Bootstrapping Recommender Systems with the Crowdsourcing

SEMESTER PROJECT REPORT

SUBMITTED BY
Cheng-Hsiang Chiu

SUPERVISED BY
Goran Radanovic

June 2014

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

TABLE OF CONTENTS

1. Introduction

Recommender system is an information filtering technique aiming at predicting meaningful information to users, which is widely implemented for the purpose of suggesting books to consumers in Amazon, movies in Netflix , music in Pandora and so on [1]. The system recommends items to users based on a prediction of users' profiles. Nowadays, recommender system faces a cold start problem which would jeopardize the precision of the prediction. Cold start problem arises from the situation in which there is not enough information to make a good recommendation. In this paper, by taking advantages of crowdsourcing we propose a technique to mitigate the consequences caused by the cold start problem.

The report is structured as follows. In Section 2 we briefly introduce background of recommender system and the approaches behind it. Then a related work is also given in this section. After that, the proposed technique is presented in Section 3. Next, Section 4 would cover the experimental results. Finally, conclusions and reference are noted.

2. Background and Related Work

We first give a brief introduction to recommender system and collaborative filtering techniques before describing the proposed idea.

2.1 Recommender System

There are two main approaches to construct the recomender system : content-based filtering and collaborative filtering [2]. Content-based filtering method is based on item profile which is a record or collection of records that describes the characteristics of that item. The item

profile of a movie could be the actors, the director, the genre, etc. Items which are featured with the identical profiles as a certain item would be recommended to users who are fond of the specific item.

The other method is collaborative filtering technique, which has two implementations. The necessity of processing users' preferences instead of analyzing item profiles explains how the term "collaborative" comes from. More details about collaborative filtering method would be given in the next section.

In recommender systems, researchers usually transform a data set into a utility matrix, shown in Figure 1 in which columns are denoted as the ratings of one item rated by some users and row are described as the rating of a user given to some items.

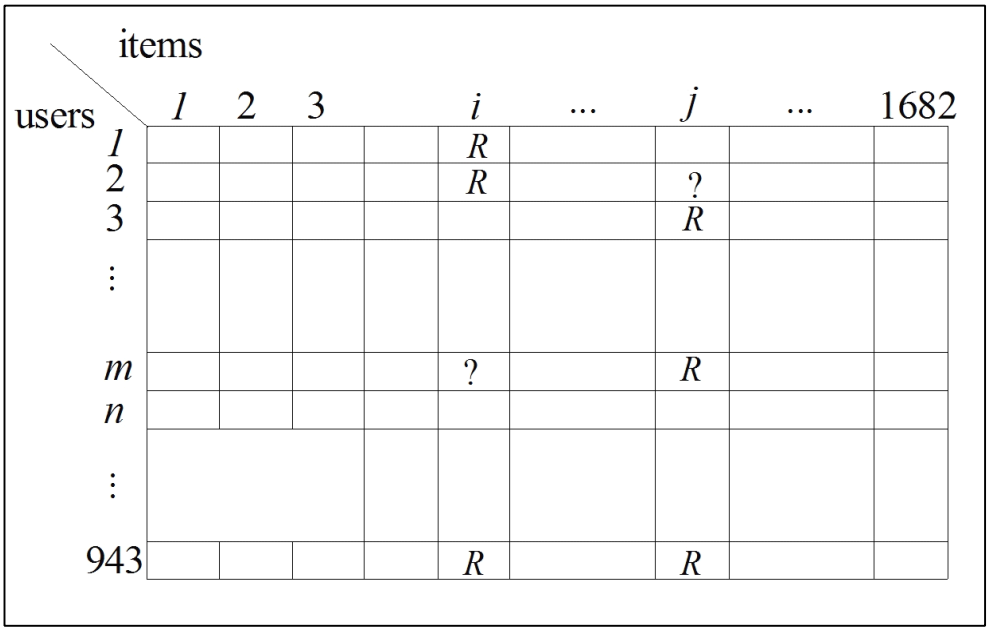

Figure 1 : A utility matrix with columns which are ratings of an item and rows are ratings which are given by a user over some items.

