Exploiting Popularity and Similarity for Link Recommendation in Twitter Networks

Jun Zou
Georgia Institute of Technology
Atlanta, GA 30332, USA
junzou@gatech.edu

Faramarz Fekri
Georgia Institute of Technology
Atlanta, GA 30332, USA
fekri@ece.gatech.edu

ABSTRACT

Twitter functions both as a social network and an information network, where users follow other users to make social connections as well as to receive information. Both popularity and similarity are important factors that drive the growth of the Twitter network. In this paper, we propose two approaches to exploiting both popularity and similarity for link recommendation. The first approach employs the rank aggregation technique to combine rankings generated by popularity-based and similarity-based recommendation algorithms. The second approach adapts the collaborative filtering algorithms to incorporate popularity in addition to similarity. The empirical evaluation results on real-world datasets confirm that combining popularity and similarity improves the recommendation performance.

Categories and Subject Descriptors

H.3.5 [**Online Information Services**]: Web-based services

Keywords

Link recommendation; Twitter; Popularity; Similarity

1. INTRODUCTION

Twitter is a popular on-line platform for social networking and information sharing. Twitter users can follow other users to receive messages, or tweets, from them. However, given the limited attention and time, most users would like to follow only the most relevant users. Due to the huge number of users in Twitter, recommendation algorithms are needed to help users automatically discover new interesting users to follow.

Many existing link recommendation algorithms for social networks are developed with focus on the link structure [5]. The simplest example is to recommend the most popular users with the largest number of connections. Other common algorithms first weigh each link by some importance score, and rank the nodes according to the sum of importance scores of their links, e.g., the PageRank algorithm [7]. Twitter provides the user recommendation service called WTF ("Who To Follow") [2], which was built on SALSA (Stochastic Approach for Link-Structure Analysis) [4], a random-walk algorithm similar to PageRank. We generally refer to those algorithms as "popularity" or "weighted popularity" based algorithms.

Yet, unlike other social networks such as Facebook, besides to establish social connections, many Twitter users follow other users to receive information interesting to them. Hence, it is promising to exploit the similarity between Twitter users for recommendation, i.e., to recommend other users similar to the followees already followed by the follower, or to recommend other users who are similar to the follower. In [3], the authors proposed a content-based algorithm that match user interests by directly analyzing the texts of user tweets. In [6], the authors proposed a collaborative filtering algorithm using the matrix factorization technique to learn latent user interests from the user feedbacks, e.g., to follow a user or not.

Indeed, a recent study has shown that popularity and similarity are the two important factors that drive the growth of a variety of networks including the Internet and social networks [8]. In this paper, we compare various popularity-based algorithms and similarity-based algorithms for Twitter user recommendation, and propose two approaches to exploiting both popularity and similarity. The first approach is to employ rank aggregation techniques [1]; the second approach is to adapt the collaborative filtering algorithms to incorporate popularity in addition to similarity.

2. POPULARITY VERSUS SIMILARITY

In this section, we describe various popularity-based algorithms and similarity-based algorithms.

2.1 Problem Description

To recommend new users for the follower to follow, the recommendation algorithm generates a personalized ranked list of users. However, due to the huge number of users in Twitter, it is too costly to compute a ranking of network-wide users for making personalized recommendations to each follower. Many algorithms instead recommend users from the local network that centers around the follower [2]. The empirical study in [10] revealed that 90% of new links in Twitter-like microblogging networks go to users two hops away from the follower, i.e., followees of the follower's followees.

As such, in the rest of this paper, we focus on recommend-

Proceedings of the 6th Workshop on Recommender Systems and the Social Web (RSWeb 2014), collocated with ACM RecSys 2014, 10/06/2014, Foster City, CA, USA.
.

ing users from 2-hop users for the follower. We denote the follower whom to recommend users for as source s, the set of its initial followees as $\mathcal{F}_1$, and the set of all followees of $\mathcal{F}_1$ as $\mathcal{F}_2$. We also denote $F^+(s)$ as the subset of $\mathcal{F}_2$ that source s follows as network grows. The recommendation problem is essentially to compute a ranked list of users from $\mathcal{F}_2$ for source s.

