RESEARCH ARTICLE

I'm Like You, Just Not In That Way: Tag Networks to Improve Collaborative Filtering [version 1; peer review: 2 approved with reservations]

Jason Boorn, Debra S Goldberg

Department of Applied Mathematics, University of Colorado, Boulder, CO, 80302, USA

v1 First published: 26 Mar 2013, 2:95
<https://doi.org/10.12688/f1000research.2-95.v1>
Latest published: 26 Mar 2013, 2:95
<https://doi.org/10.12688/f1000research.2-95.v1>

Abstract

Collaborative filtering aims to predict a person's preferences by examining the preferences of similar people. Many collaborative filtering algorithms rely on a coarse notion of similarity, which assumes that if two people are sufficiently similar in a few specific areas, each is likely to make good recommendations for the other in most areas. Our trust in the opinions of others, though, is rarely absolute; we often tend to trust recommendations from certain people in certain areas. In this paper we develop an algorithm which reflects this notion. Rather than capturing taste information at the user level, we capture taste at the topic level by making use of tags: arbitrary words or phrases which are often used to group online content. Previous attempts to improve collaborative filtering using tag information have attempted to determine tag meanings, and as a result have depended upon complex semantic analyses. Our algorithm avoids these complications by focusing instead on the clusters which tags establish. Using tags in this way provides a significant improvement in the accuracy of recommendations without a commensurate loss in coverage. These tag clusters also give rise to networks which can be exploited to further improve recommendation results.

Keywords

Collaborative Filtering, Graph Theory, Information Retrieval, Recommender Systems

Open Peer Review

Approval Status ? ?

	1	2
version 1	?	?
26 Mar 2013	view	view

1. **George Karypis**, University of Minnesota, Minneapolis, MN, USA
2. **Zi-Ke Zhang**, Hangzhou Normal University, Hangzhou, China

Any reports and responses or comments on the article can be found at the end of the article.

Corresponding authors: Jason Boorn (jason.boorn@colorado.edu), Debra S Goldberg (debra@colorado.edu)

Competing interests: No competing interests were disclosed.

Grant information: The author(s) declared that no grants were involved in supporting this work.

Copyright: © 2013 Boorn J and Goldberg DS. This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. Data associated with the article are available under the terms of the [Creative Commons Zero "No rights reserved" data waiver](#) (CC0 1.0 Public domain dedication).

How to cite this article: Boorn J and Goldberg DS. **I'm Like You, Just Not In That Way: Tag Networks to Improve Collaborative Filtering [version 1; peer review: 2 approved with reservations]** F1000Research 2013, 2:95 <https://doi.org/10.12688/f1000research.2-95.v1>

First published: 26 Mar 2013, 2:95 <https://doi.org/10.12688/f1000research.2-95.v1>

1 Introduction

As digital repositories become larger and more numerous, discovering relevant information within them becomes more difficult. While there exist numerous tools (e.g. Google) to help us locate a piece of information in which we know we are interested, fewer options exist to guide us towards items of which we might be completely unaware.

Software and algorithms which attempt to solve this problem are known as recommendation systems. Recommendation systems aim to find new items from a target repository which a user is likely to find valuable. One of the more successful and widespread methods for recommendation involves a process known as collaborative filtering. Collaborative filtering attempts to predict unknown user preferences based upon known past behavior. Most often, the process of predicting preferences for a target user proceeds in one of two ways: we either find those users whose item preference behavior most closely matches that of the target user (user-based), or we find those new items whose ratings are correlated with the target user's existing ratings across a group of users (item-based)¹. Many current collaborative filtering algorithms can be applied to both user- and item-based prediction methods^{2,3}. The decision on which method to apply is therefore often a computational one: because the problem is expressed in terms of nearest neighbors, it is preferable to limit the search space. That is, if the repository has many more users than items, item-based collaborative filtering is normally applied, and if the repository has more items than users, user-based collaborative filtering is normally applied.

A crucial benefit of collaborative filtering is that it is domain free. That is, no knowledge of the underlying items is necessary, only a knowledge of user preferences. Because of this, collaborative filtering can be applied to a wide variety of repositories in a more or less generic form. This flexibility greatly improves its popularity, but also hampers its predictive power: when we ignore qualitative characteristics of the underlying items, we lose important information about why two users might share opinions on them.

In user-based collaborative filtering, this loss of information is especially relevant. Under this method, new item suggestions are drawn from the total number of items contained in a set of similar users. Under a user-based regime (where we have many more items than users), it is likely that this suggestion list contains a great many items which are completely unrelated to those which established the initial user similarity⁴. In a movie repository, for example, I might find a set of users whose taste in documentaries so closely matches my own that a collaborative filtering algorithm would deem them good recommenders for me. These same people, though, have presumably rated more than just documentaries, and if qualitative item information (such as genre) is absent, the suggestion algorithm is

unable to separate out these unqualified results. In short, we tend to trust certain people in certain areas, but generic user-based collaborative filtering does not take this fact into account. In order to differentiate between items from different areas, some kind of item information is required. But in order to maintain domain independence, this item information must come from user behavior and not from the underlying item characteristics.

2 Tag-Based Approaches

User-generated tagging systems have become popular ways to categorize online resources. In these systems, users are given the opportunity to attach a set of words or phrases to a resource which describe it in a way that makes sense to the user. In contrast to centralized hierarchical systems, these “folksonomies” are informal and unregulated. There is no defined schema for these descriptors; rather, users determine the words or phrases which best describe the associated item and attach these to the item when saving it.

