Evaluating Peer-to-Peer Recommender Systems that Exploit Spontaneous Affinities

Giancarlo Ruffo and Rossano Schifanella
Dipartimento di Informatica
Università degli Studi di Torino
Torino, Italy
{ruffo,schifane}@di.unito.it

ABSTRACT

The validation of a recommender system is always a quite hazardous task, because of the difficulty of modeling the tastes of a given user. Novel (decentralized) recommender systems are proposed and evaluated by way of well known logs of user profiles and buddy tables, that contain lists of items with feedback ratings assigned by a given set of users. These information are cross linked, and the precision of the recommendation is compared with other well known (centralized) systems. This evaluation approach cannot be applied in the actual peer-to-peer domain: it is difficult, if not impossible, to build and maintain user profiles, and users are not required to give feedbacks to a data collector entity. Moreover, objects are poorly or not structured, and meta-information, when present, cannot be trusted because of fake files and incomplete item descriptions.
In this paper, we present an evaluation process based on a 10-fold cross validation task, that we applied to estimate accuracy of the suggestions of a P2P recommender system recently proposed in [2]. The complexity of the evaluation of this peculiar recommender is increased because of "spontaneous affinities" between users that are used instead of classical knowledge representation based strategies.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed applications*

General Terms

Algorithms, Design, Human factors, Measurement

Keywords

Peer-to-Peer, Recommender System, Complex and Social Networks, File Sharing Systems

1. INTRODUCTION

When a novel decentralized recommender system (RS) is proposed to the scientific community, we need to apply some important considerations: first of all, traditional RSs rely on a central authority that manage the complete knowledge of the domain: the users and the items they bought (or they assigned a rate to) are linked together (e.g., by means of association rules). The resulting scheme is used to derive users profiles, that are exploited for submitting recommendations to the active users. This is a very difficult task in a Peer-to-Peer system, where in principle we have not to deal with any central entities, and where no one has a complete knowledge of the domain. Moreover, profiling is strongly dependent on a preliminary tagging system, that associates each item to a set of keywords and/or high level meta information, e.g., an MP3 song can contain the name of the author, the genre (i.e., folk, rock, classical, . . .), the production year, and so on. If we focus on popular File Sharing applications, it cannot be assumed that every user inserting a file in the network properly fills the file's meta information fields. As a matter of fact, the "fake files" phenomenon is largely common in popular systems like Gnutella, eMule and BitTorrent: even the name is frequently untied from the real content of the file.

Usually, the quality of a suggestion is checked against users' feedback and/or empirical evaluations executed over well known buddy tables and users' profiles. Unfortunately, this is exactly what it is not available in the real P2P environment: the domain is open and heterogeneous, the content is poorly or unfairly structured, the users want to remain untraceable. Even if the investigation of P2P recommender systems is dedicated to forthcoming market places, actual research is constrained to deal with this kind of material, that is representative of millions of Internet users.

However, even if a trivial analysis may only bring to light the incoherent nature of the P2P world, we may use network (and social) analysis techniques in order to find useful structures in these complex networks. In particular, it can be observed that, like in the real world, even in virtual communities people create a "de facto" *word of mouth* mechanism that help both to select an item, before buying or downloading it, amongst the huge volumes of data that are available on the Web and on the P2P file sharing systems. In particular, there is the empirical evidence that *Affinity Networks*, showing a small world topology, can be built on the top of a given file sharing application. This finding has been used to provide a recommender system that can be implemented and executed "as is" in many file sharing applications [2].

The rest of the paper is organized as follows: Section 2 defines the concept of affinity networks that will be extensively used from now on. The section contains also some basic definitions from graph theory, including a brief introduction to small-world networks. After a brief survey on recommender systems known in the literature, with a focus on decentralized approaches (Section 3), the evaluation method will be presented in Section 4. DeHinter, that will be evaluated as a case study, is described in Section 5.

2. DOMAIN DESCRIPTION AND DEFINITIONS

In order to model our domain in a more formal way, let us assume that a set of users $U = \{u_1, u_2, \dots, u_n\}$ is sharing a set of items $S = \{s_1, s_2, \dots, s_l\}$. We assume a bijection between users and nodes in the system, hence the user u_i denotes both the i-th node and the i-th user. Defined $\mathcal{P}(S)$ as the power set of S, i.e. the set of all subset of S, let us introduce the function $f : U \rightarrow \mathcal{P}(S)$ in order to map users and items. In other words, $f(u_i)$ is the set of items user u_i shares. Obviously, we have that $\bigcup_{i=1}^{n} f(u_i) = S$.