2.2 Recommendation Algorithms

2.2.1 Popularity-based algorithms

For the popularity based or weighted popularity based algorithms, we consider Adamic-Adar score [5] and SALSA [4]. Adamic-Adar score is a simple and effective measure for link recommendation. We adopt it for Twitter user recommendation, and compute the score for a 2-hop user $u \in \mathcal{F}_2$ of source s as

$$AA(s,u) = \sum_{z \in \mathcal{P}(u) \cap \mathcal{F}_1} \frac{1}{|\mathcal{R}(z)|}, \tag{1}$$

where $\mathcal{P}(u)$ denotes the set of followers of user u, and $\mathcal{R}(z)$ denotes the set of followees of user z.

In the SALSA algorithm, we construct a bipartite graph $\mathcal{G}$ by assigning $\mathcal{F}_1$ as "hubs" and $\mathcal{F}_2$ as "authorities". The algorithm performs two distinct forward-backward random walks starting from the "authorities" side or the "hubs" side. The scores for the authorities and hubs are related as

$$a_i = \sum_{\{k|(k,i)\in\mathcal{G}\}} \frac{h_k}{|\mathcal{R}(k)|}, \quad h_k = \sum_{\{i|(k,i)\in\mathcal{G}\}} \frac{a_i}{|\mathcal{P}(i)|} \tag{2}$$

where a_i and h_k denote the scores of authority node i and hub node k, respectively, and (k,i) denotes an edge in bipartite graph $\mathcal{G}$. The users in $\mathcal{F}_2$ ("authorities") are ranked by the authority scores.

2.2.2 Similarity-based algorithms

For the similarity-based algorithms, we focus on the collaborative filtering recommendation algorithms using the observed follower-followee relationship between users as implicit feedback, including the neighborhood method and the MF-BPR (Matrix Factorization with Bayesian Personalized Ranking) method [9]. The basic principle is to recommend to source s the followees of the $\mathcal{F}_1$ users who share similar interests with source s. Using the terms of traditional user-item recommender systems, the set of "users" corresponds to $\mathcal{U} = \{s\} \cup \mathcal{F}_1$, and the set of "items" corresponds to $\mathcal{F}_2$. Moreover, the established follower-followee relationships between $\mathcal{U}$ and $\mathcal{F}_2$ are considered as observed implicit feedback on "items" in $\mathcal{F}_2$ from "users" in $\mathcal{U}$. The collaborative filtering approach exploits the collective implicit feedbacks to generate recommendations.

In the neighborhood method, we compute the similarity between source s and other users $u \in \mathcal{F}_1$ using Jaccard's coefficient as

$$J(s,u) = \frac{|F^+(u) \cap F^+(s)|}{|F^+(u) \cup F^+(s)|} \tag{3}$$

where $F^+(u)$ denotes the subset of $\mathcal{F}_2$ followed by user u. The score on user $i \in \mathcal{F}_2$ is predicted as

$$R(s,i) = \sum_{u \in \mathcal{F}_1} J(s,u) R(u,i), \tag{4}$$

where $R(u,i) = 1$ if u follows i and $R(u,i) = 0$ otherwise.