2.2 Collaborative Filtering

Collaborative filtering method recommends items on the basis of similarity distance between users and/or items. While two users have highly similar interests on movies, music, books, and etc., then there is a good reason to recommend some favorite items of one user to the other user, which is the theory of user-based collaborative filtering technique. In order to measure the similarity distance between two users, shown in Figure 2, there are two common approaches, Cosine distance and Pearson correlation, expressed in Equations (1) and (2) respectively. Cosine distance between user m and user n is defined as

$$sim(m,n) = \frac{\sum_{i \in S_m \cap S_n} r_{m,i} \times r_{n,i}}{\sqrt{\sum_{i \in S_m \cap S_n} r_{m,i}^2} \sqrt{\sum_{i \in S_m \cap S_n} r_{n,i}^2}}, \quad (1)$$

where S_m represents the set of items that user m has rated and $r_{m,i}$ denotes the rating of item i of user m [3][4][5].

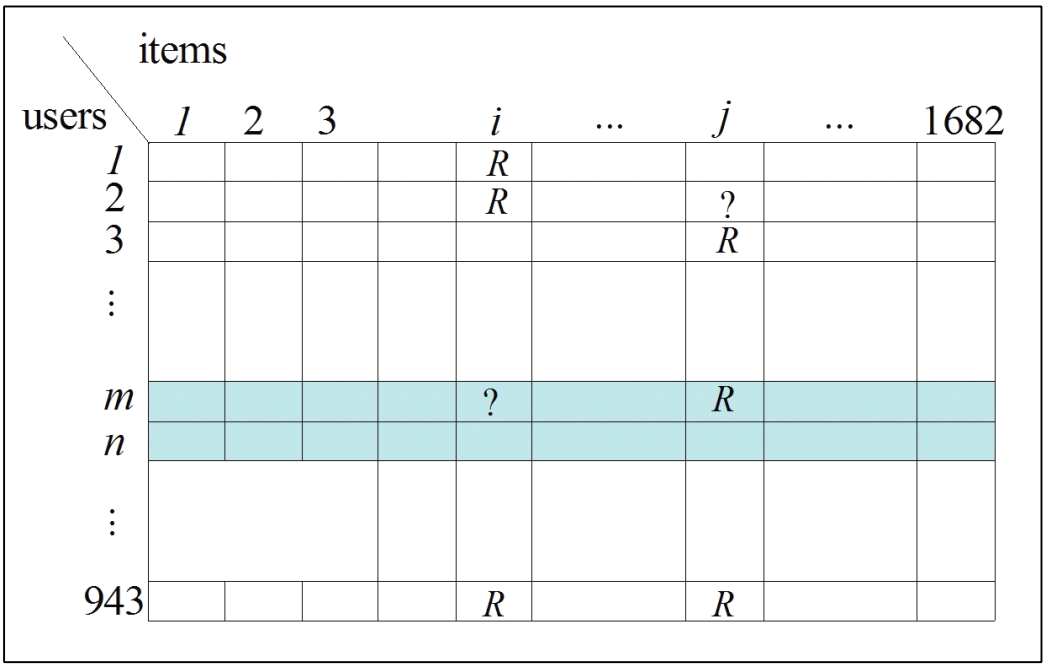

Figure 2 : A utility matrix where rows of user m and user n are marked in blue.

