Trust based Recommender Systems

Divya Gadde, Yamini Gotimukul, Sindhu Maiyya, Vatsal Shah

December 10, 2014

Contents

1 Introduction 4

2 Data and Methodology 5

3 Traditional Recommender Systems 6

4 Trust Based Recommender Systems 8

4.0.1 Properties of Trust 9

4.1 Pure Trust 10

4.2 Random TrustWalker 12

4.3 Augmented Trust 14

5 Experimental Results 15

5.1 Evaluation metrics 17

5.2 Results 18

6 Evaluating performance using Top N Recommendations 22

6.1 Experiments 23

6.2 Evaluation Metric 23

6.3 Results 24

7 Challenges 25

7.1 Extended Epinions Dataset 25

8 Conclusion 26

Symbols used	
u, v, w	Users
i, j	Items
$r_{u,i}$	Rating corresponding to user u and item i
$r_{u,i}^{(n)}$	Rating corresponding to user u and item i in the n^{th} random walk
$\hat{r}_{u,i}$	Predicted rating for user u and item i
$\bar{r}_u$	Average rating given by user u
$\bar{r}_i$	Average rating received by item i
τ_u	Trust network with user u as its source node
$t_{u,v}$	Trust between user u and user v
$dist(u, v)$	Distance between user u and user v in the trust network
$\phi(v, j, k)$	Probability of stopping at node v and item j at a depth of k
$sim(i, j)$	Pearson Correlation Co-efficients between items i and j
$\mathcal{T}_u$	Set of users in the trust network of u
$P(u, v)$	Probability of going from user u to user v in the trust network
$\mathcal{T}_{u,sim}$	Set of users in the trust network of u with similarity greater than sim
$UC_{i,j}$	Set of users who have rated both items i and j
RI_u	Set of items rated by user u
$\|S\|$	Cardinality of the set S

Abstract

When consumers are faced with numerous choices, recommender systems provide a way to tackle information overload problem meaningfully. Traditional recommender systems use collaborative filtering approach, which make use of the similarity between items or users to provide recommendations to the user or predict a rating for a given user-item pair. Trust based recommender systems use new approaches that use the social network of users to provide recommendations. User item ratings are sparse, so collaborative filtering methods can not make recommendations for 'cold start users' that have rated very small number of items. Their coverage is usually poor. Trust based systems use the trust network of cold start users to make recommendations.

We evaluated the performance of user based and item based collaborative filtering methods as well as three different trust based recommender systems - Pure Trust, Random TrustWalker, and Augmented Trust - over Epinions dataset. We use $RMSE$, $Coverage$ and $F\ Measure$ to compare the performance of these algorithms. Both Random Trustwalker and Augmented Trust methods make use of trust network as well as collaborative filtering to make recommendations, thereby improving the coverage tremendously for a marginal tradeoff in accuracy. For cold start users, we saw that the coverage almost doubles for trust based methods when compared to traditional methods. Finally, we also experimented with an algorithm that extends augmented trust to give top-N recommendations.

1 Introduction

With the explosion of internet and digitalization, the amount of information available to people has grown beyond perception. For a consumer who wants to choose a product, the choices and the information on each of the choices is tremendous. To put the explosion of choices into perspective, the coffee shop chain Starbucks boasted in 2003 that it offered each customer 19,000 beverage possibilities at every store [1]. A search for smartphone returns 1,664,253 results in Amazon products or a search for best movies to watch in Google videos returns about 219,000,000 results.

Sifting through the gamut of information to make a choice is a labor intensive process. Research on "The Paradox of Choice" and "Decision Fatigue" has shown that having more choices does not correspond to increased satisfaction among people [2]. Another fundamental limitation that consumers come across is that there simply isn't enough time to course through all the choices for every decision to be made. At this juncture, recommender systems become very important as a tool to reduce these choices meaningfully. Recommender systems are a subclass of information filtering systems. These systems shortlist the candidate items to be chosen based on user preferences and are used in recommending products, social links (for example twitter users to follow), and digital items. There are three major groups of recommender systems - collaborative filtering based, content based, and trust aware recommender systems.

Collaborative filtering based recommender systems use users' past behavior or ratings to predict newer ratings. Content based systems use the content information or features of the items to recommend items to the user. For example recommending movies or music based on users' genre preferences, recommending products based on categories, etc. For a user who does not have too many ratings or preferences recorded (cold start user), it is very hard to give recommendations.

Trust aware recommender systems use social networks and trust to recommend products to users. Similar users can be found by looking at the trust network. This augments the information available for cold start users and so these recommender systems are expected to give much better coverage for cold start users.

2 Data and Methodology

To demonstrate trust aware recommender systems, we chose Epinions data set, which is one of the few publicly available datasets that contains user trust information. Epinions is a website where registered users provide ratings for specific products from a large collection of products. Users can add other users to their 'Web of Trust'. The data, originally crawled from Epinions.com by Paolo Massa in December 2003 [3]. This dataset contains 49,290 unique users who have rated a total of 139,738 items. There are 664,824 ratings in total. Among the users, there are 487,181 trust statements that make up the trust network. There are roughly 13 ratings per user and 5 ratings per item, making the rating matrix sparse. As per our definition, a user that has issued less than 5 ratings is a cold start user. Around 24,000 or roughly half of the users in our dataset have rated less than or equal to 5 items.