The MF-BPR method learns a model for correctly ranking pairs of users in $\mathcal{F}_2$. It is assumed that if user u follows user i but not user j, $\forall i,j \in \mathcal{F}_2$, then user u prefers user i over j. We represent such relationship as (u,i,j). The training data is created as $\mathcal{D} = \{(u,i,j)|i \in F^+(u), j \in \mathcal{F}_2 \backslash F^+(u)\}$. We associate each user u with a K-dimensional latent vector $\mathbf{w}_u$, and each $i \in \mathcal{F}_2$ with a K-dimensional latent vector $\mathbf{v}_i$. Let W and V represent the collections of all $\mathbf{w}_u$ and all $\mathbf{v}_i$, respectively. The probability that user u prefers i over j is defined as

$$P(i >_u j|W,V) = \frac{1}{1+e^{-(X(u,i)-X(u,j))}}, \tag{5}$$

where $X(u,i) = \langle \mathbf{w}_u, \mathbf{v}_i \rangle$, and $\langle , \rangle$ indicates the dot product. We further introduce a normal distribution $\mathcal{N}(\mathbf{0}, \lambda^{-1} I_K)$ as the prior distribution for each $\mathbf{w}_u$ and $\mathbf{v}_i$, where $\lambda > 0$ and I_K is a $K \times K$ identity matrix. The latent vectors are learnt by maximizing the posterior probability, which can be formulated as

$$\max_{W,V} \quad \log \prod_{(u,i,j)\in\mathcal{D}} P(i >_u j|W,V) P(W) P(V). \tag{6}$$

We apply the stochastic gradient-descent algorithm with bootstrap sampling as in [9] to solve this optimization problem. The latent vectors are updated for triple $(u,i,j) \in \mathcal{D}$ with learning rate μ as follows

$$\mathbf{w}_u \leftarrow \mathbf{w}_u + \mu \left(\frac{\mathbf{v}_i - \mathbf{v}_j}{1+e^{X(u,i)-X(u,j)}} - 2\lambda \mathbf{w}_u \right), \tag{7}$$

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \mu \left(\frac{\mathbf{w}_u}{1+e^{X(u,i)-X(u,j)}} - 2\lambda \mathbf{v}_i \right), \tag{8}$$

$$\mathbf{v}_j \leftarrow \mathbf{v}_j + \mu \left(\frac{-\mathbf{w}_u}{1+e^{X(u,i)-X(u,j)}} - 2\lambda \mathbf{v}_j \right). \tag{9}$$

To recommend users for source s, we compute $X(s,i)$, $\forall i \in \mathcal{F}_2$, and rank the users in descending order of $X(s,i)$.

3. PROPOSED LINK RECOMMENDATION

We propose two approaches to exploiting both popularity and similarity: rank aggregation and popularity-biased collaborative filtering. The rank aggregation approach combines the recommendation results of the popularity-based algorithm and the similarity-based algorithm, while each individual algorithm generates recommendations independently. It is worth noting that the scores predicted by different recommendation algorithms can be very different in both the range and the form, which are not suitable for aggregation, and hence we only consider order-based rank aggregation that combines the rankings which can be easily derived by sorting users by the predicted scores.

The popularity-biased collaborative filtering approach directly adapts the original collaborative filtering algorithms to incorporate popularity in addition to similarity. We introduce two specific cases of the neighborhood algorithm and the MF-BPR algorithm.

3.1 Rank Aggregation

Rank aggregation combines several ranking lists to generate a "consensus" ranking. It has been widely applied in Web search and document retrieval, e.g., meta-search engines and multi-criteria search. There are various rank aggregation algorithms such as Borda's method, median rank aggregation,

and Markov chain methods. We choose the Markov chain method for its superior performance. In particular, we focus on the MC3 method among the four Markov chain methods proposed in [1].