To take advantage of the power of social relationships, we need to shape the concept of *affinity* among users. For this purpose, we introduce the *affinity function* $\mathit{Aff} : U^2 \rightarrow \mathcal{N}^+$ where

$$\mathit{Aff}(u_i, u_j) = |f(u_i) \cap f(u_j)| \,. \quad (1)$$

The *friendship* between users is hence defined by the number of resources they have in common.

These hypotheses enable us to introduce the idea of *"Affinity Network"* that is represented by a graph where nodes are users and an edge between users u_i and u_j exists iff they share at least m files. More formally, we define a family of graphs $G^m = (U, E^m)$, where $e_{ij}^m \in E^m \Leftrightarrow \mathit{Aff}(u_i, u_j) \geq m$.

Of course, we can define more complex *affinity functions* that consider also other kinds of user-related information such as an high level description of peer's profile or structured metadata regarding the resources shared. It is evident as the topology of G^m graphs is strongly related to degree m. Growing m, the network appears less connected since two users have to share more resources in order to be linked. On the other hand, such a stronger relationships lead to connect users with increasing level of likeness due to larger intersection of shared files.

The definition of *affinity networks* is in some way related to the concept of *data-sharing graphs* presented in [1] with some significant differences, that are deeply underlined in [2].

2.1 Small Worlds

Very briefly, Small World graphs, w.r.t. random networks, are characterized by short paths connecting most pairs of vertices and by the *transitivity* property, i.e., in many real networks if a node A is connected to a node B and B is linked to a node C, then there is a high probability that node A is also connected to C. In a social context, the friend of your friends is likely also to be a friend of you. Moreover, the transitivity property has a significant impact in the applicability of a *Affinity Network* (see Section 2) in the design of our decentralized recommender system: if a user u_a shows similar likings with user u_b and the latter shows a high affinity degree with user u_c, then users u_a and u_c almost certainly will reveal a strong likeness.

In terms of network topology, the transitivity nature leads to the presence of an elevated number of *triangles*, i.e. sets of vertices each of which is connected to each of others. In order to quantify the phenomenon we introduce the concept of *clustering coefficient* that in a small world network assumes a value much higher than in a random network.

Formally speaking, let $G = (V, E)$ be a graph, where V and E are respectively the set of vertices and the set of edges between nodes. If we define the set of *neighbors* of v_i as $V_i = \{v_j\} : v_j \in V, e_{ij} \in E$, then the *degree* of v_i is $d_i = |V_i|$, i.e., d_i is the number of neighbors of the vertex. Note that D_i, the maximum number of links between neighbors of v_i, can be defined in function of d_i; in fact, if G is a directed graph (i.e., $e_{ij} \neq e_{ji}$), then $D_i = d_i \cdot (d_i - 1)$. Otherwise (when G is undirected), $D_i = \frac{d_i \cdot (d_i - 1)}{2}$. Let $E_i = \{e_{jk}\} : v_j, v_k \in V_i, e_{jk} \in E$ be the actual set of edges between neighbors of v_i.

Watts and Strogatz [11] defines *clustering coefficient* C as the average of a local value: $C_i = \frac{|E_i|}{D_i}$.

Observe that if C_i is equal to 0, it means that the neighbors of v_i are not connected each other (i.e., $E_i = \emptyset$). Otherwise, if $C_i = 1$, then the sub-graph G_i is complete, where $G_i = (V_i \cup \{v_i\}, E_i \cup \{e_{ij} : e_{ij} \in E\})$. Furthermore, the *clustering coefficient of graph* G is defined: $C = \frac{\sum_i C_i}{|V|}$.

An alternative definition of clustering coefficient widely adopted can be found in [6], that gives different values than equation above in many networks. In general, regardless of which definition is used, the property characterizing a small world topology is a value of C considerably higher than for a random graph with the same number of vertices and edges. Anyhow, for the sake of completeness let us emphasize that our investigation is based on the definition given above and introduced by Watts and Strogatz.