Pearson correlation is given in the following equation,

$$sim(m,n)=\frac{\sum_{i\in S_m\cap S_n}\left(r_{m,i}-\overline{r_m}\right)\left(r_{n,i}-\overline{r_n}\right)}{\sqrt{\sum_{i\in S_m\cap S_n}\left(r_{m,i}-\overline{r_m}\right)^2}\sqrt{\sum_{i\in S_m\cap S_n}\left(r_{n,i}-\overline{r_n}\right)^2}}, \tag{2}$$

where $\overline{r_m}$ stands for the average rating of user *m*. After the similarity distance is measured, the subsequent job is to make predictions and suggest the top ranked recommendations to users. The predicted item can be calculated as follows :

$$p_{m,i}=\overline{r_m}+\frac{\sum_{\{u\in U\mid i\in S_n\}}sim(m,n)\times(r_{n,i}-\overline{r_n})}{\sum_{\{u\in U\mid i\in S_n\}}\left|sim(m,n)\right|}. \tag{3}$$

Besides the user-based collaborative filtering approach, item-based collaborative filtering is the other technique. The idea behind user-based collaborative filtering is the same as user-based collaborative filtering. Recommender systems suggest some similar items to users according to the similarity matrix between items and items, depicted in Figure 3.

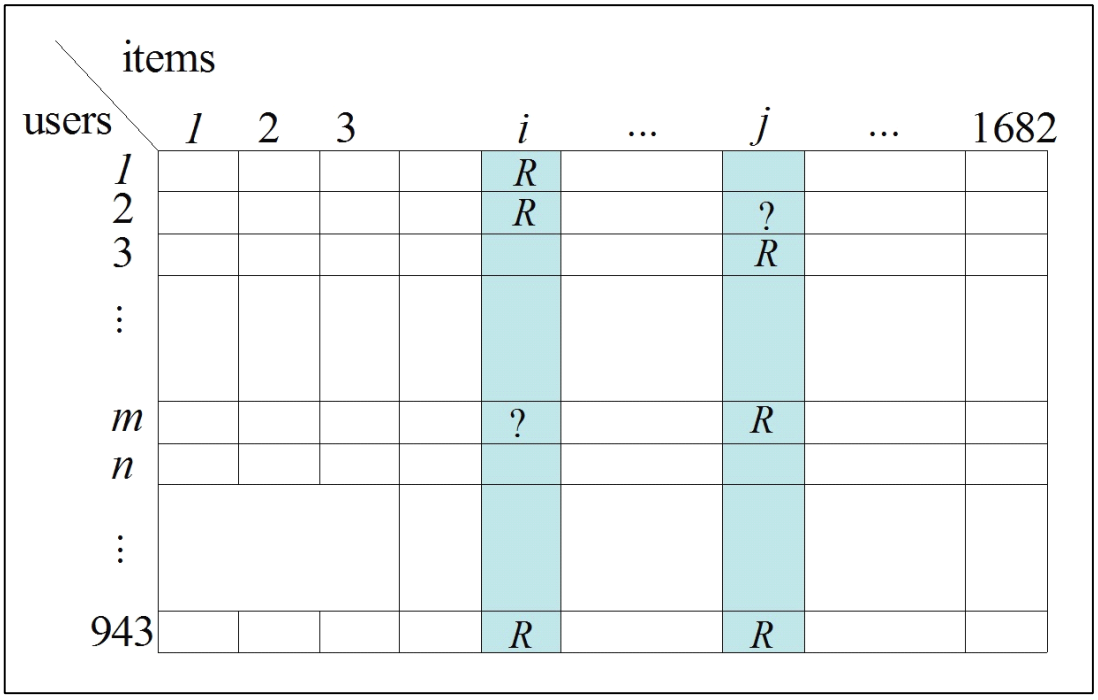

Figure 3 : A utility matrix in which columns of item *i* and item *j* are marked in blue.

The similarity distance of item-based approach between item *i* and item *j* is formulated by using either Cosine distance, shown in equation (4), or Pearson correlation, expressed in (5).

