Suggestion of True Popular Items

Huidrom Romesh Chandra Singh[1], T. kalaikumaran[2], Dr. S. Karthik[3]

[1] IInd M.E (CSE) rom_huidrom@yahoo.co.in
[2] Supervisor cum Associate Professor kalai_chennai@y ahoo.com
[3] Professor & Head kkarthikraja@yahoo.com
Department of Computer science and Engineering, S.N.S College of Technology, Coimbatore-35, Tamil nadu, India.

***Abstract*— Making of suggestion of items using user's feedback creates some problems in the actual ranking of the items. It populates some items while it suppresses some others. In short, it affects the original popularity of the items. Our goal in this paper is to make a suggestion of the items and rank them according to true/original popularity. We use some ranking and suggesting algorithms in order to achieve our goal. Our result provides a very effective performance which gives a true popularity of the items.**

Keywords – suggestion; ranking; user feedback; social tagging.

I. INTRODUCTION

Suggestion is a list of items presented to the users. This is made based on the user's feedback. The users give their preference of items (feedback) and use them in the ranking of items. The main aim of this paper is to learn the true popularity of items and suggest them to the user.

Item mentioned here can be anything like files, documents, search query keywords etc. A more specific application of this system is that of tagging where items are tags applied to the content e.g. photo (in fickr), web pages (in delicious) and video (in youtube) etc.. The users can choose the appropriate tags for an information object based on their preference. The existing tagging system is based on the history of tagging. The figure 1 shows a tagging system in delicious.com where the information object "http://www.e-pao.net" is tag with items i.e., Manipur, news, chat, e-pao, e-friends. Suggested items and popular items are also provided. Users can select items from suggestion or popular sets or create own tag items.

Suggestion of items to the users becomes complicated in the popularity of items. The user tends to select items from the suggested list more frequently. It is because of (1) Bandwagon (the user conform the choice of other users) (2) least effort (selecting from the suggested items is more easier than to think another alternative) (3) Conformance in vocabulary (no need to write whole word correctly). So the suggestion can skew the popularity over items [13]. In figure 2, the chart shows the disorder created by the suggestion. The item "news" becomes more popular if the item is suggested frequently. We see that suggesting popular item creates some problems in the popularity of the items, then why we made suggestion? There are lots of reasons; say it recalls what the candidate items are.

In order to remove this popularity skewness, we have to suggest whole items; which is not possible practically because of (1) limited user interface, (2) ability of the users to access little content, (3) not necessary to suggest less popular items. So the number of suggested items is fixed to small number, here we use seven items in the suggestion set.

In this paper, our goal is to prepare some algorithms for ranking and suggesting so that it enables to learn the users' true preference over items. The true preference is the user preference over items without any exposure to any suggestions.

In the remaining sections of the paper, we define the problem more precisely (section II), related works are discussed (section III), our ranking and suggesting algorithms (section IV), a summary of our results (section V), our numerical results (section VI), and we conclude our paper (section VII).

II. PROBLEM AND USER'S CHOICE MODEL

Here in this section we precisely formulate the problem of ranking and suggesting items, and define a user's choice model.

A. Assumption and Problem Formulation

Let we consider the users select items from the set $Z= \{1, 2, 3,\ldots, z\}$ which is the collection of all items, where $Z > 1$. Then $t = \{t_1, t_2, t_3, \ldots t_z\}$ be the users' true preference over the set Z. The t_i is the portion of the users who select the item i. The true popularity score t_i are such that (1) t_i is positive number, (2)

items are arranged such that $t_1 \geq t_2 \geq t_3 \geq \ldots \geq t_z$ (3) t is normalized i.e., $t_1 + t_2 + t_3 + \ldots + t_z = 1$.

The algorithm is formed by two rules i.e., ranking rule (the rule is to update the ranking scores of the items) and suggestion rule (the rule that specifies what item or set of items to be suggested to the users. We assign a fixed suggestion set size to s. let $r = \{r_1, r_2, r_3, \ldots r_z\}$ is the set of rank scores obtained from the algorithms.

