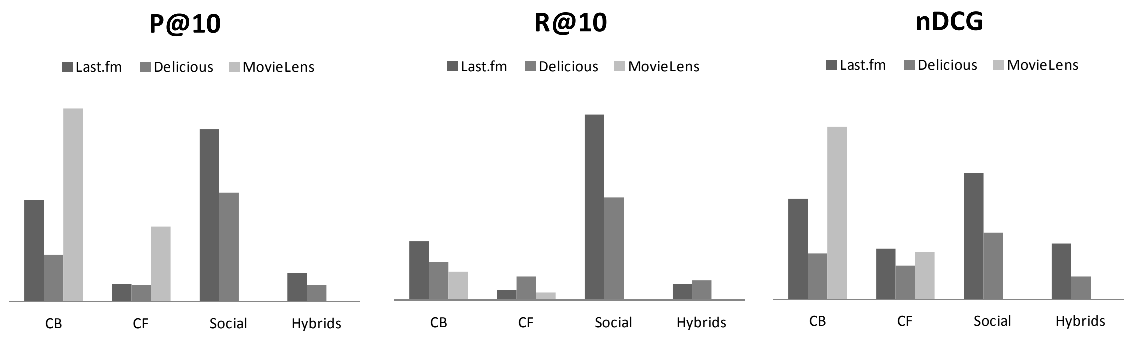

**Figure 2**. Summary of the performance results; only the best results for each recommender type are shown. Note that no social and hybrid recommenders were run on MovieLens dataset, which does not have social networking information.

Figure 3 shows a summary of the best values of non-performance metrics –coverage, diversity and novelty –for the different types of recommenders. On the three datasets, CBF had higher coverage than CF and social strategies. On the MovieLens dataset, coverage values were much higher than on Last.fm and Delicious datasets, which may be due to the way in which the latter were built, i.e., by starting the information crawling from the systems' top tags (see Sections 4.1.1 and 4.1.2).

On the Last.fm and Delicious datasets, CBF offered less diverse recommendations, while the other types of recommenders obtained very similar results. Nevertheless, for both datasets, when the length of the lists increases (see Table 4), social-based approaches and hybrids boost diversity, showing the importance of exploiting social networking information to provide diverse recommendations. In MovieLens, CBF and CF obtained similar diversity values.

Novelty was more stable with respect to the length of the recommendation lists. In Last.fm and Delicious, CBF offered higher novelty than other recommenders, whilst in MovieLens, CF was the strategy with the most novel recommendations. These opposite results may be due to the differences in the nature of the available data. Last.fm and Delicious are tag-oriented, and thus have abundant tags, which help retrieving novel information. MovieLens, on the other hand, is mainly rating-oriented, with fewer tags to procure novelty in tag-based recommendations. As we may see in Table 1, the average number of tag assignments per user in Last.fm and Delicious is indeed much larger than in MovieLens: 98.6 and 234.4 vs. 22.7, respectively.

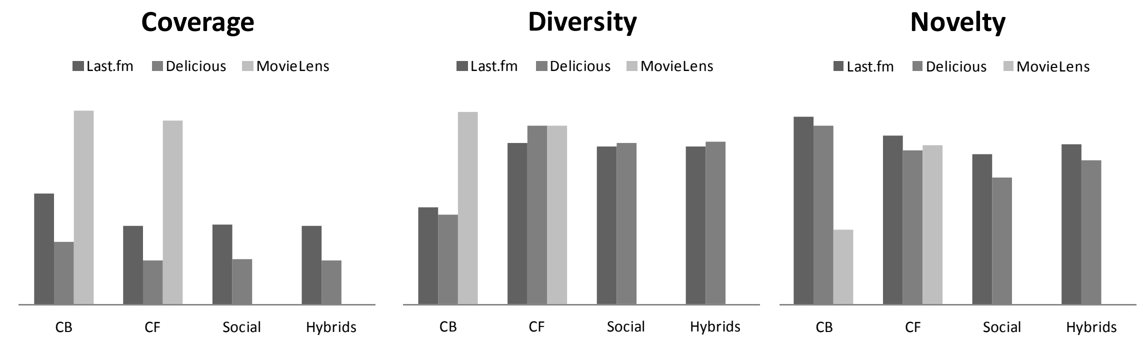

**Figure 3**. Summary of the non-performance results for N=10; only the best results for each recommender type are shown. Note that no social and hybrid recommenders were run on the MovieLens dataset, since it does not include social networking data.

