Solving the stability-accuracy-diversity dilemma of recommender systems

Article

Accepted Version

Solving the stability-accuracy-diversity dilemma of recommender systems

Lei Hou[a,*], Kecheng Liu[a,b,*], Jianguo Liu[b], Runtong Zhang[c]

[a] *Informatics Research Center, University of Reading, Reading RG6 6UD, United Kingdom.*
[b] *Data Science and Cloud Service Research Centre, Shanghai University of Finance and Economics, Shanghai 200433, China.*
[c] *School of Economics and Management, Beijing Jiaotong University, Beijing, 100044, China.*

Abstract

Recommender systems are of great significance in predicting the potential interesting items based on the target user's historical selections. However, the recommendation list for a specific user has been found changing vastly when the system changes, due to the unstable quantification of item similarities, which is defined as the recommendation stability problem. To improve the similarity stability and recommendation stability is crucial for the user experience enhancement and the better understanding of user interests. While the stability as well as accuracy of recommendation could be guaranteed by recommending only popular items, studies have been addressing the necessity of diversity which requires the system to recommend unpopular items. By ranking the similarities in terms of stability and considering only the most stable ones, we present a top-n-stability method based on the Heat Conduction algorithm (denoted as TNS-HC henceforth) for solving the stability-accuracy-diversity dilemma. Experiments on four benchmark data sets indicate that the TNS-HC algorithm could significantly improve the recommendation stability and accuracy simultaneously and still retain the high-diversity nature of the Heat Conduction algorithm. Furthermore, we compare the performance of the TNS-HC algorithm with a number of benchmark recommendation algorithms. The result suggests that the TNS-HC algorithm is more efficient in solving the stability-accuracy-diversity triple dilemma of recommender systems.

Keywords: Recommender system, recommendation stability, stability-accuracy-diversity dilemma, Heat Conduction algorithm

1. Introduction

Recent decades witnessed the explosive growth of online information, which brings a great deal of information to fit people's preferences. However, the volume of online information is considerably more than any person can possibly process [1] which has been characterised as the information overloading problem. Regarding this dilemma, the recommender system [2, 3, 4] is one of the powerful tools to offer a solution, by predicting online users' interests in terms of their historical rating or selecting behaviours.

A fundamental idea of the recommender system is to find items that are similar to the target user's historical selections. Therefore, how to properly quantify the similarity between items is of great significance for the recommendation. One of the classical techniques is the content-based method [5, 6], which detects similarity by examining whether two items share the same attributes or features. In collaborative tagging systems [7, 8] which allow users to freely assign tags on items, similarities can also be quantified using those user-created tags and this method is known as the folksonomy-based similarity [9, 10]. However, those external attributes are not always available and, in many scenarios, the data is just binary records of links from users to items. For this kind of systems, using what is

*Corresponding author.
Email addresses: lhou4397@gmail.com +44(0)7843139316 (Lei Hou), k.liu@henley.ac.uk (Kecheng Liu), liujg004@ustc.edu.cn (Jianguo Liu), rtzhang@bjtu.edu.cn (Runtong Zhang)

normally called association rules, one can mine the co-selecting patterns between items. The fundamental assumption is that two items should be similar if they are selected by the same users. For example, the Common Neighbour index defines the similarity between two items exactly as the number of users who have rated or selected both of them. By considering the popularity information of the items or the activity level of users, some variations of Common Neighbour index have been developed, such as the Salton index, the Jaccard index, the Hub-Promoted index [11], and the Leicht-Holme-Newman index[12]. Given the rapid development of network science [13], the binary relations between users and items could be modelled as the user-item bipartite networks [14, 15]. By regarding the users and items as nodes, and the accessing records as the links, the user-item bipartite network therefore simplifies the recommendation problem as predicting possible links between unconnected user-item pairs. Based on the bipartite network, physical spreading processes have been applied to quantify the similarities between nodes (users or items), such as the Resource Allocation index and Mass Diffusion index [15].

Given the enormous efforts on the investigations of similarity quantification, most of the well established recommender systems are based on similarities, such as the Amazon [16, 17], the TiVo digital video system [18] and the YouTube [19]. Despite the wide applications and investigations, the similarity-based recommender system are still with challenges, such as the cold start problem [20] and how to evaluate the effect of time [21, 22]. Notably, one of the most discussed challenges of the recommender system is how to improve the recommendations' diversity (also referred as the personalisation problem) [23, 24]. Since the popular items are welcome by most users, the recommender system could achieve high accuracy by recommending popular items, which will lead to centralised interests. However, the users always have interests that are different with each others', and hence, the recommender systems also should present diverse recommendation lists by digging out dark information. The Heat Conduction algorithm [25, 26] is one of the most discussed such methods, which has very high diversity but low accuracy. Regarding to the accuracy-diversity dilemma, numerous efforts have been devoted [27, 28, 29]. However, the performances of these algorithms highly depend on the similarity measures.

Recently, the stability property of similarity measures for user-item bipartite networks has been investigated [30]. While similarity measures are trying to reveal the real similarities in terms of the wiring patterns, they should stand in presence of noises such as random wiring. Additionally, although the real similarity between two specific items may evolve in time which leads to the instability, the similarity measures should at least be stable when evaluating with different samples of the network. However, the experimental results indicate that most of the measures may generate totally different evaluations of the similarity when using different samples even from the same period of data. Therefore, a serious question raises that, if the similarity measures are unstable evaluating the similarities, how could one be sure of that the measured similarity is the reflection of real similarity. Furthermore, given the recommender systems highly depending on the similarity quantifications, those unstable similarity measures offering inappropriate quantifications puts the system at risk, *i.e.* the recommendations will also be unstable. Recommendations to become unstable may cause risks such as 1) users finding recommendations unreasonable which leads to bad experiences, and 2) the uncontrollable performance of the recommendation algorithm in practical applications. The stability problem of similarity quantification and recommendation should be of both theoretical and practical concerns. From the theoretical perspective, if the extracted similarity is unstable, it would be hard to evaluate weather a user is interested in an item or not. From the practical perspective, the stability problem would be a gap between laboratory investigation and real-time application because practical systems are always vastly evolving.

Note that, there are some related researches on the stability problem of recommender systems, such as Adomavicius and Zhang (2012) [31, 32]. They define the stability as the consistency between the original recommendations and the recommendations using the combination of the historical data and some of the original recommendations (assuming some of the original recommendations have been adopted by the users). The stability of the present paper is defined differently from their study [31, 32]. While Adomavicius and Zhang used the output (commendations) of the first recommendation experiment as the input (historical records) of the second recommendation experiment to examine the consistency of the prediction, which could be regarded as the recommendation algorithm's self-consistency, we explore the influence of the users' real behaviour growth on the similarity quantification and recommendation change. Other studies also have discussed the systems' ability to remain stable under malicious attached (records faked for specific purpose) [33, 34], which also been referred as the robustness of the recommender systems, while we study the systems' stability with its own natural evolution.

