Recommending Friends in Local Social Networks: An Envelope of Algorithms

Yuewen Liu
School of Management
Xi'an Jiaotong University
Xi'an, China
+86-029-82668382
liuyuewen@mail.xjtu.edu.cn

Xiangyu Chang
School of Mathematics and Statistics
Xi'an Jiaotong University
Xi'an, China
+86-029-82668005
xiangyuchang@gmail.com

Wayne Wei Huang
College of Business
Ohio University
Ohio, USA
+86-029-82668382
whuang@mail.xjtu.edu.cn

ABSTRACT

As the proliferating of online social networks, a lot of researchers studied the topics of link prediction, and developed a plenty of friend recommending algorithms. The researchers try to illustrate that their algorithms perform better than other algorithms, especially in terms of prediction accuracy. However, it is much possible that no friend recommending algorithm can really beat the other algorithms based on the same data/information. Specially, in this study, we find that the friend recommending algorithms perform differently when we evaluate the algorithms using different performance indicators, or when the algorithms are applied for different groups of users classified by the user degree. Based on these findings, we propose a method to construct envelops of algorithms, to combine the "best part" of each algorithm together for specified purpose. We collect data from a Chinese dominant social network website to compare the existing algorithms and to verify the proposed envelope method. The results show that the proposed algorithm can enhance the performance of friend recommending algorithms for specified purposes.

Categories and Subject Descriptors

H.3.4 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems –*Pattern matching.*

General Terms

Algorithms, Measurement, Performance, Design.

Keywords

Friend Recommending, Social Network, Envelope Method

1. INTRODUCTION

As the proliferating of social networks, a lot of researchers studied the recommending algorithms. The researchers compare the existing algorithms, and then propose new algorithms to over perform the existing ones [1-3].

However, the idea of pursuing a "better" algorithm may be misleading. There is no algorithm can perform better than all the other algorithms in different datasets [4, 5]. Even for the same dataset, the best algorithms may not perform always the best for

(1) Different purposes: The same algorithm may serve in different situations. For example, a friend recommending algorithm may be used in a PC platform, where the users can browse a lot of recommended friend candidates simultaneously, by reading the big screen and scrolling the pages easily using mouse; it can also be used in a mobile device, such as a mobile phone, which can show only a hand of friends at the same time. The design purpose of the algorithm should be different in different situations, and the different purposes can be verified by different indicators. For example, the accuracy of recommending algorithms in a 2 slots panel (e.g., mobile phone) could be verified by Precision@2, while the accuracy of recommending algorithms in a big screen with a lot of slots (e.g., PC screen) could be verified by Precision@20, @50, or even @100. An algorithm performs better in terms of some performance indicators may not perform better in terms of other indicators. Pursuing a recommending algorithm which performs better in terms of all the indicators may be not possible, and also not necessary.

(2) Different cases: The algorithms may serve different types of users in terms of user attributes. An algorithm performs well for some types of users may not perform equivalently well for other types, because different types of users may have different demands in friend recommendation service. In this study, we focus on the user degree, i.e., the number of friends of a user. The user degree may represent a users' type, or the user's stage of "life-cycle". For example, the developing users in early stage may have less number of friends on average than the developed users in stable stage. The developing users and the developed users may search for different types of friends. An algorithm performs well for some types of users may not perform equivalently well of some other types. Pursuing a recommending algorithm which performs better for all users may be not possible.

In this study, we propose that a friend recommending algorithm may not beat all the other algorithms, in terms of (1) performance indicators and (2) degree of users. We also design a method to construct envelopes of the algorithms, to combine the "best algorithms" in different cases together for given purpose. The major goal of the envelope is to assemble the "better parts" of several algorithms to form a high-performance algorithm, and avoid the "worse parts" of the algorithms given certain purposes.

Specifically, we review 10 different friend recommending algorithms based on local-information. Then we collect data from a Chinese dominant social network website, to compare the algorithms, and verify our envelope method. We find that: (1) the algorithms perform differently in terms of different performance

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
BigDataScience'14, August 04 - 07 2014, Beijing, China.
Copyright 2014 ACM 978-1-4503-2891-3/14/08 ...$15.00.
http://dx.doi.org/10.1145/2640087.2644167

indicators; (2) the algorithms perform differently when the users have different degrees in terms of the same performance indicator; (3) some algorithms which perform not the best on average may perform the best for some groups of users. (4) Our envelope method enhances the performance of friend recommending algorithms given specific purposes.