$$sim(i,j)=\frac{\sum_{\{u\in U|i\in S_u \& j\in S_u\}} r_{u,i}\times r_{u,j}}{\sqrt{\sum_{\{u\in U|i\in S_u \& j\in S_u\}} r_{u,i}^2}\sqrt{\sum_{\{u\in U|i\in S_u \& j\in S_u\}} r_{u,j}^2}}, \tag{4}$$

and

$$sim(i,j)=\frac{\sum_{\{u\in U|i\in S_u \& j\in S_u\}} \left(r_{u,i}-\overline{r_u}\right)\left(r_{u,j}-\overline{r_u}\right)}{\sqrt{\sum_{\{u\in U|i\in S_u \& j\in S_u\}} \left(r_{u,i}-\overline{r_u}\right)^2}\sqrt{\sum_{\{u\in U|i\in S_u \& j\in S_u\}} \left(r_{u,j}-\overline{r_u}\right)^2}}, \tag{5}$$

where $r_{u,i}$ is described as the rating of user *u* on item *i* and $\overline{r_u}$ as the average rating that user *u* has rated. And the prediction of item is shown in the following equation,

$$p_{m,i}=\frac{\sum_{\{j\in S_m|j\neq i\}} sim(i,j)\times r_{m,j}}{\sum_{\{j\in S_m|j\neq i\}} |sim(i,j)|}. \tag{6}$$

After the prediction is finished, we have to evaluate the quality of recommender systems. Generally, there are two common-used performance matrics. They are Mean Absolute Errorr (MAE) and Root Mean Square Error (RMSE). MAE and RMSE are formulated in (7) and (8) respectively :

$$MAE=\frac{1}{|T|}\sum_{(u,i,r)\in T}|p_{u,i}-r|, \tag{7}$$

and

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,i,r)\in T}\left(p_{u,i} - r\right)^2}\ , \tag{8}$$

where the set of (user, item, rating) triplets is represented as T.

2.3 Related Work

In the paper [6] "Alleviating the Sparsity in Collaborative Filtering using Crowdsourcing," a technique is proposed to thwart the cold start problem by the help of crowdsourcing. The approach is to expand the dimension of users, that is, adding the extra workers' ratings to the original utility matrix. Then, the current utility matrix turns out to be the one shown in Figure 4. According to the experimental results revealed in the paper, quality of the recommender system improves. More informations indeed bolsters the precision of recommender systems. But the approach is implicit rather than explicit, because it adds new users hoping that they are similar to the existing user. In crowdsourcing, however, there is a concern regarding the existence of sloppy workers and spammers. Moreover, there is no ground truth of these workers. Therefore, the above reasons lead us to explicit strategy where we fill in the blank elements directly using crowdsourcing.

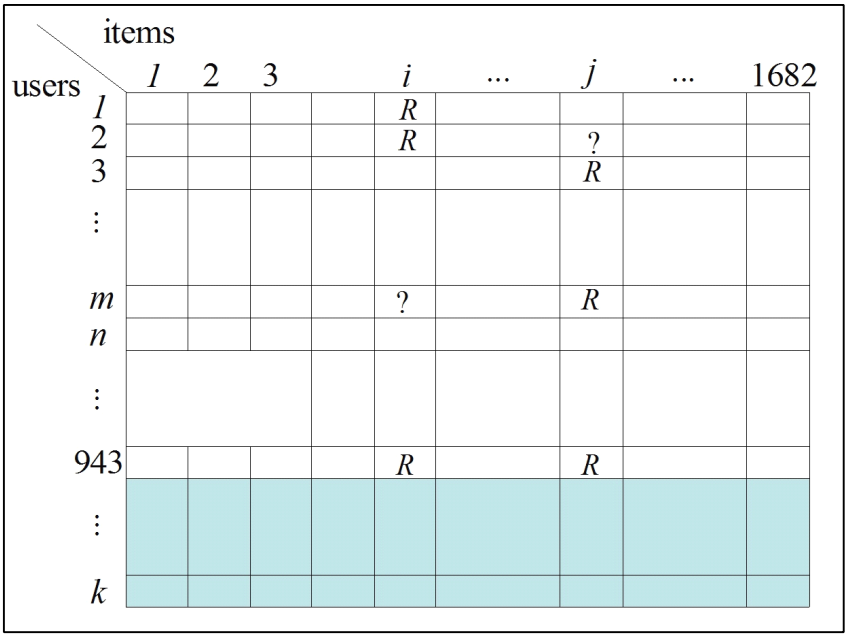

Figure 4 : An extended utility matrix where extra rows marked in blue are obtained from the crowdsourcing.