Similar to the accuracy problem, by only recommending popular items, the system could have very high stability. However, the recommender system then falls again into the dilemma that whether should the recommender system

recommends popular items to achieve high stability and accuracy or recommends unpopular items to achieve high diversity. So, there rises the triple dilemma of stability-accuracy-diversity. By ranking the similarities in terms of stability and only considering those most stable ones, the present paper applies a top-n-stability method based on the Heat Conduction algorithm to solve the stability-accuracy-diversity triple dilemma. The results suggest that, our method could not only significantly improve the recommendations' stability, but also gains both high diversity and accuracy simultaneously. The top-n-stability method based on the Heat Conduction algorithm is arguably better than classical methods as a solution to the triple dilemma.

2. Materials and Methods

2.1. Data

This paper introduces four benchmark data sets to conduct the experiments of the top-n-stability recommendation. Basic statistics of each data set are shown in Table 1. Those data sets are widely used to investigate the user-item bipartite network, and evaluate and test the recommendation algorithms. The *MovieLens* and *Netflix* are movie web sites in which users could watch and rate movies. The *Last.fm* is a music web site allowing users to collect different artists' music. The *Epinions* allows users to share reviews on products with each other. As the recommendation experiments in this study are to predict links for each user, here we ignore the rating information and only take the wiring patterns between users and items, *i.e.* which user selected which items.

Table 1: **Basic statistical properties of the utilised data sets**. In the table, M, N and T stands for the number of users, items and total links (user-item relations) respectively. The sparsity is the deviation between existed links and possible links, *i.e.* $T/(M \cdot N)$. Those data sets have different scales and sparsities.

data set	M	N	T	Sparsity
MovieLens	5547	5850	698054	2.15×10^{-2}
Netflix	8608	5081	419247	9.59×10^{-3}
Last.fm	1885	6953	82155	6.27×10^{-3}
Epinions	27923	30073	419674	5.00×10^{-4}

2.2. Heat Conduction Algorithm

A recommender system could be modelled as a user-item bipartite network as Fig. 1 shows. In the network, there are two sets of nodes, *i.e.*, a set of items denoted by $O = \{o_1, o_2, \cdots, o_N\}$ and a set of users denoted by $U = \{u_1, u_2, \cdots, u_M\}$ where N and M are the numbers of items and users respectively. Therefore, the historical data could be described by an adjacency matrix $\mathbf{A} = \{a_{uo}\}_{M,N}$ where $a_{uo} = 1$ if there is a link between user u and item o (o is collected by u), and $a_{uo} = 0$ otherwise.

Assuming an initial item's temperature as 1, and letting the temperature conducts from items to users (Fig. 1 (a) - (b)) and then back to items (Fig. 1 (b) - (c)), the Heat Conduction algorithm (we denote with HC henceforth) regards the final temperature possessed by an item as the similarity from the initial item to the target item [28, 26]. For each step, a node receives the average temperature of all its neighbour (connected nodes). Figure 1 shows an example of the conduction process. Initially, we assume the item o_1 has a temperature of 1, and that of others are 0. When the temperature conducting to users, a user's temperature will be the average of his/her connected items', *i.e.* $\sum_{o \in O} a_{uo} t_o / k_u$, where k_u is the number of items that user u selected and t_o is the temperature of item o. Similarly, when the temperature conducting back to items, an item's temperature would also be the average of all the users that connected to it, *i.e.* $\sum_{u \in U} a_{uo} t_u / k_o$ where k_o is the popularity of item o (how many users have selected it). We then have the similarity from an initial item o_i to any other item o_j, which reads,

$$s_{o_i o_j} = \frac{1}{k_{o_j}} \cdot \sum_{u \in U} \frac{a_{uo_i} \cdot a_{uo_j}}{k_u}. \tag{1}$$

Note that, the similarity calculated by the HC algorithm is directed, *i.e.* $s_{o_i o_j} \neq s_{o_j o_i}$. And the similarity from o_i to o_j means that, if a user selected o_i historically, the likelihood of o_j to be selected by him/her in the future is $s_{o_i o_j}$. The

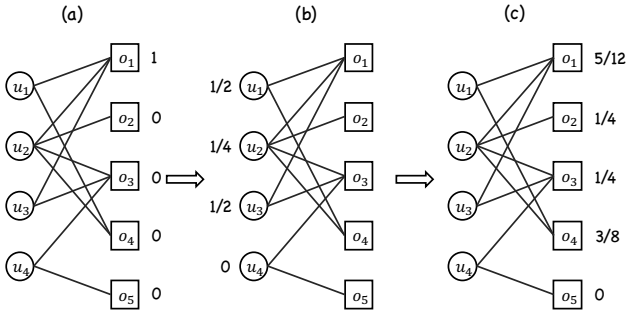

Fig. 1: **An example of the user-item bipartite network to illustrate the calculation of Heat Conduction algorithm.** In the figure, the circles represent users and the squares are items. If a user selected an item, there would be a link connecting the corresponding user and item. The numbers in the figure represent the temperature of each node to illustrate the conduction process of the heat. When calculating the similarity from an item o_1 to others, one can assume the item o_1 has a temperature of 1 while that of others are all 0 as shown in subplot (a). In the first step of the conduction, each user will receive the average temperature of his/her neighbours (items), and thusly the user temperatures are 1/2, 1/4, 1/2 and 0 for u_1, u_2, u_3 and u_4 respectively. For the second step of the conduction, the items will in turn receive the average temperature of its neighbours (users) which leads to the final temperatures of 1/4, 1/4, 3/8 and 0 for o_2, o_3, o_4 and o_5 respectively. The final temperature of an item then represents the similarity between it and the original item o_1. For example, the similarity $s_{o_1o_2} = 1/4$ while $s_{o_1o_5} = 0$.

similarity between two items calculated by the HC measure, $s_{o_io_j}$, is based on the common neighbour of those two items, *i.e.* how many users selected both of them, and weighted by each user's activity level k_u. The HC measure further normalised the similarity by dividing the target item's popularity k_{o_j}.

When one recommends to a target user u, the system calculates a score for every item that has not been selected by user u according to the similarities. The score of an item o_j for the target user u, w_{uo_j} reads,

$$w_{uo_j} = \sum_{o_i \in O} a_{uo_i} \cdot s_{o_io_j}. \tag{2}$$

The score w_{uo_j} could be regarded as the likelihood of the target user u potentially being interested in the item o_j. Normally, the system recommends L items with the highest scores to users, and those L items are what the system predicts to have the highest potential to hit the target user's interests. In this paper, we investigate the recommendation performance with recommendation length of 10, 20 and 50 respectively regarding practical systems.