2.2 Affinity Networks Analysis

In [2], in order to endorse the intuition that Affinity Networks are small worlds, we investigated the *Gnutella* file sharing network. Instead of implementing a Gnutella crawler from scratch, we modified the open-source client *Phex*, a pure Java file sharing application, multi-platform, with the multi-source download feature and able to realize an effective passive searching and snooping for files. This adapted client is forced to enter the network in *ultrapeer* mode, collecting and storing all *QueryHit* messages it forwards. The crawler ran for seven days, from 19 October to 26 October 2005, within our department laboratories, collecting (cleaned) data for 278,281 different hosts, and 714,640 distinct files.

In order to create the affinity graphs, we needed a set of pairs in the form (u_i, f_i) that means user u_i shares the resource f_i. An interesting point is to decide the criteria able to identify unambiguously both users and files in Gnutella. Instead of exploiting the name of a file, in our work we took advantage of the SHA1 hash codes that bind identifiers to the content rather than to the name of a resource. In fact, the hash codes can smooth the phenomenon of *fake files* and it can counter the presence of identical items shared with different file names. Afterward we focused our attention on the composition of file types shared by users in the Gnutella network. As expected, the majority of files shared are mp3 songs or other audio formats (58% of the overall resources), in addition we noticed that video contents represent the second largest set of items (27.8%). Finally, we generated the affinity graphs G^m from these two most popular categories in order to point out the hypothesized small-world behavior, and also generating a set of random graphs of comparable sizes (in terms of number of nodes and links). All the G^m graphs exhibited small-world patterns: in fact, we had that, for all $m = 1, \ldots, 8$, clustering coefficient is very high ($C \gg C_{rand}$) and average shortest paths are short ($L < L_{rand}$, even if the satisfaction of condition $L \approx L_{rand}$ is considered enough). The evidence of small world patterns in *affinity graphs* G^m depicts the Gnutella network as a set of strongly interconnected clusters, representing spontaneous thematic communities of users sharing kindred files. More details can be found in [2]

3. RELATED WORKS ON RECOMMENDER SYSTEMS

Recommender systems are often suggested as an effective technique to cope with the problem of information overloading in a wide range of domains. In the real life, we frequently face up a choice without having a direct experience about the feasible alternatives [9]. In the context of the huge amount of data available in the the Web and the Internet, it is evident as we need a sort of *personalized advice* able to aid the user in finding useful information. The recommender systems are in some way devoted to play this significant role.

3.1 Centralized Approaches

To date, the most relevant proposals in recommendation area are the *content-based* (e.g., [7, 5]), the *collaborative filtering*, and the *demographic* (e.g., [4]) approaches.

Each approach shows benefits and shortcomings, therefore no technique can be used effectively in all domains and for every users types. For all that, one common thread in recommender system research is the need to combine different approaches in a *hybrid* technique in order to gather the advantages of each proposal. Generally, the *content-based* and the *collaborative filtering* approaches are integrated [8, 12].

3.2 Decentralized Approaches

One common characteristic of the recommender systems described in the previous section is the use of a *centralized* client-server architecture. Focusing on the collaborative filtering approach, the information about items and ratings is stored in a central database that contains a complete knowledge of the domain. In other words, usually a recommender system creates a matrix in which rows are users and columns contain votes concerning the evaluation of items. Therefore, each vector represents a customer profile exploited to compute the correlation between users and to form good suggestions. For instance, the book store *Amazon* implements a popular centralized recommender system in which there is a whole knowledge about the books up for sale and a central repositories of user's activities and profiles.

In a decentralized environment, such as the peer-to-peer file sharing domain, these conditions are not always achievable. A first attempt to deal with a decentralized environment is proposed in [10], even if used broadcast approach to spread votes to files can bear serious scalability and efficiency issues. A different approach depicted in [3] introduces a probabilistic relevance model based on the concept of *buddies tables*.

Compared with such and other proposals, DeHinter (Section 5) shows some relevant differences:
(1) we do not employ any form of description of user's profile nor vector ratings, removing the burden of spreading this information in the network. We suppose that if a user shares a file, then she shows interest in it. In order to generate the affinity networks we need to know at least the set of files shared by a given peer, operation supported (even if often not automatically) by all file-sharing clients. Therefore, our recommender system can be easily implemented, and employed without restrictions, within any real file sharing communities, e.g. Gnutella, Emule and so on, independently to topological or structural issues. Anyhow downloading the set of *partners* of a peer along with the files list, can reduce the computational cost of making a suggestion.
(2) the system is completely *self-organizing* and *autonomous*.
(3) the recommendation engine is completely *transparent*: the suggested items are pushed to the user that just has to use the system.