Tags represent additional structure which may be used in a collaborative filtering algorithm. What is attractive about using tag information is that a tag can confer content information in a domain-independent manner. It can, in other words, give us a description of the underlying content without requiring a direct look at it. This comes at a cost, though, as the variety of tags that might be used to categorize a particular item seems endless. The flexibility which makes tagging so popular also limits its descriptive power.

Recent studies have attempted to use tag information to improve collaborative filtering results^{5,6}. In particular⁷, uses tag information to establish a “context” within which a user found the resource informative. This context becomes an additional layer through which suggestions can be filtered. Similarly⁸, uses tagging behavior to limit the number of possible items which can be suggested before applying a conventional user similarity approach. Here, tags are clustered by examining which items they are attached to across the entire dataset in an effort to characterize their meaning.

Studies such as those above are indicative of a popular strategy when working with tag information. Because a tag is fundamentally a categorization term, it seems natural to first discover a tag's semantic meaning. This, however, can prove a difficult task, and efforts to employ tag information often get bogged down in larger questions of semantic analysis.

In fact, though, tags offer useful information apart from their semantic content which can be used to improve the results of user-based collaborative filtering. We propose a novel algorithm which analyzes tagging behavior to extend collaborative filtering in a natural manner, one that is both domain independent and free of semantic context. We find that suggestions

generated from this graph outperform those generated from more conventional techniques, dramatically boosting accuracy without a commensurate decrease in coverage.

3 Similarity By Area

When a user tags an item, he makes a statement about its content. Although this statement can be highly subjective, it is assumed to be consistent. That is, other items for this user which share this tag represent one of the user's interests. We refer to all the documents tagged by an individual user with a particular tag as a tag group. If our goal is to compare two people, we might use an approach which finds the documents they have in common. If our goal is to compare the interests of two people, we might alternatively find the documents which their tag groups have in common. We take a vector t to represent a given tag group. $t_i = 1$ if the i^{th} item in the data set is included in tag group t , $t_i = 0$ otherwise. An expression for comparing different tag groups then uses cosine similarity over tag groups rather than over users:

$$\text{sim}(t, g) = \frac{\langle t, g \rangle}{\|t\| \|g\|}$$

where t and g are vectors representing items in respective tag groups, $\langle t, g \rangle$ is the dot product of these vectors, and $\|t\|$ is the magnitude (Euclidian norm) of the t vector.

It is important to note that this neighborhood does not take the semantic content of the tags into account. In this model they function in the same way that user names do in the conventional model: they serve only to delimit the group of items which are compared. In this way we sidestep the complicated issues of semantics which hamper many tag-based approaches, yet still manage to gain new information about content associations.

The most important consequence of comparing tag groups rather than users is that doing so limits the scope of the comparisons to a particular interest. This means that we have less information on which to base a comparison, but also that comparisons should be more focused. We see that these assumptions hold in a set of experiments performed using data from an online research paper repository.

4 The CiteULike Data Set

CiteULike is a social bookmarking website for researchers. It allows users to save and tag research articles. Because users may attach multiple tags to a document, it is possible (indeed highly likely) that a paper belongs to more than one of a user's tag groups, and also that it is contained in tag group(s) of other users (likely tagged with different words or phrases).

Every day, CiteULike publishes a data dump of its current repository. Here we use a dump from November 2007, because

a number of other papers in the field have used the CiteULike data dump from this period^{8,9}.

4.1 Spam Filtering

Because CiteULike publishes its data set on a daily basis, it has become an excellent resource for data in collaborative filtering. Unfortunately, it has also become an excellent target for spam. Unscrupulous marketers have seeded the data set with phony article descriptions designed to lead users to online offers. Many recent publications have used the data from CiteULike without acknowledging the amount of this material which is included in the daily data dumps. Before enacting experiments on the data set, we set out to scrub the underlying repository of extraneous spam data.

CiteULike publishes two data files: one which includes an entry for each documentid/userid/tag combination¹⁰, and one which associates these documentids with external URLs (called a linkout file)¹¹. CiteULike does not store full documents, only document descriptions. This URL information is therefore used to link a document description to its associated content. This content is usually the full text of a journal article, although any existing web page may be used.

Because setting up the target web page for an external URL requires some significant effort, we can be reasonably certain that any documentid which is associated with an external URL in the linkout file is not spam. As a first step to removing spam from the repository we simply removed all documents which did not have associated linkout entries. This made for a dramatic change in the dataset: based solely on the linkout test, roughly one third of the users, two thirds of the documents, and three quarters of the tags used in the corpus are associated with spam entries. We were somewhat surprised by this result, and so decided to verify the finding using an alternative method. A direct examination of those documents deemed questionable by the linkout test was undertaken. It is possible to query CiteULike by documentid directly. Legitimate documentids produce their corresponding descriptions, even when these descriptions do not contain a linkout entry. Documents which correspond to spam entries, however, fail to produce a document description at all. By querying the site by documentid directly, we can determine whether an individual document is spam. Note that this procedure is different from the linkout test, where we were

Table 1 CiteULike Data Before and After Filtering.

	Raw Data Set	With Linkout	Questionable Data
Users	27133	17366	36%
Documents	813186	296998	64%
Tag Groups	1226458	275233	78%

checking for the existence of an external URL associated with a given description. Here we check whether the description itself exists or not.