Cold start users do not have sufficient ratings for collaborative filtering recommender systems to predict a rating for them. Trust network provides additional information about the preferences of cold start users, thereby making the trust based recommender systems more robust. Trust network is dynamic and usually has a complex structure owing to its irregular growth[4]. Recommender systems predict a rating for an item i from the set of items I for a user u from the set of users U. The set of ratings for $(user, item)$ pairs is denoted as $U \times I \rightarrow R$, $r_{u,i}$ is

the rating for item i given by user u. The rating can be any real number, but the ratings in our dataset are integers in the range $\{1, \ldots 5\}$. We use different algorithms on our dataset to demonstrate the effectiveness of our proposed algorithm.

3 Traditional Recommender Systems

Many recommender systems use collaborative filtering technique for predicting ratings. Collaborative filtering (CF) either makes recommendation based on the past preferences or predicts the missing preferences for a given user. CF algorithms are based on the following assumptions: (1) Many users give ratings, (2) the algorithm will predict the rating by matching similar preferences, (3) User who have given rating for similar items will rate the items similarly in future and, (4) Items with similar ratings in the past will probably have similar ratings in the future. In CF the user expresses the preferences by rating items, these ratings are near approximation of their tastes for a particular domain. These ratings are matched with the ratings given by other users in the respective domain to find users with similar preferences [5].

The rating for an item for a given user can be predicted based on one of the following two similarities. One, the similarity measured for users or the similarity measured for items. The rating predicted by calculating the weighted average of the ratings given by similar users for the target item is called *user-based Collaborative Filtering*. Similarly, the prediction is a weighted average of ratings for similar item given by the target user in *item-based Collaborative Filtering*.

Similarity Computation: The most important step in the *Collaborative Filtering* algorithm is to compute the similarity between items or users and then to select the most similar items or users. Similarity is computed by taking two

items i and j (or users u and v) and then find the set of common users who have rated i and j (or items that have been rated by both u and v). Finally, compute the similarity index using the *Pearson* correlation measure. The formula used for calculating the similarity for item-based similarity is given by the following expression.

$$Corr_{item}(i,j) = \frac{\sum_{u \in \mathcal{U}_{i,j}}[(r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)]}{\sqrt{\sum_{u \in \mathcal{U}_{i,j}}(r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_{u \in \mathcal{U}_{i,j}}(r_{u,j} - \bar{r}_j)^2}} \tag{1.1}$$

In the current implementation of the item based CF algorithm, we are modifying the correlation coefficient to reflect the cardinality of the set of users who have rated both the items i and j. Thus the similarity is denoted by:

$$Sim(i,j) = Corr_{item}(i,j) * \frac{1}{1 + \exp\frac{(-|\mathcal{U}_{i,j}|}{2})} \tag{1.2}$$

$Sim(i,j)$ increases as the number of common users who have rated items i and j increases as the sigmoid function goes to 1.

The algorithm for item based CF and user based CF are very similar. Therefore, only the algorithm 1 describes the user based Collaborative Filtering Algorithm:

The limitations of CF techniques are as follows:

- Collaborative filtering techniques, particularly user based approach is not preferred due to the instability in the relationship between users.
- When this technique is applied to a large data set, even the smallest change to the data requires the calculation of similarity for the entire dataset.
- Many commercial recommender systems evaluate very large datasets. The main issue with large datasets is the data sparseness which has a negative effect on the performance of the recommendation system.

Algorithm 1 userbasedCF(u,i)

$\bar{r}_u$ = average rating for the user u
$user.rated$ = users that have rated the item i
$Corr_{u,j}$ = Pearson correlation coefficients for u and all the j users in $user.rated$

$usr.avg.local$ = average of the ratings given by j users in $user.rated$
$p.coef$ = A vector of length k that has $Corr_{u,j}$ that are positive
for $j \in length(user_rated)$ **do**
    $relative.rating_j$ = ratings given for item i by user j ($r_{j,i}$) - $usr.avg.local_j$
    $ratingb_j = Corr_{u,j} \times relative.rating_j$
end for
$$avg.ratingb = \frac{\sum_1^{length(p.coef)} ratingb_k \times p.coef_k}{\sum p.coef_k}$$
if $length\ of\ p.coed \leq 5$ **then**
    return -1
else
    return $\bar{r}_u - avg.ratingb$
end if

- The collaborative techniques cannot be useful for predicting a rating for a new user or item that has entered the system. The main issue here is to find a similar item without having much information about the item or user, which is referred to as cold start problem [5].

- CF techniques are very vulnerable to attacks that insert fraudulent users into databases

All these limitations compel us to utilize the trust networks in order to overcomes the limitations of CF algorithms.

4 Trust Based Recommender Systems

Traditional recommender systems used collaborative filtering techniques to identify the similarity between users' choices. These similarity coefficients were used to identify similar items (or similar users) using their ratings. With the advent of

Facebook, Twitter and Google Plus, it became possible to find users with similar choices via their networks. Typically, when a user 'friends' another user on Facebook, it is because they share some common interest. Similarly, a user 'follows' another user on Twitter if they follow the same soccer team or they prefer the same kind of music.

Thus, the information available from social networks can prove to be extremely vital to improve performance and outreach of recommender systems. 'Cold start users' are users that have rated very few items. It becomes impossible to find a user with a similar choice. For example, a particular user u has only rated Shawshank Redemption on Internet Movie Database (IMDb). This movie has been rated by a million other users on IMDb but it is not possible to group u with any of the other users based on just one rating.

