Published in "Physics Procedia 3(5): 1887-1896 , 2010"
which should be cited to refer to this work.

Empirical comparison of local structural similarity indices for collaborative-filtering-based recommender systems

Qian-Ming Zhang[a], Ming-Sheng Shang[a], Wei Zeng[a], Yong Chen[a], Linyuan Lü[b]

[a] *Web Sciences Center, School of Computer Science and Engineering University of Electronic Science and Technology of China, 610054 Chengdu, P. R. China*
[b] *Department of Physics, University of Fribourg, Chemin du Musée 3, CH-1700 Fribourg, Switzerland*

Abstract

Collaborative filtering is one of the most successful recommendation techniques, which can effectively predict the possible future likes of users based on their past preferences. The key problem of this method is how to define the similarity between users. A standard approach is using the correlation between the ratings that two users give to a set of objects, such as *Cosine index* and *Pearson correlation coefficient*. However, the costs of computing this kind of indices are relatively high, and thus it is impossible to be applied in the huge-size systems. To solve this problem, in this paper, we introduce six local-structure-based similarity indices and compare their performances with the above two benchmark indices. Experimental results on two data sets demonstrate that the structure-based similarity indices overall outperform the *Pearson correlation coefficient*. When the data is dense, the structure-based indices can perform competitively good as *Cosine index*, while with lower computational complexity. Furthermore, when the data is sparse, the structure-based indices give even better results than *Cosine index*.

Keywords: collaborative filtering, recommender system, classification, structure-based similarity index

1. Introduction

Recommender systems or recommendation engines, resulted from a specific type of information filtering system technique, attempt to recommend informational items (films, television, music, books, etc.) which are likely to cater to the users. Up to now, many recommendation algorithms have been proposed, such as collaborative filtering [1, 2, 3], content-based analysis [4], spectral analysis [5], latent semantic models [6], heat conduction [7, 8], opinion diffusion [9, 10], and so on.

Email address: linyuan.lue@unifr.ch (Linyuan Lü)

Collaborative filtering (CF) is one of the most successful recommendation approaches which have been widely investigated in academe and adopted in e-commerce [12, 13, 14]. The basic assumption of CF is that people who agreed in the past tend to agree again in the future. In another word, similar people will have similar preference. Thus, for a target user, his/her potential evaluation on an object is estimated based on the evaluations of his/her similar users. Therefore, the most important problem of CF is how to properly quantify the similarity between users. A standard approach is using the correlation between the ratings that two users give to a set of objects, such as the *Cosine index* [15] and the *Pearson correlation coefficient*. There are three disadvantages of these two approaches. Firstly, since the rating information is the fundamental ingredient for calculating these two methods, they can not be applied to the systems without explicit ratings [16]. Secondly, their performances may decrease when data gets sparse. An evidence is shown in Ref. [17], where they found that similarity based on the relevance information, namely whether a user has voted an object, can output a better recommendation than that based on the exact rating correlations on sparse data. Thirdly, their computational complexities are relative high, especially for the huge-size and dense data sets.

In this paper, we employ six local-structure-based indices to quantify the similarity between users, which have been widely used in link prediction problem [18, 19, 20] and the classification in partially labeled networks [21]. We compare their performances with the two benchmark indices, namely *Cosine Index* and *Pearson correlation coefficient*, on two data sets, namely MovieLens and Netflix. Experimental results demonstrate that the structure-based similarity indices overall outperform the *Pearson correlation coefficient*. Furthermore, when the data is dense, the structure-based indices can perform competitively good as *Cosine index*, while with lower computational complexity. When the data is sparse, the structure-based indices give even better results than *Cosine index*.

The rest of this paper is organized as follows: we firstly introduce the framework of standard collaborative filtering and then give an introduction of the six structure-based similarity indices. The experimental results and a detailed discussion are presented in section 3. Finally, we conclude our results in section 4.

