Clustering Recommenders in Collaborative Filtering Using Explicit Trust Information

Georgios Pitsilis[1,⋆], Xiangliang Zhang[2], and Wei Wang[3]

[1] Faculty of Science, Technology and Communication
Université du Luxembourg, Luxembourg
georgios.pitsilis@uni.lu
[2] Division of MCSE,
King Abdullah University of Science and Technology (KAUST), Saudi Arabia
xiangliang.zhang@kaust.edu.sa
[3] Interdisciplinary Centre for Security, Reliability and Trust (SnT Centre),
Université du Luxembourg, Luxembourg
wwangemail@gmail.com

Abstract. In this work, we explore the benefits of combining clustering and social trust information for Recommender Systems. We demonstrate the performance advantages of traditional clustering algorithms like *k-Means* and we explore the use of new ones like *Affinity Propagation* (AP). Contrary to what has been used before, we investigate possible ways that social-oriented information like explicit trust could be exploited with AP for forming clusters of high quality. We conducted a series of evaluation tests using data from a real Recommender system *Epinions.com* from which we derived conclusions about the usefulness of trust information in forming clusters of Recommenders. Moreover, from our results we conclude that the potential advantages in using clustering can be enlarged by making use of the information that Social Networks can provide.

Keywords: Social Trust, Clustering, Recommender Systems, Epinions.com, Affinity Propagation.

1 Introduction

Recommender systems are widely used nowadays and in simple terms they are services used for suggesting products to people who might be interested in them. Recommender systems became popular because they were able to provide personalized recommendations using as input the rating profiles of users. Despite their success and adoption in user communities, they have not shown their full potential yet. Various techniques such as *Nearest Neighborhood*, *Trust* and *Clustering* have been employed in Recommender Systems. Collaborative Filtering (CF), the best known type of Nearest-Neighborhood, has as fundamental idea the agreement on taste of people for predicting their future liking on new items.

⋆ The author carried out this work during the tenure of an ERCIM "Alain Bensoussan" Fellowship program.

I. Wakeman et al. (Eds.): IFIPTM 2011, IFIP AICT 358, pp. 82–97, 2011.

Information overloading and *Data sparsity* are two known issues [15][4] in information retrieval that Recommender Systems come to address. Being contrary to each other, the former is referred to the presence of too much information for making a choice, while the latter is caused by the lack of sufficient information during start-up. Information overload has serious implications on performance as it affects the scalability and responsiveness of a system due to the intensive processing power that is needed to correlate the increased amounts of data. Sparsity on the other hand, also known as *cold start problem*, is responsible for the poor performance of traditional Recommender Systems and appears when there is no sufficient information to correlate.

Clustering has been widely investigated in computer science as an unsupervised learning method [9,11,12]. In general, the idea of dividing the big communities of users into smaller sets (clusters) has seemed to offer advantages, such as scalability, which as a result improves the response time, due to the smaller set of data that algorithms operate on. On the other hand, the loss of prediction accuracy is not compensated at a sufficient level to render the use of clusters an attractive solution.

Trust has also been used to mitigate the problem of information overload [18], incorporating the idea of filtering out the available options to the trustworthy ones. Various models for trust propagation [21,20] have been proposed for filtering the selection of neighbors. Despite the benefits offered by employing trust in the production of recommendations, the additional computation effort for trust derivation incurs a penalty in performance, which can limit scalability.

Even though clustering and trust can both individually offer some distinguishable advantages over not using them, there is no previous work to investigate the benefits when used together. In our opinion a more systematic investigation is necessary in how trust expressed by people could be utilized for building neighborhoods, in which recommendations' performance will be enhanced by clustering those neighborhoods. The contribution of this paper is two fold. First, we demonstrate the advantages that a new clustering algorithm *Affinity Propagation* (AP) offers over the use of widely known *k-Means* approach. Second, we explore the potential benefits of using the combination of Explicit trust information with two different clustering algorithms.

The rest of the paper is organized as follows. In section 2 we refer to related work in the area. In section 3 we describe our motivation and essential knowledge about the clustering algorithms we used. Next, in section 4 we describe in more detail the setting used in our evaluation and finally our results and discussion follows in sections 5 and 6 respectively.

2 Related Work

Clustering has been the subject of research in the area of Recommender systems, but it has not been widely studied yet. We mention the research done by Sarwar et al. in [7] as the pioneer work in the field. The main idea behind clustering is to permanently partition choices into smaller sets so that making easier a future

choice for a neighbor. This is an idea very much adopted in Social Networking where information can be overwhelming. Truong et al. in [13] introduce an algorithm for producing uniform clusters of items by minimizing the variance between the items within the same clusters. Even though this work is useful, its applicability is still limited to a special algorithm of collaborative filtering (item-based), in which correlations are performed over items rather than users.

Despite a selection of neighbors that is based on trust alone does improve the quality of predictions, as quite many researches have shown [8,2], it is yet not enough to overcome the problem of information overload.