Our questionable data set contains over half a million documents. Given that the time required to query each of these would be prohibitive, we adopt a more aggressive approach: we query a given document, and if it is found to be spam we remove it as well as all the users who have included it in their saved documents. We assume that few non-spammers will include spam entries in their saved articles, and that spammers have an incentive to include the same spam document in more than one of their “false” profiles. A manual examination of the resulting data sets (potentially good and still questionable) verified the approach: tags selected at random from the questionable set proved suspicious (e.g. “free”, “cash”, “enlargement”), while tags selected at random from the potentially good set matched tags from known good profiles.

After performing this process on data determined to be questionable by the naive linkout file method, we found only 282 potentially good users, 2820 potentially good documents, and 10907 potentially good tag groups. These represented 1.0%, 0.3%, and 0.9% of the original data set, respectively. In other words, using just the linkout file to filter spam articles is a method which performs very well.

4.2 Data Set Characteristics

Figures 1(a), (b), and (c) show that spam removal tends to change the characteristics of the data set, which is to be expected. More importantly, removing spam from the data brings to light a crucial network property that can be used to dramatically improve the results of user-based collaborative filtering.

We alluded to a fundamental problem of accuracy: user-based collaborative filtering, because it does not take into account

the fact that we trust certain people in certain areas, tends to return suggestion lists which are not very accurate. While these suggestion lists do contain relevant items, they contain a large number of irrelevant items. In Figure 1 we see that this is indeed the case. In each figure, the magnitude of the slope of the regression line indicates the skewness of the distribution: lower magnitudes indicate fatter tails. Shown in Figure 1(a), for plots of number of papers per user, this value is -1.26 and -1.51 for raw and filtered data, respectively. This indicates relatively fat tails for the number of papers by user. In other words, a large proportion of users in the data set have saved a large number of papers to their profiles. Yet the data set is sparse; the number of papers shared by users is small. Therefore a suggestion list based on user similarity will contain a relatively large number of papers, many of these not included on the basis of this similarity.

We can see a similar situation develop when we examine papers by tag. Tag in this case refers to the semantic token; any papers tagged with the term “biology” for example, would be grouped together. Here the slope of the regression line is -1.67 or -1.5, indicating fat tails. If we attempt to compare based on tag, then it seems we will also return a large number of extraneous documents.

Turning to tag groups, however, we notice a change. A tag group in this case refers to just the set of papers which a user has grouped together using a tag; the semantic content of the tag itself is ignored. Here the slope of the regression line after filtering is -2.38, indicating much thinner tails. According to this, a comparison based on tag group should result in far fewer irrelevant documents. Moreover, we should get better results by ignoring the semantic content of the tags altogether, and just using them to define areas of trust.

A further result emerges from the analysis of tag groups. Not only does the slope of the regression line decrease to a value of -2.38, so does the mean squared error of its fit to the data.

Figure 1 CiteULike dataset characteristics before/after filtering. We plot the number of papers belonging to a given number of users, tags, and tag groups respectively, then plot a regression line through the data. All data are plotted in log/log format (base 10), and number of papers is plotted on the y-axis. The slope and mean squared error of the resulting regression line are displayed.

In other words, the distribution of papers by number of tag groups follows a power law much more closely after filtering. This fact might be useful in estimating the prevalence of spam in an arbitrary data set which uses tag groups.

4.3 Other Preprocessing

In order to apply the algorithms below to the data set, some additional preprocessing was required. In that singlet documents (documents saved by only one user) are impossible to analyze and represent only noise for purposes of investigation, these were removed. Additionally, because the investigation below depends upon tag group documents, we also removed tag groups which contained only one document. These two steps can depend on one another (i.e. removing tag groups with one document might create additional situations where a document is owned by only one user and vice-versa), so we ran both procedures iteratively until the number of documents converged to a representative set. Finally, we removed documents which were saved to the repository without any associated tags. Our final data set consisted of 4612 users, 32085 documents, and 40704 tag groups.

5 Experimental Results

We apply the collaborative filtering algorithms described above to the CiteULike data set after filtering with the linkout file and preprocessing as described above. We start with basic user collaborative filtering, which compares users to other users based upon which articles they share. We then apply the filtering algorithm described above which captures a more natural notion of trust in a given area. The results we obtain suggest a hybrid approach which combines the two; we apply this hybrid algorithm to the data set to achieve much improved suggestions. Results of this final experiment in turn suggest further areas of investigation which are discussed in the next section.

5.1 Evaluation Metrics

In order to evaluate the performance of our algorithm, we employ a set of metrics commonly used to validate leave-n-out suggestion algorithms. In these situations, we remove a target set of n items which we attempt to recover. *Recall* measures the percentage of items in the target set which appear in our suggestion list. *Precision* measures the percentage of the suggestion list which is made up of target set items. In a realistic suggestion environment, we care not only about how many items in the suggestion list match target list items, but also at what position in the list these matches occur⁴. We therefore measure *accuracy*, which represents the average position of a target item in the suggestion list. A common metric for evaluation which combines recall, precision, and accuracy is *mean average precision*, or *MAP*. Average precision is the sum of the precision at each relevant item in the suggestion list divided by the total number of relevant items in the collection. If we average this value over all suggestion lists, we get the MAP for the suggestion algorithm.

A perfect MAP score, then, must recall all relevant documents and place them at the top of the suggestion list for a perfect score of 1.0. A final metric which we use in evaluation we call *satisfaction*, and is defined as the percentage of users in the set for whom at least one suggestion is found to match an item in the target set. Satisfaction is useful in isolating the percentage of cases which represent complete failures of the algorithm.