2. Method

2.1. Standard Collaborative Filtering

A rating system can be represented by a bipartite network $G(U, O, E)$, where $U = \{u_1, u_2, ..., u_m\}$, $O = \{o_1, o_2, ..., o_n\}$ and $E = \{e_1, e_2, ..., e_l\}$ are the sets of users, objects and links (labeled by ratings), respectively. We denote $r_{u\alpha}$ the rating from user u on object α. Let $\Gamma(u)$ be the set of objects that user u has voted. Then the mean rating for u is $\overline{r_u} = \frac{1}{|\Gamma(u)|}\sum_{\alpha\in\Gamma(u)} r_{u\alpha}$. According to the standard collaborative filtering, the predicted rating of user u on an unselected object α is

$$r'_{u\alpha} = \overline{r_u} + \kappa \sum_{v\in\hat{U}} s_{uv}(r_{v\alpha} - \overline{r_v}) \tag{1}$$

where $\hat{U}$ denotes the set of users that are most similar to user u, s_{uv} denotes the similarity between user u and user v, and $\kappa = \frac{1}{\sum_v |s_{uv}|}$ is for normalization.

In the framework of user-based collaborative filtering, the most important thing is to properly quantify the similarity between users. Here are two benchmark indices which have been widely used in previous works.

(1)*Cosine Index* [1, 27] — Denote by $\overrightarrow{r_x}$ the scores list rated by user x, then the Cosine similarity index is defined as

$$s_{xy}^{cos} = \frac{\overrightarrow{r_x} \cdot \overrightarrow{r_y}}{\|\overrightarrow{r_x}\| \times \|\overrightarrow{r_y}\|}. \tag{2}$$

(2)*Pearson correlation coefficient* (PCC) [1, 27] — This index is defined as

$$s_{xy}^{PCC} = \frac{\sum_{i \in O_{xy}} (r_{xi} - \overline{r_x})(r_{yi} - \overline{r_y})}{\sqrt{\sum_{i \in O_{xy}} (r_{xi} - \overline{r_x})^2} \sqrt{\sum_{i \in O_{xy}} (r_{yi} - \overline{r_y})^2}} \tag{3}$$

where O_{xy} is the set of common neighbors of user x and y, namely $\Gamma(x) \cap \Gamma(y)$.

2.2. Structure-based similarity indices

Another kind of similarity indices are defined based solely on the network structure. In another word, they merely care whether a user has voted to an object, but not the explicit score. In this paper we apply six local-structure-based indices and compare their performances with the above two benchmark methods. An introduction of these six indices are shown as follows:

(1) *Common Neighbors* (CN) — By common sense, two users, x and y, are more similar if they have voted many common neighbors (i.e., objects). The simplest measure of this neighborhood overlap is the directed count, namely

$$s_{xy}^{CN} = |\Gamma(x) \cap \Gamma(y)|. \tag{4}$$

where $|Q|$ is the cardinality of the set Q.

(2) *Salton index* [22] — The Salton index is defined as

$$s_{xy}^{Salton} = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{k(x) \times k(y)}}, \tag{5}$$

where $k(x)$ is the number of objects that user x have voted, namely $k(x) = |\Gamma(x)|$.

(3) *Jaccard Index* [23] — This index was proposed by Jaccard over a hundred years ago, and is defined as

$$s_{xy}^{Jaccard} = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}. \tag{6}$$

(4)*Sørensen Index* [24] — This index is used mainly for ecological community data, and is defined as

$$s_{xy}^{Sørensen} = \frac{2|\Gamma(x) \cap \Gamma(y)|}{k(x) + k(y)}. \tag{7}$$

(5) *Adamic-Adar Index* (AA) [25] — This index refines the simple counting of common neighbors by assigning the less-connected neighbors more weight, and is defined as:

$$s_{xy}^{AA} = \sum_{z \in O_{xy}} \frac{1}{\log k(z)}. \tag{8}$$

(6) *Resource Allocation* (RA) [19] — Consider a pair of users, x and y, the user x can send some resource to y, with their common neighbors playing the role of transmitters. In the simplest case, we assume that each transmitter has a unit of resource, and will equally distribute it between

all its neighbors. The similarity between x and y can be defined as the amount of resource y received from x, which is:

$$s_{xy}^{RA} = \sum_{z \in O_{xy}} \frac{1}{k(z)}. \tag{9}$$

Clearly, this measure is symmetric, namely $s_{xy} = s_{yx}$. Note that, although resulting from different motivations, the AA index and RA index have the very similar form. Indeed, they both depress the contribution of the high-degree common neighbors in different ways. AA index takes the $\log k(z)$ form while RA index takes the linear form. The difference is insignificant when the degree, k, is small, while it is great when k is large. Therefor, RA index punishes the high-degree common neighbors heavily.

