Proc. ACM 10th International Conference on Information and Knowledge Management (CIKM'01)

Selecting Relevant Instances for Efficient and Accurate Collaborative Filtering*

Kai Yu
Institute for Computer Science
University of Munich
yu_k@dbs.informatik.uni-muenchen.de

Xiaowei Xu
Information and Communications
Corporate Technology
Siemens AG
Xiaowei.Xu@mchp.siemens.de

Martin Ester, Hans-Peter Kriegel
Institute for Computer Science
University of Munich
{ester,kriegel}@dbs.informatik.uni-muenchen.de

ABSTRACT
Collaborative filtering uses a database about consumers' preferences to make personal product recommendations and is achieving widespread success in both E-Commerce and Information Filtering Applications nowadays. However, the traditional collaborative filtering algorithms do not scale well to the ever-growing number of consumers. The quality of the recommendation also needs to be improved in order to gain more trust from the consumers. In this paper, we present a novel method to improve the scalability and the accuracy of the collaborative filtering algorithm. We introduce an information theoretic approach to measure the relevance of a consumer (instance) for predicting the preference for the given product (target concept). The proposed method reduces the training data set by selecting only highly relevant instances. Our experimental evaluation on the well-known EachMovie data set shows that our method doesn't only significantly speed up the prediction, but also results in a better accuracy.

Keywords
Collaborative filtering, Instance selection, Data Mining.

1. INTRODUCTION
The Internet is increasingly used as a channel for sales and marketing. More and more people purchase products through the Internet. One main problem that the customers face is how to find the product they like from millions of products. For the vendor, again, it is crucial to find out the customers' preferences for products. Collaborative filtering or recommender systems have emerged in response to these problems [5; 9; 12].

Collaborative filtering accumulates a database of consumers' product preferences, and then uses them to make customer-tailored recommendations for products such as clothing, music, books, furniture, and movies. The user's preference can be either explicit votes or implicit usage. Collaborative filtering can help E-commerce in converting web surfers into buyers by personalization of the web interface. It can also improve cross-sell by suggesting other products the customer might be interested in. In a world where an E-commerce site's competitors are only a click or two away, gaining customer loyalty is an essential business strategy. Collaborative filtering can improve the loyalty by creating a value-added relationship between supplier and consumer. Collaborative filtering has been very successful in both research and practice. However, there still remain important research questions in overcoming two fundamental challenges for collaborative filtering [10].

The first challenge is to improve the scalability of the collaborative filtering algorithms. Existing collaborative filtering algorithms can deal with thousands of consumers within a reasonable time, but the demand of modern E-Commerce systems is to handle millions of consumers.

The second challenge is to improve the quality of the recommendations. Consumers need recommendations they can trust to help them find products they will like. If a consumer trusts a recomender system, purchases a product, but finds out he does not like the product, the consumer will be unlikely to use the recommender system again.

1.1 Contributions of this paper
In this paper, we propose a novel collaborative filtering method to meet these challenges, and provide an information theoretical analysis for learning customers' preferences from a reduced training set. Our experiments on a real-world database show the proposed algorithm not only improves the efficiency of prediction, but also results in a better accuracy. In summary, the main contributions of this paper are:

1. Based on the nature of collaborative filtering, the dependency between items is studied and a measure of relevance is proposed.
2. An information theoretical framework is introduced to evaluate instances' (customers') relevance for training and the intimate relationship between feature selection and instance selection is interpreted in this framework.
3. A novel approach is proposed to actively select relevant instances (customers) for training to dramatically improve the efficiency and accuracy of memory-based collaborative filtering.

* The work was performed in Cooperate Technology, Siemens AG. The contact author is Xiaowei Xu: Xiaowei.Xu@mchp.siemens.de

Table 1. **memory-based collaborative filtering and instance-based learning**

	Memory-based collaborative filtering	**Instance-based learning**
Object	Customer	Instance
Target	Vote on target items (unfixed)	Label of target concept (fixed)
Features	Vote on the rest items (with missing values and different customers have different features)	Value of features
Distance function	Correlation (over common items)	Euclidean distance
Output generation	Weighted sum over all the training instances	Nearest neighbor(s) algorithm
Size of database	Over tens of thousands of customers (ever increasing)	Hundreds or thousands of instances
Number of features	Over thousands	No more than hundreds

1.2 Organization

Section 2 introduces related work, collaborative filtering algorithms and instance selection methods in lazy learning. In section 3, we motivate our work from an intuitive and a theoretical perspective, then we study the dependency between items and the relevance of instances, and describe the proposed algorithms. In section 4, we evaluate our approach on the well-known EachMovie data set. The paper ends with a summary and some interesting future work.

2. RELATED WORK

2.1 Collaborative Filtering Algorithms

The task in collaborative filtering is to predict the preference of a particular user (or active user) to a given product (target item) based on a database of customers' product preferences. There are two general classes of collaborative filtering algorithms: memory-based methods and model-based methods [5].