Can we improve the prediction if some other information was available about the user u? Let us say that some explicit information was made available by u regarding another user v. For example, u says that there is another user v whose choice in movies is quite similar to his. In this case, it is possible to look at v and utilize his ratings for all the movies that v has rated on IMDb to predict ratings for user u. This is exactly the motivation for using trust statements to improve both our recommendations and our coverage in the algorithms described in this section.

4.0.1 Properties of Trust

In the next few algorithms, we assume certain basic properties of trust [6]:

- Transitivity: Transitivity allows us to establish indirect links between users. This property implies that if u trusts v and v trusts w, then u trusts w i.e. $t_{u,w} > 0$. The transitivity property allows us to infer indirect trust between users. Trust can be inferred in a lot of ways as described in [7]. Some of

them are discussed here:

- Inferred trust between users is also binary. If $t_{u,w} = 1$ and $t_{w,v} = 1$, then $t_{u,v} = 1$. This can also be represented as follows:

$$t_{u,v} = \max_{\{w \in \mathcal{T}_u\}} \{t_{u,w} \times t_{w,v}\} \tag{1.3}$$

- Inferred trust between users at a larger distance should not be as strong as the inferred trust between users at a smaller distance. The original binary trust network to a discrete trust network where the inferred trust between two nodes u and v is inversely proportional to the $dist(u, v)$, distance between these nodes (i.e. users) in question.

$$t_{u,v} = \frac{\max_{\{w \in \mathcal{T}_u\}} \{t_{u,w} \times t_{w,v}\}}{dist(u, v)} \tag{1.4}$$

- Asymmetry: Trust is a personalized acknowledgement of belief in someone. Thus, if user u trusts user v, then it does not imply that user v trusts user$u$, i.e. the trust network is an undirected graph. Thus, the original binary trust network is converted to a discrete trust network where the inferred trust between two users u and v is inversely proportional to the $dist(u, v)$.

- Context Dependence: If a user u has similar taste to user v in movies does not mean that he will have a similar taste when buying cars or books. Thus context becomes very important when we are considering trust.

4.1 Pure Trust

In pure trust based methods or trust-aware filtering [8], the algorithm runs multiple random walks which traverse the trust network and pick a user v who has rated item i and returns that rating. As the $dist(u, v)$ increases, the probability of

picking user v has to decrease. This is approximated by a random walk over the trust network starting from node u. Each iteration of the random walk is stopped under two circumstances:

i) If we come across a user v in the trust network of u that has rated item i,

ii) If we are at a distance greater than 6 from node u.

The 'small world experiment' described in [9] states that it is possible to reach any user v starting from any user u in six steps on an average based on the theory of ' six degrees of separation'. This is the motivation for using 6 as the maximum-depth to traverse in our random walk algorithms.

A detailed flowchart describing the algorithm is given in Figure 1.1:

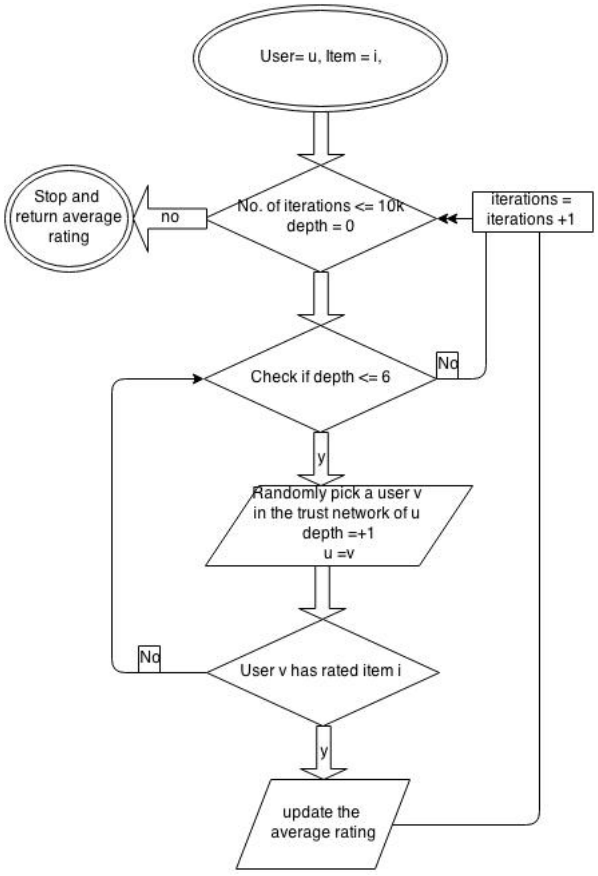

Figure 1.1: Flowchart describing the pure trust algorithm

An exhaustive breadth first search on 6 levels of trust network will be an expensive operations. For the Epinions dataset, each user trusts 10 users on an average and so going upto a depth of 6 corresponds to iterating over 10^6 users. Instead, the trust network can be treated as a Markov Chain since the probability of going to a user v starting from user u is independent of the past. The aggregated rating for user u and item i, $\sum_v P(u,v)r_{v,i}$ using the Ergodic Theorem for Markov Chains can also be equivalently denoted by:

$$\sum_v P(u,v)r_{v,i} = \frac{\sum_{n=1}^{N} r_{v,i}^{(n)}}{N} \tag{1.5}$$

The final predicted rating for user u and item i,$\hat{r}_{u,i}$, is given as follows:

$$\hat{r}_{u,i} = \frac{\sum_{n=1}^{N} r_{v,i}^{(n)}}{N} = \frac{\sum_v t_{u,v} r_{v,i}}{\sum_v t_{u,v}} \tag{1.6}$$

In the above equation, $t(u,v)$ is calculated using 1.3 and $r_{v,i}^{(n)}$ is the result of the n^{th} random walk

The biggest concern for the Pure Trust technique is to decide how much deeper should we go in the trust network. The deeper we go, lesser is the probability of coming across trustworthy users. Using these less trusted users to make predictions results in a loss of precision. Instead if we decide to stop early (i.e. maximum depth = 2) then we are losing coverage. Thus, there exists a tradeoff between coverage and precision. Our next algorithm improves the coverage without sacrificing the precision.