Implying trust from existing properties is an idea that has been employed in quite many schemes. In order to alleviate the sparsity problem, the authors in [22] proposed to build an implicit web-of-trust using information like the personal interactions between users. Massa et al. in [19] try to address the problem of information overload by utilizing the explicitly provided trust links of users. Different from ours, in their work they proposed a mechanism that used the trust values in the computation of predictions. Lathia et al. [8] used an algorithm for deriving how much users should trust each other based on their past ratings.

In recent work in [5], it has been attempted Clustering and Trust models to combine together, which at some point did improve the quality of recommendations. The same authors in [6] are driven by the idea that trust networks can be treated as random graphs, and they proposed a trust inference algorithm in which trust is derived from the connection distance in the graph. Even though there is still wide space for development into clustered networks, it so far remains unexplored how much helpful clustering can be for producing recommendations of good quality.

The development of new Clustering algorithms [3] highlights the need for a more systematic approach of Clustering in Recommender systems, similarly to other areas like intrusion detection for security systems [16] or data streaming [10].

Moreover, the emergence of social networking has given access to much more information than before, such as the explicitly expressed trustworthiness of users, and thus it has increased the potential for existing approaches to develop further. This combined with the fact that new clustering algorithms have not been explored adequately should mean that clustering trust information can be promising for further improving the performance of Recommender Systems.

3 Motivation

In traditional classic CF, only inputs from relevant users referred to as 'neighbors' or 'predictors' are employed in predictions to provide accurate recommendations. Contrary to traditional nearest neighborhood based approaches for CF, which require data to be globally available for being able to compute predictions for all users, in Clustering, user neighborhoods have more static sense. With clustering, neighbors should be pre-selected and be used the same for all predictions made thereafter. A widely adopted CF technique called k-Nearest

Neighborhood (*k-NN*) scheme, identifies the k most similar neighbors to use as inputs in the predicted recommendations. The pure *k-NN*, due to the requirement for neighbors to be selected dynamically from the whole set of users, still fails to overcome the problem of information overload. On the other hand, grouping users along with the information they carry into independent sets has the risk of deteriorating the performance. This necessitates that clustering should be done in a careful manner. A possible solution that we come to investigate in this paper is to utilize other sources of information which normally do not take part in the process of recommendation production. An interesting challenge we come to explore in this paper is to exploit the information provided by users in such a way that clusters of high quality can be built. With quality in clustering we refer to how suitable neighbors have been grouped together for computing predictions of high accuracy.

In total we attempted to test two different clustering schemes, an easily applicable, but established one, and another one which uses a newly developed algorithm. More particularly, our objective is to improve the quality of clustering and to quantify how extraneous information like social data could contribute to the performance of Recommendations. Contrary to using infered trust, as other researchers have attempted in the past [5], we used the trust information expressed by the users, as input to clustering. Furthermore, in order to quantify the contribution of either part in the performance gain, we attempted a comparison between using various sources of information over different clustering algorithms.

3.1 Clustering Algorithms

***K*-Means.** Clustering analysis with k-means regards the partition of a number of n observations $S = \{x_1, x_2, ..., x_n\}$, into a predefined number of k clusters ($k < n$). The main idea is to define k centroids, one for each cluster, as much as possible far away from each other. Next, points that belong to a given data set are associated with the nearest centroids. The allocation of each observation into a cluster is done on the basis of choosing the one which has the nearest mean. The aim is to minimize the within cluster sum of squares given by:

$$\arg\min \sum_{i=1}^{k} \sum_{x_j \in S_i} || \, x_j - m_i \, ||^2 \tag{1}$$

where m_i is the mean of points in the cluster S_i. The last step of binding the data points to clusters is repeated for a number of iterations where a new centroid is re-calculated for each cluster until the centroids have not changed their location any more.

Two key features of *k-Means* include the use of the *Euclidean distance* as a metric for measuring the distances between the points, and the number of clusters k, which is given as an input parameter to the algorithm. The latter is known as a weakness, as not well-defined values can lead to poor results.

Obviously the cluster centroids generated by *k-Means* do not necessarily represent real existing points from the cluster. Fig.1 shows an example of clustering

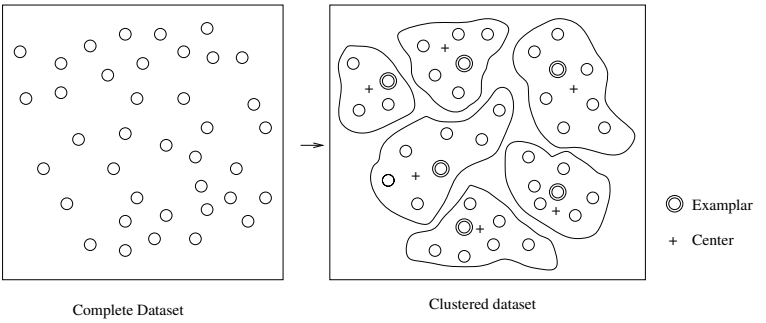

Fig. 1. Neighborhood formation with Clustering. The center of a cluster is normally generated by averaging all the data inside of the cluster, while the exemplar of the cluster is a real data point that best represents the cluster.

a number of points into 6 clusters, with the cross symbol (+) denoting the cluster center generated by *k-Means*.