4. A NOVEL EVALUATION APPROACH

The validation of a recommender system is always a quite hazardous task, because of the difficulty of modeling the tastes of a given user: we need a quantitative measure of a qualitative service, that can be fairly evaluated only after a final feedback of the final user. Novel recommender systems are proposed and evaluated by way of well known logs of user profiles and buddy tables, that contain lists of items with feedback ratings assigned by a given set of users. These information are cross linked, and the precision of the recommendation is compared with other well known (centralized) systems.
This approach cannot be applied to the pure P2P domain, if we want to use real data, because it is hard to build and maintain user profiles, and users are not required to give feedbacks to a data collector entity. Moreover, our objects are not structured, e.g., in terms of author names, genre, song or movie titles. In general, we have unique identifiers (i.e., hash values) that are not coupled with content descriptions. For example, DeHinter's recommendations (see Section 5) are made only by way of users relationships and partnership degrees. We think that this is a merit of the proposal, because it takes care of the privacy of the users without disseminating information about his/her interests; furthermore, he/she has not to waste time training his/her personal virtual assistant.

The proposed validation approach is based on a very straightforward and simple idea: the recommendation problem can be reduced to a classification task, that, for each user u, labels every file in the P2P system as "*interesting*" or "*not interesting*". Of course, we can set different recommendation criteria, in order to evaluate several degree of precisions of the proposed RS. For example, if each file s is given a numerical rate r, we can set different threshold t that divide the "*interesting*" files ($r(s) > t$) from the "*not interesting*" ones ($r(s) \leq t$).

Cross-validation is a statistical test that fits well the classification domain and that is used for validating hypotheses, especially when further data are difficult to collect. A single run of a cross validation test is made of two steps: the dataset is randomly partitioned into two parts, a training set and a test set. Training data are used to build the model, and test data are "hidden" in order to confirm and validate the initial analysis.

Our suggested empirical evaluation uses a particular kind of this test, that is called K-fold cross validation (in our analysis, $K = 10$), where the data set is partitioned in K subsets. Of the K parts, a single one is used to validate the classification, and the other $K - 1$ subsets are used for running the model. The process is repeated K times, with each of the partitions used exactly once as the test set. The results of each fold then can be averaged.

Let us consider a *dataset* containing a list of rows like this: $[f_i : u_{i1}, u_{i2}, \ldots]$, where $u_{i1}, u_{i2}, \ldots$ is the list of the peers storing file with identifier f_i. First we need to *clean* the dataset: a pre-processing phase can consist of filtering out from the set all the rows related to files owned by only one peer, because they would not be of use for discovering relationships between users. Then, we execute the 10-fold cross validation test on the cleaned dataset, that is split in 10 parts. During each fold, the recommendation outcome of all the files in the test set is calculated for each user. Of course, depending on the RS under evaluation, a given knowledge base in addition to the above dataset can be used, e.g., users' profiles, buddy tables completed with rates assigned by the active users, and so on. Notice that in traditional evaluation approaches the usage of the knowledge base is mandatory. This makes RSs not assessable without additional information.
Finally, the accuracy of the estimation is calculated as it follows: giving an user u, if file s_{k_i} is stored by u and it has not been correctly classified as "*interesting*" by a given recommendation criterion, then we have an error; if err_u is the number of errors, and n_u is the number of files stored by u and correctly classified, then the accuracy of the estimation is $\frac{n_u}{n_u + err_u}$.

5. EVALUATION OF DEHINTER: A CASE STUDY

In this section we describe *DeHinter*, a decentralized recommendation scheme that exploits spontaneous elective affinities of the users: peers in the same cluster of an affinity network share a subset of common items and are likely interested to other files popular in the cluster. The transitivity property of affinity networks may be used for enabling *reserved information lanes* between users, in order to suggest items that are potentially of interests for members of the same cluster.

Very roughly, DeHinter works as follows: Let u_x be the active user, we consider the set of neighbors $F_0(u_x)$ in the affinity network G^1. Let $F(u_x) \subseteq F_0(u_x)$ where every peer is connected to each other. If peers in $F_0(u_x)$ are called *partners* of u_x, each node in $F(u_x)$ is a *friend* of the active user. All the files stored by *friends* of u_x are candidates for a recommendation, less the files already possessed by the active user. These files are ordered by means (1) *popularity* in the cluster of partners of u_i, and (2) *partnership degree* of friends storing the missing files. The idea is that a file is very popular in $F(u_x)$, then it could be recommended to u_x, but an higher weight is given to files stored by peers that have more files in common with the active user: some friendship is stronger than others!