4.2 Random TrustWalker

The Random TrustWalker algorithm [10] considers both users in the trust network and items similar to the target item to make better predictions. This algorithm

improves the precision of recommendations by giving a higher preference to raters at a shorter distance while also enhancing the coverage at the same time.

The probability of picking an item j rated by user v at a depth of k from user u is denoted by $\phi(v, j, k)$ which is expressed as follows:

$$\phi(v, j, k) = \max_{\{j \in RI_u\}} corr(i, j) * \frac{1}{1 + \exp\left(\frac{-|UC_{i,j}|}{2}\right)} * \frac{1}{1 + \exp\left(\frac{-k}{2}\right)} \tag{1.7}$$

The final prediction is just the aggregation of these ratings weighted by $\phi(v, j, k)$. As mentioned in the previous section, this aggregation can also be approximated by running random walks over the trust network.

Each random walk starts from source user u. At depth k of a random walk, we are at a certain node v. If user v has the rated the target item i, then we stop our random walk and return $r_{v,i}$ as the result of the random walk. If v does not have a rating on i, then we have two options:

- With probability $\phi(v, j, k)$, we don't continue the random walk. The algorithm stays at v and randomly selects one of the items j, similar to the item i in consideration, and returns $r_{v,j}$ as the result of random walk.

- With probability $1 - \phi(v, j, k)$, we continue the random walk and pick a user w that is a direct neighbor of v.

Algorithm 2 describes the Random TrustWalker algorithm.

The final predicted rating for user u and item i,$\hat{r}_{u,i}$, is given as follows:

$$\hat{r}_{u,i} = \frac{\sum_{n=1}^{N} r_{v,i}^{(n)}}{N} \tag{1.8}$$

Algorithm 2 randomWalk(user, item)

$nlevels = 6$
$u = user$
for $level \in 1 : 6$ **do**
  $RI_u = item\ rates\ by\ user\ u$
  $\phi = \max_{\{j \in RI_u\}} sim(item, j) * (\frac{1}{1+\exp(\frac{-level}{2})})$
  if $rand(0, 1) < \phi$ **then**
    $pick\ any\ j \in RI_u\ with\ p_j = sim(item, j)$
    return $r_{u,j}$
  else
    $pick\ \hat{u}$ ($any\ user\ trusted\ by\ u\ with\ equal\ probability$)
    $u = \hat{u}$
    if $u\ has\ rated\ target\ item$ **then**
      return $r_{u,item}$
    end if
  end if
end for
return -1

4.3 Augmented Trust

One important property that was not considered in the previous algorithms was context dependence. Since the Epinions dataset consists of a wide range of items, trust between users may not always correspond to a similar opinion for all items[11]. The trust ratings are merely an indicator of the belief a user has in ratings of another user instead of similarity in their choices. This is evident from the histogram in Figure 4.3 for a sample of $(user, user)$ pairs in the trust network :

As is evident from the histogram, nearly 40% of the users in the trust network are not similar to the user in consideration. Thus, the ratings of these dissimilar items act as noise and adversely affect the predictions. The augmented trust algorithm tries to assimilate best parts of both the User Based Collaborative filtering technique and the Pure Trust techniques. The algorithm (similar to one described in [12]) is as follows:

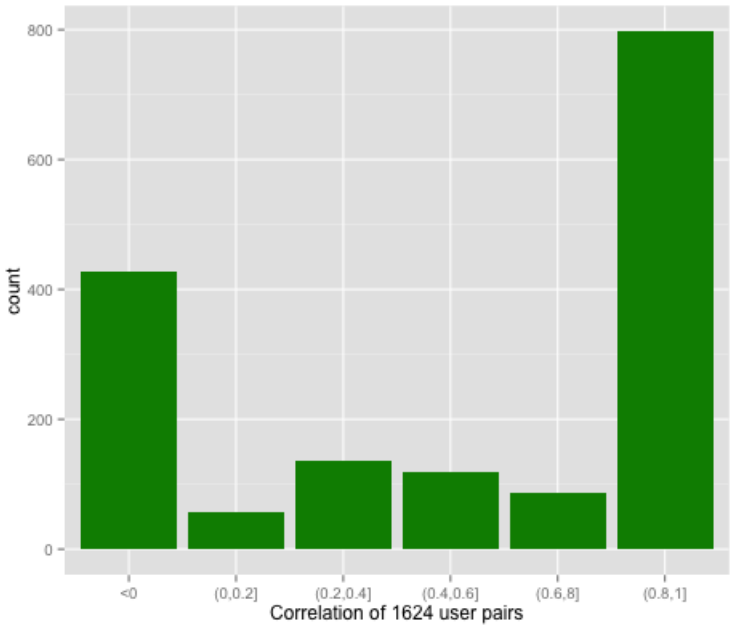

Figure 1.2: Histogram of similarity coefficients for trusted users

The augmented trust algorithm to predict a rating for user u and target item i is given as Algo:

The formula for predicted rating, $\hat{r}_{u,i}$, correspponding to user u and item i is given as follows:

$$r_{u,i} = \bar{r}_u + \frac{\sum_{v \in \mathcal{T}_{u,0.4}} t_{u,v}(r_{v,i} - \bar{r}_v)}{\sum_{v:v \in \mathcal{T}_{0.4}} t_{u,v}} \tag{1.9}$$

In the above equation, $t(u, v)$ was calculated using Equation 1.4.