The rest of the paper is organized as follows, section 2 review related literature; section 3 propose our method; section 4 show the experiments and results; section 5 discuss the implications, limitations and future research directions for this study.

2. RELATED LITERATURE

2.1. Friend Recommending Algorithms

Friend recommending means to recommend friends in a social network. In the domain of complex networks, friend recommending are also called “link prediction” [6]. Friend recommending algorithms can be classified as local-information based algorithms, quasi-local information based algorithms, and global-information based algorithms [1, 7, 8]. Local-information based algorithms only utilize the information of friends’ friends. The most common idea of friend recommending algorithms based on local information is to calculate the similarity of two nodes, *S_xy*. If the similarity of two nodes *S_xy* is high, then the two nodes are more likely to be friends [8]. These algorithms are intuitively simple, easy to interpret, and with low calculation burden, thus are widely adopted in a lot of social networks [7].

Reviews of the local-information based algorithms can be found in [1, 6, 7]. To recommend friend candidates to a user, we should firstly calculate the indices *S_xy* for each of the friend candidates, and then show the user the friend candidates descending ordered by the indices. According to the variables utilized in the algorithms, we may classify the algorithms to four types: (1) Absolute CN algorithms, (2) Relative CN algorithms, (3) Preferential Attachment Algorithm, and (4) Information Entropy Algorithms.

2.2. Evaluations of the Algorithms

The performance indicators of algorithms include ROC, P@k (Precision@k), MRR, and MAP. Since we limit the recommending results to top 100 results, ROC is not fit for our study. P@k (Precision@k) is a widely used method to evaluate information retrieval systems. P@k = n / k, where k is the number of people who are recommended by the system and n is the number of true friends in the recommended list. In this paper, we choose 1, 2, 5, 10, 20, and 50 as the value of k. MRR (Mean reciprocal rank) is a measure of navigational searching or question answering, which only focuses on the rank of the first correct one in the recommendation list. The MRR is the average of the reciprocal ranks of the first correct answer for a set of queries. MAP (Mean average precision) takes into account the rank of all the correct answers in the response list for a query. MAP is the mean of the average precision values for a set of queries, it is used to evaluate the algorithms by the ranks of all the correct results.

The evaluation of algorithms may be represent different purposes. For example, in a 2 slot panel (e.g., mobile phone), P@1, P@2, and MRR is the most proper indicators, because they are measuring the precision of the top 1 or 2 recommending results. In a web page with a long recommendation list, P@20, P@50 and MAP should be more proper, because they are measuring the average precision of a long list. Based on the interpretation, we may classify the performance indicators of the algorithms to 3 groups:

Table 1. Performance Indicators of Recommending Algorithms

Groups	Indicators	Situations	Groups
Position-Prior Indicators	P@1, P@2, MRR	Panels with 2 slots (especially on mobile devices)	Position-Prior Indicators
Accuracy-Prior Indicators	P@20, P@50, MAP	Webpages with a long list	Accuracy-Prior Indicators
Moderation Indicators	P@5, P@10	Panels with 5-10 slots (Traditional social network panel)	Moderation Indicators

2.3. Comparisons of Algorithms

There are few studies comparing these algorithms using empirical social network dataset. However, there are some studies compare the algorithms using the complex network dataset. The term they used is “link prediction”.

Huang et al. collected information from a major Chinese online bookstore, including 2,000 customers, 9,695 books and 18,771 transactions [16]. Based on this dataset, they compared some FoF algorithms with path-based algorithms. They showed that the Preferential Attachment beat other algorithms. The performance of Adamic/Adar and Common-Neighbors were also acceptable.

Liben-Nowell and Kleinberg compared most of the algorithms we mentioned previously in this paper and conducted experiments on future interactions in the co-authorship networks [7]. The experiments showed the relative effectiveness of these algorithms. The data they used were 23,589 authors who had written more than 3 articles during a 3-year period from www.arxiv.com [7].

Chen et al. introduced the SONAR recommendation system developed by IBM [17]. In order to evaluate this new system, they first conducted a personalized online survey on Beehive and then randomly picked 3,000 users to compare this system with several baseline algorithms, including FoF. They found that FoF is not as good as SONAR, but it performed better than the content-based algorithm and the content-plus-link algorithm [17].