The memory-based algorithm [9; 12] is the most popular prediction technique in collaborative filtering applications. Its basic idea is to predict the active user's vote of an item as a weighted average of the votes given to that item by the other users. Specifically, the prediction $P_{a,j}$ of active user a on item j is given by:

$$P_{a,j} = \bar{v}_a + k\sum_{i=1}^{n} w(a,i)(v_{i,j} - \overline{v_i}) \qquad (2.1)$$

where n is the number of the users who rated item j, $\overline{v_i}$ is the mean vote for user i, $v_{i,j}$ is the vote cast by user i on item j, $w(a,i)$ is the similarity measure between active user a and user i and k is a normalizing factor such that the absolute values of the weights sum to unity. There are two popular similarity measures: Pearson correlation coefficient and cosine vector similarity. Since the correlation-based algorithm outperforms the cosine vector based algorithm [5], we apply the former one as the similarity measure. The Pearson correlation coefficient was first introduced in [9]. The correlation between users a and i is defined as:

$$w(a,i) = \frac{\sum_j \left(v_{a,j} - \bar{v}_a\right)\left(v_{i,j} - \bar{v}_i\right)}{\sqrt{\sum_j \left(v_{a,j} - \bar{v}_a\right)^2 \sum_j \left(v_{i,j} - \bar{v}_i\right)^2}} \qquad (2.2)$$

Memory-based methods have the advantage of being able to rapidly incorporate the most up-to-date information and yield relatively accurate predictions [5], but they suffer from poor scalability for large numbers of users. This is because the search for similar customers is slow in large databases.

Model-based collaborative filtering, in contrast, uses the customer's preference database to learn a model, which is then used for predictions. The model can be built off-line over several hours or days. The resulting model is very small, very fast, and essentially as accurate as memory-based methods [5; 3]. Model-based methods may prove practical for environments in which consumer preferences change slowly compared to the time needed to build the model but they are not suitable for environments in which consumer preference models must be updated rapidly or frequently.

In this paper, we will focus on memory-based algorithms and present a novel approach to improve their efficiency and accuracy.

2.2 Memory-based Collaborative Filtering and Lazy Learning

Memory-based collaborative filtering belongs to a class of lazy learning methods [1] which simply store all the training instances instead of producing any explicit generalization during the training phase and reply to information requests by combining their stored training instances. In this paper, our study on collaborative filtering focuses on actively selecting training instances (customers), which is also a topic related to lazy learning algorithms. Therefore it is necessary to give a brief comparison (as shown in table 1) between memory-based collaborative filtering and typical lazy learning algorithms, such as instance-based learning [2]. The content of table 1 should help to understand our work in this paper. In the following, we don't distinguish between 'customer and instance', 'feature items and features', 'target item and target concept' etc.

2.3 Instance Selection in Lazy Learning

Since lazy learning algorithms search through all available instances to classify (or predict) a new input vector, it is necessary to decide what instances to store for generalization in order to reduce excessive storage and time complexity. Therefore instance selection has become an important topic in lazy learning [2; 14; 8]. Some algorithms seek to select representative instances, which could be border points [2] or central points [16]. The intuition behind retaining border points is that "internal" points do not affect the decision boundaries as much as border points,

Table 2. Four customers' votes on four movies in Example 1

	Superman	Titanic	Dances with Wolves	Batman
Jason	5			5
Karen			3	4
Fred	2	5		2
Tom	4	3	4	?

and thus can be removed. However, noisy points are prone to be judged as border points and added to the training set. If the central points are chosen as representations, the selection should be carefully done since the decision boundary lies halfway between two nearest instances of different classes. Another class of algorithms attempt to remove noisy points before selecting representative instances [14]. For example, DROP3 used a simple noise-filtering pass: any instance misclassified by its *k* nearest neighbors is removed [14]. For almost all the algorithms mentioned above, classification is performed at least once in each step of removing or adding an instance, so it is somewhat expensive w.r.t. computational complexity.

In a very large training data set, there might be many "poor" instances for which their target concept are not sufficiently and effectively described by their features. This could be caused by missing feature values or irrelevant features. These "poor" instances should also be removed. In the following sections, we will study the "quality" of the instances and propose to select the instances of high "quality" for faster and more accurate collaborative filtering.

3. SELECTION RELEVANT INSTANCES

3.1 Motivation

In this paper, our work is mainly focused on memory-based collaborative filtering algorithm [5]. This algorithm simply stores all the training instances (customers) in database and defer processing of its inputs until it receives requests for prediction. Although memory-based collaborative filtering had been widely studied and turned out a success in applications [5; 9; 12], it still remains several important questions:

(1) Since it becomes more and more expensive for the algorithm to search the whole database with explosively growing consumers, how to speed up the prediction?

(2) Is every instance equally useful for the learning process?

(3) If an instance is not well described by its features, how will this instance impact the prediction?