Affinity Propagation. Affinity Propagation (AP) is a newly developed clustering algorithm proposed by Frey et al. [3]. AP takes as input negative similarities between data points which exchange two types of messages called *Responsibility* and *Availability*. These two messages are communicated between the candidate exemplar points.

An *Availability* message from a candidate k to another point i regards the supporting evidence that k has collected from other points for being an exemplar and is announced to i. The point i in response to the candidate k replies his opinion for this candidature taking into account other candidate exemplars. This type of message is called *Responsibility* and reflects to how well-suited point k is to serve as exemplar for i.

Finally after a few iterations of message exchanges, a set of exemplars emerges from the data points. An exemplar is a selected point of a cluster used as the center. Opposite to what happens in *k-Means*, in AP the exemplars are real data points (See exemplars in Fig.1).

Given a fitness function:

$$E(c) = \sum_{i=1}^{n} S(x_i, x_{c(i)}) \tag{2}$$

where $c(i)$ is the index of the exemplar that represents the point x_i in the cluster of n points, AP finds the mapping c that maximizes the fitness function $E(c)$. $S(x_i, x_j)$ is set to $-d(x_i, x_j)^2$ if $i \neq j$, and otherwise is set to a small constant $-s^*$ ($s^* \geq 0$). $-s^*$ represents a preference that x_i itself be chosen as an exemplar.

Rather than requiring a predefined number of clusters (e.g., *k-Means*), in AP the number of exemplars will emerge from the procedure of message passing. In detail, AP specifies the preference s^* for allowing a data point to become an exemplar. Note that for $s^* = 0$, the best solution is the trivial one, selecting every item as an exemplar.

4 Experimental Setting

Since the observations we intended to use from Recommender Systems could not be represented as points in n-space, we used derived quantities instead to map into the necessary attributes that clustering algorithms required. For the *k-Means* we mapped each pair of Correlation Coefficient ($w_{a,b}$) value between each two users a and b into distances from the hypothetical mean. In Collaborative Filtering the $w_{a,b}$ values are derived by applying the user's liking (ratings) for products to Pearson's formula:

$$w_{a,b} = \frac{\sum (r_{a,k} - \bar{r}_a)(r_{b,k} - \bar{r}_b)}{\sqrt{\sum (r_{a,k} - \bar{r}_a)^2 \sum (r_{b,k} - \bar{r}_b)^2}} \tag{3}$$

where $\bar{r_a}$ and $\bar{r_b}$ denote as the average of all ratings of user a and b respectively while $r_{a,k}$ and $r_{b,k}$ are the ratings for item k given by users a and b.

Finally, the derived values $w_{a,b}$ are then used in Resnick's prediction formula [17] along with the ratings of the selected predictors $r_{i,j}$ for the item of interest i to provide a prediction of likeness for user a.

$$p_{a,i} = \bar{r_a} + \frac{\sum [w_{a,j}(r_{j,i} - \bar{r_j})]}{\sum |w_{a,j}|} \tag{4}$$

where $\bar{r_j}$ denote as the average value of the ratings given by a predictor.

Eqns. (3) and (4) describe the traditional CF approach which we use as baseline for our comparison.

Since *k-Means* algorithm does not necessarily find the most optimal configuration, as far as to minimize the objective function (1), we applied *k-Means++* algorithm [1], an algorithm proposed for optimized selection of the cluster centers. Rather than arbitrarily selecting the initial centers of clusters, *k-Means++* can be used to seed *k-Means* with a series of suitable candidate centers. More specifically, with *k-Means++* the first cluster center is chosen uniformly at random among the data points. The remaining centers are chosen from the remaining points with probability proportional to the distance $D(x)^2$ of this point x from the nearest cluster center. Once all new cluster centers have been determined, the standard *k-Means* is applied. As a result, the speed and accuracy is improved over the random selection, as the *k-Means* converges quite faster after this seeding.

For *Affinity Propagation* we considered employing properties used to study the structural characteristics of Social Networks. The explicit trust information that is available in *Epinions.com* makes easier to derive properties from the structural characteristic of the network of users. The information that was provided as explicit trust of a user for another user is given in binary form, with a value of 1 denoting trust, or it is unknown if no value exists.

Vertex Similarity is an important concept used in social network analysis and data mining. The need for quantifying the similarity between vertices in a network can be approached from various perspectives. Consequently, it is reasonable to consider the structure of the network as a way to capture this.

Similarities between the vertices in a social network graph can possibly be identified by analyzing the patterns of edges between the vertices. A common heuristic that can be used for deciding whether two vertices (or users in the case of a social network) are similar with each other is to assume their similarity to be proportional to the number of neighbors they share.