Yin et al. conducted an experiment to compare several recommendation systems [18]. The data were collected from DBLP (Digital Bibliography Project, a website provides bibliographic information on major computer science journals and proceedings) and IMDB (Internet Movie Database, an online database of information related to movies, actors, television shows, etc.). Totally 2,500 authors’ and 6,750 actors’/actresses’ information were used [18].

2.4. Dynamics of User Demand

Moreover, User in different stage may have different demand. the developing users in early stage may have less number of friends on average than the developed users in stable stage. The developing users and the developed users may search for different types of friends. For example, Kenis et al. indicate that, in different stages of community, the tie formation rates are different [19]. Similarly, in different stage of a user, the friend adding behavior could be different. It is meaningful to study the recommending algorithms in different stage of users.

3. THE PROPOSED METHOD

3.1. Envelope of Algorithms

To overcome the weaknesses of single algorithm for a given purpose, we propose an envelope of algorithms. For example, as shown in the figure 1(a), algorithms 1, 2, and 3 perform differently when the served users have different degrees. Algorithm 1 fits to the initial users in the social network, thus performs best when the served users have relatively less friends. Algorithm 2 fits to the developing users, thus performs best when the served users have moderate number of friends. Algorithm 3 fits to the developed users, thus performs best when the served users have relatively more friends.

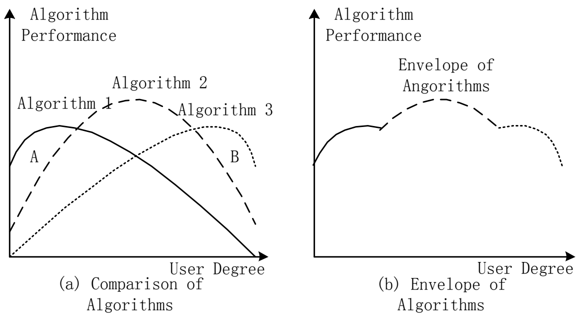

Figure 1. An Illustration of Envelope of Algorithms

Among the 3 algorithms, suppose algorithm 2 performs best on average. However, if we use algorithm 2 to recommend friends for users, the initial users and developed users may not be properly served, since algorithm 2 has weaknesses in recommending friends for initial users and developed users. It is natural to take the "best part" of each algorithm to compose a new algorithm. As shown in figure 1(b), the new algorithm uses algorithm 1 to serve initial users, algorithm 2 to serve developing users, and algorithm 3 to serve developed users. Since the new algorithm forms the envelope of the existing algorithms in the user degree-algorithm performance figure, thus we name the new algorithm as "envelope of algorithms". The "envelope of algorithms" performs better than algorithm 2 in terms of area A and B in figure 1(a).

4. EXPERIMENT

4.1. Data Collection

We collected data from a Chinese dominant social network website to compare the existing algorithms, and verify the envelope of the algorithms. The majority of the users in the website are college students and graduates. The website has strong influence in the young generation in China.

In detail, we developed a web crawler (in Java) to collect social network data from the social network website. The data collection process is a snowball process. First, we got 2 volunteers' account (denoted by D0), then their 240 friends (D1), then 51,340 friends' friends (D2), and then 7,158,934 friends' friends' friends (D3). To get the D3 users' degree information, we also "visited" each D3 user's public homepage and downloaded relative information.

4.2. Experiment Design

We first try to recommend friends for the 240 D1 users using each of the algorithms. To verify the performance of the algorithms, we split each user's friends to two parts, 9/10 of them as the training set, and 1/10 of them as the test set. We use the 9/10 friends and the corresponding friends' friends to construct a social network, and then recommend friends based on the social network. This 9/10 method is a well-known method in the literature [1]. The recommending method is to calculate a score for each friend candidate based on a given algorithm, and then to sort all the friend candidates according to a descending order of the scores, and then to select the top 100 friend candidates as the recommendation results set. Since a user needs efforts to identify and judge the recommendation results, we believe that recommending top 100 friend candidates is sufficient in practice. Finally, the recommending results set are compared with the 1/10 friends set, and the overlap of the two sets are recognized as correct recommend results. The experiment is conducted by developing Stored Procedure in SQL Server. The detailed pseudo code is as shown in Figure 2.