5 Experimental Results

We have used the Epinions dataset published by the authors of [3]. Each of the 5 algorithm was coded using Python. We ran each of the for three different datasets:

- Dataset 1: Epinions data set: This dataset has $49k$ users and $139k$ items with $664k$ ratings and $487k$ trust statements. Since, this is a sparse dataset,

Algorithm 3 AugmentedTrust(user, item)

```
1: ratings = [ ]
2: for u ∈ trustNetwork[user] do
3:    sim = pearson(user, u)
4:    if sim > 0.4 then
5:       if u has rated item then
6:          ratings.add(r_{u,item}, sim)
7:       end if
8:    end if
9:    for v ∈ trustNetwork[user] do
10:      sim = pearson(user, v)
11:      if sim > 0.4 then
12:         if v has rated item then
13:            ratings.add(r_{v,item}, sim)
14:         end if
15:      end if
16:   end for
17: end for
18: return (Σ_i ratings[i]*sim[i]) / (Σ_i sim[i])
```

adjacency lists were use to optimize the space required for storing data. The dataset contains $24k$ cold start users who have rated less than 5 items. The Random TrustWalker algorithm acted as a bottleneck and so a test set of size 5000 was considered.

- Dataset 2: Epinions Cold Start dataset: This is very similar to Dataset 1 with one major difference. All the users in the test set are cold start users.

- Dataset 3: Trimmed Epinions dataset: All the items that were rated by less than 10 users were removed. Similarly, all the users who had rated less than 10 items were removed. Thus, the Trimmed dataset had $10k$ users, $10k$ items and $300k$ ratings. The use of this dataset showed that the algorithm gives tremendous performance gains even for marginally dense datasets. The trust dataset was not reduced mainly because of two reasons:

- If the trimmed user trusts a lot of users (i.e. densely connected) then removing this user may adversely affect the performance of our algorithm
- Even if the trimmed user is sparsely connected, it may be the only link to a densely connected user (since we are traversing upto a depth of 6 this becomes important) and so removing trimming this user may disrupt the trust network.

5.1 Evaluation metrics

Three different evaluation metrics were used to compare the performance of various algorithms:

- **RMSE** is the root mean square of the error terms for all the $(user, item)$ pairs in the test set. The error term is the difference between actual and predicted ratings.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(r_{u,i} - \hat{r}_{u,i})^2}{N}} \tag{1.10}$$

  Precision defines a confidence score for the predictions based on $RMSE$. Lower the value of the $RMSE$, more precise is the prediction and consequently the algorithm.

  The maximum possible error for our dataset where the ratings can take any integer values between 1 and 5is given by $\frac{Range^2}{4} = \frac{(5-1)^2}{4} = 4$

  Thus, for our dataset, precision can be defined as:

$$Precision = 1 - \frac{RMSE}{Maximum\ possible\ error} = 1 - \frac{RMSE}{4} \tag{1.11}$$

- **Coverage** is defined as the percentage of pairs of $(user, item)$ in the test set

for which we can predict a rating. Higher the value of *Coverage*, better is the outreach of an algorithm.

- **F Measure** is defined as the harmonic mean of *Precision* and *Coverage* and can be denoted as:

$$F\ Measure = \frac{1}{\frac{1}{2}\left(\frac{1}{Precision} + \frac{1}{Coverage}\right)} = \frac{2 \times Precision \times Coverage}{Precision + Coverage} \quad (1.12)$$

F Measure is a technique which enables us to attach equal importance to both *Precision* and *Coverage*. *F Measure* thus acts as a balancing act and best captures the tradeoff between *Precision* and *Coverage*. Thus, an algorithm with a very high precision, but poor coverage will have a low *F Measure*. Similarly, an algorithm with very high coverage but low precision will also have low *F Measure* but an algorithm with reasonably good values of precision and coverage will have a high *F Measure*.

5.2 Results

In order to test the performance of our algorithms, we decided to test the algorithms on trimmed dataset. Here are the results:

Algoirthm	**RMSE**	**Coverage**(%)	**F Measure**
Item Based CF	1.01	73.1	0.74
User based CF	1.03	54.9	0.63
Pure Trust	1.13	75.4	0.74
Random TrustWalker	1.22	100	0.82
Augmented Trust	0.48	99	0.92

Table 1.1: Performance of recommender systems on the Dataset 3

For collaborative filtering techniques, we expect better *RMSE* and poor *Coverage* compared to trust based methods. Our experiments on the reduced dataset (Ta-

ble 1.1) confirm the same. The coverage for CF techniques is higher than expected since the dataset is dense. The $RMSE$ for augmented trust is very low because we are considering ratings from users with a $similarity > 0.4$ in addition to the fact that the data is dense.