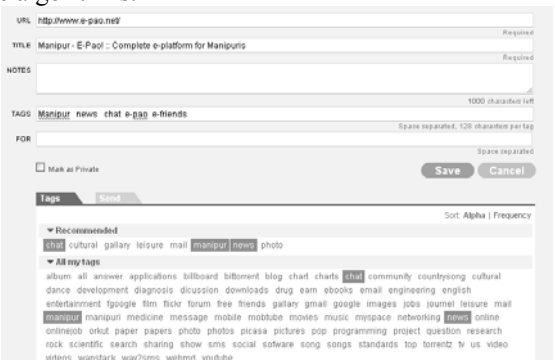

Figure 1. User tag entry form. The suggested tags are presented to the users.

The objective is to learn true popularity ranking of items which means the ranking produced by ranking scores r(p) at the time p is the proportional to that produced by true popularity ranking scores t, i.e., $t_i \geq t_j \Rightarrow r_i \geq r_j$ for any two items i and j.

We define the precision of set S of size s [11], [14] given by

$$\text{Prec}(S) = \frac{|\{i \in S : t_i \geq t_s\}|}{s} \quad (1)$$

B. User's Choice Model

In this model, the user can choose items either from suggestion set with probability p_s or from entire set of items by sampling with probability of $1\text{-}p_s$. The user's choice axiom was introduced by Luce [8] and also related to Luce – Shepard model [9], [1].

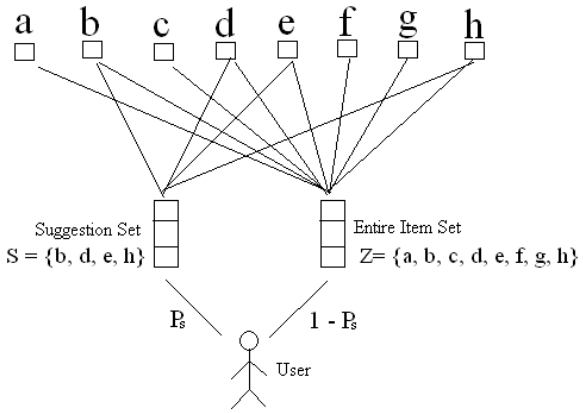

Figure 3. User's Choice Model

III. . RELATED WORKS

The way a user feed his/her choice is related to approval voting [5] where the users can select their desired candidate

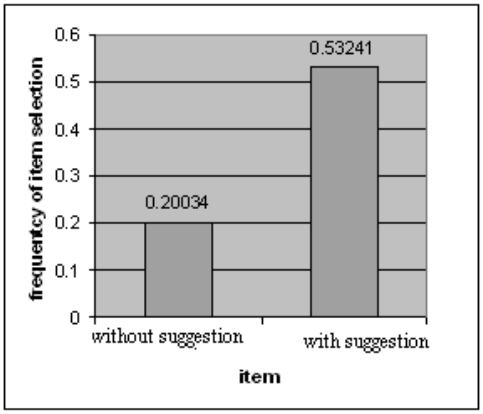

Fig. 2. Item popularity is boosted due to suggestion.

from a list of electing candidates. Suggestion in this paper is related to the way recommendation is made [6] where some popular things are presented to the users. Suggestion made some reinforcement in the popularity order which can be seen in preferencetial attachment [2]. Bandwagon and Underdog affects the voting results; see simon [12]. Our paper is related to the mathematical learning problem of the multiarmed bandit type [7] where each user is presented with an item that is selected by this user with probability specific to this item. A better rule to decide which item to present was founded by Lai and Robbin [7] to allow presenting more than one item. Entrenchment problem studied by Pandey et al [10] and Cho et al [3] gives a way where the search engine result lock down to a set of popular URLs. Imitation probability involved in this paper is related to the work of

Suchanek at al [13] for his finding of imitation rate. Finally we study on the social tagging system, Golder and Huberman [4].

IV. ALGORITHMS

In this section, we are going to present some algorithms for ranking and suggesting items which give a near approximate result of true popularity.