5.2 User Collaborative Filtering

To see how user-based collaborative filtering performs on the CiteULike data set, and to establish a baseline for subsequent experiments, we apply a conventional form of user-based collaborative filtering. The procedure is as follows:

- 1) Select a user at random who has saved at least 6 items; call this the target user.
- 2) Remove half of the documents belonging to this user; call this the target set.
- 3) After removing the target set documents from the target user document vector, find the k users in the repository whose document vectors are most similar to its remaining document vector using cosine similarity.
- 4) The documents belonging to the document vectors of these k users, but not currently contained in the target user's document vector, represent the suggestion list. Score the suggestion list according to criteria described in section 5.1.

The above procedure depends upon a single parameter k : the size of the user neighborhood that is used to generate the suggestion documents. Previous studies on this data set suggest that generic user-based collaborative filtering algorithms perform best at values of k between 5 and 12⁹. Table 2 lists results for values of k in (5,8,12). We see that satisfaction (percentage of users who received at least one good suggestion) is fairly high, but accuracy is dismal (recall lower accuracy scores are better). This agrees with our earlier assumptions about user-based filtering in the context of trust. The algorithm returns suggestions from all of a particular user's interests – not just those areas in which his interests align with the target user's. Because of this, a large number of the returned suggestions are irrelevant, and accuracy suffers.

Table 2 User Collaborative Filtering Results.

Metric	k=5	k=8	k=12
Recall	0.351	0.384	0.438
Precision	0.067	0.051	0.040
MAP	0.213	0.191	0.158
Satisfaction	0.804	0.859	0.887
Accuracy	57.9	89.1	129.7

Table 3 Tag-Based Collaborative Filtering Results.

Metric	k=5	k=8	k=12
Recall	0.229	0.278	0.329
Precision	0.108	0.089	0.072
MAP	0.249	0.255	0.251
Satisfaction	0.508	0.600	0.681
Accuracy	8.3	11.3	16.1

5.3 Tag-Based Filtering

We now apply the tag-based algorithm that implicitly specifies the interest area of a target user by comparing tag groups. We start with the natural approach detailed above; instead of comparing users to other users, we compare user interests to other user interests. The procedure is as follows:

- 1) Select a tag group at random which contains at least 6 items; call this the target group.
- 2) Remove half of the documents belonging to this group; call this the target set.
- 3) After removing the target set documents from the target group document vector, find the k tag groups in the repository whose document vectors are most similar to its remaining document vector using cosine similarity.
- 4) The documents belonging to the document vectors of these k tag groups, but not currently contained in the target group's document vector, represent the suggestion list. Score the suggestion list according to criteria described in section 5.1.

Again, we run the procedure for values of k in (5,8,12). Results are listed in Table 3. The assumptions relating to trust appear again to hold. Here we have paid a nominal price in satisfaction for a huge improvement in accuracy. The improvement in accuracy is sufficient to raise the overall MAP score between 25% and 65% depending on k .

6 Analysis

The above results show that similarity comparisons performed with respect to particular subject areas outperform those performed on a simple user basis. They imply that, with respect to suggestions, we do tend to trust certain people in certain areas. Although we are less likely to recall as many of the target documents which represent good suggestions, those we do find appear much earlier in the suggestion list. In a practical setting, one could argue that the latter benefit greatly outweighs the former loss; few users will look further than 20 items deep in a suggestion list to ferret out an interesting recommendation⁴.

Looking more closely at the results above, a fundamental trade-off emerges. User-based collaborative filtering on this data set

produces suggestions which are high in recall and satisfaction, but poor with respect to accuracy. Tag-based collaborative filtering on this data set produces suggestions which are lower in recall and satisfaction, but high with respect to accuracy. Tag-based filtering produces more accurate results because the denominator of the cosine similarity measure includes a term for the size of the tag group – effectively this penalizes larger tag groups. As a result, the suggestion list returned is smaller. This also explains the loss in recall we see above.

It is important to point out that the loss in recall we see with the tag-based algorithm lies mostly in a loss in satisfaction. Comparing the two algorithms, we see that using tags to drive similarity results in a loss of recall of 12%, 11%, and 11% for various k . We see a corresponding loss of satisfaction of 30%, 26%, and 21% respectively. In that satisfaction measures the number of groups for which any good recommendation was found, it appears that the price we pay in recall is primarily due to an inability to establish any good recommendations for a fairly large subset of the tag groups. That is, for roughly one-third of the tag groups tested, the algorithm is unable to find other tag groups which contain any of the hidden documents. The data set was constructed such that every document is guaranteed to exist in at least one other tag group. This seems, therefore, a rather large number of failures.

Our network analysis of the data suggests why, and underscores the degree to which sparsity can be a problem in collaborative filtering. Recall that our algorithms hide half of the target data set; for user-based filtering we hide half of the user's documents and for tag-based filtering we hide half of the documents within that tag group. Because a user will be associated with at least as many documents as his largest tag group, removing half of his overall documents leaves us with a larger set on which to base recommendations. When we remove half of the documents which belong to a tag group, by contrast, this can leave us with far fewer (in some cases only 3) documents on which to base a similarity judgment. In an extremely sparse environment, this can have major consequences. It is quite probable, for example, that the documents removed from a tag group are part of a disconnected cluster: the other tag groups which contain these hidden documents do not also contain the non-hidden documents required to make them a part of the similarity neighborhood. This circumstance, moreover, is much more likely when dealing with smaller tag group sets than with larger user document sets.