For the purpose of motivation, let us study a simple example.

Example 1. A list of people and the movies they voted as well as the values of the votes are shown in Table 2. (A real database, such as EachMovie, would have tens of thousands of customers and thousands of movies, but we use a smaller set for illustration.) Our task is to predict Tom's preference for the movie Batman based on other three customers' preferences. From the lazy learning point of view, Jason, Karen and Fred are stored training instances while Tom is regarded as an input instance for the query. The vote on Batman is the target concept and votes on other movies are the features. A common phenomenon in this kind of database is that every instance has a large number of features with missing values. So a question arises: for the prediction of Batman, are all the three training instances equally important? Our answer is "No", if we could find following rules: 1. There is a statistical relationship between votes on Superman and Batman, anyone who likes Superman always likes Batman with a high probability. 2. There is a dependency between votes on Titanic and Batman, anyone who likes Titanic tends to dislike Batman. 3. There is no clear statistic relationship between votes on Dances with Wolves and Batman. As shown in Table 1, Fred voted on two relevant movies, Jason voted on one relevant movie, but Karen only voted on the irrelevant movie, Dance with Wolves. Therefore, Fred's preference data is the most important instance for the prediction of Batman, and Jason is also a useful instance, while Karen may mislead the prediction. If we removed Karen from the training set, it might be expected that prediction time could be shortened to 2/3 and the accuracy might be improved.

The above example indicates that instance selection and feature selection are closely related. Blum pointed out that more studies need to be conducted to help understand this relationship [4]. In the machine learning community, feature selection has received wide attention [4; 13; 8]. However, we believe that instance selection has not been pushed to the same level yet. Especially, in data mining tasks such as collaborative filtering, data acquisition is performed automatically and there is no human expert involved to manually label the instances or select relevant instances. There are potentially a tremendous number of irrelevant instances whose target concept is not adequately and effectively described by given features, and those irrelevant instances would dramatically degrade the performance of learning both w.r.t. efficiency and accuracy. Thus, instance selection is very desirable for collaborative filtering.

3.2 Dependency between Items

As indicated above, instance relevance and feature dependency seem to be intimately related. From intuitive perspective, the dependency between target concept and other features has played an important role in instance selection in example 1. In this section, we will study this dependency and see how to measure it.

Example 2. There are 50 users who give scores for two movies, *i* and *j*. The scores take the value ranged from 0 (bad) to 5 (good). Let us consider two different situations, case 1 and case 2 respectively, as shown in fig.1. In case 1, users are nearly uniformly distributed in the movie-movie score space. *A* and *B* are two arbitrary users who have close interests to movie *i*. However, it does not necessarily mean that they also have a similar preference for movie *j*. But in case 2, the situation is quite different. We can find the following rule: for users who dislike movie *i*, movie *j* is always their favorite. Users who like movie *i* always rate the other one just above the average.

Memory-based collaborative filtering algorithms are built on the following assumption:

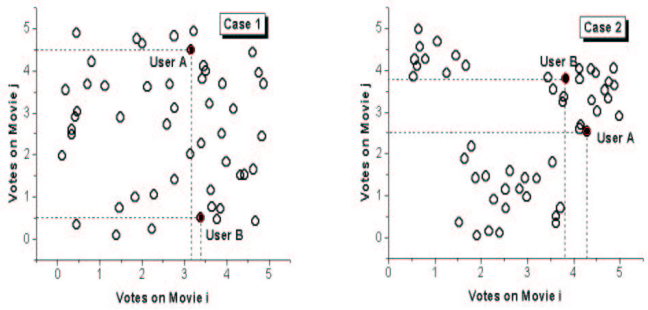

Figure 1. 50 users' votes on movie *i* and *j* in example 2

People who have similar preferences for some other items (feature items) would also show similar preferences for the target item.

Thus, the success of collaborative filtering greatly depends on whether the above assumption is true in practice. But from example 2 we can see that this assumption does not always hold. In case 1, e.g., if we know user *A* and user *B* have given very similar votes on movie *i*, and want to predict user *A*'s attitude towards movie j given user *B*'s vote, we can't do that with a high confidence because votes on the two movies seem to be statistically independent. In contrast, votes on the two movies in case 2 have shown some kind of interesting dependency and the assumption holds with a high confidence.

Suppose that the votes on each feature item independently influence the votes on the target item, based on the above assumption, a straightforward way to calculate the dependency between feature item *i* and target item *j* is :

$$p(|v_{j,u_A}-v_{j,u_B}|<e \,\big|\, |v_{i,u_A}-v_{i,u_B}|<e) \tag{3.2.1}$$

where u_A and u_B represent two arbitrary users and e is the threshold. If the difference between two votes is below e, these two votes are regarded to be similar. The above conditional probability expresses the probability that two arbitrary users have similar preferences for item j given the condition that those two users have similar preference for item i. Obviously this measure provides a way to select relevant feature items for predicting someone's votes on target item. However, this would be very expensive since its runtime complexity is $O(n^2m^2)$ if n is the number of consumers and m the number of products.

