Improving Recommendation Accuracy by Clustering Social Networks with Trust

Tom DuBois
Computer Science Department
University of Maryland, College Park
College Park, MD 20741
tdubois@cs.umd.edu

Jennifer Golbeck
Human-Computer Interaction Lab
College of Information Studies
University of Maryland, College Park
College Park, MD 20741
jgolbeck@umd.edu

John Kleint
Computer Science Department
University of Maryland, College Park
College Park, MD 20741
jk@cs.umd.edu

Aravind Srinivasan
Computer Science Department
University of Maryland, College Park
College Park, MD 20741
srin@cs.umd.edu

ABSTRACT

Social trust relationships between users in social networks speak to the similarity in opinions between the users, both in general and in important nuanced ways. They have been used in the past to make recommendations on the web. New trust metrics allow us to easily cluster users based on trust. In this paper, we investigate the use of trust clusters as a new way of improving recommendations. Previous work on the use of clusters has shown the technique to be relatively unsuccessful, but those clusters were based on similarity rather than trust. Our results show that when trust clusters are integrated into memory-based collaborative filtering algorithms, they lead to statistically significant improvements in accuracy. In this paper we discuss our methods, experiments, results, and potential future applications of the technique.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Keywords

recommender systems, trust, social networks

1. INTRODUCTION

Trust between users in a social network indicates similarity in their opinions [25, 24, 11]. Hundreds of millions of people are members of social networks online [10] and many of those networks contain trust data. With access to this information, trust has potential to improve the way recommendations are made.

Existing work has used trust to make recommendations by treating it as a weight in collaborative filtering algorithms [3, 9, 15]. This has been effective, but it is not the only way in which trust can be used. If users can be clustered into trusted groups, these may be useful for improving the quality of recommendations made with a variety of algorithms. While clustering can be computationally difficult with social networks, a new trust metric we have developed makes it easy to apply clustering techniques to build these groups. Since trust is related to similarity, we use correlation clustering to identify groups of trusted people.

This leads to the question: since we know that trust can be useful for making recommendations directly, and if we can effectively cluster users by trust, can those clusters be used to improve recommendation accuracy further? There has been some work on using clusters for improving recommendations already. These techniques cluster based on similarity, much as collaborative filtering techniques rely on user similarity. Unfortunately, research has not found an improvement in accuracy using these methods and, in many cases, the recommendation techniques using clusters actually lead to worse performance.

Previous experiments with trust have shown that it leads to improvements over similarity-based recommendation techniques in certain cases [8], and that user-assigned trust ratings capture sophisticated types of similarity [11]. Thus, it is possible that trust clusters may have benefits that are not found when similarity-based clusters are used. Our results show that incorporating trust clusters into collaborative filtering algorithms - including algorithms that use Pearson correlation coefficients and algorithms that use trust - does indeed lead to statistically significant improvemnets.

In this paper, we will present a discussion of our new trust metric and its applications to clustering, the integration of these clusters into recommendation algorithms, the experiment where we tested these algorithms and found the improvement in accuracy, and finally discuss a range of applications where this technique may be beneficial.

RecSys '09 New York, NY USA

2. RELATED WORK

User clustering has been applied to the task of collaborative filtering before, but our work is the first that uses trust as a basis for forming clusters. In this section, we describe work that has been done on using clustering for collaborative filtering and on using trust for collaborative filtering. Then, we introduce the dataset used in our computations and experiments.

2.1 Clustering and Collaborative Filtering

Breese et al. [5] used a Bayesian clustering model to cluster users based on their ratings. Their work showed mixed results; in some cases the clustering approach was competitive in terms of accuracy of the ratings and in others it performed poorly. Ungar and Foster [23] also used a Bayesian approach to cluster users based on their preferences. Their results also showed that clustering users was not a particularly successful approach. Graph theoretic methods for clustering users based on preferences were discussed in [19], however they do not evaluate the impact these clusters have on recommendation accuracy or quality. Finally, in [22] users were clustered using a scalable neighborhood algorithm and, once again, the clustering approach had a higher MAE than the standard collaborative filtering method.