The idea of transforming the process of community detection in social networks into clustering is a concept which has also been studied by other researchers [23]. Nevertheless, here we attempt to study it within a particular application domain.

We used a function proposed by *Jaccard* [25] to express the similarity between entities based on their connectivities with their neighbors. The formula is given by Eqn. 5. It was chosen as a simple and straightforward way to start our experimentation on transforming social behavioral information into similarity.

$$S_{jaccard}(i,j) = \frac{\mid N_i \cap N_j \mid}{\mid N_i \cup N_j \mid} \tag{5}$$

where N_i and N_j denote as the sets of users trusted by i and j respectively. The $S_{jaccard}$ similarity metric has a value range in the interval [0,1], and maximizes when $N_i = N_j$. $\mid \quad \mid$ indicates the cardinality (i.e., the number of elements in the set). *Jaccard* metric is typically used in the field of data mining to measure the diversity or similarity of sample sets. In our particular case with *Jaccard* metric is meant the level of potential direct trust that might exist between two entities a and b, given their explicit trust to third parties whom they trust in common. The intuition behind this formula can be phrased as: *Two users i and j are more similar in taste as the more users they trust in common.*

$$w_{new(a,b)} = w_{a,b}(1 + \frac{S_{jaccard(a,b)}}{2}) \tag{6}$$

Finally, the value of $S_{jaccard(a,b)}$ is reduced by the Correlation Coefficient $w_{a,b}$ to weighted similarity $w_{new(a,b)}$, which is then applied on Resnick's Eqn. 4 to predict ratings.

We consider the computation of N_i and N_j feasible in a social networking environment for the reason that, even though in many cases social links are meant strictly private, they can still be computable by a third trusted party such as the social networking service provider itself.

4.1 Test Schemes - Dataset

In performing our experimental validation we used data taken from a real Recommender System *Epinions.com*. We chose this particular Recommender system because it provides both ratings of users for products as well as trust information for users. In that system people can rate and write reviews for products. Ratings regard the evaluation of experiences of products by users and are expressed in a five star scale. Also, the members of *Epinions.com* can rate each other based on the reviews they have written for products, forming in this way a web-of-trusted recommenders. The trust information relates to the level of confidence expressed

by users for other users based on the reviews they have written for products, and it is provided in binary form. Yet, in the current form of *Epinions* the trust information is not used for computing personalized recommendations. Instead, users are expected to digest manually the reviews coming from sources trusted by users.

The special set we used was collected by Paolo Massa [14] in Nov-Dec 2003. This data set contained 664K ratings, 49K users and 139K products and thus being very sparse (99.025 %). To avoid poor performance due to the noisy behavior of Correlation Coefficient in sparse data sets, we selected a subset of the 1500 most experienced users on the basis of number of ratings given by each other (no matter how many outward and inward trust links they have). This was also done to ensure that the Pearson similarity value between the users is computable as long as there is adequate number of commonly trusted items.

In order to be able to compare in a fair way the results produced between the tested schemes and the baseline *k-NN* approach, we selected predictors for *k-NN* equal in number with those actually selected for the contrasted clustered method. In this way, both clustered and CF techniques used almost the same amount of information from the available knowledge that is expressed as ratings. As far as comparing with classic CF technique, we used all the available information for computing predictions by selecting all the available predictors (i.e., all users which have rated the item in question and the $w_{a,b}$ value with the querying user is computable).

In our experiments, we assumed random selection of predictors for the baseline technique. Since the traditional CF requires no trust information, we made no use of the trust values for the *k-NN* and the classic CF. We neither used the trust as input to the original *k-Means* scheme, which used only the traditional Pearson's similarity ($w_{a,b}$). In addition, for being able to investigate the contribution of social trust data in the performance we attempted to use the social data on both the traditional Clustering *k-Means* scheme as well as the new one which employs the AP algorithm. We called *Jaccard-AP* and *Jaccard-k-Means* the additional two test schemes which used the w_{new} quantity as input information in the clustering.

The static dataset we used limited our options of simulating the prediction creation process and the submission of feedback, as would normally happen in a real environment. Therefore, we tried a 'cross validation' scheme for being able to know how good our predictive model would be in estimating some rating that a user would give for an item before he had experienced that item. More specifically we applied a technique called *leave-one-out*. That is, each rating that was already provided in the dataset by some users was kept hidden and its value was predicted using the rest of the data. This method produces results of acceptable accuracy with that achieved by k-fold cross validation scheme. The reason for not using k-fold was mainly due to the small data samples we had available and which would lead to poor results with clustering.