2.3. Top-n-Stability Method

Despite that, item similarities have been widely applied in recommender systems, it has been found to be unstable as the network structure changes. To measure the stability of similarity, we firstly divide the data randomly into two subsets, i.e. each link of user-item has a probability of 50% to be assigned to one subset, or the other subset otherwise. Hence, we can calculate the similarities for two subsets respectively using an arbitrary similarity measure (in this paper, the HC measure). We denote the similarities between two items o_i and o_j in two subsets of data as $s'_{o_io_j}$ and $s''_{o_io_j}$ respectively. A proper measure should result in the same similarity for two subsets, *i.e.* $s'_{o_io_j} = s''_{o_io_j}$. Therefore, the more difference between the two similarities $|s'_{o_io_j} - s''_{o_io_j}|$, the more unstable the similarity is. Considering the similarities may be of totally different scales, we further normalise the difference, leading to the definition of the stability of similarity from o_i to o_j, $\delta_{o_io_j}$ which reads,

$$\delta_{o_io_j} = \frac{|s'_{o_io_j} - s''_{o_io_j}|}{s'_{o_io_j} + s''_{o_io_j}}. \tag{3}$$

Consequently, the stability of similarity between any pair of items would be in the range [0, 1], and the larger the value of $\delta_{o_io_j}$ is, the more unstable the similarity should be considered. Additional to the unstable nature of HC similarity as reported recently [30], the similarity stability has a strong correlation with the item popularity, as shown in Figure 2 (a-d). The results suggest that, the similarities directing to the unpopular items are generally less stable.

Considering that, some of the similarities are quite unstable when the network structure changes, the top-n-stability method is to only consider the most stable similarities when making recommendations.

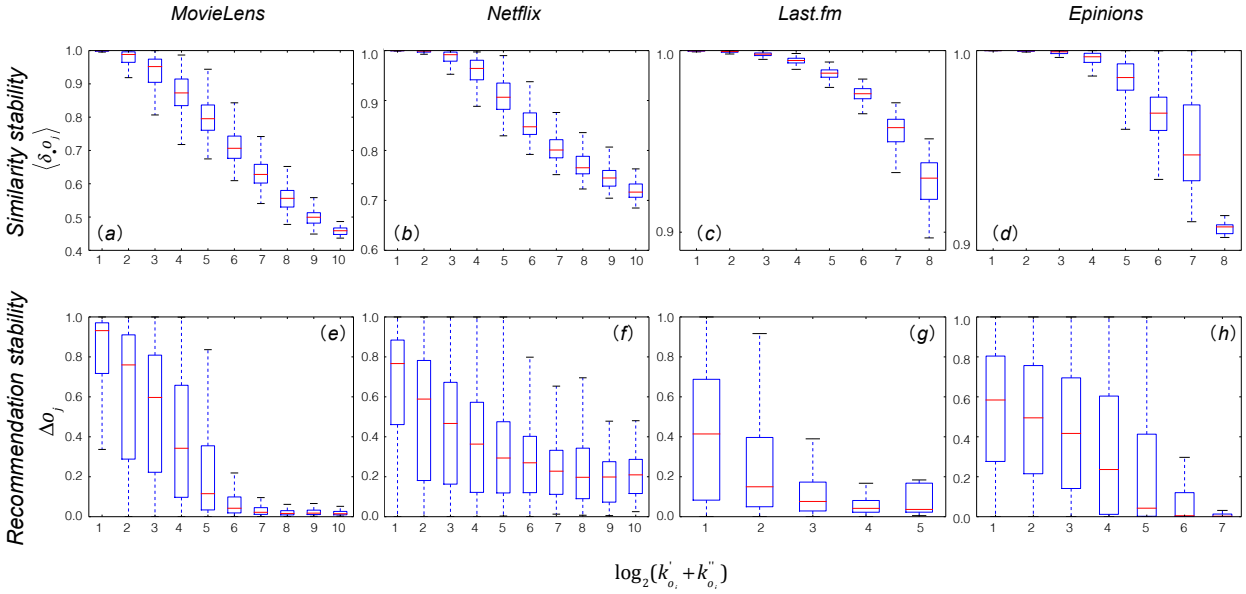

Fig. 2: (Colour online.) **Box-plot of item popularity versus similarity stability (top row a-d) and recommendation stability (bottom row e-h) respectively.** The horizontal axis is the popularity of the items, where k'_{o_j} and k''_{o_j} represent the items o_j's popularity in two subsets respectively. In the top row, $\langle \delta_{\cdot o_j} \rangle$ means the average stability of similarities directing to item o_j, *i.e.* $\sum_{o_i \in O} \delta_{o_i o_j}/N$, where N is the number of items. Generally, both of the similarity stability and the recommendation stability are mainly caused by the unpopular items.

In the traditional recommendation procedure, if the target user u has selected an item o_i in the training set, the similarity to any other item o_j, $s_{o_i o_j}$, will be contributed to the score of the item w_{uo_j}. Here in the top-n-stability method, we assume the item o_i only contribute to n items that have most stable similarities with it. Consequently, one can firstly rank the possible similarities starting from an item o_i, *i.e.*, $\{s_{o_i 1}, s_{o_i 2}, \cdots, s_{o_i M}\}$, in terms of stability from low to high values and then only consider the n most stable similarities of those. In other words, we update the similarities as,

$$s_{o_i o_j} = \begin{cases} s_{o_i o_j} & if\ Rank(\delta_{o_i o_j}) \leq n \\ 0 & if\ Rank(\delta_{o_i o_j}) > n \end{cases} . \tag{4}$$

Then, we could use the updated similarity matrix to make recommendations. In the recommendation experiments, n is a tunable parameter which could be gradually changed to explore that to what extent should the unstable similarities be removed. As the similarity stability is highly correlated with the item popularity, one may be interested in another similar technic of making recommendations which is to rank the popularity instead of stability. Thusly, a Top-n-Popularity (TNP) method could be conducted by letting the historically selected items only contribute to the n most popular items instead of most stable items. However, the ranking of popularity would be the same for every user while the stability ranking is specific to each item.

2.4. Evaluation Metrics

The present paper evaluates the recommendation performance from three aspects, namely the stability, accuracy and diversity.