A. Base Algorithm (BA)

It is very simple and foundation of all ranking and suggestion algorithms. It contains algorithm both for ranking and suggestion. It is given as follow:

Init $c_i = 0$ *for each i item*
If item i is selected :
$c_i \leftarrow c_i + 1$
$S \leftarrow$ *a set of s items with largest c values*

Ranking is made on the number of selection of items in the past. If an item i is selected then its count i.e., c_i is incremented by 1. Initially all the items' count $c_i = 0$ for each item i. The suggestion set S containss s items.

B. Ranking Algorithm (RA)

This algorithm is similar to the above algorithm i.e., base algorithm where the count of an item is incremented by 1 whenever that item is selected. The algorithm is as follow:

Init c_i *:=0 for each item i*
At the p^{th} *item selection:*
If item i is selected :
$c_i \leftarrow c_i + 1$
$r_i \leftarrow c_i / p$

The ranking score is obtained from the counter of that item divided by p. It updates the ranking score of the item when the user selects it from the suggestion set or entire set of items.

C. Suggestion Algorithms

We present two algorithms for suggesting popular item.

1) Suggestion Algorithm 1 (SA1)

It is a randomized algorithm which updates suggestion set items iteratively. It updates suggestion set when a user selects an item which is not in the suggestion set. The algorithm is as follows:

At p^{th} *item selection*
If i item is selected and i is not in suggestion set, S
Randomly remove an item from S
Add i to S

Randomly removal of item from the suggestion set gives a chance to the less popular items to become popular. It does not require any counter to update suggestion set. The main feature of this algorithm is that it suggests last used item. Any item would frequently enter into the suggestion set provided only when the user select it frequently with positive probability. It does not properly maintain true popularity in the suggestion set as it allows entering less popular item into S.

2) Suggestion Algorithm 2 (SA2)

Our next algorithm for suggesting popular items is very much similar to the previous one except it has a counter. The algorithm checks the counter of the selected item with the counters of each item in the suggestion set. If the selected item's counter is enough suitable to suggest, then the algorithm remove one item from suggestion set S randomly and add that new item to S.

Init: N_i *:=0 for each item i*
At p^{th} *item selection*
If i is selected and i not in S
$N_i \leftarrow N_i + 1$
If N_i *greater than any N values of items in S*
Randomly remove one item from S
Add i to S

The counter N_i is incremented whenever the item i is selected but not suggested to the user. Moreover, a selected item that was not suggested does not immediately qualify to enter into suggestion set (as in suggestion algorithm 1) but only if its counter exceeds that of any item that is already in the suggestion set.

V. ANALYTICAL RESULTS

In this paper we present some algorithms for ranking and suggesting items. All these algorithms are based on the user's feedback. In this section we analyze these algorithms one by one.

First we take the base algorithm which suggests a fixed number of the topmost popular items, and this algorithm fail to learn the true popularity of item if the imitation probability is sufficiently large. There is a threshold on the imitation probability below which the algorithm shows true popularity, and otherwise this may not hold. The threshold is a function of the suggestion set size s and true popularity rank score t.

The threshold imitation is given by,

$$r_{thres}(t,s) = \min_{0 \le i \le s \le z} \frac{a(i,j)}{1+a(i,j)} \quad (2)$$

Where $a(i,j) = \left(\sum_{k=1}^{i} t_k + \sum_{k=j-s+i+1}^{j} t_k \right) \left(\frac{t_{i+1}}{t_j} - 1 \right)$

Suppose the imitation probability is $p_s < 1$, the top popular ranking are induced by ranking score which is given by

$$r_i(S) = \begin{cases} (1-p_s)t_i + p_s \dfrac{t_i}{\sum_{j \in s} t_j} & , i \in S \\ (1-p_s)t_i & , i \in Z \setminus S \end{cases} \quad (3)$$

Our randomized suggestion algorithms recursively update the suggestion set based on the item selected by users. The suggestion algorithm 1 (SA1) biases to show recently used items in the suggestion set. It does not require any counter for updating suggestion set. Under the user's choice model, for any imitation probability smaller than 1, the algorithm guarantees the frequencies of item selections give a popularity ranking that is proportional to the true popularity ranking. The main disadvantage is that if an item is selected at some frequency then it appears in the suggestion set in the same frequency. It boots the popularity of that item.