The plots of evaluation metrics corresponding to Table 1.1 is given in 1.3

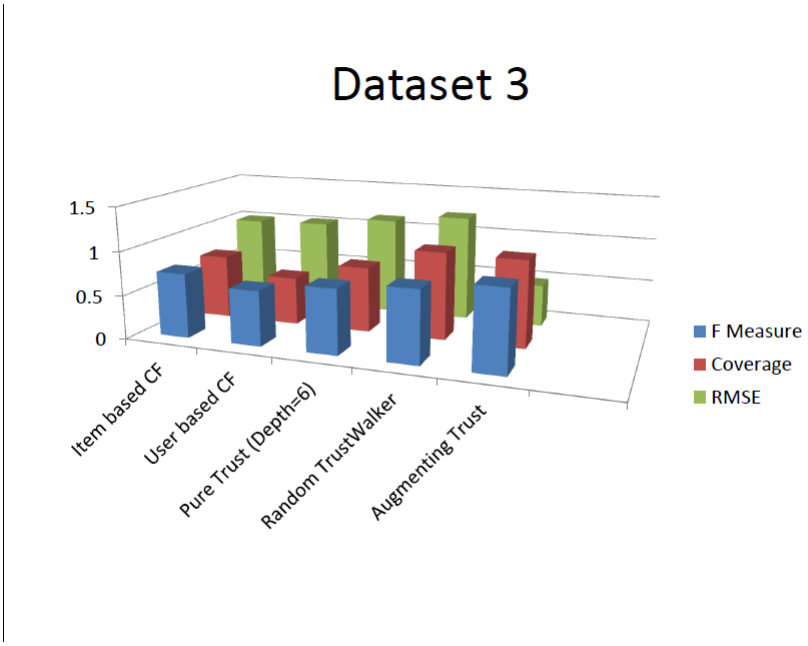

Figure 1.3: Plot of evaluation metrics on the Dataset 3

Algorithm	**RMSE**	**Coverage**(%)	**F Measure**
Item Based CF	1.06	31.9	0.44
User based CF	1.10	24.4	0.36
Pure Trust (depth =6)	1.25	61.1	0.64
Random TrustWalker	1.29	98.9	0.80
Augmented Trust	1.23	45.6	0.55

Table 1.2: Performance of recommender systems on Dataset 2

For cold start users (Table 1.1), CF techniques give very poor *Coverage*. Pure Trust algorithm almost doubles the *Coverage* but at the cost of higher $RMSE$.

Random TrustWalker gives the highest coverage as well as relatively low RMSE since it combines trust-based as well as CF techniques. Augmented trust algorithm does not perform as well as it did on the previous dataset for two main reasons: 1) Since the entire test set consists of cold start users, it may be difficult to find users in the trust network with similarity greater than 0.4. 2) The algorithm only traverses upto a maximum depth of 2. It should be noted that this performs better than the Pure Trust algorithm with depth 2 ($RMSE = 1.24, Coverage = 35.3\%$) validating augmented trust with user similarity.

The plots of evaluation metrics corresponding to Table 1.2 is given in 1.4

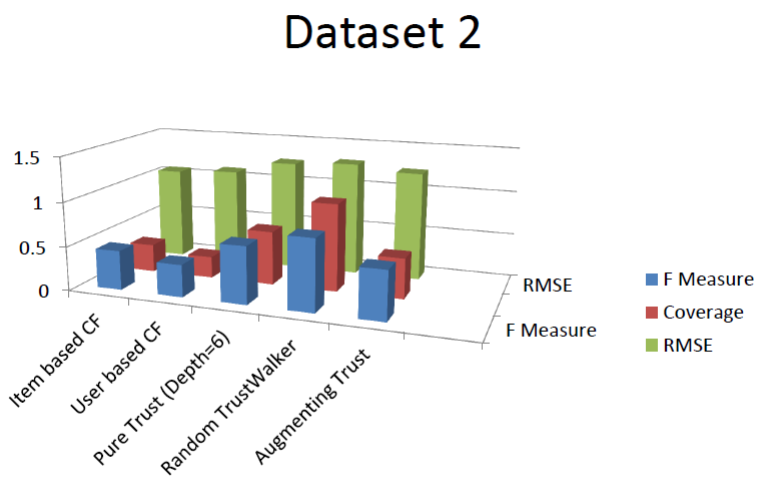

Figure 1.4: Plot of evaluation metrics on the Dataset 2

As is evident from Table 1.3, both $RMSE$ and $Coverage$ show significant improvements as compared to Dataset 2. The coverage of Augmented trust is not as good as expected since 50% of the dataset consists of cold start users and it might be difficult to find users with similarity greater than 0.4 in the trust network. One important observation from the Results in that the TrustWalker algorithm

Algoirthm	**RMSE**	**Coverage**(%)	**F Measure**
Item Based CF	1.00	38.0	0.50
User based CF	1.04	40.5	0.52
Pure Trust (depth =6)	1.21	68.4	0.69
Random TrustWalker	1.26	98.4	0.80
Augmented Trust	1.18	56.8	0.62

Table 1.3: Performance of recommender systems on Dataset 1

gives identical *Coverage* for all three datasets.