Even if, in principle, every file in the P2P system can be of interest of active user u_i, we focus only to files stored by friends of u_i; in fact, we consider Co-f(u_i) as the set of all the files shared by friends of u_i minus the objects she already got. For each file $s_{k_h} \in$ Co-f(u_i), let us define with $pop(s_{k_h})$ the *popularity* of s_{k_h} in Co-f(u_i). Moreover, if the *partnership degree* between two users is calculated by means of the affinity function (1), given a file $s_{k_h} \in$ Co-f(u_i), we want to compute the average partnership degree of the friends (users in $F(u_i)$) that stores s_{k_h}. Let $deg(s_{k_h})$ be this second parameter.

The *recommendation list* is defined as the ordered sequence:
$R(u_i) = (s_{k_1}, s_{k_2}, \ldots, s_{k_\ell})$, where $\ell = |\text{Co-f}(u_i)|$, and $\forall h = 1, \ldots, \ell : s_{k_h} \in$ Co-f(u_i). Files in $R(u_i)$ are sorted (and, hence, recommended), on the basis of the *weight* $w(s_{k_h}) = pop(s_{k_h}) \cdot deg(s_{k_h})$. Hence, $\forall s_{k_d} \in R(u_i) : w(s_{k_{d-1}}) \leq w(s_{k_d}) \leq w(s_{k_{d+1}})$.

The recommendation criteria we want to map into the classification task (see Section 4) are defined as follow. First of all, we compute the weight for each file in the test set. A first classification is made in terms of the following (**normal recommendation**) criterion: *if* $w(s_{k_i}) > 0$*, then file* s_{k_i} *is considered "interesting" for the active user, otherwise it is considered "not interesting".*
In the case we have a set of recommended files, i.e., $R(u) = (s_{k_1}, s_{k_2}, \ldots, s_{k_\ell})$, that is sorted by weight, we may want to define a stronger criterion, such that the given item is considered "interesting" only if it is in the higher half of the list. Therefore, if $med_w(u)$ is the median of $w(s_{k_i})$ values, with $1 \leq i \leq \ell$, then we can use the following (**strong recommendation**) criterion: *if* $w(s_{k_i}) \geq med_w(u)$*, then file* s_{k_i} *is considered "interesting" for the active user, otherwise it is considered "not interesting".*

In our experiment, we considered the set containing all the data presented in Section 2.2, that we transformed in a list of rows like this: $[f_i : u_{i1}, u_{i2}, \ldots]$, where $u_{i1}, u_{i2}, \ldots$ is the list of the peers storing file f_i. After cleaning the file as requested by the pre-processing phase, we obtained a set of $105,995$ rows. Then, we executed the 10-fold cross validation test: the dataset has been split in 10 parts, giving a test set of approximately $10,600$ rows. During each fold, the recommendation weight of all the files in the test set has been calculated for each user. The weights were calculated considering only the data in the training set (see an example in Figure 1). The classification was made using both recommendation criteria we defined above.

Table 1 reports the averaged accuracies for all the users and for each different fold of the test. The results are really good, with an average accuracy of 81% (with a confidence interval of $[66\%, 96\%]$) if the normal criterion is used. As expected the strong recommendation criterion has a lower average accuracy, even if it is still quite high (66%).

It is quite clear that only spontaneous relationships between peers were used for recommending files. Moreover, the evaluation of the system is based on what effectively the users have: it is like guessing which file a user is storing *now*. DeHinter is expected to provide good guessing even for items that the user may want to store *in the future*: a classification method that has good accuracy in the test set has statistical evidence for behaving well on unseen cases.

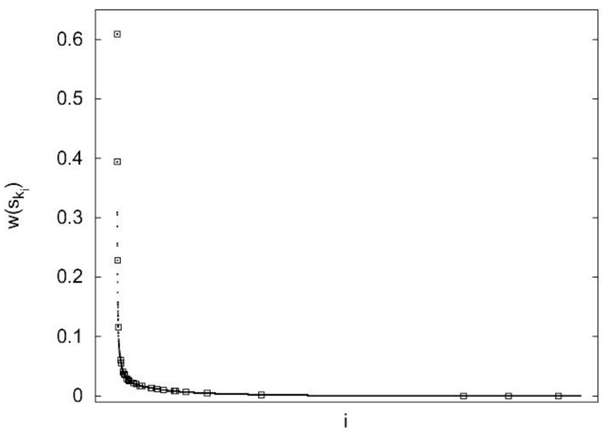

Figure 1: Given an user u, files s_{k_i} in the testing set are sorted by $w(s_{k_i})$. Files s_{k_i} are displayed with dots. In addition, if the file is owned by u, then it is also displayed with a square.