2.2 Trust and Collaborative Filtering

Social networks, and trust in particular, have been used to generate recommendations for users. In these cases, trust is used directly to generate the recommendation. This work follows from the fact that people tend to develop connections with people who have similar preferences [1]. Trusting the opinion of another particularly speaks to this type of similarity. The applicability of this effect to recommender systems has been established in several papers. Ziegler and Lausen [25] that showed a correlation between trust and user similarity in an empirical study of a real online community. Using All Consuming [1], an online community where users rate books. The authors showed that users were significantly more similar to their trusted peers than to the population as a whole. This work was extended in [24] which augmented the analysis of the All Consuming community and added an analysis. The second result in [24] used the FilmTrust system[9] (described below) where users have stated how much they trust their friends in a social network and also rated movies. Within that community, results also showed a strong correlation between trust and similarity in movie ratings. Further work in [11] shows that trust captures similarity in more nuanced ways, such as similarity on items with extreme ratings and large differences.

Empirical results show that using trust from social networks can improve recommendations. O'Donovan and Smyth [20] performed an analysis of how trust impacts the accuracy of recommender systems. Using the MovieLens dataset [18], they create trust-values by estimating how accurately a person predicted the preferences of another. Those trust values were then used in connection with a traditional collaborative filtering algorithm [14], and an evaluation showed significant improvement in the accuracy of the recommendations. Massa and Bhattacharjee [16] also conducted a study on the applicability of trust in recommender systems. Their study relied on the user ratings of products and trust ratings of other users from epinions [2] as their dataset. Using a trust propagation algorithm, similar to that described in section 3, they showed that trust based recommendations could perform significantly better than those based on similarity alone.

In the FilmTrust recommender system mentioned above, trust is used in place of the Pearson correlation coefficient to generate predictive ratings. Results showed that when the user's rating of a movie is different than the average rating, it is likely that the recommended rating will more closely reflect the user's tastes. As the magnitude of this difference increases, the benefit offered by the trust-based recommendation also increases. Moleskiing [3], at http://moleskiing.it, is another real system built to utilize trust ratings in a recommender system. Using a similar approach, it recommends routes to users based on information supplied by trusted peers.

2.3 Dataset

For these experiments, we needed our dataset to have two components:

1. A social network with trust ratings between individuals so we could apply the trust inference algorithm and clustering methods discussed in section 3.

2. Ratings of items by the members of that social network.

We used the FilmTrust dataset [9] for these experiments because it had both these features - a trust network and a set of ratings on movies. Because trust assigned by one user to another is a value that is kept very private, there are no other publicly available datasets with this information. Thus, while FilmTrust provides a good basis for this initial analysis, further analysis with privately held trust networks will likely lead to additional insights.

At the time of analysis, the FilmTrust movie rating dataset has 29,551 ratings by 1,254 unique users over 1,946 movies. That is an average of 15.2 ratings per movie, though some movies have hundreds of ratings. Trust values are assigned on a 1 to 10 scale and are asymmetric; Alice may trust Bob at level n, but Bob may have no trust or a different value of trust in return for Alice. The entire FilmTrust social network has 712 nodes with 1,465 edges and an average of trust rating is 6.83. Many of these nodes are in small groups of two or three disconnected from the main component. For our algorithms, we selected the the giant component and removed nodes with a degree of 1. This left 348 nodes with 1,059 edges in the FilmTrust network. Since our recommendation technique required nodes to be in the social network, we used only the ratings from these 348 nodes. They had 8,457 ratings on 1,558 movies.

3. A PROBABILISTIC TRUST INFERENCE ALGORITHM

As part of our previous work [6], we developed a probabilistic trust inference algorithm that leads nicely to clustering applications. In this section we present an overview of that work and discuss the clustering techniques used.

[1] http://allconsuming.net/

[2] http://epinions.com

3.1 The Trust Inference Algorithm

Our work takes a trust network, which may be very sparse since most people will know only a small fraction of the network, and generates inferred trust values between all pairs in the network. We then use these trust values as the basis in a trust distance metric space, where the more trust between a pair, the closer they are in the space. One of the major benefits of our approach, and the benefit we exploit here, is the ability to group people into clusters within this metric space. In this section we present our probabilistic trust inference algorithm and describe the properties of its output which make it easy to cluster.

A very intuitive idea motivates this trust inference model. Consider the following scenario:

- Alice knows Bob and thinks he has a $p_{a,b}$ chance of being trustworthy.
- Bob knows Eve and thinks she has a $p_{b,e}$ chance of being trustworthy, and he tells this to Alice if he is trustworthy. If Bob is not trustworthy, he may lie about $p_{b,e}$ and give any value to Alice.
- Alice reasons that Eve is trustworthy if Bob is trustworthy and gives her the correct value $p_{b,e}$ and Eve is trustworthy with respect to Bob.
- This combination happens with probability $p_{a,b}p_{b,e}$ if Bob's trustworthiness and Eve's trustworthiness are independent.