3. Experiments

3.1. Data sets

Two data sets are used to test the algorithms: i) MovieLens [1] is a movie recommendation website, which uses users' ratings to generate personalized recommendations. ii) Netflix [2] is an online DVD and Blu-ray Disc rental service in the US. The data we used is a random sample that consists of 3000 users and 3000 movies. The basic properties of these two data sets are shown in table 1.

Table 1: The basic properties of the tested data sets.

Data set	Users	Objects	Links	Sparsity
MovieLens	943	1682	100000	6.3×10^{-2}
Netflix	3000	3000	197248	2.2×10^{-2}

3.2. Metrics

To test the algorithm's performance, the observed ratings (links), E, is randomly divided into two parts: the training set, E^T, is treated as known information, while the probe set, E^P, is used for testing and no information in this set is allowed to be used for prediction. Clearly, $E = E^T \bigcup E^P$ and $E^T \bigcap E^P = \varnothing$. In our experiment, for each user we randomly select q (the ratio between the training set and the whole data set) of his/her ratings as the training set, and the remaining $(1 - q)$ constitute the probe set.

We employ three metrics to measure each algorithm's performance: *Precision*, *Diversity* and *Popularity*. A short introduction is shown as follow:

Precision — This metric considers only the top-L objects of the recommendation list. For a target user i, the precision of recommendation, $P_i(L)$, is defined as

$$P_i(L) = \frac{R_i(L)}{L}, \tag{10}$$

[1] http://www.movielens.org
[2] http://www.netflix.com

where $R_i(L)$ indicates the number of relevant objects (namely the objects collected by u_i in the probe set) in the top-L places of recommendation list. Averaging over all the individual precisions, we obtain the precision of the whole system, as

$$P(L) = \frac{1}{m}\sum_{i=1}^{m} P_i(L), \tag{11}$$

where $m = |U|$ is the number of users in the system. Clearly, higher precision means higher recommendation accuracy.

Diversity [26] — Inter-user diversity is defined by considering the uniqueness of different user's recommendation list. Given two users i and j, the difference between their recommendation lists can be measured by Hamming distance,

$$H_{ij}(L) = 1 - \frac{Q_{ij}(L)}{L}, \tag{12}$$

where $Q_{ij}(L)$ is the number of common objects in the top-L places of both lists. Clearly, if user i and j have the same list, $H_{ij}(L) = 0$, while if their lists are completely different, $H_{ij}(L) = 1$. Averaging $H_{ij}(L)$ over all pairs of users we obtain the mean distance $H(L)$, for which greater or lesser values mean, respectively, greater or lesser personalization of users' recommendation lists. Thus, a more personalized recommendation algorithm may lead to a larger diversity.

Popularity [26] — This metric quantifies the capacity of an algorithm to generate novel and unexpected results, that is to say, to recommend less popular items unlikely to be already known about. Denote that, higher popularity corresponds to lower novelty. The simplest way to calculate popularity is to use the average collected times over all the recommended items, as:

$$N(L) = \frac{1}{mL}\sum_{i=1}^{m}\sum_{o_r \in O_R^i} k(o_r) \tag{13}$$

where O_R^i is the recommendation list of user i and $k(o_r)$ denotes the degree of object o_r.

3.3. Results

In order to investigate the effects of different parameters on the recommendation results, we perform the experiments for each group (q, N, L), where q denotes the density of the data, N represents the amount of the nearest neighbors and L is the length of the recommendation list for each user. Here we use CN, Sal, Jac, Sor, AA, RA, Cos and PCC as the abbreviations of *Common Neighbors*, *Salton Index*, *Jaccard Index*, *Sørensen Index*, *Adamic-Adar Index*, *Resource Allocation*, *Cosine Index* and *Pearson correlation coefficient* respectively.

The results of *Precision* and *Diversity* are shown in Fig. 1 and Fig. 2 respectively, where the results for $q = 0.2$ and $q = 0.8$ are selected as the representative cases of sparse and dense data respectively.

From Fig. 1, we can find that the precision rises with the increasing of N, while decreases when L goes up. This indicates that the objects that we recommend tend to be presented in the front of the recommendation list. Because the differences of the denominators of the six structure-based indices, their performances are also slightly different. A detailed discussion about correlation effect of users' degree on personalized recommendation can be found in Ref. [28]. Overall speaking, the *Salton Index*, *Jaccard Index*, *Sørensen Index* and *Cosine Index* perform better than others, and PCC performs the worst. Although the *Cosine Index* can indeed

Table 2: Popularity of MovieLens data for different sets of parameters (N, q). Here we set $L = 10$.