In each of the experiments, we equally divide the data set into two subsets at random as reported in section 2.3. We then take those two subsets of data as the training set and the probe set by turns. Consequently, the recommendation is to use the training set as the users' historical records to try to retrieve the missing records from the probe set. To do that, we generate a ranking list for every user based on the training set according to the recommendation algorithm as shown by eq. (2) (in this paper, the top-n-stability based on Heat Conduction algorithm). The higher an item ranks in the list, the more likely the recommendation algorithm considers it to fits the target user's interest. Furthermore, the L items at the top of the ranking list would be recommended to the user.

2.4.1. Stability

As the two subsets are taken as the training set and the probe set by turns, there would be two recommendation lists for every user. For a specific user u, if an item o_j ranked at the top L in one of the lists, it means the system predicts the item o_j to be potentially interested by the user u according to that subset. Therefore, we define the item o_j's rank in another list as the recommendation stability of item o_j for user u, denoted with Δ_{uo_j}, which could be described as

$$\Delta_{uo_j} = \begin{cases} r'_{uo_j}/(N - k''_u) & if\ r''_{uo_j} \leq L \\ r''_{uo_j}/(N - k'_u) & if\ r'_{uo_j} \leq L \end{cases}, \tag{5}$$

where r'_{uo_j} and r''_{uo_j} is the rank of item o_j in user u's two recommendation lists respectively, and N is the total number of items. To further quantify the stability of each recommendation experiment, we use the average value of all the recommended items' stability to describe the recommendation's performance, which reads,

$$\langle \Delta \rangle = \frac{1}{|U|} \sum_{u \in U} \sum_{o \in \Omega_u} \frac{\Delta_{uo}}{|\Omega_u|}, \tag{6}$$

where Ω_u is the set of items that are recommended (rank at the top L in the list) to user u according to either subset but have not been collected by the user in the another subset. Note that, if an item o_i has been selected by a user in one of the subsets, it would be ranked at the bottom of the list. If in the another list, the item o_i is in the top L positions (which is an accurate prediction), the stability would be approximately equals to 1 but cannot be regarded as unstable. Therefore, we don't include those items that have already been selected by the user u historically in either subset to the set Ω_u.

According to this definition, a low value of recommendation stability $\langle \Delta \rangle$ means the system gives similar evaluation of the potentials of items being selected using two subsets of data. On the other hand, a large value of recommendation stability $\langle \Delta \rangle$ would indicate that, for a particular user, the potential of a particular item would be evaluated to be high in one subset but low in another. Accordingly, the lower value of stability $\langle \Delta \rangle$ means the recommendation is more stable.

Similar to the similarity stability, the recommendation stability of each item also has strong correlation with item popularity. As shown in Figure 2 (e-h), although the recommendation stabilities Δ empirically distribute in a wide range (sometimes almost from 0 to 1) for each popularity level, the unpopular items are relatively less stable to be recommended.

2.4.2. Accuracy

One of the most used accuracy metrics is the ranking score RS. Assuming the users' selecting behaviours are rational which could represent the users' preference, those records in the probe set should be ranked at the top of the ranking list. If we denote the rank of an item o in user u's list with r_{uo_i}, the ranking score RS reads,

$$RS = \frac{1}{|\Gamma^{probe}|} \sum_{u \in U} \sum_{o \in \Gamma_u^{probe}} \frac{r_{uo}}{N - k_u^{training}}, \tag{7}$$

where Γ_u^{probe} is the set of items in the probe set that user u collected, $|\Gamma^{probe}|$ is the size of the whole probe set and $k_u^{training}$ is the number of items that user u selected in the training set. Consequently, the accuracy RS ranges in (0,1) and the larger value of RS represents the less accurate recommendation and the smaller value of RS stands for the more accurate recommendation.

Additional to the ranking score RS, the present paper considers two more metrics that evaluate the accuracy of the recommendation list with length L, namely, the precision and recall. Both of the two metrics are based on how many items in the probe set that been retrieved by the recommendation list with length L for a target user u, denoted with $h_u(L)$. One may interest in how many items in the list with L recommendations are accurate or how many of the user u's k_u^{probe} deleted selections have been retrieved. Hence, we could have the precision $p_u(L) = h_u(L)/L$ and recall $r_u(L) = h_u(L)/k_u^{probe}$. The overall Precision $P(L)$ and Recall $R(L)$, therefore, could be calculated by averaging over all the users.