Thus we infer that Alice's trust in Eve should be $p_{a,b}p_{b,e}$. More formally we view any path through the network as a Bayesian chain. Define X_{Bob}, X_{Eve} to be the respective random events that Bob and Eve are trustworthy from Alice's perspective. This is explained in more detail in Figure 1.

The same analysis can be used if trust is a proxy for similarity. Specifically Alice and Bob's mutual trust can be a measure of how similar their tastes in movies are. If trust is interpreted as probability of liking the same film, then Alice will agree with Eve about a movie if (but not necessarily only if) Alice and Bob agree on it and Bob and Eve agree as well.

Our model would not be interesting if it required simple probabilities which can be computed exactly. Fortunately we can quickly estimate trust between individuals in a more complicated network, one with exponentially many, highly correlated paths between pairs of nodes. In these examples, the Bayesian chain view still applies. If there exists a path from Alice to Eve in a random network constructed from trust values, then that path is a chain of people from Alice to Eve who each trust their successor, and Alice can trust Eve. Therefore Alice trusts Eve with the probability that there is a path from Alice to Eve in the random graph.

We define $t_{u,v}$ to be the direct trust between u and v, and $T_{u,v}$ to be our inferred trust value. The direct trust values may be arbitrary, however the inferred trust should obey the axioms in Table 1.

The idea that trust networks can be treated as random graphs underlies this algorithm. For every pair (u, v), we place an edge between them with some probability that depends on $t_{u,v}$. We then infer trust between two people from the probability that they are connected in the resulting graphs. Formally we choose a mapping f from trust value to probabilities. We then create a random graph G where each edge (u, v) exists with probability $f(t_{u,v})$. We then use this graph to generate inferred trust values $T_{u,v}$ such that $f(T_{u,v})$ equals the probability that there is a path from u to v in the random graph. We give a small illustrative example graph in Figure 2. This model is one of many that satisfies our trust axioms.

Axioms of inferred Trust

Local Pessimism	Since $t_{u,v}$ is a pessimistic estimate, indirect information can only increase trust, thus $T_{u,v} \geq t_{u,v}$.
Bottleneck	If all paths from u to v use (a, b), then $T_{u,v} \leq t_{a,b}$, and in general the lower $t_{a,b}$ is, the lower $T_{u,v}$ should be.
Identity	Individuals should completely trust themselves: $T_{u,u} = T_{\max}$.
Complete Trust	If there exists a path $(a_0, a_i, \ldots, a_n)$ such that for all i from 1 to n : $t_{a_{i-1},a_i} = T_{\max}$, then $T_{a_0,a_n} = T_{\max}$.
Monotonicity	For any u, v such that $T_{u,v} < T_{\max}$, augmenting a graph with a new trust path from u to v, or increasing a $t_{a,b}$ value along an existing trust path should increase $T_{u,v}$.
No Trust	For any u, v with no path from u to v, $T_{u,v} = 0$.

Table 1: These rules should apply to any pessimistic system which derives inferred trust from direct trust information.

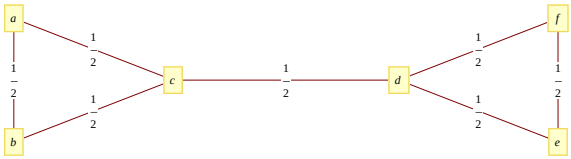

Figure 2: This is an example network with a critical edge. No one from the set $\{a, b, c\}$ can trust anyone in $\{d, e, f\}$ except through the mutual trust between c and d.

3.2 Trust Based Clustering

Given that $f(T_{u,v})$ is the probability of a path connecting u and v. The function $d(u, v) = \log \frac{1}{f(T_{u,v})}$ defines a metric space on the nodes because it satisfies four conditions:

- $d(u, v) \geq 0$
- $d(u, v) = d(v, u)$ (though this condition is not necessary for asymmetric metrics).
- $d(u, u) = 0$
- $d(u, v) + d(v, w) \geq d(u, w)$

Since we have a metric space on the nodes where the further apart two nodes are, the lower the probability of a path between them, we can make use of existing metric clustering algorithm to partition the nodes into groups. A clustering algorithm takes a set of points in a metric space and groups

$$\begin{aligned} Pr[X_{Eve}] &= Pr[X_{Eve}|X_{Bob}] \cdot Pr[X_{Bob}] + Pr[X_{Eve}|\overline{X_{Bob}}] \cdot Pr[\overline{X_{Bob}}] \\ &\geq Pr[X_{Eve}|X_{Bob}] \cdot Pr[X_{Bob}] = Pr[X_{Bob} \wedge X_{Eve}]. \end{aligned}$$

Figure 1: The second term drops out because Alice has no information about Eve if Bob is not trustworthy. Furthermore, if Eve and Bob are independent, this probability becomes $Pr[X_{Bob}]Pr[X_{Eve}]$.