4.2 Evaluation Metrics

As far as evaluating accuracy we considered both *Predictive* and *Classification Accuracy* as being important to be shown. *Predictive Accuracy* demonstrates the efficiency of the system to predict accurately the liking of users to products. Two metrics, the first called *Mean Absolute Error* (MAE), and the second *Root Mean Squared Error* (RMSE) are used to demonstrate this. RMSE is useful for quantifying undesirably large errors. *Classification Accuracy* measures the ability for a Recommender system in creating personalized lists of suggested products to users. In other words, it means the frequency at which the system decides correctly or not about if a product would be a good choice for a user. *F-Score* (f) is a metric, known also as *Harmonic Mean*, for measuring the efficiency of retrieval with respect to the cost of retrieval. *F-Score* became popular in measuring Recommender systems performance because it reflects the ability of the studied system to produce personalized top-k lists of liking products for users.

$$p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}, \quad f = \frac{2pr}{p + r} \tag{7}$$

True Positive (tp) or a *hit* denotes the case that a product is of user's liking and the Recommender system has predicted as such. Similarly, False Positive (fp) denotes the case of an item that has wrongly been predicted to be of liking of a user. False Negative (fn) is when an item has been predicted of not being of user's liking, but in reality it was. We used the value of 4 stars in a 5 star scale of our data as the threshold for classifying a bad experience from a good one (values 4 and 5 considered as Positive p). We considered as such, since rate 3 may even be used by users who are not happy enough with their choice.

F-score (f) requires another two metrics, *Precision* (p), which is the success in retrieving items that is of users interest, and *Recall* (r), which is the success in retrieving items that are truly of interest in relation to the number of all items that claim to be of interest.

To capture the implication on the number of items that can be predicted, we used the metric of *Coverage*. This metric (shown as C_a in Eqn.8) is specific to a particular user a who has in total rated a set of items I_a. I_b is referred to the items rated by some neighbor b and for which predictions can be made by a.

$$C_a = \frac{1}{\mid I_a \mid} \mid I_a \cap \{\cup_{b \in K} I_b\} \mid \tag{8}$$

5 Test Results

We present the most interesting results from our experimentation. In AP, it was not possible to directly generate any exact user-specified number of clusters for the set of data provided. Therefore we were able to experiment only with numbers of clusters that were possible to produce for our data. Conversely, for *k-Means* we provide results for all sizes of cluster communities, as it was possible to make

Table 1. Comparative Results of Prediction Accuracy

k	k-NN			k-Means			Jaccard K-means			Jaccard AP		
	MAE	RMSE	F-Score	MAE	RMSE	F-Score	MAE	RMSE	F-Score	MAE	RMSE	F-Score
5	0.1756	1.236	0.7256	0.1747	1.223	0.7348	0.1734	1.216	0.7372	-	-	-
10	0.1709	1.233	0.7237	0.1705	1.190	0.7383	0.1699	1.126	0.7417	-	-	-
15	0.1691	1.228	0.7217	0.1668	1.163	0.7456	0.1674	1.166	0.7447	-	-	-
16	-	-	-	-	-	-	-	-	-	0.1641	1.133	0.7481
20	0.1665	1.228	0.7173	0.1634	1.137	0.7492	0.1647	1.147	0.7556	-	-	-
25	0.1615	1.211	0.7309	0.1587	1.118	0.7652	0.1616	1.128	0.7599	-	-	-
26	-	-	-	-	-	-	-	-	-	0.1632	1.128	0.7578
29	-	-	-	-	-	-	-	-	-	0.1549	1.077	0.7682
30	0.1612	1.217	0.7315	0.1581	1.111	0.7696	0.1628	1.140	0.7603	0.1530	1.067	0.7717
40	0.1594	1.206	0.7382	0.1545	1.087	0.7742	0.1573	1.106	0.7717	0.1479	1.024	0.7798
44	-	-	-	-	-	-	-	-	-	0.1471	1.020	0.7891
45	0.1596	1.206	0.7357	0.1547	1.084	0.7764	0.1556	1.096	0.7778	0.1459	1.015	0.7889
47	-	-	-	-	-	-	-	-	-	0.1480	1.027	0.7869

the appropriate adjustments to the algorithm so as to generate the number of clusters we preferred.

Detailed results for prediction accuracy are provided in Table 1, with k denoting as the number of clusters. MAE is expressed in percentage as a fraction of 1.

In the comparison diagram in Fig. 2 for all schemes tested, we observe for MAE the following: first, it follows a decreasing trend for all figures as the number of clusters increases, and second, the error measured in the original *k-Means* clustering has lower figure (with a small exception at 5 clusters) than in both *k-NN* and *Jaccard-k-Means*. The former can be likely due to the way