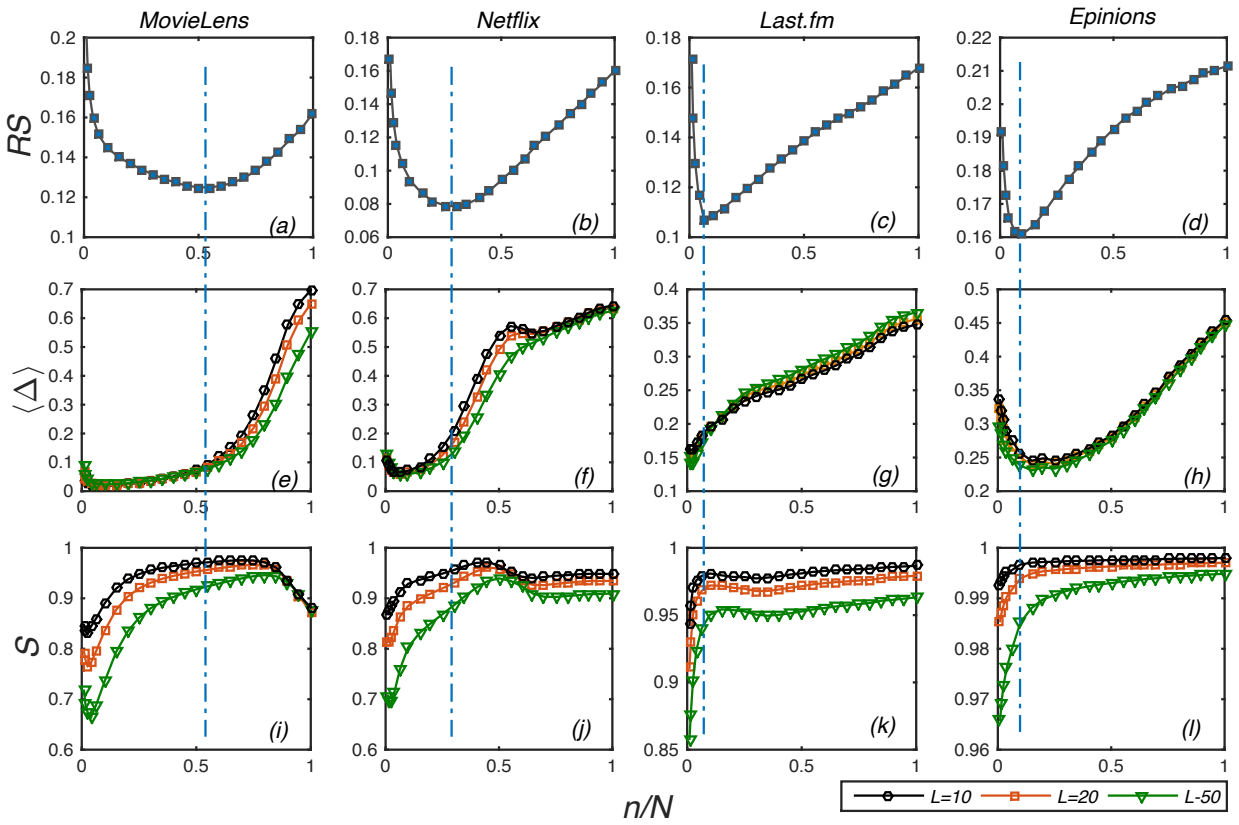

Fig. 3: (Colour online.) **Recommendation performances, namely, the accuracy RS, stability $\langle\Delta\rangle$ and diversity S of the TNS-HC algorithm on each of the four data sets.** As the number of items N varies for each data set, we take the normalised top-n, i.e. n/N to illustrate the horizontal axis. And $n/N = 1$ reveals the original HC recommendation algorithm. Note that, due to the definitions of the three metrics, lower values of the stability $\langle\Delta\rangle$ and the accuracy RS represent better performances, while higher values of the diversity S mean better performances. While each of the metrics has the optimised value of n/N, we take the n/N minimising the accuracy RS as the optimised value for the overall performance, as identified by the red dashed lines in the figure. In practical cases, one can take value optimising any function of metrics. Note that, all the results of the recommendation are averaged over 20 independent experiments.

2.4.3. Diversity

As shown in Figure 2, recommending unpopular items is more likely to be unstable. To enhance the recommendation stability, the system can recommend only the popular items which will also guarantee the accuracy. However, recommending popular items would result in that, all the users' recommendation lists are very similar to each other. Therefore, many studies have argued that, the diversity should be an important objective of recommender system [24, 23]. To evaluate the variety of different users' recommendation list, the diversity S is defined as the average value of the Hamming distance between every pair of users' recommendation lists, which reads,

$$S = \langle H_{ij} \rangle = 1 - \sum_{i,j \in U} \frac{Q_{ij}}{L}, \tag{8}$$

where Q_{ij} is the number of common items in user i and user j's recommendation list with length L. Hence, $S = 1$ represents the totally diverse condition, in which every user's list is completely different with others, and $S = 0$ means every user has totally the same recommendation list.

3. Results

3.1. Top-n-Stability Recommendation

By considering only the most stable similarities, we perform the top-n-stability method based on the HC algorithm (we denote this recommendation method with TNS-HC henceforth). We gradually change the number of stable similarities that to be considered in the recommendation to explore that, to what extent should the unstable similarities be removed to gain high recommendation stability $\langle\Delta\rangle$, accuracy RS and diversity S simultaneously. The recommendation performances on four data sets are reported in Fig. 3. In the figure, $n/N = 1$ gives the standard HC algorithm, which uses all the similarities in the recommendation.

Table 2: **Numerical results of standard HC and the optimised TNS-HC algorithm.** In the table, $(n/N)^o$ represents the optimised value of n/N minimising the accuracy RS. For the ratios, the numerical value is calculated as $|m^{HC} - m^{TNS-HC}|/m^{HC}$ for every metric m. And if for a metric, the performance of the TNS-HC algorithm is better than the one of HC algorithm, the value would be marked as positive (+), and vice versa.

		MovieLens			*Netflix*			*Last.fm*			*Epinions*		
L		10	20	50	10	20	50	10	20	50	10	20	50
HC	RS		0.162			0.16			0.131			0.211	
	$\langle\Delta\rangle$	0.697	0.651	0.552	0.643	0.637	0.625	0.347	0.356	0.366	0.453	0.451	0.446
	S	0.882	0.871	0.874	0.948	0.935	0.909	0.986	0.979	0.963	0.997	0.997	0.994
TNS-HC	$(n/N)^o$		0.55			0.3			0.06			0.1	
	RS		0.125			0.078			0.107			0.161	
	$\langle\Delta\rangle$	0.092	0.083	0.077	0.208	0.17	0.134	0.184	0.179	0.173	0.256	0.249	0.238
	S	0.971	0.957	0.923	0.955	0.93	0.885	0.979	0.968	0.94	0.996	0.994	0.985
Ratio %	RS		+22.83			+51.25			+18.32			+23.7	
	$\langle\Delta\rangle$	+86.8	+87.3	+86.1	+67.1	+73.3	+78.6	+46.9	+49.7	+52.7	+43.5	+44.8	+46.6
	S	+10.1	+9.9	+5.6	+0.73	-0.53	-2.64	-0.7	-1.12	-2.38	-0.1	-0.3	-0.9

While the standard HC ($n/N = 1$) gives highly diverse recommendations with diversity S generally larger than 0.9, the recommendation lists are quite unstable (low stability $\langle\Delta\rangle$). In the *MovieLens* and *Netflix* data set, the stability $\langle\Delta\rangle$ even goes beyond the value of random scenario. If recommending items uniformly at random, the items that recommended by one list would randomly distribute in the another list, which leads to the random stability $\langle\Delta^{rand}\rangle = 0.5$. According to the definition, the HC algorithm experts in recommending unpopular items. However, the unpopular items are generally less stable in the system as illustrated in Fig. 2. Hence, the standard HC algorithm recommending unpopular items would lead to the poor stability $\langle\Delta\rangle$ of the recommendation.

The TNS-HC algorithm can largely improve the recommendation stability $\langle\Delta\rangle$ by removing the unstable similarities. Taking $L = 20$ as an example, with only the stable information, the stability $\langle\Delta\rangle$ is improved by 87%, 73%, 50% and 45% for *MovieLens*, *Netflix*, *Last.fm* and *Epinions* data set respectively, as shown in Table 2. More surprisingly, the recommendations are more accurate when the unstable similarities been removed. On the other hand, the high-diversity advantage of the HC algorithm is retained by the TNS-HC algorithm. For the *MovieLens* data set, the diversity is also improved by 9.8% to a quite diverse level $S(20) = 0.96$. As to the other three data sets, the diversity of the standard HC algorithm has already reached a quite high level ($S > 0.9$) due to the sparsity of the data set (Table 1). Hence, the TNS-HC dose not improve the diversity in these three data sets further. Nevertheless, the TNS-HC algorithm simultaneously gaining high stability, high accuracy as well as high diversity, is an efficient method to solve the stability-accuracy-diversity triple dilemma. Additionally, the stability $\langle\Delta\rangle(L)$ and the diversity $S(L)$ for recommendation with different recommendation list length L have very similar behaviour against the change of the parameter n. In other words, the length of recommendation list does not significantly influence the top-n-stability method on the HC algorithm.