3. The Proposed Technique

The quality of the recommender system is highly compromised because of existence of the cold start problem. To mitigate the effect caused by the problem, the utility matrix should be as less sparse as possible. Therefore, we propose an approach that tries to fill in some blank elements by taking the advantage of crowdsourcing. With the help of crowdsourcing, the recommender system is able to suggest items to users with more precisions.

3.1 System Overview

The following figure is the overall system which has two subparts. The left flow in Figure 5 has the purpose of checking the quality of the recommender system by the use of performance matrix introduced in equations (7) and (8). The right flow focuses on gathering data with crowdsourcing.

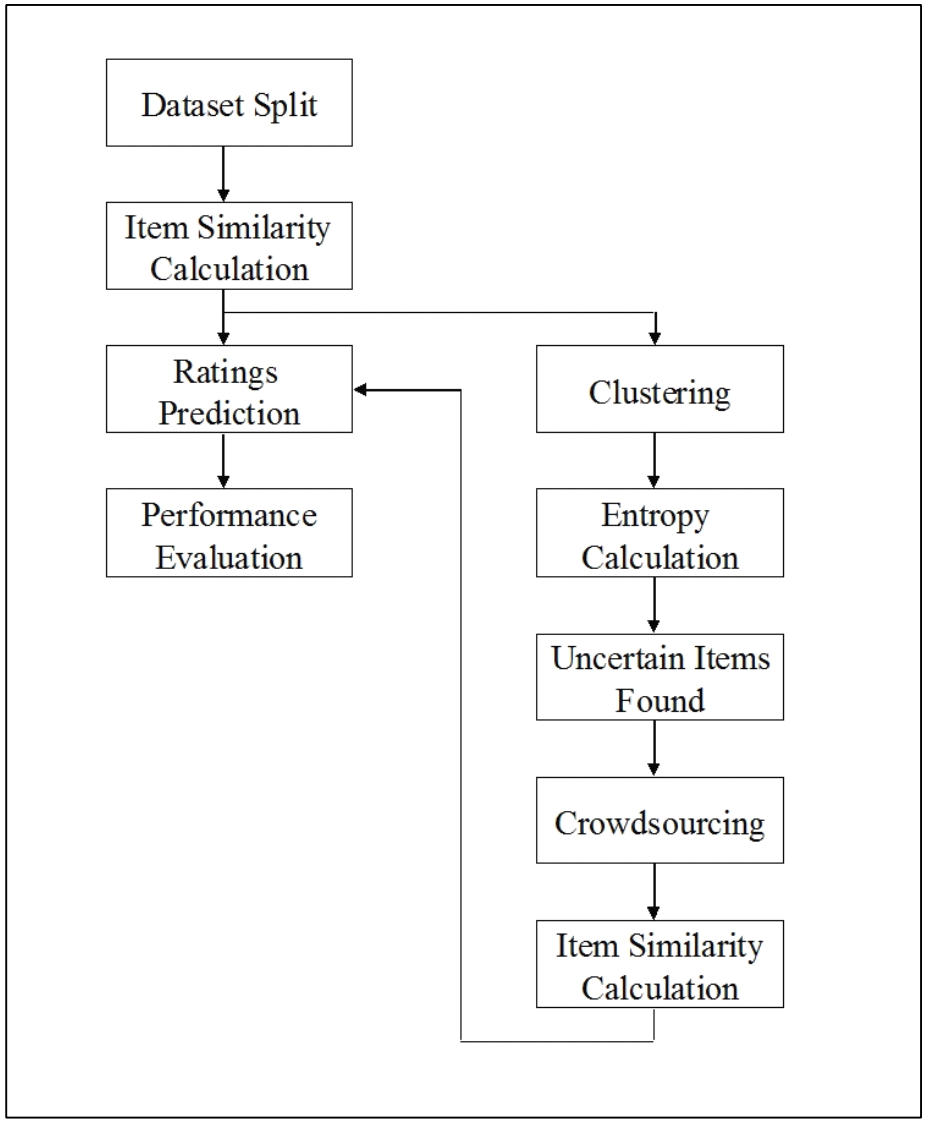

Figure 5 : System Flowchart

3.2 The Left Flow - Performance Evaluation

The left flow is basically the job of evaluating performance of all the recommender system and works in the following ways :

1. Measure the item-item similarity distances by the help of equation (5).

2. Predict the ratings which appears in the testing set only by using formula (6) and the similarity distances obtained in the first step.