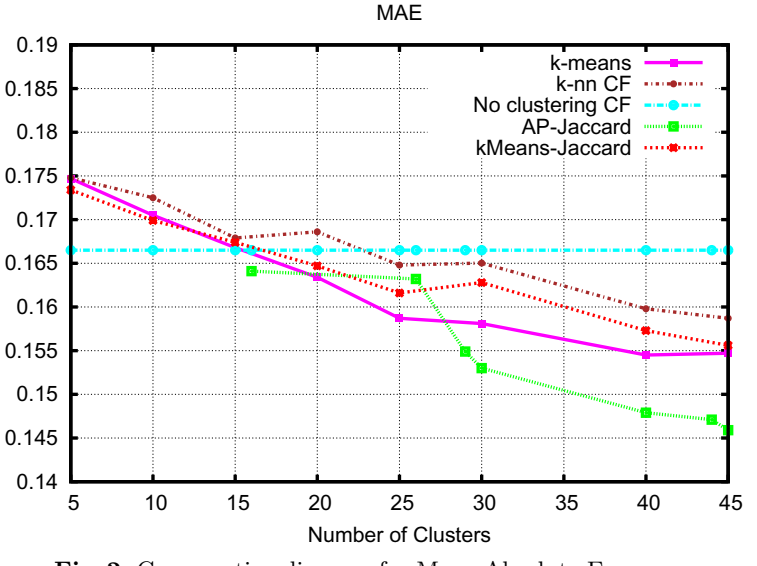

Fig. 2. Comparative diagram for Mean Absolute Error

that clustering works. Increasing the number of clusters while keeping the whole population constant has as a result clusters produced of smaller number of users, but still highly similar with each other. The high similarity of users within the same clusters is the reason of the improvement of the quality in predictions. Besides, selecting the most influential users has been the fundamental idea of *k-NN* filtering in Recommender Systems [24]. For instance we mention that the Intra-cluster distance for clusters generated with AP ranged between 0.75 to 0.82 for the whole range of Cluster sizes we experimented with. For the CF field as Intra-cluster distance is meant the average squared similarity between the members of the same cluster. For K-means clustering respectively Intra-cluster distance remained significantly lower between 0.35 and 0.60.

In the same diagram, it is also depicted the case of using all available predictors for each rating prediction (no-clustering) which roughly considers all users in a unique cluster. As can be seen, the no-clustering approach outperforms all the other techniques for small number of clusters (less or equal to 15). However, the distinctive difference with the no-clustering is on the higher computational cost needed, due to the larger quantities of data used to achieve the same result (Information Overload problem).

Our findings regarding the prediction accuracy of *k-Means* are in line with those reported by pioneer researchers in the field, like Sarwar et al. in [7]. More specifically, in our figures for MAE, it is shown a decreasing trend for accuracy as the number of clusters increases. We identify the reason why in our results *k-Means* clustering seems to outperform the baseline *k-NN* approach on the fact that in our case we assumed the standard CF algorithm as the baseline rather the no-Clustering approach. Also in the latter scheme, no filtering has been applied on the less similar predictors. We considered that option because neighbors who might have bad influence on prediction quality can still be potential candidates.

We omitted the RMSE diagram as it follows a similar pattern with MAE and hence it is still in line with our previous finding. As far as comparing the clustering algorithms with each other, from the results it can be seen that, for both the MAE and RMSE of predictions, in all our experiments the *Affinity Propagation* algorithm, that used explicit trust, is the only algorithm that outperforms all the traditional approaches that used or didn't use clustering.

Nevertheless, clustering has the drawback of limiting the number of predictors that can actually be chosen to those which belong to the same cluster with the querying user. That appears as reduced coverage, and responsible for this is the smaller set of experiences that can be utilized by making them shared into a small community rather than a bigger one. For *AP* responsible also for the low coverage is the fact that less users finally get allocated to a cluster for receiving help from their neighbors. A comparative diagram of the Coverage for *k-Means*, AP, *k-NN* as well as the other clustering approaches can be seen in Fig. 3. In this diagram, it is depicted the drop in the number of predictions for items that can be computed for increasing numbers of clusters. Very interestingly, coverage decreases fourfold with clustering when the number of clusters is large, regardless of the particular clustering method used. It is worth noticing that

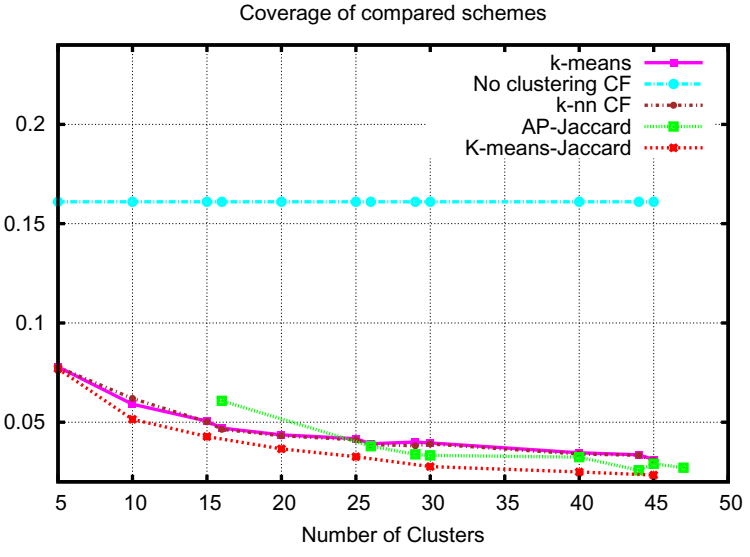

Fig. 3. Coverage in % of predictions

Coverage does not decrease sharply over 15 clusters and it is maintained at a constant level further on.