As to the two other accuracy metrics, namely the precision and recall of the recommendation, the results are similar to the ranking score RS, as shown in Fig. 4. The standard HC algorithm ($n/N = 1$) gives relatively low precision and recall values. When the unstable similarities are removed, the precision $P(L)$ and recall $R(L)$ could be largely improved.

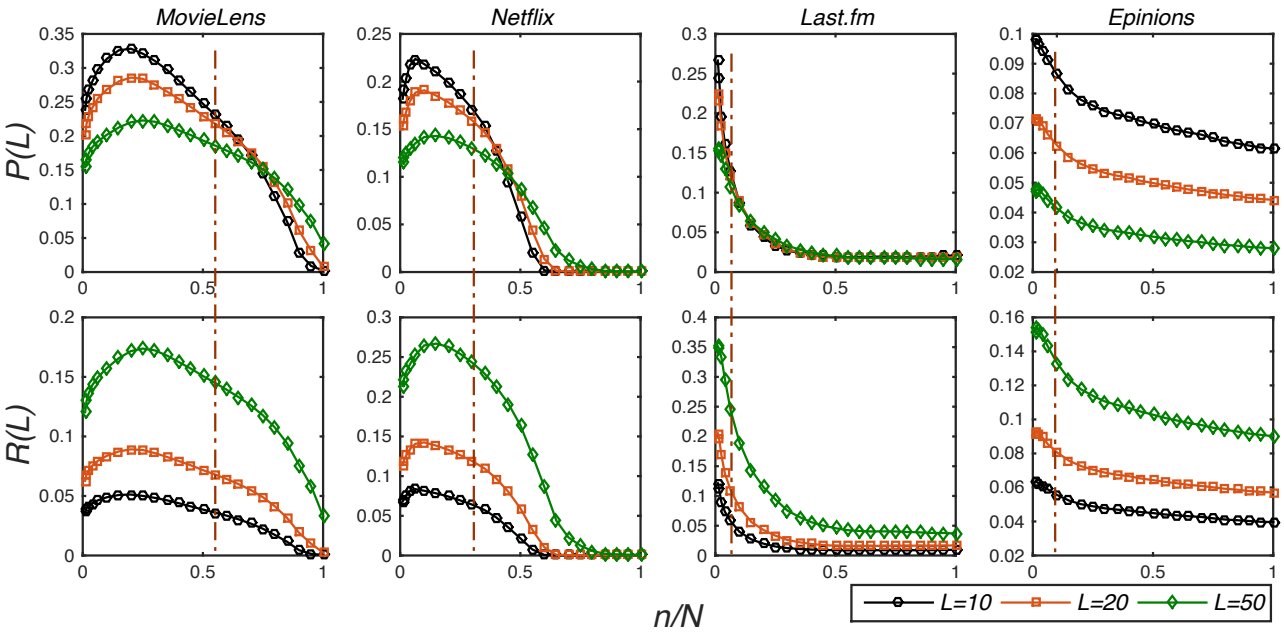

Fig. 4: (Colour online) **Precision $P(L)$ and recall $R(L)$ of the TNS-HC with different lengths of recommendation list.** The dashed line in each column is the optimised value of n/N minimising the ranking score RS which is reported in Fig. 3. As the length of recommendation list L does not influence the value of ranking score RS, those results with different lengths have the same optimised value of n/N. Note that, the higher values of precision and recall represent more accurate performances.

3.2. Comparison with Other Algorithms

The success of the TNS-HC algorithm gives a solution to the triple dilemma, which could generate accurate, diverse and most importantly, stable recommendations. Here we further compare the TNS-HC algorithm with some of the benchmark recommendation algorithms.

As the TNS-HC algorithm is based on the HC algorithm, we firstly compare the proposed method with some of the other improvement of the HC algorithm such as the HC itself, the Improved Heat Conduction (IHC) [25], the Top-n-popularity algorithm based on HC (TND-HC) as discussed in the section 2.3, the Biased Heat Conduction (Biased-HC) [28] and a hybrid algorithm combining the Heat Conduction and Mass Diffusion algorithms (HC+MD) [29]. Note that, the Biased-HC and HC+MD algorithms also involve tunable parameters similar to the TNS-HC algorithm. To be a baseline for the comparison, some of the classical recommendation techniques are also included. The Common Neighbour (CN) is the most fundamental method in the similarity-based recommendation, which is widely used in practical systems due to its accessibility for both users and system administrators. Considering the popularity information of the items, two variations of the CN index, namely the Salton index (SAL) and the Jaccard index (JAC) are also widely discussed. Another diffusion-based method is the Mass Diffusion (MD) [15], which could achieve very high accuracy. While the CN-based methods consider the population of two items' neighbourhood (users who selected both of them), it has been argued that, the density of the neighbourhood is also very important for the evaluation of the similarity [35, 36]. Accordingly, an algorithm has been proposed by considering both the number of the common neighbours and the number of local links among those common neighbours (denoting with LCP) [36]. Note that, the LCP algorithm calculates directly the similarity between users and items and thusly could rank the items in terms of the similarity to a target user as the recommendation list. Considering that the ratings are available in most online systems, here we consider two rating-based recommendation algorithms, namely the Cosine index (COS) and the Pearson Correlation index (PC). For the algorithms referred here, the detailed techniques and definitions could be found either in the corresponding original article or in review articles such as *Ref. [4]*.

Table 3 reports the stability $\langle\Delta\rangle$, accuracy RS, and diversity S of the recommendations resulted from each of those algorithms referred in this study. One can find from the numerical comparison with the HC-based algorithms that, the proposed TNS-HC algorithm has relatively good performances for all the three metrics. As to other HC-based

Table 3: **Comparisons of recommendation performances among different algorithms.** The results of TNS-HC, TNP-HC, HC+MD and Biased-HC are based on the optimised parameter respectively. The recommendations are all based on the list length $L = 20$. Note that, the Last.fm data set has no rating information for the bipartite network, and consequently the rating-based COS and PC algorithms have no results for the data set. The random recommendation is to ranking the items randomly for each user as their recommendation list. All the results are averaged over 20 independent experiments.