Suppose we have two ranked lists of users denoted by τ_1 and τ_2, which are generated using the popularity-based algorithm and the similarity-based algorithm, respectively. The Markov chain rank aggregation method assigns a unique state to each of the users, and specifies the transition matrix as follows: If the current state is user i, randomly pick a ranking list τ with probabilities $P(\tau = \tau_1) = \alpha$ and $P(\tau = \tau_2) = 1 - \alpha$, then uniformly pick a user j from τ, and go to j if $j >_\tau i$ else stay in i. Here, $j >_\tau i$ means the list τ ranks j closer to the top than i. Let T_k represent the transition matrix for the Markov chain derived from individual ranking list τ_k, $k = 1, 2$, where $T_k(i, j)$ is the transition probability from state i to j,

$$T_k(i,j) = \begin{cases} \frac{1}{|\mathcal{F}_2|}, & \text{if } j >_{\tau_k} i \\ 1 - \frac{|\{j | j >_{\tau_k} i\}|}{|\mathcal{F}_2|}, & \text{if } j = i \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

The transition matrix T of the rank aggregation Markov chain can be written as

$$T = \alpha T_1 + (1 - \alpha) T_2. \quad (11)$$

The aggregated ranking is obtained by ranking users according to the stationary probabilities of the Markov chain.

By varying the parameter α between 0 and 1, we can bias the aggregation rank towards similarity or popularity, with $\alpha = 0$ for similarity and $\alpha = 1$ for popularity. Further, α can be personalized for individual users to better predict their behavior and thus meet individual preferences.

3.2 Popularity-biased Collaborative Filtering

We adapt the two cases of the neighborhood method and the matrix factorization method, which are the most popular collaborative filtering recommendation algorithms, for popularity-biased collaborative filtering. It should be noted that the specific techniques proposed here might not be directly applicable to other collaborative filtering algorithms, as they may have very different prediction models. Nevertheless, we stress the viability and benefits of incorporating popularity into collaborative filtering link recommendation.

3.2.1 The popularity-biased neighborhood method

The neighborhood method predicts the score on user $i \in \mathcal{F}_2$ for source s using (4), where the feedback $R(u, i)$ from user u in $\mathcal{F}_1$ (set as 1 for following i and 0 for not following i) is weighted by the similarity of user u to source s. However, the similarity computed using the Jaccard's coefficient assigns zero to users who do not have common followees with the source. To bias the neighborhood method towards popularity, we modify the similarity measure as follows

$$J_p(s,u) = \frac{|F^+(u) \cap F^+(s)| + c}{|F^+(u) \cup F^+(s)|}, \quad (12)$$

where $c > 0$ ensures that every user $u \in \mathcal{F}_1$ is counted with $J_p(s, u) > 0$. In this way, the popular users who are followed by many users in $\mathcal{F}_1$, though not necessarily highly similar to the source, can have a higher predicted score than other users followed by only a few users with high similarity to the source.

3.2.2 The popularity-biased MF-BPR method

The MF-BPR method represents each user $i \in \mathcal{F}_2$ using a latent vector as described in Sec. 2.2.2. The similarity of user interests is modelled by the angle between latent vectors. We now adapt the model to also incorporate the popularity of each user by the length, or magnitude, of the latent vector. We assume a prior distribution $\mathcal{N}(\gamma \mathbf{1}, \lambda^{-1} I_K)$ for the latent vectors, where $\mathbf{1}$ denotes a vector with all entries one, and randomly initialize the latent vectors following the normal distribution $\mathcal{N}(\gamma \mathbf{1}, \beta I_K)$, where γ needs to be chosen relatively large. Using the stochastic gradient descent learning algorithm, the latent vectors are updated for triple $(u, i, j) \in \mathcal{D}$ as follows

$$\mathbf{w}_u \leftarrow \mathbf{w}_u + \mu \left(\frac{\mathbf{v}_i - \mathbf{v}_j}{1 + e^{X(u,i) - X(u,j)}} - 2\lambda(\mathbf{w}_u - \gamma \mathbf{1}) \right), \quad (13)$$

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \mu \left(\frac{\mathbf{w}_u}{1 + e^{X(u,i) - X(u,j)}} - 2\lambda(\mathbf{v}_i - \gamma \mathbf{1}) \right), \quad (14)$$

$$\mathbf{v}_j \leftarrow \mathbf{v}_j + \mu \left(\frac{-\mathbf{w}_u}{1 + e^{X(u,i) - X(u,j)}} - 2\lambda(\mathbf{v}_j - \gamma \mathbf{1}) \right). \quad (15)$$

The latent vector $\mathbf{v}_i$ of user i is updated through (14), where the angle between $\mathbf{v}_i$ and $\mathbf{w}_u$ is expected to be small, as they are both close to the vector $\gamma \mathbf{1}$ if γ is relatively large. Hence, $\mathbf{v}_i$ increases in length after each update. As popular users are updated more often in the stochastic optimization process, their associated latent vectors will have larger length.