Concerning a comparison between *Affinity Propagation* clustering and the traditional *k-Means*, it can be seen from Figure 3 that the penalty paid for clustering remains at the same level, no matter if the social trust information is actually used or not. Apparently though, the AP approach is undoubtedly better, given the significant weakness of *k-Means* over AP, as far as the prediction accuracy. A comparison between the two schemes which used social trust information (AP and Jaccard *k-Means*) shows that the better quality of clusters achieved with AP finally does have an impact on the number of experiences that users can finally find useful within the clusters. This can be interpreted as saying: *A user clustered with AP is more likely to be allocated to a cluster whose members can contribute most useful experiences for him/her*. In that respect Affinity Propagation is the winner.

The general observation is that the use of explicit trust information in fact does not help Coverage to improve, as in the best case it remains at the same levels achieved with *k-Means*. The use of sparse trust data is mainly responsible for this. As a consequence, it becomes less likely for the distance between two users (expressed as w_{new} in Eqn. (6)) to be computed. That means the ratio of useful neighbors over all neighbors remains almost constant for 15 clusters and above. *Jaccard k-Means* fails to compensate the loss of useful neighbors, whereas *Jaccard AP*, by clustering together users which are more useful to each other, has been more successful. Using other sources of information, such as: implicit trust derived from trust propagation, might be a solution to overcome this weakness. Abstracting clusters and using representative values of rates across clusters

for users who have not enough neighbors might also be way to overcome the problem.

As far as classification accuracy is concerned, it is worth pointing out that similarly to what has been observed for MAE and RMSE (seen in Fig. 2 and table 1), F-Score does improve as the number of clusters increases. That means, clustering is becoming more helpful indeed. This advantage is distinguishable from small numbers of clusters.

The no-Clustering scheme instead fails to predict better the items of user's liking, and therefore achieves F-Score=0.752. The *k-NN* selection instead performs even worse as F-Score remains almost at a constant level, achieving a score around 0.725, regardless of the size of the selected neighborhood. The reason for this might be that: selecting at random a set of constant size, of not necessarily the best predictors, does not differ in finding the top products of user's liking, no matter how big the group of candidate predictors is.

In addition, very interestingly, the clustering approaches which make use of explicit trust, outperform the *k-NN* approach. This advantage becomes obvious from early on when the number of clusters exceeds 15 for *Jaccard* k-Means. For *Jaccard AP*, similarly as observed for MAE and RMSE, the accuracy improves dramatically beyond the 26 clusters. The reason for this performance improvement is the same as for predictive accuracy.

In conclusion, clustering can clearly give predictions of higher precision than the conventional method can achieve. This advantage is enhanced when the explicit trust information is used for forming the clusters. Moreover, the successful selection of neighbors that novel clustering schemes, like AP, can achieve is found more beneficial when applied to small clusters rather than large ones.

6 Concluding Remarks

Trust and Data Clustering have been the subject of investigation in the research community in recent years, as a solution to improving the accuracy of Recommender Systems and to overcoming the sparsity and information overload problem. Nevertheless, little effort has been put on exploring the potential of new clustering algorithms and exploiting the information that users provide explicitly for the people they trust.

We performed a series of experiments in the context of Recommender Systems with the purpose to investigate our central question of whether clustering can be benefited by the use of trust information that users provide explicitly. We tested two clustering schemes, *k-Means* and *Affinity Propagation* and contrasted with the baseline Collaborative Filtering and *k-NN* approaches which make no-use of clusters. To the best of our knowledge, this is the first time that *Affinity Propagation* algorithm has been tested in Recommender Systems. Since it is known that clustering has potential advantages in Recommender Systems, in this work we also came to answer the question if the use of social trust can provide benefits to clustered users.

Apparently, our experimentation with the clustering algorithms showed that the trust information which people express explicitly for the people they know can undoubtedly be useful for improving the accuracy of their recommendations. At the same time, with clustering the problem of information overload is overcome. The fact that such trust data we used as input to the clustering algorithm is already provided in social networks as core information, makes profound what the benefit from social networking can be. We should note though that there is a drawback of clustering in general, which reflects the number of predictions that can be produced for the clustered users.

An interesting motivation for justifying the behavior of the results we received for Affinity Propagation algorithm could be useful to further investigate possible correlations between the principal objectives of the clustering algorithms we tried and the prediction algorithm used in CF. However, it is equally interesting to further investigate the reasons why particularly clusters formed up by using explicit trust information can be more useful than when the traditional Pearson's similarity is used.

Our choice of trying different sources of information into conventional clustering algorithms helped our understanding on the key factors for achieving good performance when applying clustering in Recommender Systems.

As far as the drawbacks of clustering which came up by the use of explicit trust and which sparsity of the social network is mainly responsible for, there is much hope to overcoming this with the application of cluster abstraction techniques. There is also much to learn from other disciplines related to Social Networking including Behavioral Science.