The frequency at which an item i is suggested f_i has the following properties: (1) the larger the item true popularity r_i , the larger the frequency f_i , and (2) the frequency f, is sublinear in t_i i.e., $f_i / t_i \le f_j / t_j$, for any items i and j such that $t_i \ge t_j$.

If we combine with ranking algorithm, the limit ranking scores induce the true popularity ranking i.e.,

$t_i \ge t_j \Rightarrow r_i(p) \ge r_j(p)$, for sufficiently large p.

Next is suggestion algorithm 2 (SA2) which is designed to suggest only popular items. It allows suggesting a new item if and if only it is likely to be more popular than atleast one item already in the suggestion set. It has a counter which is incremented by 1 when user selects that item which is not selected. Use of this counter is to mitigate the popularity biases. We show that it tends to display only sufficiently popular items with respect to their true popularity and fully determine this set of items in term of the suggestion set size and true popularity rank score of items.

The algorithm tends to suggest only a subset of sufficiently true popular items ("competing set"). It is characterized as follows:

1. **Competing set**. Suppose a subset of items {1, 2, 3, … c'} are suggested with strictly positive probability, where c' is larger integer i such that $s \le i \le z$ and $t_i > (1 - \frac{s}{i}) h_i(t)$
2. **Frequency**. The frequency at which an item is suggested:

$$S_i = \begin{cases} 1 - \left(1 - \dfrac{s}{c'}\right) \dfrac{h_{c'}(t)}{t_i}, & i = 1,2,...,c', \\ 0, & i = c'+1,...,z. \end{cases} \quad (4)$$

In summary, both suggestion algorithms are simple, lightweight with respect to storage and computation. There is no specific system configuration except suggestion set size.

VI. Numerical Results

In this section, all the numerical results are analyzed and
depicted in more favourable graphical format.

DATA COLLECTION. We collect data from the 352 students. The item here is food item name available in hostel mess menu list. We collect data in four phases; first we collect students' choice of food without any suggestion. In the next second, third, and fourth, we again collect using BA, SA1 and SA2 algorithms respectively. We get 7032 food item selections from these phases. Figure 4 summarizes for choosing an effective suggestion set size.

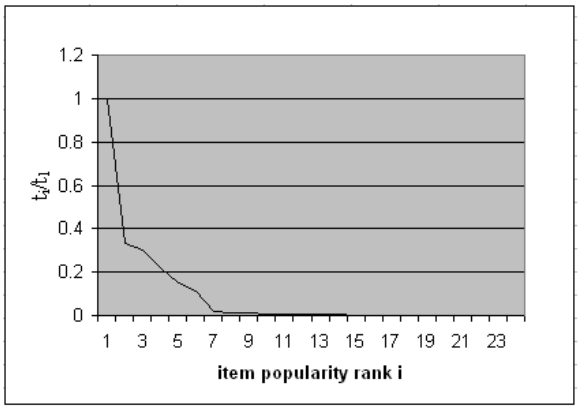

Figure 4. The graph shows the median ratio of the rank i score and rank 1 score of samples obtained for each student.

We observed that more exponential decay of the graph from rank 1 to 7 as compared from 7 to 24 rank. This provides a justification for limiting the suggestion set to 7 items.

LEARNING TRUE POPULARITY. True popularity refers to the popularity ranking obtained without any suggestion. In our paper, we used three algorithms for suggesting popular item to the users. In Figure 5. Shows that Base Algorithm (BA) and Suggestion algorithm 1 (SA1) are not suitable for obtaining true popularity while Suggestion Algorithm 2 (SA2) graph is proportional to that of true popularity.

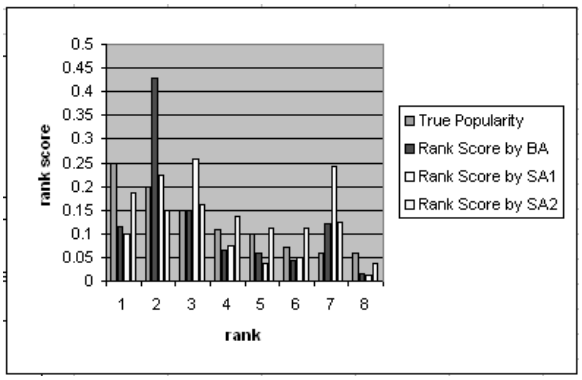

Figure 5. Rank scores obtained from all suggesting algorithms.