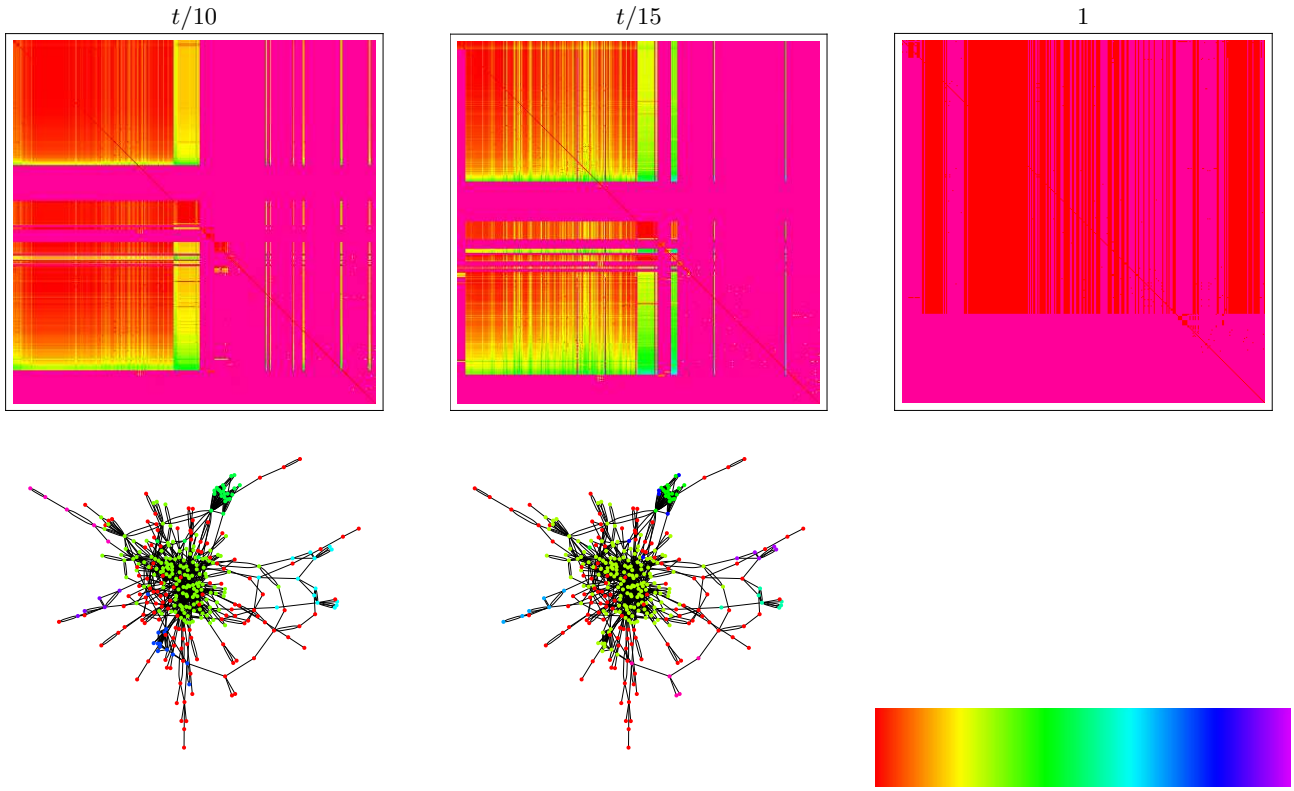

Figure 3: Here we show distance grids, partitionings for the FilmTrust dataset, and the color code for the grids. The i, j pixel in each grid encodes the distance between the i^{th} and j^{th} people accoring to our metric. The color code is given by the bottom rightmost image, with red being 0 distance and violet being distances of 10 and above. The rightmost grid is not probabilistic, but instead shows the transitive closure of the edge set.

them in a way that tries to optimize some criteria. Examples include, k-centers which finds a set of points S of k points which minimizes the distance from any point to its closest point in S, k-means which partitions the points into k sets in a way that minimizes the variance within each group, and correlation clustering which partitions the points in a way that minimizes the sum of distances within groups minus the sum of distances across groups. Each of these clustering algorithms have good approximation algorithms when applied to points in a symmetric metric space [12, 13, 4], and some even have good approximations in an asymmetric metric space [2].