References

1. Arthur, D., Vassilvitskii, S.: k-means++: The Advantages of Careful Seeding. In: Proc. of Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), pp. 1027–1035. Society for Industrial and Applied Mathematics, Philadelphia (2007)
2. O'Donovan, J., Smyth, B.: Trust in recommender systems. In: Proc. of 10th International Conference on Intelligent User Interfaces (IUI 2005), pp. 167–174. ACM, New York (2005)
3. Frey, B., Dueck, D.: Clustering by Passing Messages Between Data Points. Science 315, 972–976 (2007)
4. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommendation algorithms for e-commerce. In: Proceedings of the 2nd ACM conference on Electronic commerce (EC 2000), pp. 158–167. ACM, New York (2000)
5. DuBois, T., Golbeck, J., Kleint, J., Srinivasan, A.: Improving Recommendation Accuracy by Clustering Social Networks with Trust. In: ACM RecSys 2009 Workshop on Recommender Systems & the Social Web, New York (2009)
6. DuBois, T., Golbeck, J., Srinivasan, A.: Rigorous Probabilistic Trust-Inference with Applications to Clustering. In: Proc. of 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2009), vol. 01, pp. 655–658. IEEE Computer Society, Washington, DC, USA (2009)

7. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Recommender Systems for Large-Scale E-Commerce: Scalable Neighborhood Formation Using Clustering. In: The Fifth International Conference on Computer and Information Technology, ICCIT 2002 (2002)
8. Lathia, N., Hailes, S., Capra, L.: Trust-Based Collaborative Filtering in Trust Management II. In: IFIP International Federation for Information Processing, vol. 263, pp. 119–134. Springer, Boston (2008)
9. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Simoudis, E., Han, J., Fayyad, U. (eds.) Proc Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231. AAAI Press, Menlo Park (1996)
10. Zhang, X., Furtlehner, C., Sebag, M.: Data streaming with affinity propagation. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 628–643. Springer, Heidelberg (2008)
11. Kaufman, J., Rousseeuw, P.J.: Clustering by means of medoids. In: Dodge, Y. (ed.) Statistical Data Analysis Based on the L Norm. Elsevier, Amsterdam (1987)
12. MacQueen, J.B.: Some Methods for Classification and Analysis of MultiVariate Observations. In: Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
13. Truong, K., Ishikawa, F., Honiden, S.: Improving Accuracy of Recommender System by Item Clustering. IEICE - Trans. Inf. Syst. E90-D 9, 1363–1373 (2007)
14. A cooperative environment for the scientific research of trust metrics on social networks, `http://www.trustlet.org/` (retrieved May 2009)
15. Bollen, D., Knijnenburg, P.B., Willemsen, M.C., Graus, M.: Understanding choice overload in recommender systems. In: Proc. of fourth ACM conference on Recommender systems (RecSys 2010), pp. 63–70. ACM, New York (2010)
16. Wang, W., Zhang, X., Pitsilis, G.: Abstracting audit data for lightweight intrusion detection. In: Jha, S., Mathuria, A. (eds.) ICISS 2010. LNCS, vol. 6503, pp. 201–215. Springer, Heidelberg (2010)
17. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Dechter, R., Kearns, M., Sutton, R. (eds.) Proc. 18th National Conference on Artificial Intelligence, pp. 187–192. American Association for Artificial Intelligence, Menlo Park (2002)
18. Walter, F., Battiston, S., Schweitzer, F.: Coping with Information Overload through Trust-Based Networks. In: Helbing, D. (ed.) Managing Complexity: Insights, Concepts, Applications, vol. 32, pp. 273–300. Springer, Heidelberg (2008)
19. Massa, P., Avesani, P.: Trust-aware recommender systems. In: Proceedings of the 2007 ACM conference on Recommender systems (RecSys 2007), pp. 17–24. ACM, New York (2007)
20. Josang, A.: A logic for uncertain probabilities. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 9(3), 279–311 (2001)
21. Avesani, P., Massa, P., Tiella, R.: A trust-enhanced recommender system application: Moleskiing. In: Liebrock, L.M. (ed.) Proc. of 2005 ACM symposium on Applied computing (SAC 2005), pp. 1589–1593. ACM, New York (2005)
22. Liu, H., Lim, E., Lauw, H.W., Le, M.-T., Sun, A., Srivastava, J., Kim, Y.A.: Predicting trusts among users of online communities: an epinions case study. In: Proc. of 9th ACM conference on Electronic commerce (EC 2008), pp. 310–319. ACM, New York (2008)

23. Liu, Z., Li, P., Zheng, Y., Sun, M.: Community Detection by Affinity Propagation, Technical Reports No. 001, 200, Dept. of Computer Science and Technology, Tsinghua University,Beijing,China (2008)
24. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proc. of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 1999), pp. 230–237. ACM, New York (1999)
25. Jaccard, P.: Bulletin de la Société Vaudoise des Sciences Naturelles. Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines 37, 241–272 (1901)