We used a more realistic notion of trust to drive the tag-based algorithm. Taking a cue from¹³, the question we pose next is whether or not this notion also carries with it the prospect of transitivity. That is, can we use the transitive aspect of trust to establish “trust networks” and extend our model of similarity? If so, does such an extension present new alternatives which can achieve both high accuracy and high coverage?

7 Network Methods

To frame the basic similarity question in terms of a network, we first define the trust network. This structure is a graph, where tag groups are represented by nodes and similarities between tag groups by links. We use the cosine similarity measure described above to establish weights on these links. This structure can be represented as an adjacency matrix, where the (i, j) element of the matrix represents the cosine similarity between tag groups i and j . Our naive tag-based similarity algorithm can use this matrix to generate the k tag groups in a group's neighborhood by looking at the row corresponding to the target tag group. That is, if we are looking to generate the k tag groups which are most similar to a particular tag group t , we order the elements of the row corresponding to t and choose the first k of them. Our decision to use a network approach to boost satisfaction (and thus recall) derives from an analysis of the tag based algorithm's failures. The vast majority of tag groups for which no good suggestion can be found have a particular network structure, as illustrated in Figure 2. From our target group t we have hidden a set of documents. Documents of this hidden set, h , are also found in a set of tag groups G . The problem arises when an insufficient number of the documents in G are also contained in the non-hidden set of documents in t . Without this similarity, no links can be made between t and G using the algorithm above, and therefore no tag group of G is positioned to make recommendations. In that the set G is uniquely qualified to make recommendations on the hidden set of documents, the algorithm fails for t . It is this type of failure which most impacts the loss in recall seen above.

We know, however, that every document in the set G must also exist in some other tag group. Although it may be the case that all the hidden documents of t exist only between t and G , it is also the case that G contains many other documents, and these must link to the larger graph in some way. Moreover, even though a failure indicates that an insufficient number of the documents in G are also non-hidden documents of t , it is likely that some of the documents in G are also documents within k^* (the neighborhood of similar tag groups initially

constructed). If we suppose that strong links are likely to exist between k^* and G , we can attempt to find these and use this information to generate suggestion lists of better recall.

7.1 A Hybrid Approach

The network methods we seek to apply here depend upon tag group comparisons over the entire tag group set. Attempting to analyze the entire network of tag groups would be either computationally intractable or at the very least extremely expensive. In order to produce a viable algorithm, we must find some way to limit the number of tag groups, and the resulting tag group network structure.

Our hybrid approach can be broken down into two steps. In the first step, we calculate a neighborhood of users based upon overall document similarity (as in User-Collaborative Filtering, above). Once this neighborhood is established, we can filter the suggestion results by employing tag-based filtering. Here, instead of comparing over the entire data set, we limit our comparisons to tag groups included in the initial user neighborhood.

- 1) Select a tag group at random which contains at least 6 items; call this the target group.
- 2) Remove half of the documents belonging to this group; call this the target set.
- 3) For the user associated with the target group, run the procedures detailed in section 5.2 to produce a neighborhood of k similar users.
- 4) After removing the target set documents from the target group document vector, find the k tag groups *in this user neighborhood* whose document vectors are most similar to its remaining document vector using cosine similarity.
- 5) The documents belonging to the document vectors of these k tag groups, but not currently contained in the target group's document vector, represent the suggestion list. Score the suggestion list according to criteria described in section 5.1.

Here, we have limited the set of tag groups on which we compare to those in the initial user neighborhood. This approach is similar to⁸, except that the process is reversed. In⁸, a set of tags is employed to limit the neighborhood under which user similarity is calculated. Here, a set of users is employed to limit the neighborhood under which tag similarity is calculated. Additionally, our method does not make explicit use of tagging semantics, which are required in Zanardi's model in order to generate a tag neighborhood⁸.

The results of this modification are detailed in Table 4 at user neighborhood (k) values in (5,8,12).

Clearly, this method does not outperform the simple tag-based approach above. What is interesting, though, is that it does not

Figure 2 A Typical Suggestion Failure.

Table 4 Hybrid Collaborative Filtering Results.

Metric	k=5	k=8	k=12
Recall	0.265	0.320	0.371
MAP	0.237	0.246	0.248
Satisfaction	0.509	0.594	0.661
Accuracy	10.5	13.8	17.5

suffer terribly. In other words, while limiting tag groups to a neighborhood of similar users does not improve the algorithm, it doesn't hamper it considerably, even when we consider relatively small user neighborhood sizes (e.g. 5). Under these conditions, the resulting tag group graph becomes a feasible target of network methods.

We also see that the naive user-based algorithm implemented above still offers considerably higher satisfaction and recall. In other words, the user neighborhood established under this naive approach contains a substantial number of documents that are not being found under the tag-based approach; there is still room for improvement.

7.2 Mutual Clustering Coefficient

Now that we have established a neighborhood under which to apply network approaches feasibly, we return to the problem sketched above. Our goal is to find those tag groups in G that are not found using the basic tag-based algorithm. Because the user-based algorithm of section 5.2 delivers much higher levels of satisfaction and recall, we can be fairly confident that some of the tag groups in G lie within the neighborhood we've created.

Generally, this problem can be thought of in terms of clustering: what we are attempting to do is cluster k and G so that we can draw from documents in G even though G itself does not contain non-hidden documents from t . Many different attempts at graph clustering have been proposed for problems ranging from sociology to biology. Here, we first employ a straightforward graph metric which has been used to identify protein interactions in small-world networks¹².