		MovieLens			*Netflix*			*Last.fm*			*Epinions*		
		$\langle\Delta\rangle$	*RS*	*S*	$\langle\Delta\rangle$	*RS*	*S*	$\langle\Delta\rangle$	*RS*	*S*	$\langle\Delta\rangle$	*RS*	*S*
HC-based	TNS-HC	0.083	0.124	0.956	0.17	0.078	0.93	0.179	0.107	0.968	0.249	0.16	0.993
	HC	0.65	0.162	0.871	0.636	0.16	0.934	0.356	0.167	0.979	0.451	0.211	0.997
	IHC	0.622	0.216	0.848	0.584	0.227	0.93	0.419	0.171	0.955	0.43	0.226	0.997
	TNP-HC	0.616	0.141	0.961	0.637	0.093	0.936	0.508	0.157	0.973	0.474	0.207	0.996
	HC+MD	0.251	0.125	0.899	0.071	0.08	0.754	0.14	0.104	0.912	0.376	0.181	0.997
	Bised-HC	0.235	0.129	0.885	0.048	0.085	0.712	0.034	0.111	0.785	0.36	0.183	0.996
CN-based	CN	0.002	0.152	0.435	0.006	0.09	0.422	0.012	0.116	0.64	0.207	0.203	0.956
	SAL	0.102	0.15	0.76	0.174	0.096	0.8	0.249	0.107	0.902	0.398	0.225	0.982
	JAC	0.026	0.145	0.824	0.039	0.091	0.83	0.131	0.106	0.91	0.369	0.212	0.991
	MD	0.003	0.139	0.512	0.012	0.082	0.569	0.03	0.108	0.763	0.308	0.182	0.993
Rating-based	COS	0.026	0.146	0.753	0.056	0.094	0.797	*	*	*	0.404	0.214	0.994
	PC	0.398	0.21	0.953	0.196	0.13	0.964	*	*	*	0.294	0.263	0.984
LCP		0.002	0.153	0.39	0.006	0.091	0.422	0.009	0.114	0.672	0.211	0.201	0.951
Random recommendation		0.499	0.519	0.996	0.5	0.508	0.996	0.503	0.503	0.997	0.5	0.504	0.999

algorithms, some still have low stability and somehow low accuracy such as HC itself, IHC algorithm and the TNP-HC algorithm, and some sacrifice the diversity when trying to gain better stability and accuracy such as the HC+MD and the Biased-HC algorithms especially in the Netflix and Last.fm data sets. The TNS-HC algorithm, on the other hand is able to improve the stability and accuracy without sacrificing the high diversity, and thereby has very balanced, yet good, performances in terms of the stability, accuracy and diversity. The CN-based algorithms have generally very good stabilities due to the apparent popularity correlation of the CN nature which is, popular items are more likely to have more common neighbours with others. However, the accuracy and especially the diversity of CN-based algorithms are somehow low in comparison with the proposed TNS-HC algorithm. The COS and LCP algorithm behave very similar to the CN-based algorithms with very good stability but low accuracy and diversity. On the other hand, the PC algorithm has very good diversity but poor stability and accuracy.

4. Conclusion and Discussions

The scale of the Internet nowadays is increasing vastly in both the amount of information and the number of users. Billions of new records being created everyday brings abundant data to study with to make future recommendations to users. However, the fast evolution leaves us the challenge that, how could we guarantee the stability of the similarity quantification and recommendation? The stability problem will result in a great gap between the laboratory investigations and the practical applications of recommender systems. Furthermore, there arises the dilemma between diversity and stability. While high-diversity requires the system to recommend those unpopular items, the local structures of those unpopular items are generally unstable. Considering additionally the basic accuracy requirement of recommender systems, the challenge lies in how to solve the triple dilemma of stability-accuracy-diversity.

Focusing on the stable similarities, the present paper proposed the top-n-stability method based on the Heat Conduction algorithm. The TNS-HC algorithm significantly improves the recommendation's stability and accuracy simultaneously, and yet retains the high-diversity nature of the HC algorithm. Although the TNS-HC is an efficient solution comparing with some classical recommendation algorithms to the triple dilemma of stability-accuracy-diversity, yet it still offers recommendations not stable enough. The stabilities $\langle\Delta\rangle$ of the TNS-HC in some data sets are still larger than 0.1.

The idea of top-n-stability method is to remove the unstable similarities that quantified by existing measures. However, the question remains, that how could we stably quantify the similarity between items by mining the meaningful underlying association patterns. Actually, the content-based similarities [5, 6] won't have the stability problem, because such measures examines the similarities by extracting the items' pre-defined properties regardless of the their association patterns. Therefore, a good way to overcome the stability problem may be combining the structural simi-

larities with content-based similarities. For items with abundant records to stably mine their association patterns, the similarities could be defined accordingly, while for other items, the similarity could be defined using content-based methods. In other words, instead of removing unstable similarities, one could replace those unstable similarities with content-based evaluations. As the stability problem is mainly caused by the fast evolution of the bipartite network, another possible way is to pay closer attention to the evolution mechanisms of the user-item bipartite networks [37, 38, 39]. Although the evolution of most of social systems are believed to be governed by the preferential attachment [40, 41], *i.e.* popularity-driven, recent studies argued that, the similarity plays also a significant role [42]. For evolving recommender systems driven by both popularity and similarity, the question remains that how could we use the knowledge of its evolving mechanisms to enhance its stability and at the same time find the balance of accuracy and diversity. Furthermore, while the popular items could naturally gain more attention which leads to the popularity-driven evolution, the similarity-driven evolution is the actual task for recommender systems to deal with. Therefore, the ideal recommender system should stably quantify the similarity and offer users unpopular-but-relevant recommendations to guide the system's similarity-driven evolution.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China under grant No. 71532002 (key project), 71371125 and 61374177. Jianguo Liu acknowledges the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning and the Shuguang Program Project of Shanghai Educational Committee (Grant No. 14SG42).