While any of these clusterings can be applied, we focus on a variant of correlation clustering. Its goal - finding clusters maximizing agreement within and minimizing agreement between clusters - fits naturally with our application. Also, Unlike most clustering algorithms, no k representing the number of clusters is provided as input, since optimizing agreement is independent of the number of clusters. In our application, the trust value from one node to another can be treated as a measure of similarity, with high trust indicating agreement and low trust indicating disagreement. Using the complete graph output from the trust inference algorithm, we can perform a correlation clustering over the graph, grouping people together who have more trust for one another.

Figure 3 shows the results from applying our algorithm to the FilmTrust network. The distance grid shows one large mutually trusting group, as well as several progressively smaller mutually trusting groups. The largest of the groups is trusted by a large portion of the network. The second largest group is well trusted by this largest group. Beyond that, the plot where $f(t) = t/15$ brings out the most difference within the groups.

Finding an optimal correlation clustering is NP-hard [7], but there are efficient constant factor approximation algorithms. We use a variant where while there are nodes left to cluster, we choose one at random to be a "'center"' and

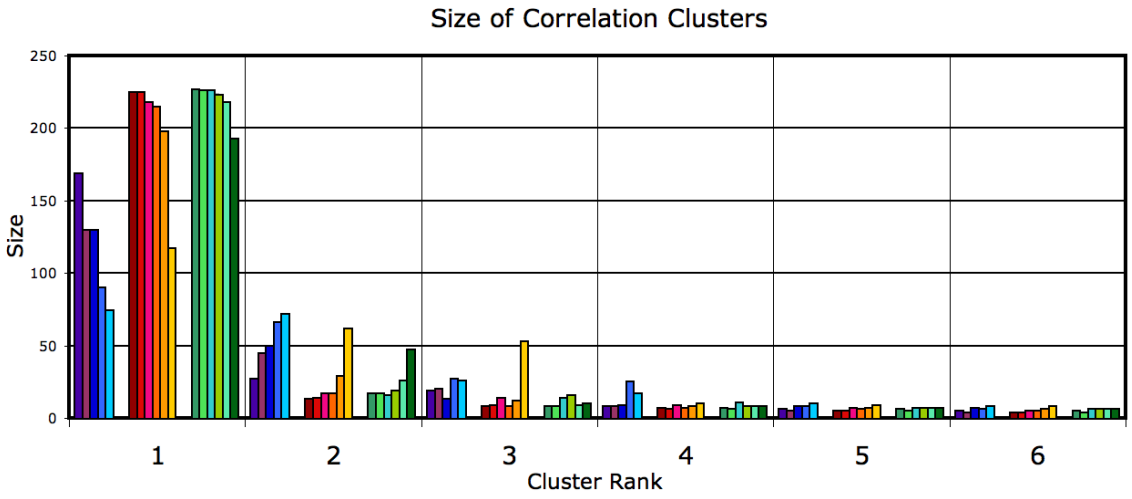

Figure 4: The size of the six largest clusters in for each iteration of the correlation clustering algorithm. The purple/blue bars indicate when the algorithm was run with a maximum radius of 1, the red/orange bars a maximum radius of 2, and the green bars a radius of 3. Six iterations are shown as the bars within each color group.

create a new cluster out of all nodes within a fixed radius of this center. Since this algorithm is randomized, the output can vary from one execution to another. Thus, in clustering our inferred trust network, we ran several iterations of the algorithm to produce a representative set of clusters to work with.

In order to obtain clusters for our recommender system, we ran the correlation clustering algorithm on the network output by running our trust inference algorithm over the FilmTrust data. We used a maximum radius of 1, 2, and 3, and ran six iterations of the algorithm for each. With this dataset, the algorithm generates one very large cluster, two or three medium sized clusters, and many small clusters. Figure 4 shows the size of the six largest clusters in each iteration for all three maximum radii.

4. EXPERIMENTS AND RESULTS

To test whether or not using trust-based clusters drawn from social networks could be used to improve the quality of recommendations, we ran several experiments. In this section, we discuss the datasets, experimental design, and results that show clusters can indeed improve the accuracy of recommendations.