The clustering coefficient of a graph measures its cohesiveness. For an individual vertex, we count the number of links among its immediate neighbors and divide by the total number of possible links in this set. An average of this value for all vertices gives us the clustering coefficient. The mutual clustering coefficient extends this idea by measuring how many neighbors are shared between two vertices. For two vertices, we use a cumulative hypergeometric to generate a p value which corresponds to the likelihood of this or a more extreme shared configuration occurring by chance.

Formally, for tag groups a and b , we take the neighborhood of each ($N(a)$, $N(b)$), the number of neighbors in common ($N(a) \cap N(b)$), and the total number of tag groups in the graph ($Total$) we calculate:

$$MCC_{ab} = -\log \sum_{i=N(a) \cap N(b)}^{\min(N(a), N(b))} \frac{\binom{N(a)}{i} \cdot \binom{Total - N(a)}{N(b) - i}}{\binom{Total}{N(b)}}$$

The summation represents the p value cited above. Taking the negative log allows us to compare different p values more intuitively.

Higher mutual clustering coefficients signify a greater likelihood that a link should exist between two tag groups. This is useful in the situation we're addressing; a large number of shared neighbors among t and G should imply the existence of a similarity where the cosine algorithm has been unable to find one for reasons cited above.

We apply this measure to our graph, this time generating a neighborhood k by finding the tag groups with the highest mutual clustering coefficients to the target tag group t . Results for running this procedure for various k are listed in Table 5.

The algorithm delivers in one sense: we've gained in recall and satisfaction. However, we've paid a steep price in accuracy which serves to bring our overall MAP score down significantly. A closer look at our algorithm suggests a reason why: whereas the naive tag-based similarity network weighs document suggestions based directly on link weights, here we use link topology. In some sense, each individual similarity link here counts equally in the overall assessment of similarity. Once any type of similarity is encoded as a link, the weight of that link is ignored in the topological calculation above. Recall that the main strength of the naive tag-based algorithm was its ability to filter according to tag group size. This is precisely the information that is lost in the topological assessment above.

It is possible that the above algorithm could be modified to introduce a notion of threshold similarity and take link weight

Table 5 Mutual Clustering Results.

Metric	k=5	k=8	k=12
Recall	0.300	0.369	0.428
Precision	0.050	0.040	0.031
MAP	0.173	0.156	0.151
Satisfaction	0.562	0.662	0.729
Accuracy	21.5	43.1	44.9

into account. That is, if we only include those links which are sufficiently powerful, we might recapture some of the lost information. Doing so would introduce another parameter to the model, though, one whose value appears on first glance to be somewhat subjective.

It is also possible that a version of mutual clustering coefficient which takes link weights into account directly might be applied. There have been efforts to establish a weighted version of the clustering coefficient, but to date no weighted version of the mutual clustering coefficient has been developed.

7.3 A Random Walk

In our discussion of recent developments above, we noted a method which has been used in networks to leverage the putative transitivity of trust relationships. A notion of spreading activation is used in Massa to model trust in online networks¹³. Most models of spreading activation resemble Hopfield networks. In these, nodes are binary threshold units which can be activated if input to the node exceeds some threshold. Nodes are connected with weighted links, and the weight of a given link determines how much input it provides to its target once it has been activated.

While a spreading activation model can be used to capture important qualities of a network based on transitive principles, its main benefit lies in its simplicity. Moreover, it is unclear why a binary mechanism better models a real-world notion of trust than would a continuous mechanism. Here, instead of a spreading activation model we employ a model which simulates a random walk on the similarity graph. This graph model allows for continuous similarity values in (0,1) and can be extended to an arbitrary number of steps.

We begin with the basic graph described above, which details the similarity of every tag group to every other within our user neighborhood. From this we generate n tag neighborhoods. Our first neighborhood is identical to the one we computed for the naive tag-based algorithm: it is generated by reading the top k similarities from the t row of the adjacency matrix. If we remove the (t, t) entry, and normalize the matrix to be row stochastic, this first set of tag groups can be seen as the most likely destinations in a one-step random walk from the t node in the similarity graph. Successive neighborhoods are simple matrix

products: if we multiply the matrix generated in step n by the original (step 0), the t row gives us the probability of moving from t to each other tag group in the network in $n + 1$ steps. The probability of moving to a given tag group other than the target in $n + 1$ steps is added to the probability of moving to it in steps 1 through n and the tag groups with the highest probability scores overall are selected for the final neighborhood. Each document in a tag group receives that tag group's probability score (these are cumulative for each document across tag groups). The final suggestion list orders documents according to these scores.

With this algorithm, we have two parameters – one for the size of the initial user neighborhood, k , and one for the size of the final tag group neighborhood (with the highest probability scores) which we call z . We can also walk the graph for an arbitrary number of steps n . We try the algorithm for $n = 2$ and $n = 3$.

We first experiment with various values of (k, z) where $n = 2$ (Table 6). Here a random walk appears to perform better than mutual clustering. MAP scores for $k = z$ are close to what they are for the hybrid model without network analysis, and accuracy scores are also roughly on a par. Unfortunately, we have not achieved a significant improvement in satisfaction or recall at $k = z$. When we move up to larger tag neighborhoods, we see that the accuracy diminishes to such an extent that the overall MAP score is decreased. While any practical algorithm for suggestion would not be based solely on this score, the benefit of MAP is that it puts a premium on accuracy – a metric which can be argued is more important than recall in a real system. As mentioned above, users are unlikely to peruse a list of suggestions beyond the first 20 or so items.