Alternatively, we use mutual information as the measure of dependency between a feature item and the target item. In the following theorem, we will show that it is consistent with the relevance measured by (3.2.1) and is significantly more efficient to calculate. In information theory [11], mutual information represents a measure of statistic dependence between two random variables X and Y with associated probability distributions $p(x)$ and $p(y)$ respectively. Following Shannon, the mutual information between X and Y is defined as:

$$MI(X;Y)=\sum_x\sum_y p(x,y)\log\left(\frac{p(x,y)}{p(x)\,p(y)}\right) \tag{3.2.2}$$

Furthermore, mutual information can be equivalently transformed into the following formulas:

$$MI(X;Y)=H(X)-H(X\mid Y) \tag{3.2.3}$$

$$MI(X;Y)=H(Y)-H(Y\mid X) \tag{3.2.4}$$

$$MI(X;Y)=H(X)+H(Y)-H(X,Y) \tag{3.2.5}$$

where $H(X)$ is the entropy of X, $H(X|Y)$ is the conditional entropy of X given Y and $H(X,Y)$ is the joint entropy of two random variables. The definitions of entropy, and the proof of the above equations can be found in [6]. The above equations indicate that mutual information also represents reduction of entropy (uncertainty) of one variable given information of another.

Theorem: Given two items i and j, as well as distributions of votes on them, $P(V_i)$ and $P(V_j)$. Let e be the interval of discrete scores. If user A and user B are two arbitrary users who have voted for both items, then

$$\frac{d\left[p(|v_{j,u_A}-v_{j,u_B}|<e \,\big|\, |v_{i,u_A}-v_{i,u_B}|<e)\right]}{d\left[MI(V_j;V_i)\right]}>0 \tag{3.2.6}$$

Proof:

Since $P(V_i)$ and $P(V_j)$ are given, we have:

$$\begin{aligned} d\left[MI(V_j;V_i)\right] &= d\left[H(V_j)-H(V_j|V_i)\right] \\ &= -d\left[H(V_j|V_i)\right] \end{aligned} \tag{3.2.7}$$

Inequation (3.2.6) can be rewritten as:

$$\frac{d\left[p(|v_{j,u_A}-v_{j,u_B}|<e \,\big|\, |v_{i,u_A}-v_{i,u_B}|<e)\right]}{d\left[H(V_j|V_i)\right]}<0 \tag{3.2.8}$$

Next, we have

$$H(V_j\mid V_i)=\sum_{v\in\aleph} p(V_i\equiv v)H(V_j\mid V_i\equiv v) \tag{3.2.9}$$

and

$$\begin{aligned} &p(|v_{j,u_A}-v_{j,u_B}|<e \,\big|\, |v_{i,u_A}-v_{i,u_B}|<e) \\ &=\frac{\sum_{v\in\aleph}p(V_i\equiv v)^2\,p\left(\left|v_{j,u_A}-v_{j,u_B}\right|<e\,|\,v_{i,u_A}=v_{i,u_B}=v\right)}{\sum_{v\in\aleph}p(V_i\equiv v)^2} \end{aligned} \tag{3.2.10}$$

where $\aleph$ is the set of all discrete scale of votes. From equation (3.2.9) and equation (3.2.10) we can easily derive inequality (3.2.8) (Detailed steps are skipped here). Therefore, inequality (3.2.6) holds. •

The above theorem clearly shows that mutual information is consistent with the relevance measure between items presented by expression (3.2.1). Therefore, we use the following equation to estimate the mutual information as a dependency measure between two items:

$$MI(V_j;V_t)=H(V_j)+H(V_t)-H(V_j,V_t) \tag{3.2.11}$$

where V_j and V_t are the votes on item j and target item t respectively, and $H(V_j,V_t)$ is the joint entropy between two items. Since not all the users have voted for the two items, calculation is done over the overlap. If the average number of overlapping users between two items is n, and there are totally m items in training database, the computational complexity for mutual information between all pairs of items is $O(nm^2)$.

3.3 Relevance of Instances

In the last section, we defined the feature dependency based on information theory. In this section, we continue to study the relevance of instances in an information theoretical framework. We want to answer the question: Given an instance, has it been described adequately and effectively by the features?

The basic idea is, for an instance with its features and target concept, if the features can't provide enough information to explain why the target concept has the labeled value, then the instance will be not useful in aiding supervised learning algorithms to search the hypothesis space. The following definitions are necessary to introduce our method.

Definition 3.3.1 (Description of instance) If an instance I with a labeled target concept C is described by a set of feature-value pairs A, then A is called the description of instance I, denoted by $D(I)$. If another set of feature-value pairs A_1 is a subset of A, then A_1 is called a sub-description of I, $D_1(I)$, the relation between two descriptions is denoted by $D_1(I) \subseteq D(I)$.