In Figure 6 (a), (b), and (c) show comparison between different algorithms and true popularity. The BA and SA1 graphs do not show proper decay as compared to true popularity graph. So they are not suitable to represent true popularity of items. SA2 graph i.e. Figure 6 (c) shows the best result as it declines with respect to true popularity.

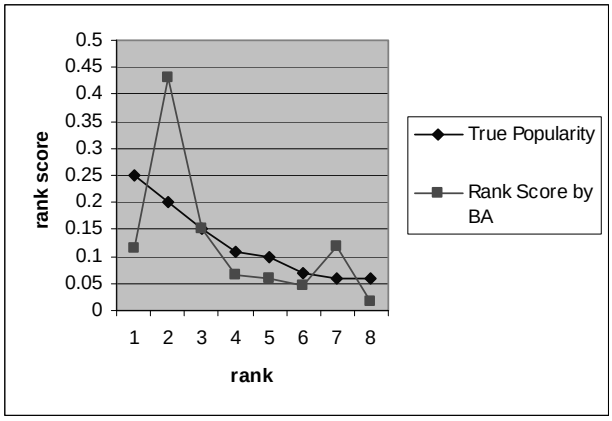

Figure 6. (a) Comparison between True popularity and BA

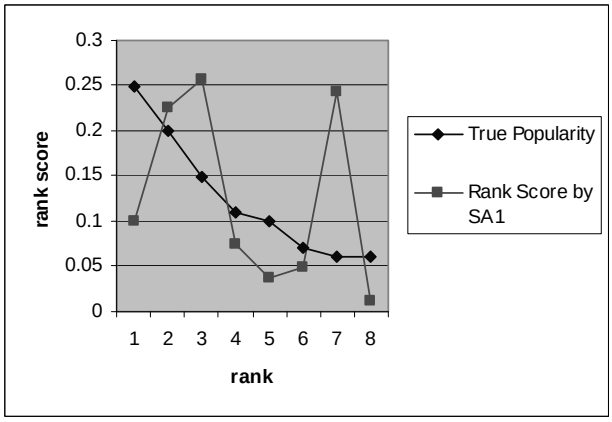

Figure 6. (b) Comparison between True popularity and SA1

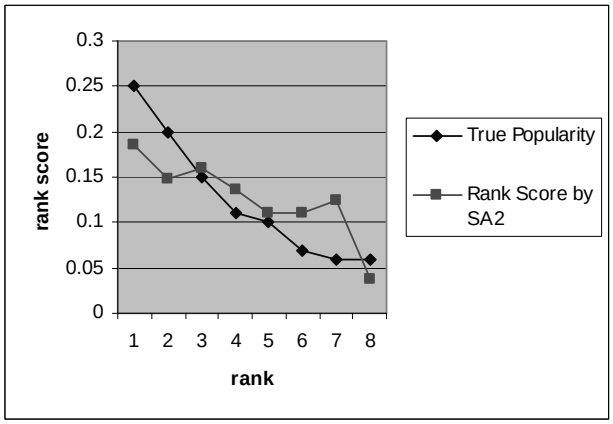

Figure 6. (c) Comparison between True popularity and SA2

VII. CONCLUSION

We present simple algorithms for suggesting true popular items. There are some limitations on these algorithms, say BA fails to show true popularity when the imitation probability is higher than its

threshold value. SA1 suggests less popular items if the user select them. SA2 with our ranking algorithm confine to display only sufficiently popular items. It removes the reinforcement in the popularity order due to suggestion, i.e., it maintains the true popularity of items.

ACKNOWLEDGEMENTS

Many thanks to the Director cum Secretary, Correspondent, Principal of SNS College of Technology for their motivation and constant encouragement. The authors would like to thank the Faculty Members of Department of Computer Science and Engineering for their helpful discussion. And also extending gratitude to the family members and friends who rendered their support throughout this research work.