The plots of evaluation metrics corresponding to Table 1.3 is given in 1.5

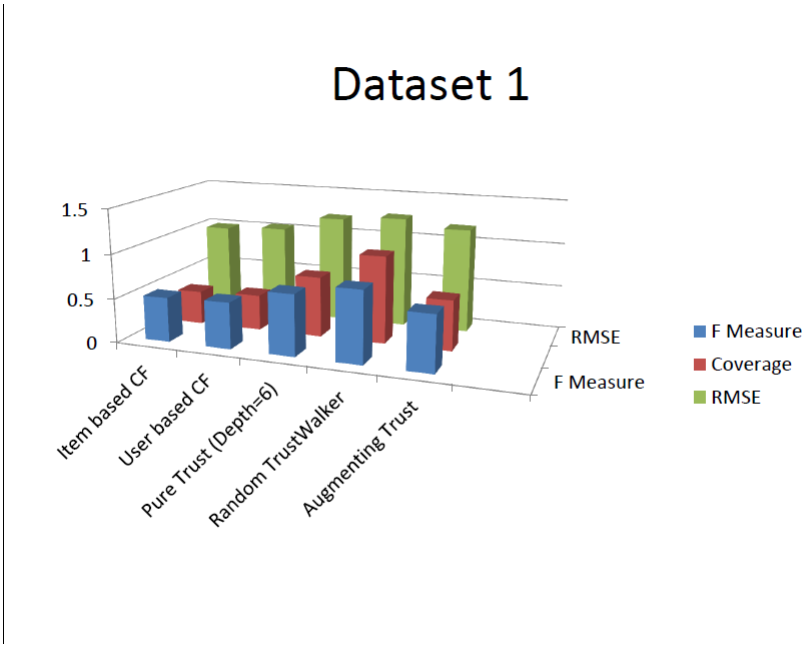

Figure 1.5: Plot of evaluation metrics on the Dataset 1

It is evident from Table 1.4 that $RMSE$ and *Coverage* under small decrements and increments respectively as the number of random walks increases from 10 to 100 while the $F\ Measure$ undergoes even smaller decrements.

Apart from the number of random walks, a host of minor tweaks were introduced in the Random TrustWalker and Pure Trust algorithms such as varying the

Dataset	Number of walks	RMSE	Coverage(%)	F Measure
Dataset 1	10	1.32	98.5	0.73
	100	1.26	98.9	0.748
Dataset 2	10	1.37	98.4	0.734
	100	1.31	98.9	0.745
Dataset 3	10	1.28	99.8	0.81
	100	1.22	100	0.82

Table 1.4: Performance of Random TrustWalker for different random walks

maximum depth, using the inferrred trust values from 1.4 in 1.6, including the negative correlation coefficients, etc. The best parameters were then chosen such that they corresponding to the best F *measure*.

6 Evaluating performance using Top N Recommendations

In practical systems, a user does not request a rating for a particular item, but wishes to see a ranked list of items that he is not aware but is likely to rate highly. This is a more natural problem definition for recommender systems and is called 'top-N item recommendation'. All the methods presented till now can be used to come up with a list of top-N recommendations. We have decided to extend Augmented trust algorithm to predict top-N recommendations. Augmented trust algorithm performs a BFS on 2 levels of the trust network to pick users with high degree of similarity. Based on our experience with trust-based recommender systems so far,we think this algorithm will give best performance in predicting top-N recommendations. We call this algorithm 'Top-N recommendations with Augmented Trust'.

'Top-N recommendations with Augmented Trust' is inspired from [13] in which the authors combined user based collaborative filtering and random walk on trust

network to predict top-N recommendations. From our experiments, we observed that an exhaustive BFS on the first two levels of the trust network with user similarity performs better than random walk on 6 levels. So in 'Top-N recommendations with Augmented Trust', to predict the required list for user u, we propose to do the following:

1 Perform a BFS on 2 levels of the trust network of user u and pick top-K users with highest similarity

2 Consider the items rated highly by the top-K users. Keep track of the ratings received by those items from top-K users and the similarity of these users with the target user.

3 The estimated rating for each item rated highly by trusted neighbors is computed as follows:

$$\hat{r}_{u,i} = \bar{r}_u \frac{\sum_{v \in NT_u} sim(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in NT_u} sim(u,v)} \tag{1.13}$$

Algorithm 4 gives the detailed algorithm:

where NT_u denotes the top-K users found by the BFS

6.1 Experiments

The augmented trust algorithm was implemented on Dataset 3 and the following metric was used to evaluate the performance of the algorithm:

6.2 Evaluation Metric

Typically, we use the leave-one-out method to evaluate recommender systems where a particular rating is withheld and then predicted by the algorithm us-

Algorithm 4 TopN(user)

```
topKusers = []
for i ∈ trustNetwork[user] do
    sim = pearson − corr(i, user)
    if sim > minimum similarity of (topKusers) then
        replace minSimilarity in topKuser with i
    end if
    for p ∈ trustNetwork[i] do
        sim = pearson − corr(i, user)
        if sim > minimum similarity of (topKusers) then
            replace minSimilarity in topKuser with p
        end if
    end for
end for
topItems = []
for i ∈ topKusers do
    topItems.add[topItemsofi]
end for
topItems − predicted − rating = average(topItems)
return topItems − predicted − rating − sorted[1 : N]
```

ing the trust network and other remaining ratings. It is necessary to modify this approach in order to use this leave-one-out technique for top-N recommendations. In this case, the recommender is asked to predict its top-N items for this user and if the withheld rating lies in this top-N recommendation then a hit has occurred. *Recall* is then used to evaluate the performance of the recommender and can be computed as follows:

$$Recall = \frac{Number\ of hits}{Size\ of\ test\ set} \tag{1.14}$$

6.3 Results

As expected, performance of top-N recommender systems improves with K and N. The performance of top-N recommender systems is not satisfactory (Table 1.5) and is an area of active research.