Regarding overlap and relative diversity, which are non-performance metrics used to compare pairs of recommendations lists, we may conclude the following. The evaluated CBF approaches overlap significantly between them; and CF and social approaches have a high degree of intersection with their corresponding hybrids. CBF approaches, however, barely overlap with CF and social recommenders, and thus leave room for combining them through more complex hybridisation strategies. In fact, the obtained relative diversity values show that recommendation lists from CBF provide new information to other types of recommenders.

## 6. Conclusions and Future Work

We have presented a comparative study on the influence of existing sources of information in Social Web systems on recommendation. We have empirically evaluated and compared a representative sample of content-based, collaborative filtering, and social recommenders, by using a variety of performance and non-performance metrics on three datasets. The datasets were obtained from the Last.fm, Delicious, and MovieLens systems, containing user data of different sorts, such as manual tags, social networking information, and item consumption records (music play counts, Web pages bookmarks, and movie ratings).

Our study provides empiric evidence on the comparative qualities of different families of recommendation methods on different input data sources. Specifically, we addressed two research questions, aiming to 1) identify which sources of information available in social systems are more valuable for recommendation; 2) prove whether recommendation approaches exploiting different sources of information in social systems really offer heterogeneous item suggestions, from which hybrid strategies could benefit.

To address the first question, we computed ranking-based metrics –*precision*, *recall* and *nDCG*– of the different recommenders on the above datasets. From the obtained results, we conclude that when explicit social networks are available, incorporating characteristics of the social graphs into the computation of user neighbourhoods in memory-based collaborative filtering significantly improves recommendation in terms of ranking quality. On all the datasets, social tagging is found to be a source of information that can easily be exploited to provide precise item recommendation ranking lists.

To address the second question, we propose a number of non-performance metrics capturing different item recommendation properties, namely *coverage*, *diversity* and *novelty*, defined as the capability of a recommender to suggest a user with relevant but non popular items), and metrics comparing pairs of ranked recommendation lists, namely *overlap* and *relative diversity*. Analysing the obtained results, we conclude that exploiting social tagging information by content-based recommenders offers high coverage and novelty, and combining social networking and collaborative filtering information by hybrid recommenders provides high diversity. This, along with the fact that recommendations from the different approaches –content-based, collaborative filtering, and social– have low overlap and relative diversity values between them, leads to the conclusion that meta-hybrid recommenders combining the above strategies may provide valuable, balanced item suggestions in terms of performance and non-performance metrics, for different contexts depending on the needed level of personalisation.

Our study provides empiric evidence on the comparative qualities of different families of recommendation methods on different input data sources. A precise knowledge of the nuanced strengths of alternative recommendation inputs and methods provides for tailoring the configuration of hybrid recommendation approaches to different domain-, business-, and/or task-dependent requirements. Beyond the study reported here, there is room for investigation with further recommenders, input data, and hybridisation strategies. In particular, handling implicit evidence of user preferences (as opposed to explicit preference values) is an open research issue in the field. Seeking high performing recommenders for each source of user preferences (in our case, manual tags, social contacts, and item consumptions), and dynamically combining them to balance the non-performance characteristics of final item recommendations, is a research direction we aim to continue as well [9]. In this context, we envision the exploration of other definitions of the metrics proposed in this paper, such as those studied in [46]. For such purpose, we also plan to take into consideration the time dimension on recommendation and its evaluation [12]. Furthermore, user studies would be a valuable complement of offline experiments to obtain further insights and conclusions.

## Acknowledgments

The work presented here was supported by the Spanish Ministry of Science and Innovation (TIN2011-28538-C02), and the Autonomous Community of Madrid (CCG10-UAM/TIC-5877).

## References

[1] P. Adamopoulos, A. Tuzhilin. On Unexpectedness in Recommender Systems: Or How to. Expect the Unexpected, In: Proc. ACM RecSys'11 Workshop On Novelty and Diversity in Recommender Systems, DiveRS'11, 11–18.

[2] G. Adomavicius, A. Tuzhilin, Toward the Next Generation of Recommender Systems: A Survey and Possible Extensions, IEEE Transactions on Knowledge & Data Engineering 17(6) (2005), 734–749.

[3] K. Ali, W. Van Stam, TiVo: Making Show Recommendations Using a Distributed Collaborative Filtering Architecture, In: Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'04, 2004, 394–401.

[4] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison Wesley, 1999.

[5] L. Baltrunas, X. Amatriain, Towards Time-dependant Recommendation based on Implicit Feedback, In: Proc. RecSys'09 Workshop on Context-aware Recommender Systems, CARS'10, 2010.