4.1 Recommendation Algorithms

There are many methods for generating predictive recommendations. Our goal in this work was not to create the next best recommendation algorithm but rather to demonstrate that using clusters based on trust has the potential to improve the accuracy of recommendations.

We used several basic recommendation algorithms to test our hypothesis. The first is a basic ACF algorithm computes a weighted average of ratings using the Pearson Correaltion coefficient between the recommendee and the rater as a weight. To compute a recommendation we required pairs of nodes to have at least four movies in common so we could compute a meaningful correlation coefficient. We also tested a trust-based recommender algorithm. There are a number of approaches to using trust for recommendations [9, 21, 17, 16], and we used a simple variation on user-user automated collaborative filtering (ACF), replacing the correlation coefficient with the inferred trust value computed using the method described above. Thus, people the recommendee trusts more will receive more weight. This approach has been used before and shown to produce equivalent results to ACF overall, and improved results in certain cases [9].

Both algorithms were modified to give more weight to ratings from nodes in the same cluster as the recommendee. Considering *only* ratings by nodes in the same cluster would exclude so much information that recommendations would suffer. However, if we believe that the clustered nodes are more valuable, we can give them more weight than would be afforded using only the trust value. In these experiments, gave an additional 5% weight to nodes in the same cluster. All ratings for a movie were considered and weighted by the inferred trust from the recomendee to the rater. Ratings by nodes in the same cluster as the recommendee had their weight multiplied by 1.05. This approach was used with the Pearson corrleation-based ACF method and the trust-based recommendation.

4.2 Experimental Setup

To test our hypothesis that using the trust clusters improves the accuracy of recommendations, we ran the following process.

1. Select an iteration of the clustering algorithm to obtain clusters

2. For each user-movie pair, generate a predictive rating for the movie in two ways:

   (a) Using the standard trust-based recommendation algorithm

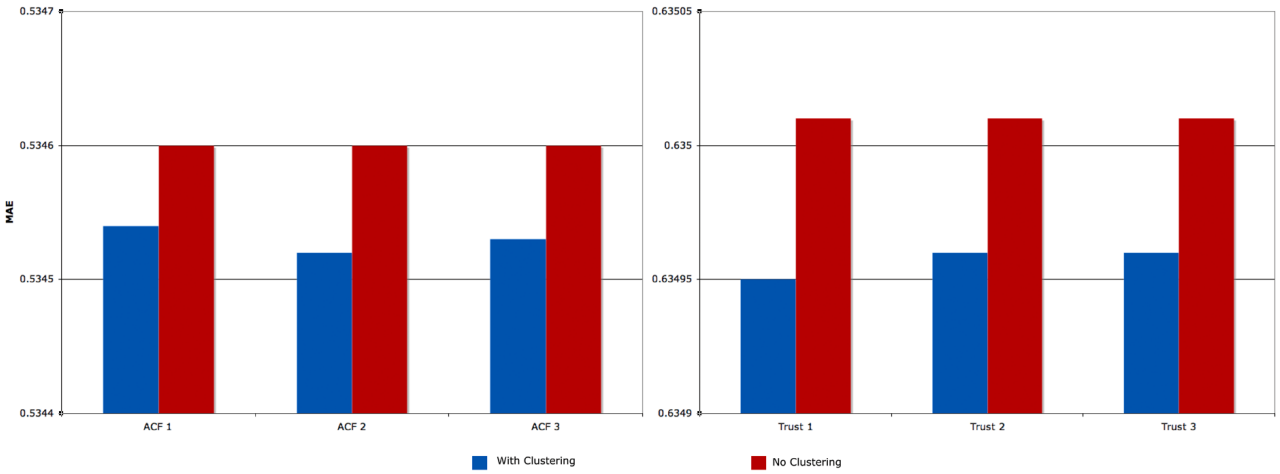

Figure 5: This illustrates the improvement in accuracy. The y-axis indicates Mean Absolute Error (MAE) so lower values are better. Blue bars indicate the results from the cluster-enhanced algorithms and red bars are the control. The numbers on the x-axis indicate the maximum radius of the clusters.

Table 2: Experimental Results of Cluster-Enhanced Recommendations. The cluster-enhanced method significantly outperforms the control in all cases.