According to the above definition, different descriptions are distinguished not only by their features (description spaces) but also by the values of the features (description regions). But in this paper, our algorithm only considers description spaces. That means, two descriptions are judged different only when they are represented by different feature sets. One reason is that the customer transaction databases always have a large part of missing values (e.g. up to 97% in EachMovie data set) and hence each customer has a different voted item list. However, we believe our work can be extended to taking into account description regions in the near future. The extension will be necessary for us to distinguish descriptions for the data set that does not have any missing values.

Definition 3.3.2 (Rationality of description for instance) Given an instance I represented by its description $D(I)$ and a labeled target concept C. If the entropy $H(C)$ is the a-pirori uncertainty of the target concept C when the value of C is assumed to be unknown, then the rationality of the instance I with the description $D(I)$ is the uncertainty reduction of C given knowledge of the description $D(I)$. Its value is calculated by

$$R_{I,C,D} = H(C) - H(C|D(I)) \tag{3.3.1}$$

The definition of rationality expresses how sufficient a description is to represent an instance with a labeled target concept. In the extreme case, if the uncertainty of the target concept is reduced to zero, the given description is completely sufficient. For convenience, if not specified, $D(I)$ is replaced by D in the rest parts of this paper, and the rationality of the description $D(I)$ for the instance I is simplified by the rationality of the instance I or the rationality of the description D with no difference.

Theorem 3.3.1 The rationality of an instance has the following properties:

Non negative: $\forall D, R_{I,C,D} \geq 0$

Irrelevant description: if D has no effect on C, then $R_{I,C,D} = 0$

Monotonicity: if $D_1 \subseteq D_2$, then $R_{I,C,D_1} < R_{I,C,D_2}$

Theorem 3.3.2 For an instance I, if its description D is a combination of n independent feature-value pairs (or sub-descriptions), $D_1, D_2, \ldots D_n$, then the following formula about the rationality of the instance I in these descriptions holds:

$$R_{I,C,D} = \sum_j^n R_{I,C,D_j} = \sum_j^n MI(C, D_j) \tag{3.3.2}$$

Proof : At first we proof the formula for the case of $n = 2$:

$$\begin{aligned}
&R_{I,C,D} = H(C) - H(C|D_1, D_2) \\
&= \sum_c \sum_{d_1} \sum_{d_2} p(c, d_1, d_2) \log\left(\frac{p(c, d_1, d_2)}{p(c)\, p(d_1, d_2)}\right) = \sum_c \sum_{d_1} \sum_{d_2} p(c, d_1, d_2) \log\left(\frac{p(c, d_1)\, p(c, d_2)}{p(c)^2\, p(d_1)\, p(d_2)}\right) \\
&= \sum_c \sum_{d_1} \sum_{d_2} p(c, d_1)\, p(d_2|c) \log\left(\frac{p(c, d_1)}{p(c)\, p(d_1)}\right) + \sum_c \sum_{d_1} \sum_{d_2} p(c, d_2)\, p(d_1|c) \log\left(\frac{p(c, d_2)}{p(c)\, p(d_2)}\right) \\
&= \sum_c \left[\sum_{d_2} p(d_2|c) \sum_{d_1} p(c, d_1) \log\left(\frac{p(c, d_1)}{p(c)\, p(d_1)}\right)\right] + \sum_c \left[\sum_{d_1} p(d_1|c) \sum_{d_2} p(c, d_2) \log\left(\frac{p(c, d_2)}{p(c)\, p(d_2)}\right)\right] \\
&= H(C) - H(C|D_1) + H(C) - H(C|D_2) \\
&= R_{I,C,D_1} + R_{I,C,D_2} = MI(C, D_1) + MI(C, D_2)
\end{aligned}$$

Let us assume that formula (3.3.2) holds for the case of $n = k$. We can then easily prove that (3.3.2) also holds for the case of $n=k+1$ in a similar way as for the case of $n = 2$. Consequently, (3.3.2) holds for all n. •

Theorem 3.3.1 shows that adding new features to a description doesn't decrease the rationality of an instance, no matter how relevant those features are. It also indicates that the more features we have, the more rationally we can explain why an instance has its label. Theorem 3.3.2 provides a way to calculate the rationality under the assumption that all the features are assumed to be independent. This assumption has been widely adopted in many articles on relevant feature selection [4]. However, does a large number of features really mean a better description? The answer is "No". For instance, suppose we already have a good feature A_1 to classify all the instances by using nearest neighbor classification method, if we introduce another irrelevant feature A_2, the distances between the instances can be biased by A_2. In such a situation, the performance of nearest neighbor classification will be degraded. Thus besides the sufficiency of descriptions, we also should consider the effectivity issue.