3. Evaluate the performance matrics, RMSE and MAE, over the testing data set and the predicted ratings with equations (7) and (8).

After finishing the above three steps, we can tell whether the recommender system is performed well or bad by comparing with the results of different approaches.

3.3 The Right Flow - Crowdsourcing

The proposed technique is implemented in the right flow and is consisted of seven steps :

1. Measure the item-item similarity matrix with equation (5).

2. Implement K-means clustering algorithm on the training set to put similar items together.

3. Calculate entropy of each item with respect to all the centroids. We wish to attain good learning performance without demanding too many items. Hence, entropy, one of the common-used approaches in active learning, is utilized to label the most uncertain items [7]. Entropy of item j is formulated in the following :

$$Entropy(j) = -\sum_{\{x \in centroids\}} \left| p_{j,x} \times \log p_{j,x} \right|, \tag{9}$$

where
$$p_{j,x} = \frac{\left| \vec{j} - \vec{x} \right|}{\sum_{\{y \in centroids\}} \left| \vec{j} - \vec{y} \right|} \tag{10}$$

and $\vec{j} = \{sim(j,k), k = 1,2,\cdots,1682\}$ denotes a vector of item j with respect to all the items.

4. Pick some items which are top-ranked in descending entropy. Since the higher entropy value an item has, the higher uncertainty it possesses. The top uncertain items are the red dots demonstrating in Figure 6.

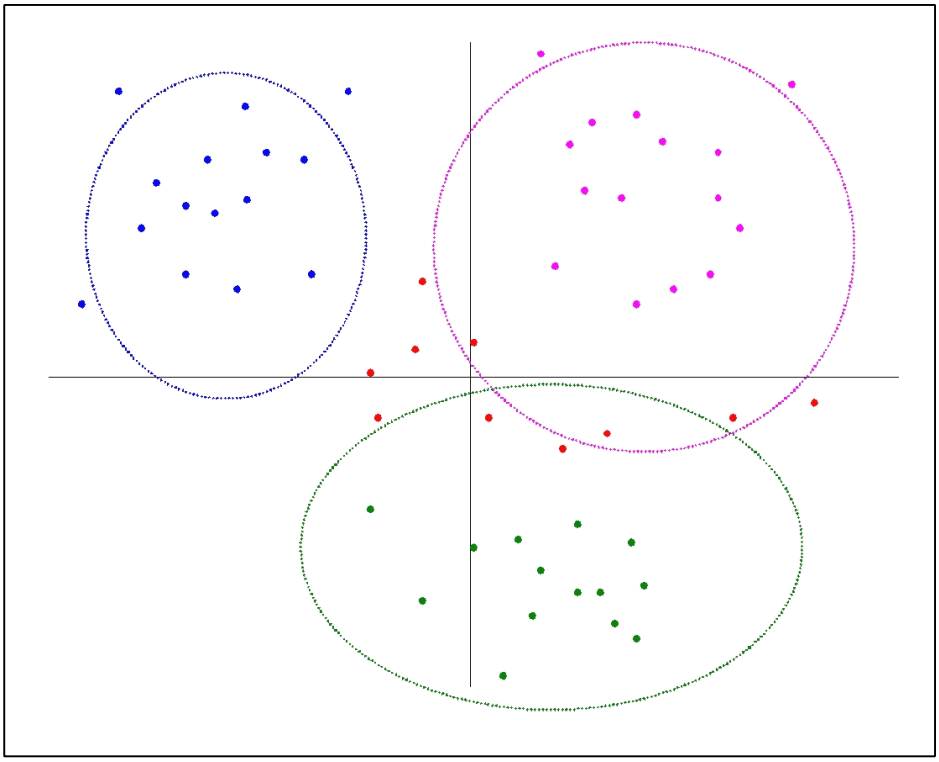

Figure 6 : An demonstration in which items are clustered. The most uncertain items are those dots marked in red.