Method	Cluster Radius	MAE (method)	MAE (control)	p-value	RMSE (method)	RMSE (control)	p-value
ACF	1	0.53454	0.53460	**0.0031**	0.70199	0.70204	**0.0273**
ACF	2	0.53452	0.53460	**0.0012**	0.70196	0.70204	**0.0022**
ACF	3	0.53453	0.53460	**0.0069**	0.70194	0.70204	**0.0004**
Trust	1	0.63495	0.63501	**0.0018**	0.82614	0.82620	**0.0076**
Trust	2	0.63496	0.63501	**0.0274**	0.82615	0.82620	**0.0493**
Trust	3	0.63496	0.63501	**0.0342**	0.82614	0.82620	**0.0356**

   (b) Using a modified trust-based recommendation algorithm that gives more weight to the nodes in the same cluster as the user as described above

3. Compare the MAE and RMSE for the two recommendation methods

This was repeated for each iteration and configuration of the clustering algorithm and for all of the cluster-enhanced algorithms described above. The clustering algorithms produced several large clusters and many very small clusters. We used only clusters with five or more nodes. To run the experiments using trust-based recommendations, it was necessary that we had an inferred trust value between the considered nodes in the network. Thus, nodes that were outside the giant component and thus had no inferred trust values were excluded. For consistency, we used the same set of nodes in all of our experiments.

4.3 Results

Our results showed that both cluster-enhanced recommendations (giving 5% extra weight to the ratings from people in the same cluster as the recommendee) offered a small but statistically significantly improvement in accuracy over the algorithms that did not consider the clusters. All significance results were computed using a Student's t-test for paired samples and were significant for $p < 0.05$.

Since the correlation clustering algorithm is randomized, we ran six iterations of the algorithm to obtain a representative sample. We ran the experiment on each iteration and then took the average rating for each user-movie pair over the six iterations to compare to the known value and judge the impact of the cluster-enhanced approach. This ensured that we could see the true impact of the approach and not be misled by an unusually good or bad clustering.

Table 4 shows the results of our different cluster-enhanced algorithms on the dataset. For both the mean absolute error (MAE) and root mean squared error (RMSE), the algorithm that took advantage of the clusters significantly outperformed the control, which ignored clusters.

4.3.1 *Coverage*

Clusters will not affect recommendations in all cases; it is possible that no one in the same cluster as the recommendee has rated the movie in question. In those cases, the result will be the same as the control method. However, analysis shows that at least one person who rated the movie is in the cluster approximately 70% of the time, and thus the clustering technique will have an impact.

5. DISCUSSION

These results show a small but statistically significant improvement in accuracy when correlation clusters generated from a trust network are incorporated into a recommender system. While the magnitude of the improvement is small,

these results are promising when we consider that similarity-based clustering approaches typically perform significantly worse that their non-clustered counterparts.

Furthermore, we believe that the results of this approach will become more practically significant on larger social networks. Our method requires a social network with trust values and item ratings created by the people in the network. We ran these experiments on a network with 348 nodes, which is small relative to most web-based social networks[3]. We used this network because it is the only one we had access to with the necessary data; since trust values must be kept private to be effective, other datasets are not made public by the large social networks that have them. This lack of *public* data does not limit the applicability of the technique; it can be applied within privately held networks where access to trust data is not a problem. The needed data exists internally on many large networks, such as Orkut. It also can be estimated from rating data on items a pair of users have in common [11].

With the larger social networks, we will see more large clusters. Since the experimental network has one large trusted cluster, and several smaller ones, our results show a significant improvement essentially from giving less weight to nodes outside the cluster. We expect to see this effect magnified when there are more large clusters.

The fact that considering trust-clusters can improve recommendations also suggests that is has potential to help with other applications. By relying on connections in the social network, it is possible to eliminate many types of attacks or gaming in rating systems that rely on creating multiple accounts. While these accounts could all connect to one another with high trust, they would only be clustered wiht "good" users if some of these good users assigned them high trust ratings as well. However, previous work has shown that it is possible to eliminate these confused nodes from consideration [15]. These approaches together have the potential to very effectively eliminate forged ratings and reviews at the same time as they highlight those most relevant to the user.

5.1 Conclusions

Trust is strongly correlated with how similar two users are in their preferences. It reflects similarity in nuanced ways that has been shown to be useful for making recommendations. In this paper, we looked at taking trust a step further. We clustered users based on the trust between them using correlation clustering and then modified a collaborative filtering algorithm to use these clusters.