4.3. Comparisons of Algorithms

The comparison results of the algorithms (the Algorithm-P@k-MRR-MAP table in the step 4.3) are shown in Table 2. The comparison results show that, in terms of the position-prior indicators (i.e., P@1, P@2, P@5, and MRR), the Adamic/Ada algorithm performs the best; while in terms of the accuracy-prior indicators (i.e., P@10, P@20, P@50, and MAP), the RA algorithm performs the best. It is reasonable that these two algorithms perform the best. While other algorithms treat the common neighbors simply as a number, these two algorithms integrate and utilize the degree information of each common neighbor. From this perspective of view, these two algorithms seem to have advantages over the other algorithms, and the other algorithms seem to have no chance to beat these two algorithms.

Since we do not possess any priori information on the relationship between algorithms and user degree, we simply classify the users to 5 groups according to their degree (i.e., the number of friends): 0 to 99, 100 to 200, 200 to 300, 300 to 400, and 400 to infinite. The benefit of this intuitively simple classification method is that we have close-to-balanced samples among each of the groups. The comparison results (the Algorithm-Group-P@k-MRR-MAP table in Step 4.4) show more details of algorithm performance. Table 3 illustrates the results. To make the results concise, we removed the algorithms with no chance to beat other algorithms in any groups.

Although we find that the Adamic/Ada algorithm performs the best in terms of the position-prior indicators, it is not always the best algorithm when we split the users into groups. As shown in the P@1, P@2, and MRR rows in the following table, The Adamic/Ada algorithm only performs the best in the [0,99] and [100,199] groups. In the [200,299] group, the RA algorithm performs the best. In the [300,399] and [400,Inf) groups, Jaccard and Sorensen algorithms perform the best. In terms of the accuracy-prior indicators, as shown in the P@20, P@50 and MAP rows in the following table, even after we break down the users to groups, the RA algorithm performs always the best.

```
Step 1: Data Preparation
1.1 Select all D1-D2
1.2 Randomly split D1-D2 to D1-D2' (1/10 of D2) and D1-D2''
    (9/10 of D2)
Step 2: Generate Friends of Friends
2.1 Select all D1-D2''-D3
2.2 Delete D1-D2''-D3 where D3=D1
2.3 Delete D1-D2''-D3 where D3 in the collection of D2'' given
```

the same D1

2.4 Get D1-D2''-D3' (D3' is a sub collection of D3 after step 1.4 and 1.5)

2.5 Select all the D1-D3' pairs, and calculate the number of friends of D1, D3', and the number of their common friends

Step3: Recommendation

3.1 For each D1-D3', calculate each of the recommendation algorithm index, get D1-D3'-Algorithm-Score

3.2 For each D1-Algorithm, sort and select top 100 D1-Algorithm-D3' according to the score, and generate D1-Algorithm-D3'' (D3'' is a sub collection of D3' after step 3.2)

Step4: Evaluation

4.1 For each D1-Algorithm, mark the D3'' when it is overlap with D2' given the same D1

4.2 For each D1-Algorithm, calculate P@k, MRR and MAP, get D1-Algorithm-P@k-MRR-MAP

4.3 For each Algorithm, calculate the average P@k, MRR and MAP, get Algorithm-P@k-MRR-MAP

4.4 Classify each D1 to groups according to her number of friends; for each Algorithm-Group, calculate the average P@k, MRR and MAP, get Algorithm-Group-P@k-MRR-MAP

Notes: (1) D1 means the users, D2 means the users' friends, and D3 means the users' friends' friends; (2) A-B means a table with two related fields A and B.

Figure 2. The Steps of Calculation

Table 2. Comparison Results of Algorithms

Method	p@1	p@2	p@5	p@10	p@20	p@50	MRR	MAP
CN	0.620	0.596	0.504	0.402	0.299	0.181	0.723	0.433
Salton	0.476	0.443	0.397	0.341	0.272	0.174	0.604	0.356
Jaccard	0.576	0.524	0.461	0.391	0.294	0.184	0.673	0.392
Sorensen	0.576	0.524	0.461	0.391	0.294	0.184	0.673	0.392
HubPromoted	0.087	0.090	0.086	0.089	0.088	0.089	0.215	0.169
HubDepressed	0.585	0.526	0.473	0.375	0.286	0.179	0.675	0.385
LHN	0.052	0.059	0.057	0.061	0.064	0.072	0.166	0.124
PA	0.004	0.007	0.007	0.007	0.006	0.006	0.029	0.029
RA	0.611	0.607	0.530	**0.446**	**0.335**	**0.202**	0.723	**0.459**
Adamic/Ada	**0.646**	**0.618**	**0.534**	0.424	0.312	0.190	**0.745**	0.454
Envelope1	**0.681**	**0.638**	**0.534**	0.434	0.316	0.192	**0.766**	**0.460**
Envelope2	0.611	0.607	0.530	**0.446**	**0.335**	**0.202**	0.723	0.459