N, q	CN	Sal	Jac	Sor	AA	RA	Cos	PCC
N=10, q=0.2	48	45	46	46	47	45	44	46
N=10, q=0.4	108	99	99	100	107	102	97	99
N=10, q=0.6	169	153	153	153	169	163	152	147
N=10, q=0.8	227	202	201	201	226	220	200	194
N=20, q=0.2	54	50	50	50	53	51	48	54
N=20, q=0.4	119	110	109	110	118	114	108	112
N=20, q=0.6	185	167	167	167	184	179	166	166
N=20, q=0.8	245	216	215	215	244	238	215	217
N=50, q=0.2	62	58	58	58	61	58	58	63
N=50, q=0.4	131	123	123	123	131	127	122	128
N=50, q=0.6	200	182	181	181	199	195	181	186
N=50, q=0.8	261	232	231	231	259	254	231	243

give good results, the computational complexity for this method is too high to be applied to very huge-size systems. Table. 3 presents the computational time of one implement of six structure-based indices as well as *Cosine Index* and PPC. The results show that PCC has the highest computational time and followed by *Cosine Index*, and the simplest method CN consumes the least. When the data set is very huge and dense, the computation advantage of structure-based indices is prominent. As can be seen in Fig. 1, three of the six structure-based indices, namely *Salton Index*, *Jaccard Index*, *Sørensen Index*, perform competitively good as *Cosine Index*, and even better when the data is sparse. Since these three indices only involve the local information, their computational complexities are relatively low, and thus have great applications in huge-size systems.

Table 3: Computational time (in second) for one implement of eight methods. All computations were carried out in a desktop computer with Pentium(R) Dual-Core CPU E5300 (2.60 GHz) and 3GB EMS memory.

MovieLens	CN	Sal	Jac	Sor	AA	RA	Cos	PCC
q=0.2	0.172	0.172	0.172	0.173	0.182	0.172	0.179	0.221
q=0.4	0.342	0.344	0.343	0.344	0.390	0.344	0.358	0.412
q=0.6	0.516	0.516	0.518	0.518	0.641	0.523	0.522	0.641
q=0.8	0.686	0.687	0.687	0.688	0.921	0.719	0.703	0.875
q=1	0.831	0.831	0.832	0.833	1.172	0.875	0.874	1.078
Netflix	CN	Sal	Jac	Sor	AA	RA	Cos	PCC
q=0.2	1.256	1.266	1.268	1.266	1.359	1.278	1.282	1.968
q=0.4	2.505	2.505	2.508	2.506	2.891	2.562	3.016	3.921
q=0.6	3.772	3.774	3.778	3.778	4.688	3.895	5.266	6.313
q=0.8	5.015	5.016	5.031	5.044	6.641	5.249	7.703	8.844
q=1	6.234	6.235	6.251	6.258	8.875	6.594	10.23	11.33

However, *Salton Index*, *Jaccard Index*, *Sørensen Index* and *Cosine Index* are not always perform the best. As shown in Fig. 1. II (a), CN, AA and RA have higher scores, and the PCC performs the best. The reason is for the Netflix data when $q = 0.2$ (the sparsity is only 4.4×10^{-3})