[6] K. Barman, O. Dabeer, What is Popular Amongst Your Friends? arxiv:1006.1772, 2010.

[7] A. Bellogín, I. Cantador, P. Castells, A Study of Heterogeneity in Recommendations for a Social Music Service, In: Proc. RecSys'10 Workshop on Information Heterogeneity and Fusion in Recommender Systems, HetRec'10, 2010, 1–8.

[8] A. Bellogín, P. Castells, I. Cantador, Precision-oriented evaluation of recommender systems: an algorithmic comparison, In: Proc. 5th ACM Conference on Recommender Systems, RecSys'11, 2011, 333-336.

[9] A. Bellogín, P. Castells, I. Cantador, Self-adjusting Hybrid Recommenders Based on Social Network Analysis. In: Proc. 34th ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'11, 2011, 1147-1148.

[10] D. Ben-Shimon, A. Tsikinovsky, L. Rokach, A. Meisles, G. Shani, L. Naamani, Recommender System from Personal Social Networks, In: Proc. 5th Atlantic Web Intelligence Conference AWIC'07, 2007, 47–55.

[11] P. Bonhard, M. A. Sasse, Knowing Me, Knowing You - Using Profiles and Social Networking to Improve Recommender Systems, BT Technology Journal 25(3) (2006), 84–98.

[12] P. G. Campos, F. Díez, M. Sánchez-Montañés, Towards a More Realistic Evaluation: Testing the Ability to Predict Future Tastes of Matrix Factorization-based Recommenders, In: Proc. 5th ACM Conference on Recommender Systems, RecSys'11, 2011, 309-312.

[13] I. Cantador, A. Bellogín, D. Vallet, Content-based Recommendation in Social Tagging Systems, In: Proc. 4th ACM Conference on Recommender Systems, RecSys'10, 2010, 237-240.

[14] I. Cantador, P. Brusilovsky, T. Kuflik, Second Workshop on Information Heterogeneity and Fusion in Recommender Systems, In: Proc. 5th ACM Conference on Recommender Systems, RecSys'11, 2011, 387-388.

[15] O. Celma, Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space, Springer-Verlag, 2010.

[16] O. Celma, P. Herrera, A New Approach to Evaluating Novel Recommendations, In: Proc. 2nd ACM Conference on Recommender Systems, RecSys'08, 2008, 179–186.

[17] T. M. Cover, J. A. Thomas, Elements of Information Theory, Wiley-Interscience, 1991.

[18] A. S. Das, M. Datar, A. Garg, S. Rajaram, Google News Personalization: Scalable Online Collaborative Filtering, In: Proc. 16th International Conference on World Wide Web WWW'07, 2007, 271–280.

[19] J. Gemmell, T. Schimoler, M. Ramezani, L. Christiansen, M. Mobasher, A Fast Effective Multi-Channeled Tag Recommender, In: Proc. ECML PKDD Discovery Challenge'09, 2009.

[20] S. A. Golder, B. A. Huberman, Usage Patterns of Collaborative Tagging Systems, Journal of Information Science 32(2) (2006), 198-208.

[21] I. Guy, L. Chen, M. X. Zhou, Workshop on Social Recommender Systems, In: Proc. 2010 International Conference on Intelligent User Interfaces IUI'10, 2010.

[22] J. He, W. W. Chu, A Social Network-Based Recommender System (SNRS), In: N. Memon, J. J. Xu, D. L. Hicks, H. Chen, (Eds.), Data Mining for Social Network Data, 2010, 47–74.

[23] L. Herlocker, J. A. Konstan, A. Borchers, J. Riedl, An Algorithmic Framework for Performing Collaborative Filtering, In: Proc. 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'99, 1999, 230–237.

[24] A. Hotho, R. Jäschke, C. Schmitz, G. Stumme, Information Retrieval in Folksonomies: Search and Ranking, In: Proc. 5th International Semantic Web Conference, ISWC'06, 2006, 411–426.

[25] Y. Hu, Y. Koren, C. Volinsky, Collaborative Filtering for Implicit Feedback Datasets. In: Proc. 8th IEEE International Conference on Data Mining, ICDM'08, 2008, 263–272.

[26] K. Jarvelin, J. Kekalainen, Cumulated Gain-based Evaluation of IR Techniques, ACM Transactions on Information Systems 20(4) (2002), 422–446.

[27] I. Konstas, V. Stathopoulos, J. M. Jose, On Social Networks and Collaborative Recommendation, In: Proc. 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'09, 2009, 195–202.