Definition 3.3.3 (Strength of description for an instance) For an instance I, if its description D is represented by a combination of n independent single-feature descriptions, $D_1, D_2, \ldots D_n$, the strength of description D for instance I is defined by

$$S_{I,C,D} = \frac{1}{n} \sum_j^n R_{I,C,D_j}$$

Definition 3.3.4 (Strong descriptions) For two different descriptions D_1 and D_2 for an instance I, the description D_1 is stronger than the other description D_2, if and only if $S_{I,C,D_1} > S_{I,C,D_2}$.

Definition 3.3.5 (Strong instances) For two instances I_1 and I_2 described by two different descriptions $D(I_1)$ and $D(I_2)$ respectively, the instances I_1 is stronger than the other instance I_2 if and only if $S_{I_1,C,D(I_1)} > S_{I_2,C,D(I_2)}$.

Based on the above definitions and theorems, we can interpret two widely studied topics in the machine learning community, feature selection and instance selection:

- Feature selection: Given a training data set in which each instance is described by the same set of features, if we don't distinguish the descriptions by their feature values, all the instances have the same description. Therefore, all combinations of the features form a space of the sub-descriptions. The task of the feature selection is to search the description space for a minimum sub-description which holds the strongest strength of the description while (approximately) has enough rationality for the instances. Although the presence of many irrelevant features does not decrease the rationality of the description, it can significantly weaken the strength of the description.

- Instance selection: Given a training data set in which each instance is described by a different description, the problem of the instance selection is to select the relevant instances which are strong instances with enough rationality by comparing the strength of the descriptions between instances. As indicated before, since an irrelevant feature can degrade the performance of learning, a weak instance with many irrelevant features is not as useful as a strong instance. More specifically, the target concept of a weak instance is not adequately described. Therefore, we should remove the weak instances from the training data set.

3.4 Proposed Algorithm

Our goal is to predict the active user's vote on a given target item based on the other customers' preference for the target item. Now we would like to evaluate every training customer's relevance to the task. Given a training customer i and his rated item set D_i as well as the corresponding votes, according to Theorem 3.3.2 and Definition 3.3.3, the customer's rationality and strength w.r.t. the target item t are:

$$R_{i,V_t,D_i} = \sum_{j \in D_i, j \neq t} MI(V_j, V_t) \tag{3.4.1}$$

$$S_{i,V_t,D_i} = \frac{\sum_{j \in D_i, j \neq t} MI(V_j, V_t)}{|D_i| - 1} \tag{3.4.2}$$

where V_t is the vote on item t. As mentioned before, we should select customers with enough rationality first, and from the selected customers we pick out relatively strong ones. However, since every training customer has voted on a fairly large number of items, if someone's rationality is low, he could not have high strength of the description. Thus, we need only select customers based on the strength of the descriptions. As a result of the instance selection, besides the original user preference database (which is unchanged) we maintain another reduced database containing only selected relevant customers for every target item. Moreover, for further improvement of accuracy, we also apply mutual information based feature weighting to modify the correlation function eq. (2.2) [15]. Our algorithm for customer selection and prediction proceedes as follows:

Based on the training database, estimate the mutual information between votes on different items.

For each possible target item, compute the strength of the description for all the customers who have voted on the target item, then sort them in descending order of the strength and select the top customers from the list according to a sampling rate r. For each target item, we create an index table of selected training customers.

In the prediction phase, given the target item and an input customer, calculate the correlation between input customer and every selected training customer, and then use the weighted average of the training customers' votes on the target item as the result.

If we have n customers and m items in the original training data set, the computational complexity of training phase (step 1 and 2) is $O(n \cdot m^2)+O(n \cdot m)+O(n \cdot \log n)$. With a sampling rate r, the prediction phase has a speedup factor of 1/r.

4. EXPERIMENTAL RESULTS

In this section, we report the results of the experimental evaluation of our novel algorithm. We describe the data set used, the experimental methodology as well as the performance improvement compared with the memory-based collaborative filtering algorithms [5; 9; 12].

4.1 The EachMovie Database

We used the data set from the *EachMovie* collaborative filtering service which was part of a research project at the Systems Research Center of Digital Equipment Corporation[1]. The database contains ratings from 72,916 users on 1,623 movies. User ratings were recorded on a numeric six-point scale.

Although data from 72,916 users is available, we restrict our analysis to the 35,527 users who voted on at least 20 out of the 1623 movies. Users who voted on fewer than 20 movies do not have a clear profile and are not suitable to be used in the evaluation. Moreover, to speed up our experiments, we randomly selected 10,000 from the 35,527 users and divided them into a training set (8000 users) and a test set (2000 users).

4.2 Metrics and Methodology

Since we are interested in a system that can accurately predict a user's vote on a specific item, we use the mean absolute error (MAE), where the error is the difference between the actual vote and the predicted vote, to evaluate the performance of our algorithm. This metric has been widely used in the literature [5; 7; 9; 12].