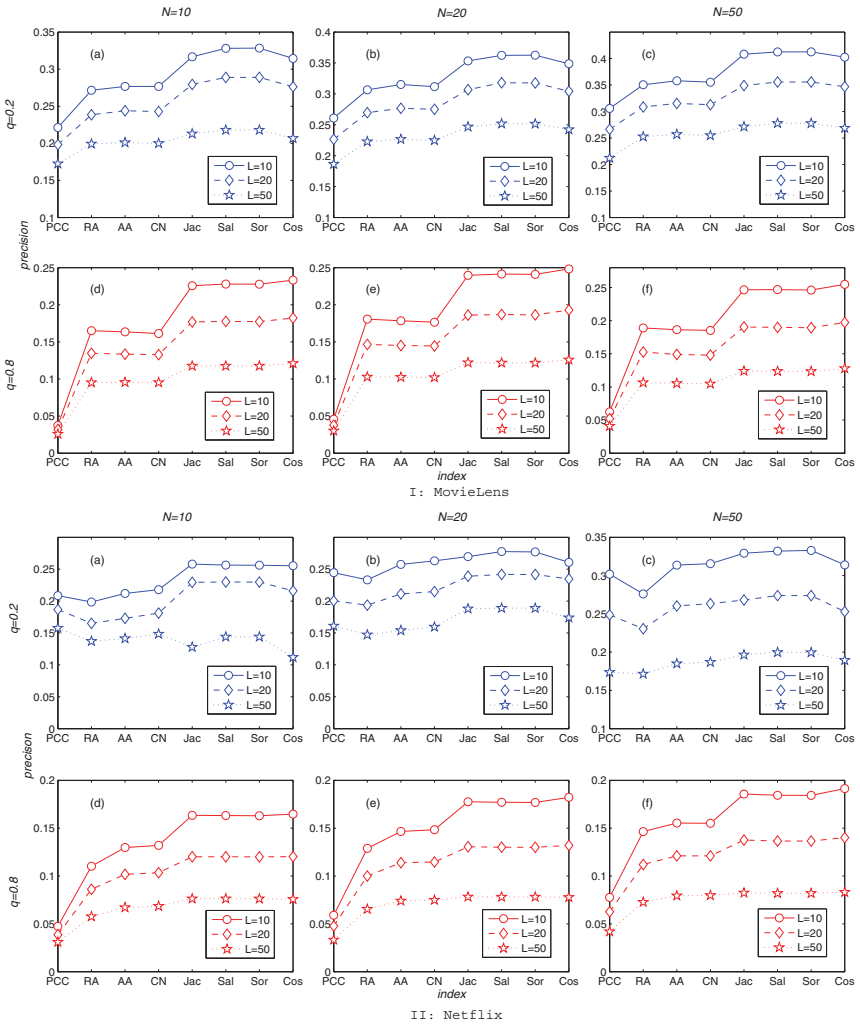

Figure 1: (Color online) Precision of the eight methods on MovieLens and Netflix. For both two data sets, (a), (b) and (c) are the results of precision when q=0.2 while (d), (e) and (f) are for q=0.8. The subgraphs (a) and (d) are the results of precision when N=10, (b) and (e) are for N=20, while (c) and (f) are for N=50.