To test our approach we used a traditional Pearson correlation collaborative filtering algorithm and a recommendation algorithm that used trust for generating recommendations independently of the clusters. In both, we modified the algorithms to give extra weight to ratings from nodes in the same cluster as the user for whom the rating was being generated. We compared the accuracy of these recommendations to those made by the unmodified version of the algorithm. In both cases, our results show a small but statistically significant improvement in the accuracy of recommendations when clusters are used.

This improvement is particularly interesting since previous work on clustering, which was based on user similarity, failed to outperform non-clustered methods and often performed significantly worse. It suggests that trust captures more sophisticated information about the similarity between two people and that it is particularly useful for highlighting more relevant information in recommendation environments. We believe this effect will be magnified in bigger networks and that it has applications to limiting gaming and other attacks in online rating systems.

[3]Among the 250 social networks listed at http://trust.mindswap.org/ the mean size is over 4,600,000 and the median is 22,000.

6. ACKNOWLEDGMENTS

Supported in part by NSF ITR Award CNS-0426683 and NSF Award CNS-0626636, and DARPA.

7. REFERENCES

[1] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2000.

[2] Aaron Archer. Two $O(log * k)$-approximation algorithms for the asymmetric k-center problem. In *Proceedings of the 8th Conference on Integer Programming and Combinatorial Optimization*, pages 1–14. Springer-Verlag, 2001.

[3] Paolo Avesani, Paolo Massa, and Roberto Tiella. Moleskiing.it: a trust-aware recommender system for ski mountaineering. *International Journal for Infonomics*, 2005.

[4] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. In *Machine Learning*, pages 238–247, 2002.

[5] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. pages 43–52, 1998.

[6] Thomas DuBois, Jennifer Golbeck, and Aravind Srinivasan. Rigorous probabilistic trust-inference with applications to clustering. In *IEEE / WIC / ACM Conference on Web Intelligence*, 2009.

[7] M.R. Garey, D.S. Johnson, et al. *Computers and Intractability: A Guide to the Theory of NP-completeness*. wh freeman San Francisco, 1979.

[8] Jennifer Golbeck. *Computing and Applying Trust in Web-based Social Networks*. PhD thesis, University of Maryland, College Park, MD, April 2005.

[9] Jennifer Golbeck. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the Fourth International Conference on Trust Management*, 2006.

[10] Jennifer Golbeck. The dynamics of web-based social networks: Membership, relationships, and change. *First Monday*, 12(11), 2007.

[11] Jennifer Golbeck. Trust and nuanced profile similarity in online social networks. *ACM Transactions on the Web*, in press.

[12] Dorit S. Hochbaum and David B. Shmoys. Best possible heuristic for the k-center problem. *Mathematics of Operations Research*, (2):180–184, May 1985.

[13] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *SCG '02:*

Proceedings of the eighteenth annual symposium on Computational geometry, pages 10–18, New York, NY, USA, 2002. ACM.

[14] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. Grouplens: applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, 1997.

[15] Raph Levien and Alex Aiken. Attack-resistant trust metrics for public key certification. In *7th USENIX Security Symposium*, pages 229–242, 1998.

[16] P. Massa and B. Bhattacharjee. Using trust in recommender systems: an experimental analysis. In *Proc. of 2nd Int. Conference on Trust Management, 2004.*, 2004.

[17] R. Matthew, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the Second International Semantic Web Conference.*, 2003.

[18] Bradley N. Miller, Istvan Albert, Shyong K. Lam, Joseph A. Konstan, and John Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 263–266, New York, NY, USA, 2003. ACM.

[19] B.J. Mirza, B.J. Keller, and N. Ramakrishnan. Studying recommendation algorithms by graph analysis. *Journal of Intelligent Information Systems*, 20(2):131–160, 2003.

[20] John O'Donovan and Barry Smyth. Trust in recommender systems. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174, New York, NY, USA, 2005. ACM.

[21] John O'Donovan and Barry Smyth. Trust in recommender systems. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174, New York, NY, USA, 2005. ACM.

[22] B.M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the Fifth International Conference on Computer and Information Technology*, pages 158–167, 2002.

[23] L.H. Ungar and D.P. Foster. Clustering methods for collaborative filtering. In *AAAI Workshop on Recommendation Systems*, pages 112–125, 1998.

[24] Cai-Nicolas Ziegler and Jennifer Golbeck. Investigating Correlations of Trust and Interest Similarity. *Decision Support Services*, 2006.

[25] Cai-Nicolas Ziegler and Georg Lausen. Analyzing correlation between trust and user similarity in online communities. In *Proceedings of the Second International Conference on Trust Management*, 2004.