The score on user $i \in \mathcal{F}_2$ by source s is predicted by

$$X(s,i) = \langle \mathbf{w}_s, \mathbf{v}_i \rangle = \cos\theta_{si} |\mathbf{w}_s| |\mathbf{v}_i|, \quad (16)$$

where θ_{si} is the angle between $\mathbf{w}_s$ and $\mathbf{v}_i$, which reflects the similarity of interests between user i and source s. Moreover, with the popularity-biased MF-BPR method, the magnitude $|\mathbf{v}_i|$ is large for popular users. Therefore, both similarity and popularity are taken into account. Also, increasing γ biases the algorithm more towards popularity.

4. EVALUATION

In this section, we compare the empirical performance of all recommendation algorithms on the Twitter dataset and the Tencent Weibo dataset. Tencent Weibo is a Twitter-like online microblogging service widely used in China. The Twitter dataset is downloaded from Twitter.com using the Twitter API. The Tencent Weibo dataset is a subset of the dataset provided by KDD Cup 2012 Track 1[1]. The source users are selected such that they have at least 100 followees. For each source user, we use its earliest 50 followees as the initial set $\mathcal{F}_1$, and construct the set $\mathcal{F}_2$ from the followees of $\mathcal{F}_1$. The statistics of the two datasets are shown in Table 1, where the number of $\mathcal{F}_2$ users is the total of $\mathcal{F}_2$ users from all source users, and the number of edges refers to the total number of follower-followee relationships between users.

In the experiment, we use 70% of the subset of $\mathcal{F}_2$ followed by each source user for training, 15% for validation,

[1] http://www.kddcup2012.org/c/kddcup2012-track1

Table 1: Statistics of the two datasets.

Dataset	# of sources	# of $\mathcal{F}_2$ users	# of edges
Twitter	158	3,195,481	6,822,720
Tencent	200	22,904	195,760

Table 2: Popularity versus similarity.

Algorithms	Twitter dataset		Tencent dataset	
	AUC	P@10	AUC	P@10
Adamic-Adar	0.7822	0.0144	0.7351	0.0934
SALSA	0.7458	0.0258	0.7168	0.1757
Neighborhood	0.6498	0.0186	0.6253	0.1326
MF-BPR	0.6363	0.0196	0.6730	0.1459

and 15% for testing. To reduce computational complexity, we further prune the $\mathcal{F}_2$ users that are followed by less than 5 users, as they are very unlikely to be followed by the source users. The metrics used for evaluating recommendation performance are AUC (Area Under the ROC Curve) and P@10 (Precision at top 10). The AUC metric measures the algorithm's overall ability to rank positive instances above negative instances, whereas the P@10 metric emphasizes more on the quality of the top 10 recommended instances. We report results of all algorithms on the testing set.

4.1 Popularity versus Similarity

We first compare the performance of the popularity-based algorithms (Adamic-Adar and SALSA) and the similarity-based algorithms (Neighborhood and MF-BPR) as shown in Table 2. The parameters of the MF-BPR algorithm are set as $K = 20$, $\lambda = 0.01$ and $\mu = 0.001$. The results show that the popularity-based algorithms generally achieve better AUC than the similarity-based algorithms. However, the similarity-based algorithms have quite good P@10 performance. Overall, combining the results suggests that while Twitter users tend to follow popular users, they also like to follow users with similar interests to them. Hence, there is a trade-off between popularity and similarity.