Our next results detail a walk with various values of (k, z) where $n = 3$ (Table 7). Our results for a 3-step random walk appear slightly worse than those for the 2-step. Satisfaction and recall are roughly similar, but accuracy is slightly lower for the 3-step walk. This is probably due to the fact that our 3-step algorithm tends to weigh tag groups which are further out in the similarity graph a fraction higher than those closer to the target. As a result, those documents are slightly overweighted, and will appear higher in the suggestion list than is probably warranted.

Table 6 2-step random walk results for various sizes of user neighborhood and tag group neighborhood (k,z).

Metric	(5,5)	(8,8)	(12,12)	(5,7)	(8,12)	(12,18)	(5,10)	(8,16)	(12,24)
Recall	0.271	0.309	0.369	0.282	0.349	0.405	0.313	0.382	0.433
MAP	0.228	0.232	0.233	0.214	0.213	0.220	0.215	0.213	0.213
Satisfaction	0.512	0.602	0.670	0.538	0.631	0.702	0.578	0.659	0.723
Accuracy	11.0	15.6	19.6	14.3	20.7	25.9	18.1	24.8	31.1

Table 7 3-step random walk results for various sizes of user neighborhood and tag group neighborhood (k,z).

Metric	(5,5)	(8,8)	(12,12)	(5,7)	(8,12)	(12,18)	(5,10)	(8,16)	(12,24)
Recall	0.252	0.309	0.358	0.286	0.356	0.405	0.310	0.379	0.437
MAP	0.219	0.218	0.216	0.213	0.207	0.215	0.203	0.201	0.208
Satisfaction	0.499	0.584	0.659	0.542	0.637	0.707	0.567	0.657	0.734
Accuracy	11.4	15.4	20.4	14.4	20.9	26.2	18.2	24.6	31.5

The results above show that we can achieve good recall without leaving the initial user neighborhood. However, doing so often involves increasing the size of the tag group neighborhood significantly, and with such an increase comes a decrease in accuracy. While the accuracy of both random walk models is significantly better than that of the naive user-based algorithm, high levels of recall (e.g. over .4) are associated with low list position (e.g. over 30). Again, in a practical situation this is probably not a reasonable tradeoff.

We believe there exists a remedy, however. Overall, our strategy has been to look at the data set in sets of successively smaller neighborhoods: we first confine the results to a group of similar users, then confine this set according to tag groups. The former strategy does not appear to severely limit the number of good recommendations that can be made, and the latter strategy helps to remove items which do not relate to the target user's area of interest. If we continue this approach, and look to characterize the resulting tag group network in terms of its items, it is possible that we can improve the accuracy of recommendations and so improve the overall MAP score. If, for example, we look to find those items which are most important for maintaining cohesiveness of the tag graph, it would make sense that these items appear higher in the suggestion list. Approaches similar to this one are reserved for future study.

8 Conclusions and Future Work

We have seen that a suggestion algorithm that takes into account a natural conception of trust outperforms one which does not. Moreover, recommendations from a user community should be sensitive to the fact that we trust certain people in certain areas. We can use the most widespread method of on-line categorization, tagging, to help delineate these areas of trust without resorting to semantic analysis of the tags themselves.

In a sparse environment, the network which emerges from a comparison based on tag groups is reasonably well-confined. That is, most of its structure is contained within a small set of similar users. But even in a sparse environment, suggestions made on the basis of similar users alone has the potential to be highly inaccurate. The act of categorization itself, however, provides a useful filter which does not depend on the content of the underlying items. Using tags, we can gain a huge

improvement in accuracy with a modest decrease in recall. It is assumed that tags need not be used per se – that is, any system which allows users to group items into categories would aid in recommendation, even when that system does not derive from a shared ontology.

A similarity model based on coarse user-user comparisons has very little cohesiveness: people outside a user's immediate neighborhood are unlikely to be similar enough to be useful as recommenders. However, because tag-based similarity is much more focused, it is possible to construct a similarity model which exhibits transitivity: if I trust person A in a particular area, and he trusts person B in the same area, the probability of B representing a good recommender is reasonably high. This property results in networks with more structure, and we believe this structure can be exploited to generate better suggestions.

Here we relied upon a take-out-half approach, as we felt this method best reflected the circumstance of a typical user. In the future it would be helpful to apply a takeout-one approach, which is also common in this field, to compare the results. As mentioned above, we would also like to attempt a modification to mutual clustering which allows for weighted edges, as this would not require that we ignore this crucial information when making assessments of network topology.

We made attempts above to exploit the structure of the resulting tag group graph with mixed results. Mutual clustering posed accuracy problems because it did not take similarity measurements into account when arriving at link probabilities. A weighted version of this algorithm might fare better on this count.

We also suspect that a more thorough analysis of the networks which emerge would suggest additional methods to test. An understanding of how these graphs cluster, for instance, might shed light on when to look only at the initial neighborhood of tag groups and when to look deeper into the graph. A comparison of different networks across different data sets might also be helpful in understanding how the underlying network structure can help produce effective suggestions. An analysis of which items are most central to the tag graph structure might

also prove valuable in determining their position in the recommendation list.

Author Contributions

JB conceived the project. JB and DSG designed the algorithms. JB wrote the code. DSG supervised the work. JB prepared the original manuscript. JB and DSG contributed to the final draft of the manuscript.

Competing Interests

No relevant competing interests disclosed.