We use the protocol of *All but One* [5] in our experiments. We randomly hide an existing vote for each test user and try to predict this vote given all the other votes the user has voted on. The All but One protocol measures the algorithms' performance when given as much data as possible from each test user and is indicative of what might be expected of the algorithm under steady state usage where the database has accumulated a fair amount of data about a particular user.

4.3 A Case Study

In this section, we study the performance of the proposed algorithm for the target movie *Dances with Wolves*. In the training set with 8000 customers there are 6474 ones who have voted on the target movie. We would like to select a subset of customers from the 6474 customers and to form a new training set. Then,

[1] For more information see http://www.research.digital.com/SRC/EachMovie/.

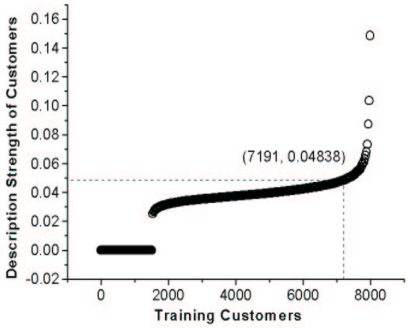

Figure 2. Training customers' description strengths for votes on *Dances with Wolves* sorted in ascending order.

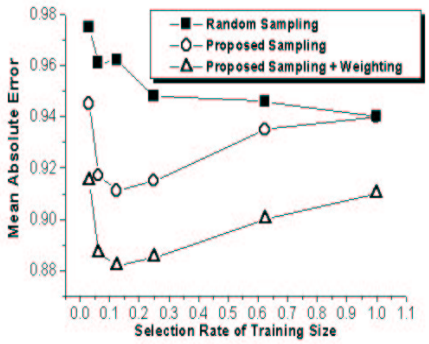

Figure 3. Prediction accuracy of training customer selection, the target movie is fixed to be *Dances with Wolves*

based on the selected training set, we evaluate our collaborative filtering algorithm using the 1618 customers from the test set who have voted on the movie *Dances with Wolves*. In the test phase, we assume the 1618 customers' votes on the target movie to be unknown. In our experiments, the mutual information between movies is calculated based on the whole training set of 8000 users.

As shown in Fig.2, we calculate every training customer's description strength and sort them in ascending order. There are 1526 users with zero description strength which have no vote on the target movie. After these customers, the description strengths are slightly increasing until about 7000 customers, where the description strength starts to dramatically increase. We select the top customers with different selection rates: 3.125%, 6.25%, 12.5%, 25%, 62.5% and 100%. We evaluate our proposed algorithm on the 6 training sets. We compare our algorithm with random sampling applying the same selection rates. Except for the cases of 62.5% and 100%, the results are averaged over 4 runs. Moreover, in order to further improve the prediction accuracy we applied mutual information as the feature weighting method. The results are depicted in Fig.3. The mean average error (MAE) is averaged over the 1618 testing customers who have cast a vote on the target movie. As shown in Fig.3, the proposed instance selection method outperforms random sampling w.r.t. accuracy. With the selection rate decreasing from 100% to 12.5%, the accuracy is getting better and better, and reaches an optimum at the selection rate of 12.5%. Note that the accuracy of collaborative filtering without any instance selection is equal to the accuracy of collaborative filtering with a selection rate of 100%. When the selection rate further decreases, the mean absolute error begins to increase. But the performance of the selection rate 6.25% is still comparable with collaborative filtering without any instance selection. Furthermore, we observe that mutual information based feature weighting can further improve the accuracy by about 4% with respect to MAE. The most interesting point about Fig.2 and Fig.3 is the existence of an optimal selection rate of 12.5%: the customer labeled by (7191, 0.04838), who is the lower bound of the description strength for the optimal training subset, is almost just the start point from which the customers' description strength begins to dramatically increase.

4.4 Overall Performance

In this section, we evaluate the overall performance of our method of selecting relevant customers for any possible movies. The results are given in Fig. 4 and Fig. 5. As described in section 3.4, we sort users in descending order of their description strengths for each target movie, and select the top users for the prediction using different selection rates of 3.125%, 6.25%, 12.5%, 25%, 62.5% and 100%. Since different target movies have different size of training sets, a small selection rate may result in too few training customers for some target movie. So we set a minimum number of training users to be 200. For each test customer, we randomly select a voted movie as the target movie and predict its vote. The results are compared with the outcomes of random sampling. It is obvious that our method outperforms random sampling w.r.t. accuracy, and the combination with feature weighting results in a further 4~5% improvement. As shown in Figure 5, the computational complexity is linear to the number of users in the training data set. If we set the selection rate to be 3.125%, the average prediction time for each vote is decreased from 222 ms to 22 ms, corresponding to a speed-up factor of 10, and accuracy is improved by a factor of 5%, while random sampling degrades the accuracy by a factor of 6%. More interestingly, from Figure 4, we observe that the accuracy is constantly getting better when the training size getting smaller until reaches a point which is 12.5%. This result clearly shows the customers with weak description strength can deteriorate the memory-based collaborative filtering algorithm.