4.2 Combining Popularity and Similarity

For the rank aggregation approach, we experiment with different combinations of popularity-based and similarity-based algorithms, including AA & Neighb (Adamic-Adar & Neighborhood), AA & MF-BPR (Adamic-Adar & MF-BPR), SLS & Neighb (SALSA & Neighborhood), and SLS & MF-BPR (SALSA & MF-BPR). The parameter α is selected independently for each individual user such that the AUC is maximized on the validation set for that user.

For the popularity-biased collaborative filtering approach, we evaluate the Pop-Neighb (Popularity-biased Neighborhood) algorithm and the Pop-MF-BPR (Popularity-biased MF-BPR) algorithm. The parameter c is set to 1 for Pop-Neighb, and the parameters for Pop-MF-BPR are set as $\gamma = 1$ and $\beta = 0.01$, while other parameters are set the same as in the original MF-BPR algorithm.

The results in Table 3 show that the popularity-biased MF-BPR algorithm achieves the best performance in terms of both AUC and P@10. Comparing the results of rank aggregation algorithms with those of the individual algorithms in Table 2, we can see that rank aggregation can combine the advantages of the popularity-based algorithm and the similarity-based algorithm. The popularity-biased neighborhood method also improves over the original neighborhood method. In summary, the evaluation results confirm that combining popularity and similarity can improve the overall recommendation performance in terms of AUC as well as the quality of the top ranked recommendations.

Table 3: Empirical performance of algorithms combining popularity and similarity.

Algorithms	Twitter dataset		Tencent dataset	
	AUC	P@10	AUC	P@10
AA & Neighb	0.7873	0.0155	0.7458	0.1365
AA & MF-BPR	0.7835	0.0227	0.7540	0.1519
SLS & Neighb	0.7493	0.0247	0.7156	0.1746
SLS & MF-BPR	0.7606	0.0309	0.7392	**0.1862**
Pop-Neighb	0.7746	0.0186	0.7236	0.1657
Pop-MF-BPR	**0.7940**	**0.0536**	**0.7734**	**0.1818**

5. CONCLUSIONS

We proposed two approaches, the rank aggregation approach and the popularity-biased collaborative filtering approach, to exploiting both popularity and similarity for Twitter user recommendation. Through experimental evaluation on two real-world datasets, we showed that the rank aggregation algorithms can combine the advantages of popularity-based and similarity-based algorithms, and the popularity-biased collaborative filtering algorithms improve upon the original algorithms in terms of both AUC and P@10.

6. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1115199.

7. REFERENCES

[1] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW'10*, pages 613–622, May 2010.

[2] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. WTF: The who to follow service at Twitter. In *WWW'13*, pages 505–514, 2013.

[3] J. Hannon, M. Bennett, and B. Smyth. Recommending Twitter users to follow using content and collaborative filtering approaches. In *RecSys'10*, pages 199–206, 2010.

[4] R. Lempel and S. Moran. Salsa: The stochastic approach for link-structure analysis. *ACM Trans. on Info. Syst.*, 19(2):131–160, 2001.

[5] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM'03*, pages 556–559, November 2003.

[6] A. K. Menon and C. Elkan. Link prediction via matrix factorization. In *ECML PKDD'11*, pages 437–452, 2011.

[7] L. Page, S. Brin, R. Motwani, and T. Winograd. *The PageRank citation ranking: Bringing order to the Web.* Technical report, Stanford InfoLab, 1999.

[8] F. Papadopoulos, M. Kitsak, M. A. Serrano, M. Boguna, and D. Krioukov. Popularity versus similarity in growing networks. *Nature*, 489:537–540, September 2012.

[9] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI'09*, pages 452–461, 2009.

[10] D. Yin, L. Hong, X. Xiong, and B. D. Davison. Link formation analysis in microblogs. In *SIGIR'11*, pages 1235–1236, 2011.