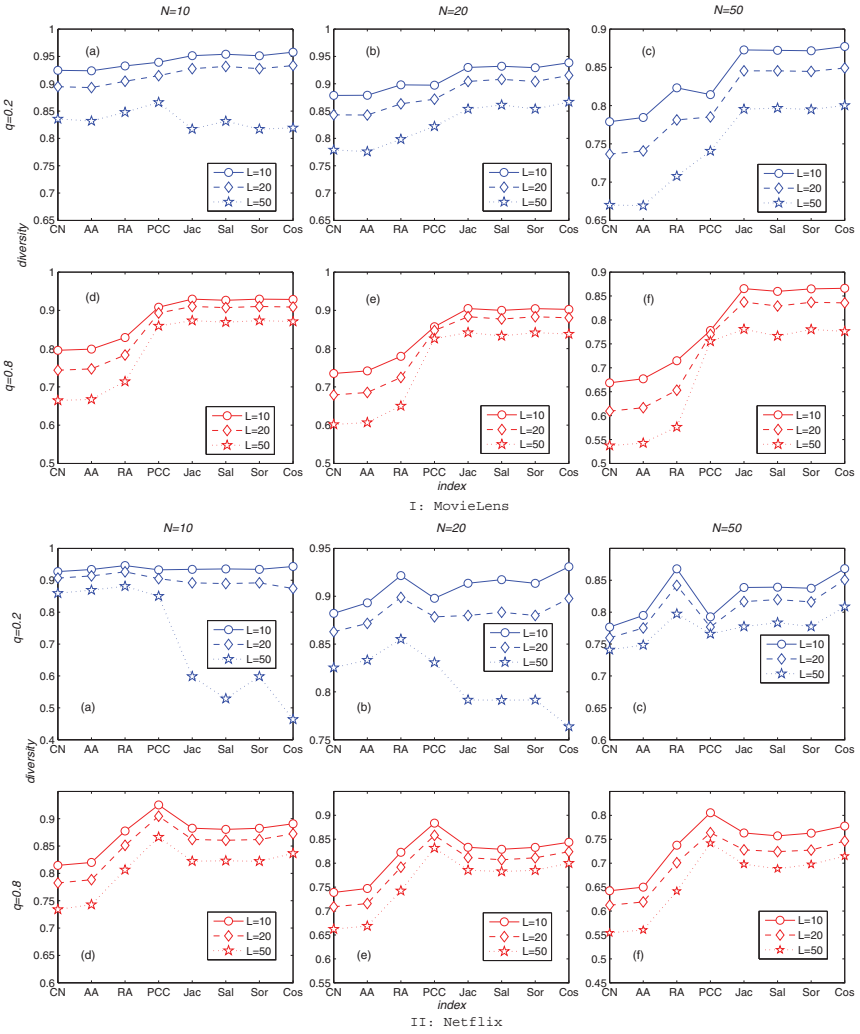

Figure 2: (Color online) Comparison of performances of the eight methods measured by diversity on two data sets, MovieLens and Netflix. For both two data sets, (a), (b) and (c) are the results of diversity when q=0.2 while (d), (e) and (f) are for q=0.8. The subgraphs (a) and (d) are the results of precision when N=10, (b) and (e) are for N=20, while (c) and (f) are for N=50.

and $N = 10$, there is too few available information (i.e., links) in the training set to generate distinguishable values of similarity.

The comparisons of diversity on two data sets are shown in Fig. 2. Generally speaking, the diversity decreases with the increasing of N or L. As shown in most subgraphs, the *Salton Index*, *Jaccard Index*, *Sørensen Index* and *Cosine Index* have higher diversity scores. However, in the subgraphs I(a), II(a) and II(b), their diversity scores are very small. In addition, comparing with Fig. 1, we find some correlations between the results of diversity and precision. As we have pointed out above, personalized objects tend to be ranked in the front of the recommendation list with the supporting of the nearest neighbors. Thus when the data is too sparse and the available nearest neighbors are too few, both the precision and diversity are striked. This might be the reason why *Salton Index*, *Jaccard Index*, *Sørensen Index* and *Cosine Index* are so frustrated when the training set is sparse.

The dependence of popularity on two parameters in MovieLens data is shown in Table. 2, where we set the length of the recommendation list $L = 10$. We find that the CN, AA and RA tend to give a little bit higher popularity. That is to say they would like to recommend popular but less novel objects. However, the differences between these eight methods are not large. Comparatively, *Consine* similarity gives the lowest popularity, and the *Salton Index*, *Jaccard Index* and *Sørensen Index* can give almost the same popularity as *Cosine* similarity. This indicates that these three indices can indeed improve the precision without damaging the novelty. Since the main findings for novelty in Netflix are almost the same, here we only show the results of MovieLends as an example.

4. Conclusions

In this paper, under the framework of user-based collaborative filtering, we adopted six structure-based similarity indices, namely *Common Neighbors*, *Salton Index*, *Jaccard Index*, *Sørensen Index*, *Adamic-Adar Index*, *Resource Allocation*, to quantify the similarity between users. We compared their performances, measured by precision, diversity and popularity, with two benchmark methods, the *Cosine Index* and the *Pearson correlation coefficient*. Experimental results on two data sets show that three structure-based similarity indices, including *Salton Index*, *Jaccard Index* and *Sørensen Index*, always have good performances unless the available information is too few, which may be caused by either the extremely sparse training set (i.e., small q) or the very few number of nearest neighbors (i.e., small N). Although the Cosine similarity can indeed give good results, the computational complexity for this method is too high to be applied to very huge-size systems. By contrast, the *Salton Index*, *Jaccard Index* and *Sørensen Index* can perform competitively good as *Cosine Index*, and even better when the data is sparse, while with much lower computational complexity. Thus these three indices have great applications in online social systems which usually contain more than ten million users or even more objects.

In recommender systems, the diversity and accuracy are usually competitive, namely they are not easily to be improved at the same time. However, this diversity-accuracy dilemma was challenged by a recent algorithm [8], where a hybrid diffusion-based method simultaneously improves accuracy and diversity. Therefore, their relationship may be far more complex than our previous understanding, yet an algorithm is undoubtedly good if it can improve both of them. The empirical results in this paper show that the *Salton Index*, *Jaccard Index* and *Sørensen Index* can give not only accurate, but also diverse and novel recommendations.

5. Acknowledgments

This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 60973069, 90924011 and 10635040.