[1] U. Shardanand, P. Maes, Social information filtering: algorithms for automating word of mouth, In: Proc. SIGCHI Conf. Human Factors in Computing Systems, 1995 May, pp. 210-217.
[2] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl-Based Syst. 46 (2013) 109-132.
[3] D. Goldberg, D. Nichols, B.M. Oki, D. Terry, Using collaborative filtering to weave an information tapestry, Commun. ACM 35 (1992) 61-70.
[4] L.Y. Lü, M. Medo, C.H. Yeung, Y.C. Zhang, Z.K. Zhang, T. Zhou, Recommender systems, Phys. Rep. 519 (2012) 1-49.
[5] C. Basu, H. Hirsh, W. Cohen, Recommendation as classification: Using social and content-based information in recommendation, In: Proc. 15th Nat. Conf. Artificial Intelligence 1998 pp. 714-720.
[6] P. Lops, M. de Gemmis, G. Semeraro, Content-based recommender systems: State of the art and trends, In: Recommender systems handbook pp. 73-105. Springer, US, 2011.
[7] C. Cattuto, V. Loreto, L. Pietronero, Semiotic dynamics and collaborative tagging, Proc. Nat. Acad. Sci. 104 (2007) 1461-1464.
[8] H.N. Kim, A. Alkhaldi, A. El Saddik, G.S. Jo, Collaborative user modeling with user-generated tags for social recommender systems, Exp. Syst. Appl. 38 (2011) 8488-8496.
[9] B. Markines, C. Cattuto, F. Menczer, D. Benz, A. Hotho, G. Stumme, Evaluating similarity measures for emergent semantics of social tagging, In: Proc. 18th Int. Conf. World Wide Web, ACM 2009 Apr., pp. 641-650.
[10] P. Symeonidis, A. Nanopoulos, Y. Manolopoulos, A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis, IEEE Trans. Knowledge and Data Engineering 22 (2010) 179-192.
[11] E. Ravasz, A.L. Somera, D. Mongru, Z.N. Oltvai, A.L. Barabási, Hierarchical organization of modularity in metabolic networks, Science 97 (2002) 1551-1555.
[12] E.A. Leicht, P. Holme, M.E. Newman, Vertex similarity in networks, Phys. Rev. E 73 (2006) 026120.
[13] M. Newman, A.L. Barabasi, D.J. Watts, The structure and dynamics of networks, Princeton University Press, 2006.
[14] J.G. Liu, L. Hou, Y.L. Zhang, W.J. Song, X. Pan, Empirical analysis of the clustering coefficient in the user-object bipartite networks, Int. J. Mod. Phys. C 24 (2013) 1350055.
[15] T. Zhou, J. Ren, M. Medo, Y.C. Zhang, Bipartite network projection and personal recommendation, Phys. Rev. E 76 (2007) 046115.
[16] G. Linden, B. Smith, J. York, Amazon.com recommendations: Item-to-item collaborative filtering, Internet Computing IEEE 7 (2003) 76-80.
[17] J.B. Schafer, J.A. Konstan, J. Riedl, E-commerce recommendation applications, In: Applications of Data Mining to Electronic Commerce, Springer, US, 2001, pp. 115-153.
[18] K. Ali, W. van Stam, TiVo: making show recommendations using a distributed collaborative filtering architecture, In: Proc. 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, ACM, 2004, pp. 394-401.
[19] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, D. Sampath, The YouTube video recommendation system, In: Proc. 4th ACM Conf. Recommender Systems, ACM, 2010, pp. 293-296.
[20] B. Lika, K. Kolomvatsos, S. Hadjiefthymiades, Facing the cold start problem in recommender systems, Exp. Syst. Appl. 41 (2014) 2065-2073.
[21] Q. Guo, W.J. Song, L. Hou, Y.L. Zhang, J.G. Liu, Effect of the time window on the heat-conduction information filtering model, Physica A 401 (2014) 15-2.
[22] L. Hou, X. Pan, Q. Guo, J.G. Liu, Memory effect of the online user preference, Sci. Rep. 4 (2014) 6560.
[23] N. Hurley, M. Zhang, Novelty and diversity in top-n recommendation – analysis and evaluation, ACM Transactions on Internet Technology (TOIT) 10 (2011) 14.
[24] S.M. McNee, J. Riedl, J.A. Konstan, Being accurate is not enough: how accuracy metrics have hurt recommender systems, In: CHI'06 Extended Abstracts on Human Factors in Computing Systems, ACM, 2006, pp. 1097-1101.
[25] Q. Guo, R. Leng, K. Shi, J.G. Liu, Heat conduction information filtering via local information of bipartite networks, Eur. Phys. J. B 85 (2012) 1-8.

[26] Y.C. Zhang, M. Blattner, Y.K. Yu, Heat conduction process on community networks as a recommendation model, Phys. Rev. Lett. 99 (2007) 154301.
[27] K. Lee, K. Lee, Escaping your comfort zone: A graph-based recommender system for finding novel recommendations among relevant items, Exp. Syst. Appl. 42 (2015) 4851-4858.
[28] J.G. Liu, T. Zhou, Q. Guo, Information filtering via biased heat conduction, Phys. Rev. E 84 (2011) 037101.
[29] T. Zhou, Z. Kuscsik, J.G. Liu, M. Medo, J.R. Wakeling, Y.C. Zhang, Solving the apparent diversity-accuracy dilemma of recommender systems, Proc. Nat. Acad. Sci. 107 (2010) 4511-4515.
[30] J.G. Liu, L. Hou, X. Pan, Q. Guo, T. Zhou, Stability of similarity measurements for bipartite network, Sci. Rep. 6 (2016) 18653.
[31] G. Adomavicius, J. Zhang, Stability of recommendation algorithms, ACM Transactions on Information Systems (TOIS) 30 (2012) 23.
[32] G. Adomavicius, J. Zhang, Iterative smoothing technique for improving stability of recommender systems, In: Proc. Workshop Recommendation Utility Evaluation: Beyond RMSE, CEUR Workshop Proceedings, 910 (2012) 3-8.
[33] R. Burke, M.P. O'Mahony, N.J. Hurley, Robust Collaborative Recommendation, In: Recommender Systems Handbook, pp. 805-835, Springer, US, 2011.
[34] M. O'Mahony, N. Hurley, N. Kushmerick, G. Silvestre, Collaborative recommendation: A robustness analysis, ACM Transactions on Internet Technology (TOIT) 4 (2004) 344-377.
[35] C. V. Cannistraci, G. Alanis-Lobato, T. Ravasi, From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. Sci. Rep. 3 (2013) 1613.
[36] S. Daminelli, J. M. Thomas, C. Duran, C. V. Cannistraci, Common neighbours and the local-community-paradigm for topological link prediction in bipartite networks. New J. Phys. 17(11) (2015) 113037.
[37] M.B. Díaz, M.A. Porter, J.P. Onnela, Competition for popularity in bipartite networks, Chaos 20 (2010) 043101.
[38] L. Ji, J.G. Liu, L. Hou, Q. Guo, Identifying the role of common interests in online user trust formation, PloS one 10 (2015) e0121105.
[39] J.G. Liu, Z. Hu, Q. Guo, Effect of the social influence on topological properties of user-object bipartite networks. Eur. Phys. J. B 86 (2013) 1-11.
[40] A.L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (1999) 509-512.
[41] Z.M. Ren, Y.Q. Shi, H. Liao, Characterizing popularity dynamics of online videos Physica A 453 (2016) 236-241.
[42] F. Papadopoulos, M. Kitsak, M.Á. Serrano, M. Boguñá, D. Krioukov, Popularity versus similarity in growing networks, Nature 489 (2012) 537-540.