Folds	Normal Recommendation		Strong Recommendation	
	Accuracy	σ	Accuracy	σ
0	0,8145	0,145113	0,6682	0,189547
1	0,8115	0,152825	0,6549	0,194969
2	0,8174	0,149422	0,6657	0,189540
3	0,8215	0,137835	0,6766	0,186365
4	0,8261	0,143064	0,6629	0,192164
5	0,8022	0,148108	0,6453	0,199965
6	0,8097	0,148670	0,6517	0,191519
7	0,8066	0,150512	0,6595	0,194688
8	0,8035	0,166106	0,6560	0,199304
9	0,8029	0,154615	0,6574	0,204398
Tot	0,8116	0,149630	0,6598	0,194250

Table 1: 10-fold cross validation results: average accuracy and variance σ for the given recommendation criteria.

6. CONCLUSIONS

The reader should observe that we considered only one side errors, i.e., items that are erroneously considered as "not interesting": of course, we cannot make any deduction from files that are not owned by the users. Only feedback rating can say the last word on the "satisfaction degree" of the user. However, this can erroneously let the reader think that a system that recommends every file, would fare excellent according to our evaluation. Nevertheless, we have made the tacit assumption that a *flat* recommendation criterion is not applicable, and that different votes must aid the user to discriminate between more or less interesting objects. The importance of the shown results with the given strong criterion is the following: the most weighty files are considered interesting with an high probability (66%), given evidence to the intuition that a more interesting file has a greater weight. For instance, Figure 1 shows that items with higher recommendation votes are very likely interesting: squared objects correspond to files owned by the active user, and the majority of them are manifestly distributed over the median threshold (that in this case, is slightly above 0).

As a final remark, we want to stress that spontaneous relationships between actors in given systems, detected by means of network analysis, are going to be exploited in many other application domains. This implies that present and future systems' evaluations must seriously consider the constraints that can be found in complex networks that deal with unstructured data. We have the sensation of being just over the top of an iceberg.

Acknowledgment

This work has been financially supported by the Italian FIRB 2001 project number RBNE01WEJT WebMiNDS.

7. REFERENCES

[1] I. F. A. Iamnitchi, M. Ripeanu. Small-world file-sharing communities. In *The 23rd Conference of the IEEE Communications Society (InfoCom 2004)*, Hong Kong, 2004.

[2] G. Ruffo, R. Schifanella, E.Ghiringhello A Decentralized Recommendation System based on Self-Organizing Partnerships In *IFIP-Networking'06, May 2006*, LNCS 3976:618-629, Coimbra (Portugal), 2006.

[3] R. L. Jun Wang, Johan Pouwelse and M. R. J. Reinders. Distributed collaborative filtering for peer-to-peer file sharing systems. In *Proc. of the 21st Annual ACM SAC*, New York, NY, USA, 2006. ACM Press.

[4] B. Krulwich. Lifestyle finder: Intelligent user profiling using large-scale demographic data. *AI Magazine*, 18(2):37–45, 1997.

[5] K. Lang. NewsWeeder: learning to filter netnews. In *Proc. of the 12th ICML*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.

[6] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167, 2003.

[7] M. J. Pazzani, J. Muramatsu, and D. Billsus. Syskill webert: Identifying interesting web sites. In *AAAI/IAAI, Vol. 1*, pages 54–61, 1996.

[8] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proc. of UAI '01*, pages 437–444, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[9] P. Resnick and H. R. Varian. Recommender systems - introduction to the special section. *Communication ACM*, 40(3):56–58, 1997.

[10] A. Tveit. Peer-to-peer based recommendations for mobile commerce. In *WMC '01*, pages 26–29, New York, NY, USA, 2001. ACM Press.

[11] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

[12] Y. Z. Wei, L. Moreau, and N. R. Jennings. A market-based approach to recommender systems. *ACM Trans. Inf. Syst.*, 23(3):227–266, 2005.