Grant Information

This work was funded by CU-Boulder and by National Science Foundation award MCB-0630250 to DSG.

Acknowledgements

We thank Jem Corcoran and Juan Restrepo for their feedback.

References

1. Herlocker JL: Understanding and Improving Automated Collaborative Filtering Systems. Ph.D. Dissertation, University of Minnesota, 2000.
2. Adomavicius G, Tuzhilin A: Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*. 2005; 17: pp. 734–749.
3. Bell R, Koren Y, Volinsky C: Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems. ATT Research, Netflix Research Track Paper, 2007.
4. McNeel S, Konstan JA: Meeting User Information Needs in Recommender Systems. Ph.D. Dissertation, University of Minnesota, 2006.
5. Li X, Guo L, Zhao YE: Tag-based Social Interest Discovery. *Proc. of the 17th International Conference on World Wide Web*. 2008; pp. 675–684.
6. Tso-Sutter KH, Marinho LB, Schmidt-Thieme L: Tag-aware Recommender Systems by Fusion of Collaborative Filtering Algorithms. *Proc. of the 2008 ACM symposium on Applied Computing*. 2008; pp. 1995–1999.
7. Nakamoto R, Nakajima S, Miyazaki J, *et al*: Tag-Based Contextual Collaborative Filtering. 18th IEICE Data Engineering Workshop, 2007.
8. Zanardi V, Capra L: Social Ranking: Uncovering Relevant Content Using Tag-based Recommender Systems. *Proc. of the 2008 ACM conference on Recommender systems*. 2008; pp. 51–58.
9. Bogers T, van den Bosch A: Recommending Articles Using CiteULike. *Proc. of the 2008 ACM Conference on Recommender systems*. 2008; pp. 287–290.
10. CiteULike, "Who-posted-what data" [Data File], <http://static.citeulike.org/data/2007-11-02.bz2>, Retrieved November 8, 2008.
11. CiteULike, "Article linkout data" [Data File], <http://static.citeulike.org/data/linkouts-2008-02-02.bz2>, Retrieved November 8, 2008.
12. Goldberg DS, Roth FP: Assessing experimentally derived interactions in a small world. *Proc Natl Acad Sci U S A*. 2003; 100: pp. 4372–4376.
13. Massa P, Avenasi P: Trust-aware Collaborative Filtering for Recommender Systems. International Conference on Cooperative Information Systems (CoopIS), 2004.

Open Peer Review

Current Peer Review Status: ? ?

Version 1

Reviewer Report 27 August 2013

<https://doi.org/10.5256/f1000research.1150.r1578>

© 2013 Zhang Z. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Zi-Ke Zhang

Institute of Information Economy, Hangzhou Normal University, Hangzhou, China

This paper, entitled 'I'm Like You, Just Not In That Way: Tag Networks to Improve Collaborative Filtering', describes an algorithm to improve the performance of collaborative filtering, by introducing tag groups. In addition, in order to improve the performance of former naive tag-based approaches in metrics recall and satisfaction, the author also constructed trust tag networks and proposed several methods: hybrid approach, mutual clustering coefficient and random walk. It is interesting and innovative to introduce tag groups and tag networks which can avoid complex semantic analyses, and improve the performance of collaborative filtering in some aspects. The author also did well in the data pre-processing and methods discussion. However, the article needs to improve its clarity in the methods introduction. Also, the background of training steps is fuzzy. My technical comments are as follows:

1. In most of the method introductions, the author may pay too much attention to semantic description of these approaches, but ignores their simplicity and clarity, which may seem somewhat confusing. For example, in the section of 'a random walk', much semantic description of random walk method in network may push readers to a confusing state. It would be clear and simple by introducing matrix multiplication and probability equations. The same situations happens in the section 'a hybrid approach', 'mutual clustering coefficient', and so on.
2. My key concern is that all of approaches in this article have a step in the method where half of the documents belonging to target user or tag group are removed, which called the target set. In my mind, this step may mean separating training and testing sets for the recommendation task. However, it seems somewhat obscured. Based on prior experiences, all of training and testing samples are selected from a data set by specific or random ordering at one time. I do not sure whether this removing step is sound, without strong evidence cited to support this method.
3. There are a few minor concerns in this article. For instance, values of metric precision in Table 4 are missing, without specific reasons.

Above all, I suggest this paper is revised and submitted again.

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Reviewer Report 19 August 2013

<https://doi.org/10.5256/f1000research.1150.r1348>

© 2013 Karypis G. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

George Karypis

Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA

The paper explores an interesting use of tags within the context of recommender systems. My reservations regarding the paper have to do with:

1. The baseline comparisons

The baseline approach is a user-based content filtering of similar users. Though this is an acceptable baseline, it would also be nice to see how a pure collaborative-filtering approach works as well (i.e., no content, just article IDs).

2. The interpretation of the results

It is not fair to compare the results coming from the first set of experiments, as they use a different approach when doing a train/test split. The results between the two methods can only be compared if the first step of both methods is the same. My suggestion is to use the "Select a tag group at random which contains at least 6 items; call this the target group." approach for all experiments. The reason that the reported results are not comparable, is the the train/test scheme used for the tag-based results has created a very homogeneous test set relative to the training set, and as such, the prediction problem is easier. On the other hand, the train/test scheme used in the first set of experiments, when the user has rated many items, will be significantly more diverse, and as such it would be harder for the prediction method to identify the hidden articles, as they can cover only a subset of the user's interests.

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of

expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research