REFERENCES

[1] J.R. Anderson, "The Adaptive Nature of Human Categorization," Psychological Rev., vol. 98, no. 3, pp. 409-429, 1991.
[2] S. Chakrabarti, A. Frieze, and J. Vera, "The Influence of Search Engines on Preferential Attachment," Proc. Symp. Discrete Algorithms (SODA), 2005.
[3] J. Cho, S. Roy, and R.E. Adams, "Page Quality: In Search of and Unbiased Web Ranking," Proc. ACM SIGMOD '05, 2005.
[4] http://en.wikipedia.org/wiki/Approval_voting.
[5] R. Kumar, P. Rajagopalan, and A. Tomkins, "Recommendation Systems: A Probabilistic Analysis," Proc. 39th Ann. Symp. Foundations of Computer Science (FOCS), 1998.
[6] T.L. Lai and H. Robbins, "Asymptotically Efficient Adaptive Allocation Rules," Advances in Applied Math., vol. 6, pp. 4-25, 1985.
[7] R.D. Luce, Individual Choice Behavior: A Theoretical Analysis. Dover, 1959.
[8] S. Pandey, S. Roy, C. Olston, J. Cho, and S. Chakrabarti, "Shuffling Stacked Deck: The Case for Partially Randomized Ranking of Search Engine Results," Proc. 31st Int'l Conf. Very Large Data Bases (VLDB), 2005.
[9] R.M. Phatarfod, "On the Matrix Occurring in a Linear Search Problem," J. Applied Probability, vol. 18, pp. 336-346, 1991.
[10] H.A. Simon, "Bandwagon and Underdog Effects and the Possibility of Election Predictions," Public Opinion Quarterly, vol. 18, pp. 245-253, 1954.
[11] G. Salton and M.J. McGill, Introduction to Modern Information Retrieval. McGraw-Hill Education, 1983
[12] S. Sen, S.K. Lam, A.-M. Rashid, D. Cosley, D. Frankowski, J. Osterhouse, F.M. Harper, and J. Riedl, "Tagging, Communities, Vocabulary, Evolution," Proc. 2006 20th Anniversary Conf. Computer Supported CooperativeWork (CSCW), 2006.
[13] H.A. Simon, "Bandwagon and Underdog Effects and the Possibility of Election Predictions," Public Opinion Quarterly, vol. 18, pp. 245-253, 1954.
[14] F. Suchanek, M. Vojnovic, and D. Gunawardena, "Social Tagging: Meaning and Suggestions," Proc. 17th ACM Conf. Information and Knowledge Management (CIKM), Oct. 2008.

AUTHORS

Huidrom Romesh Chandra Singh received his Master of Computer Applications degree from Dr. MGR University, India in 2008 and currently he is pursuing his Master of Engineering in Computer Science and Engineering in SNS College of Technology affiliated to Anna University, Coimbatore, India. He presented two papers in National Conferences.

T.Kalaikumaran received his M.E. degree in computer Science and Engineering from Anna University, Chennai, Tamilnadu, India in July 2006 and pursing PhD degree in Computer Science and Engineering in Anna University, Coimbatore, Tamilnadu, India. He is currently an associate professor at the Computer Science and Engineering department of SNS College of technology, Coimbatore, Tamilnadu, India. His research includes various techniques of data mining, data warehousing, online analytical processing and database systems. He has published 20 technical papers in various conferences. He has served on the program committees for a number of conferences and workshops.

Dr.S.Karthik is presently Professor& Head, Department of Computer Science & Engineering, SNS College of Technology, affiliated to Anna University-Coimbatore, Tamilnadu, India. He received the B.Sc degree from Bharathiar University in 1998 and M.Sc degree from the Periyar University in 2000, M.E Degree from Anna University Chennai and Ph.D degree from International University, Washington. His research interests include network security, web services and wireless systems. In particular, he is currently working in a research group developing new Internet security architectures and active defense systems against DDoS attacks. Dr.S.Karthik published many papers in international journal and conferences and has been involved many international conferences as Technical Chair and tutorial presenter. He is the guest editor and reviewer for 7 international journals. He is a active member of IEEE, ISTE and Indian Computer Society.