Parameters, (K, N)	**Recall**
(30,100)	4.08
(70,100)	4.08
(100,100)	6.12
(70,500)	12.07
(150,500)	16.32

Table 1.5: Experimental results for top-N recommendation with Augmented trust

7 Challenges

Dataset 1 is an example of sparse dataset. Storing this data in the form of a matrix with $49k$ rows and $140k$ items requires $66GB$ of storage space and thus it is expensive. This issue was resolved using adjacency lists. Another issue that cropped up was the computation time required for running multiple iterations of random TrustWalker algorithm. This acted as a bottleneck for the maximum size of the test set. As a result, a smaller test set of size 5000 was used.

7.1 Extended Epinions Dataset

Extended epinions dataset contains trust and distrust information to addition to ratings data. This dataset also has the dates at which items were rated or trust statements were issued. This dataset has 13,668,319 ratings, 1,560,144 items and 132,000 users. These users have expressed 717,667 trusts and 123,705 distrusts. This network is very large to be processed with the limited computational power accessible to us. Even if we can reduce the size of user-item rating matrix by random sampling, it is difficult to reduce the size of the trust network accordingly [14]. The sampled network has to preserve the qualities of a the original network such as average degree, average clustering coefficient, and diameter of the network. More importantly, since the performance of our recommender systems depends on presence of a sizeable portion of users from the test set in the training set and the

trust network, reducing the size of trust network ran into many challenges. We tried a few techniques to sample the dataset which hit roadblocks:

- Random Sampling : Random sampling causes the data to become too sparse. More than 90% of the users are cold start users with less than 2 ratings. As a result, it is not posssible to use collaborative filtering techniques.
- Contiguous chunk of ratings: Since the data are ordered by the date at which the data entry was created, there is once again little overlap between training and test mainly because of the sparse nature of the data.
- Trim item/users with ≤ 100 : This causes the data to become too dense which is no longer realistic. As a result, the results we obtained are impractically optimistic

Thus, the biggest challenge for the Extended Epinions dataset is to keep the data size manageable and realistic without making it too dense or too sparse or too disconnected.

8 Conclusion

We compared and contrasted collaborative filtering and trust-based algorithms for three datasets. Collaborative filtering techniques gave poor coverage for cold start users. On the other hand, trust based algorithms provided good *Coverage* even for sparse data. In random TrustWalker, we looked at six levels of trust network. As a result of noisy ratings from users that are far away in the trust network, we got poor values of $RMSE$. However, Trustwalker outperforms all the other algorithms in terms of *Coverage*. In Augmented trust, we only looked at the first two levels of trust network which led to an improvement in the $RMSE$ values but with reduced *Coverage*. Both these algorithms are better equipped to deal with

the tradeoff between *Coverage* and $RMSE$ as shown by their *F measure* values. Their performance for cold start users is superior to CF techniques.

We have extended augmented trust for providing top-N recommendations but the performance of top-N systems has scope for improvement. Another algorithm to consider for top-N recommendations is the one combining Augmented Trust and Item Based CF which is expected to improve both Coverage and Precision as compared to the Augmented trust technique.

There are many avenues that can be explored in future: one of them is to use trust based methods in presence of additional data features. For example, recency of ratings can be used to further improve the precision of recommender systems. Trust information can be split into different trust networks based on their context. The effect of distrust on the performance of recommender systems can also be investigated. Limited computational capacity and issues with network sampling have prevented us from further exploring these techniques on the Extended Epinions dataset.

Bibliography

[1] Waldman, Steven, *The Tyranny of Choice: Why the Consumer Revolution Is Ruining Your Life*, New Republic 206 (1992): 22-25.

[2] Iyengar, Sheena S., and Mark R. Lepper, *When choice is demotivating: Can one desire too much of a good thing?*, Journal of personality and social psychology 79.6 (2000) 995.

[3] Massa, Paolo, and Paolo Avesani, *Trust-aware bootstrapping of recommender systems*, ECAI Workshop on Recommender Systems. 2006.

[4] Yuan, Weiwei, et. al., *Improved trust-aware recommender system using small-worldness of trust networks*, Knowledge-Based Systems 23.3 (2010): 232-238.

[5] Su, Xiaoyuan, and Taghi M. Khoshgoftaar, *A survey of collaborative filtering techniques*, Advances in artificial intelligence 2009 (2009): 4.

[6] Abbasi, Mohammad Ali, Jiliang Tang, and Huan Liu. *Trust Aware Recommender Systems.*

[7] Golbeck, Jennifer Ann. *Computing and applying trust in web-based social networks*, 2005.

[8] O'Donovan, John, and Barry Smyth, *Trust in recommender systems*, Proceedings of the 10th international conference on Intelligent user interfaces. ACM, 2005.

[9] S. Milgram, *The small world problem*, Psychology Today, 2, 1967.

[10] Jamali, Mohsen, and Martin Ester, *TrustWalker: a random walk model for combining trust-based and item-based recommendation*, Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009.

[11] Ziegler, Cai-Nicolas, and Georg Lausen. *Analyzing correlation between trust and user similarity in online communities*, Trust management. Springer Berlin Heidelberg, 2004. 251-265.

[12] Nazemian, Ali, Hoda Gholami, and Fattaneh Taghiyareh, *An Improved Model of Trust-aware Recommender Systems Using Distrust Metric*, Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012). IEEE Computer Society, 2012.

[13] Jamali, Mohsen, and Martin Ester, *Using a trust network to improve top-N recommendation*, Proceedings of the third ACM conference on Recommender systems. ACM, 2009.

[14] Mohammad Al Hasan, Nesreen Ahmed, Jennifer Neville, *Network Sampling and Applications*, KDD 2013.