5. CONCLUSIONS

In this paper, we present an information theoretical framework to explain the intimate relationship between feature selection and instance selection for collaborative filtering, and propose the description rationality and the description strength to measure the relevance of an instance with respect to a target concept. Based on this work, we propose a novel instance selection method to reduce the computational complexity of memory-based collaborative filtering algorithm and possibly improve its accuracy. We conduct an empirical evaluation of our proposed algorithm on the well known EachMovie database. The results show that our method can significantly reduce the size of the training data set and speed up the collaborative filtering algorithm. Furthermore, our method even achieves an improvement on the accuracy using a rather small training set: for example, in our overall evaluation the best accuracy is obtained for the selection rate of 6.25%.

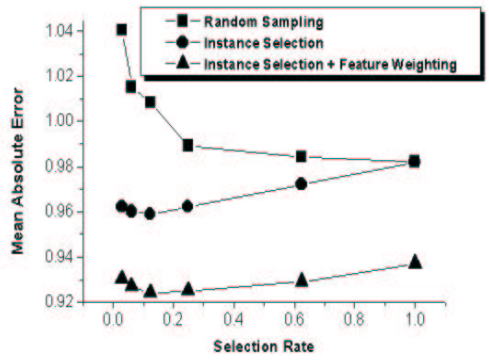

Figure 4. Prediction accuracy of training customer selection, target movie for each test customers is randomly selected

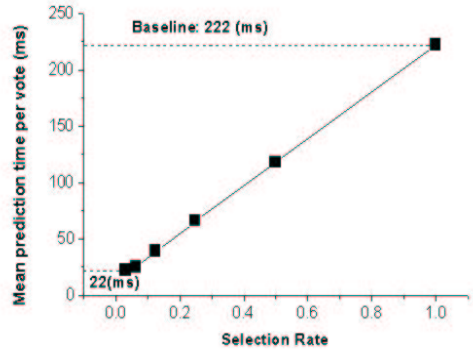

Figure 5. Prediction time for different selection rates

Our work provides a novel study on the problem of instance selection, which could be potentially useful in other data mining applications, especially for business transaction databases with a very large number of missing values. However, there are still two important questions on which we are going to work:

- As shown in Fig.2 and Fig.3, the observation about the optimal selection rate is very interesting. Does there exist any reason for that? And how to determine the optimal selection rate automatically?

- So far, our work has been targeted towards databases where every instance has a large number of missing feature values. Can we extend the proposed method to deal with cases where the instances have no missing values?

6. ACKNOLEDGEMENT

We would like to thank the System Research Center of Digital Equipment Corporation for making the EachMovie database available for research.

7. REFERENCES

[1] D. W. Aha, "Lazy Learning", *Artificial Inteligence Review*, 11: 7-10, 1997.

[2] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based Learning Algorithms", *Machine Learning*, 6: 37-66, 1991.

[3] D. Billsus and M. J. Pazzani, "Learning Collaborative Information Filters", In *Proceedings of the International Conference on Machine Learning*, 1998.

[4] A. L. Blum and P. Langley, "Selection of Relevant Features and Examples in Machine Learning", Artificial Intelligence, 97:245-272, 1997.

[5] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering", In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998.

[6] G. Deco, and D. Obradovic, *An Information-Theoretic Approach to Neural Computing*, Spinger-Verlag Inc., New York, 1996.

[7] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering", in *Proceedings of the Conference on Research and Development in Information Retrieval*, 1999.

[8] S. Pradhan and X. Wu, "Instance Selection in Data Mining", Technical Report, 1999.

[9] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews", In *Proceedings of the 1994 Computer Supported Collaborative Work Conference.*

[10] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Analysis of Recommender Algorithms for E-Commerce", In *Proceedings of ACM E-Commerce 2000 Conference*.

[11] C. E. Shannon, "A Mathematical Theory of Communication", *Bell Sys. Tech. Journal*, vol. 27, 1948

[12] U. Shardanand, and P. Maes, "Social Information filtering Algorithms for Automating 'Word of Mouth'", In *Proceedings of CHI'95*.

[13] D. Wettschereck, D. W. Aha and T. Mohri, "A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms", *Artificial Intelligence Review*, 11: 273-314, 1997.

[14] D. R. Wilson and T. R. Martinez, "Reduction Techniques for Instance-Based Learning Algorithms", *Machine Learning*, 38-3, pp. 257-286, 2000.

[15] K. Yu, Z. Wen, X. Xu and M. Ester, "Feature Weighting and Instance Selection for Collaborative Filtering", *2nd International Workshop on Management of Information on the Web*, in conjunction with the *12th International Conference on DEXA'2001*, Munich, Germany, 2001.

[16] J. Zhang, "Selecting Typical Instances in Instance-Based Learning", in *Proceedings of the Ninth International Conference on Machine Learning*, Aberdeen, Scotland: pp. 470-479,1992.