[28] Y. Koren, R. M. Bell, Advances in Collaborative Filtering. In F. Ricci, L. Rokach, B. Shapira, P. B. Kantor (Eds.) Recommender Systems Handbook, 2011, 45–186.

[29] R. Kumar, S. Vassilvitskii, Generalized Distances between Rankings, In: Proc. 19th International Conference on World Wide Web, WWW'10, 2010, 571–580.

[30] T. Lee, Y. Park, Y. T., Park, A Time-based Approach to Effective Recommender Systems using Implicit Feedback, Expert Systems with Applications 34(4) (2008), 3055–3062.

[31] F. Liu, H. J. Lee, Use of Social Network Information to Enhance Collaborative Filtering Performance, Expert Systems with Applications 37(7) (2010), 4772–4778.

[32] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008.

[33] P. Massa, P. Avesani, Trust-aware Recommender Systems, In: Proc. 1st ACM conference on Recommender Systems, RecSys'07, 2007, 17–24.

[34] S. M. Mcnee, J. Riedl, J. A. Konstan, Being Accurate is not Enough: How Accuracy Metrics Have Hurt Recommender Systems, In Proc. CHI 2006 Extended Abstracts on Human Factors in Computing Systems, CHI EA'06, 2006, 1097–1101.

[35] K. Musial, Recommender System for Online Social Network, Lambert Academic Publishing, 2009.

[36] S. Niwa, T. Doi, S. Honiden, Web Page Recommender System based on Folksonomy Mining for ITNG'06 Submissions, In: Proc. 3rd International Conference on Information Technology, ITNG'06, 2006, 388–393.

[37] G. Noll, C. Meinel, Web Search Personalization via Social Bookmarking and Tagging, In: Proc. 6th International Semantic Web Conference, ISWC'07, 2007, 367–380.

[38] D. W. Oard, J. Kim, Implicit Feedback for Recommender Systems, In: Proc. AAAI'98 Workshop on Recommender Systems, 1998, 81–83.

[39] I. Pilászy, D. Tikk, Recommending New Movies: Even a Few Ratings Are More Valuable Than Metadata, In: Proc. 3rd ACM Conference on Recommender Systems, RecSys'09, 2009, 93–100.

[40] S. Sen, J. Vig, J. Riedl, Tagommenders: Connecting Users to Items through Tags, In: Proc. 18th International Conference on World Wide Web, WWW'09, 2009, 671–680.

[41] A. Seth, J. Zhang, A Social Network Based Approach to Personalized Recommendation of Participatory Media Content, In: Proc. 2nd International AAAI Conference on Weblogs and Social Media, ICWSM'08, 2008.

[42] G. Shani, A. Gunawardana, Evaluating Recommender Systems, In: F. Ricci, L. Rokach, B. Shapira, P. B. Kantor (Eds.) Recommender Systems Handbook, 2011, 257-297.

[43] A. Shepitsen, J. Gemmell, B. Mobasher, R. Burke, Personalized Recommendation in Social Tagging Systems using Hierarchical Clustering, In: Proc. 2nd ACM Conference on Recommender Systems, RecSys'08, 2008, 259–266.

[44] K. Spärck-Jones, S. Walker, S. E. Robertson, A Probabilistic Model of Information Retrieval: Development and Comparative Experiments (parts 1 and 2), Information Processing and Management 36(6) (2000), 779–840.

[45] D. Vallet, I. Cantador, J. M. Jose, Personalizing Web Search with Folksonomy-Based User and Document Profiles, In: Proc. 32nd European Conference on Information Retrieval, ECIR'10, 2010, 420-431.

[46] S. Vargas, P. Castells, Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems, In: Proc. 5th ACM Conference on Recommender Systems, RecSys'11, 2011, 109-116.

[47] J. Wang, S. Robertson, A. de Vries, M. Reinders, Probabilistic Relevance Ranking for Collaborative Filtering, Information Retrieval 11(6) (2008), 477–497.

[48] S. Xu, S. Bao, B. Fei, Z. Su, Y. Yu, Exploring Folksonomy for Personalized Search, In: Proc. 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'08, 2008, 155–162.

[49] V. Zanardi, L. Capra, Social Ranking: Uncovering Relevant Content using Tag-based Recommender Systems, In: Proc. 2nd ACM Conference on Recommender Systems, RecSys'08, 2008, 51–58.

[50] T. Zhou, Z. Kuscsik, J. G. Liu, M. Medo, J. R. Wakeling, Y. C. Zhang, Solving the Apparent Diversity-accuracy Dilemma of Recommender Systems, National Academy of Sciences of the United States of America 107(10) (2010), 4511–4515.