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possiable extention, IEEE Trans. Knowl. Data Eng. **17**, (2005) 734.
[2] D. Goldberg, D. Nichols, B. M. Oki, D. Terry, Using collaboraive filtering to weave an information tapestry, Commun. ACM **35**, (1992) 61.
[3] J. B. Schafer, D. Frankowski, J. Herlocker, S. Sen, Collaborative filtering recommender systems, Lect. Notes Comput. Sci. **4321**, (2007) 291.
[4] M. J. Pazzani, D. Billsus, Content-based recommendation systems, Lect. Notes Comput. Sci. **4321**, (2007) 325.
[5] K. Goldberg, T. Roeder, D. Gupta, C. Perkins, Eigentaste: A constant time collaborative filtering algorithm, Inform. Retrieval **4**, (2001) 133.
[6] T. Hofmann, Latent semantic models for collaborative filtering, ACM Trans. Inf. Syst. **22**, (2004) 89.
[7] Y.-C. Zhang, M. Blattner, Y.-K. Yu, Heat conduction process on community networks as a recommendation model, Phys. Rev. Lett. **99**, (2007) 154301.
[8] T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, and Y.-C. Zhang, Diversity versus accuracy: Solving the apparent dilemma facing recommender systems, Proc. Natl. Acad. Sci. U.S.A. **107**, (2010) 4511.
[9] Y.-C. Zhang, M. Medo, J. Ren, T. Zhou, T. Li, F. Yang, Recommendation model based on opinion diffusion, EPL **80**, (2007) 68003.
[10] T. Zhou, J. Ren, M. Medo, Y.-C. Zhang, Bipartite network projection and personal recommendation, Phys. Rev. E **76**, (2007) 046115.
[11] Z. Huang, D. Zeng, H.-C. Chen, A comparison of collaborative-filtering recommendation algorithms for e-commerce, IEEE Intell. Syst. **22**, (2007) 68.
[12] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, *GroupLens: an open architecture for collaborative filtering of netnews, In: Proceedings of the 1994 ACM conference on Computer Supported Cooperative Work* (pp. 175-186. New York, 1994).
[13] M. Deshpande, G. Karypis, Item-based top-N recommendation algorithms, ACM Trans. Inf. Syst. **22**, (2004) 143.
[14] G. Linden, B. Smith, J. York, Amazon.com recommendations: Item-to-item collaborative filtering, IEEE Internet Comput. **7**, (2003) 76.
[15] J. S. Breese, D. Heckerman, C. Kadie, *Empirical analysis of predictive algorithms for collaborative filtering, In: Proceedings of 14th Conf. Uncertainty in Artificial Intelligence (UAI 98)* (pp. 43–52. Morgan Kaufmann, 1998).
[16] M. Claypool, D. Brown, P. Le, M. Waseda, Inferring user interest, IEEE Internet Comput. **5**, (2001) 32.
[17] M.-S. Shang, L. Lü, W. Zeng, Y.-C. Zhang, T. Zhou, Relevance is more significant than correlation: Information filtering on sparse data, EPL **88**, (2009) 68008.
[18] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, J. Am. Soc. Inf. Sci. Technol. **58**, (2007) 1019.
[19] T. Zhou, L. Lü, Y.-C. Zhang, Predicting missing links via local information, Eur. Phys. J. B **71**, (2009) 623.
[20] L. Lü, C.-H. Jin, T. Zhou, Similarity index based on local paths for link prediction of complex networks, Phys. Rev. E **80**, (2009) 046122.
[21] Q.-M. Zhang, M.-S. Shang, L. Lü, Similarity-based classification in partially labeled networks, Int. J. Mod. Phys. C (to be published), arXiv:1003.0837.
[22] G. Salton, M. J. McGill, *Introduction to modern information retrieval* (MuGraw-Hill, Auckland, 1983).
[23] P. Jaccard, Étude comparative de la distribution florale dans une portion des Alpes et des Jura, Bulletin de la Société Vaudoise des Science Naturelles **37**, (1901) 547.
[24] T. Sørensen, A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons, *Biol. Skr.* **5**, (1948) 1.
[25] L. A. Adamic, E. Adar, Friends and neighbors on the web, Soc. Netw. **25**, (2003) 211.
[26] T. Zhou, L.-L. Jinag, R.-Q. Su, Y.-C. Zhang, Effect of initial configuration on network-based recommendation, EPL **81**, (2008) 58004.
[27] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. **22**, (2004) 5.
[28] J.-G. Liu, T. Zhou, B.-H. Wang, Y.-C. Zhang, Effects of user tastes on personalized recommendation, Int. J. Mod. Phys. C, **21**, (2010) 137.