Table 3. Comparison Results of Algorithms between Groups

	User Degree	0,99	100,199	200,299	300,399	400,Inf
	Users Count	54	55	44	31	45
p@1	CN	0.370	**0.727**	0.705	0.710	0.644
	Jaccard	0.222	0.582	0.659	**0.742**	**0.800**
	Sorensen	0.222	0.582	0.659	**0.742**	**0.800**
	RA	0.278	0.655	**0.795**	0.677	0.733
	Adamic/Ada	**0.407**	**0.727**	0.750	0.710	0.689
p@2	Jaccard	0.148	0.518	0.602	**0.742**	**0.756**
	Sorensen	0.148	0.518	0.602	**0.742**	**0.756**
	RA	0.315	0.636	**0.750**	0.677	0.733
	Adamic/Ada	**0.352**	**0.673**	0.693	0.710	0.733
p@20	RA	**0.108**	**0.290**	**0.390**	**0.461**	**0.521**
p@50	RA	**0.053**	**0.162**	**0.235**	**0.266**	**0.354**
MRR	Jaccard	0.319	0.679	0.772	**0.860**	**0.867**
	Sorensen	0.319	0.679	0.772	**0.860**	**0.867**
	RA	0.478	0.741	**0.850**	0.785	0.830
	Adamic/Ada	**0.548**	**0.779**	0.834	0.807	0.812
MAP	RA	0.351	**0.468**	**0.487**	**0.498**	**0.524**
	Adamic/Ada	**0.379**	0.458	0.476	0.485	0.498

4.4. Performance of Envelope Algorithm

Based on the comparison results, we designed two envelopes. The first envelope is designed for the position-prior indicators:

$$Envelope\ 1\ of\ Algorithms = \begin{cases} Adamic\ /\ Ada, & d < 200 \\ RA, & 199 \le d < 300 \\ Jaccard\ or\ Sorensen, & otherwise \end{cases}$$

The performance of the envelope 1 is then calculated, and shown in the Table 2. The results shown that, the envelope 1 of the algorithms performs better over all the algorithms. The second envelop is designed for the accuracy-prior indicators. It is simply the RA algorithm, since the RA algorithm performs always the best in terms of the accuracy-prior indicators. The performance of envelope 2 is the same with the RA algorithm, as shown in table 2.

It is valuable to note that the envelope algorithms developed here are merely illustrations to the envelope algorithm method. For other datasets, we need to develop other envelope algorithms using the same method.

5. DISCUSSIONS

This study empirically compared the existing friend recommending algorithms in the literature, and find that: (1) the algorithms perform differently in terms of different performance indicators; (2) the algorithms perform differently when the users have different degrees in terms of the same performance indicator; (3) some algorithms which perform not the best on average may perform the best for some groups of users. We also show that user degree is a proper moderating variable when comparing the performance of friend recommending algorithms. This study also proposed an envelope method to combine the advantages and avoid the weaknesses of the existing algorithms. The envelope method may help to enhance the performance of friend recommending algorithms given specific purpose. To the best of our knowledge, this study is the first one which compares the friend recommending algorithms in deep. While the previous studies simply stopped by the average performance of the algorithms, this study show that the algorithms may perform differently for different purposes, and in different cases. Our findings extend our understandings on the performance of algorithms. This study also develops a new method which is essentially different from the methods in the existing literature. Our method may also be practically meaningful.

6. ACKNOWLEDGMENTS

This research was supported by the Xi'an Jiao Tong University – Tencent Inc. Joint Research Project for Big Data and Social Network, National Natural Science Foundation of China (Project No. 71301128, 7131005, and 71371151), China Postdoctoral Science Foundation (Project No. 2012M527182) and the Key Lab of the Ministry of Education in China for Process Control & Efficiency Engineering.

7. REFERENCES

(The references could be acquired from the corresponding author.)