5. Post the built web page on the Amazon Mechanical Turk for workers to rate the uncertain items. What we want to do is to add more ratings in the original utility matrix. Hence, the workers who are hired to help us rate items should pretend to a specific user. For example, in Figure 3 the rating of user *m* on item *i* is blank and item *i* is supposed to be a uncertain item. After referring to some known ratings of user *m* on items other than item *i*, workers should predict the rating of item *i* on behalf of user *m*; that is to say, workers have to figure out the preference profile of user *m* and give the most probable rating for item *i*.

6. Deploy a quality control mechanism in the web page to filter out sloppy workers or spammers. Since they rate items casually, the variance between the ratings they

give and the real ratings would be undoubtedly huge . Hence, the mechanism is to calculate a score based on the known ratings and predicted ratings and the score is given in the follwoing :

$$score = 200 \times (1 - \left| \frac{known\_rating - predicted\_rating}{4} \right|). \quad (11)$$

The score decreases as the variance increases. And the higher the score is, the more reliable the worker's predictions should be. Therefore, the predicted ratings are aggregated in the end with the following equation :

$$r_\alpha = \frac{\sum_{\beta \in W} R_\beta \times \frac{score(\beta)}{1000}}{\sum_{\beta \in W} \frac{score(\beta)}{1000}}, \quad (12)$$

where r_α is represented as the rating of uncertain item α , W={all workers who rated item α } and β is a worker. The idea of equation (12) is that the workers of the higher scores gain more dominant portion in the aggregation to the final predicted ratings.

7. Add the extra ratings into the original utility matrix and repeat the left flow depicted in Figure 5.

After these steps, the utilty matrix becomes more dense than the original matrix. And the predicted ratings will be more precise since more information are presented.

4. Experimental Results

Before presenting the experimental results, we explain the data set and how we process it.

4.1 Data Set

The data set is from MovieLens which provides data set of three different sizes, 100K, 1M and 10M. Not only ratings but users' profiles and items' profiles are all included in the data set. We used the set of size 100K in which there are 100,000 ratings from 943 users on 1682 movies. And the data set can be converted into a utility matrix with 943 rows and 1682 columns, as shown in Figure 1. Since the technique we adopt is item-based collaborative filtering method, only the ratings matter to us.

4.2 Preprocessing of Data Set

The data set is divided into two subsets, a training set of 80, 000 ratings and a testing set of 20,000 ratings. The item-based similarity distance is calculated with Equation (5). Since only 1,000 ratings were supposed to be obtained via crowdsourcing, the improvement caused by the additional 1,000 ratings will not be significant while the training set is consisted of 80,000 ratings. Hence, we had to choose a proper size of the training set. Therefore, we followed the steps introduced in Section 3.2 to calculate the performance metrics of various size of training set, which is presented in Table 1.

While examining Table 1, we found additional 1,000 ratings makes significantly greater improvement when the training set is composed of 10,000 ratings and, therefore, chose 10,000 as the size of training set. A mechanism to pick the 10,000 ratings is described in the following formula :

$$N_{10,000}(m) = 10{,}000 \times \frac{\sqrt{N_{80,000}(m)}}{\sum_{u=1}^{u=983} \sqrt{N_{80,000}(u)}}, \tag{13}$$

where $N_{80,000}(m)$ is denoted as the number of ratings user m has given in the training set of size 80,000. Implementation of (13) can assure all users make contributions to the 10,000 ratings.

Table 1 : Performance Metrics of different size of training set

	MAE	RMSE
80,000	0.8453750918	1.0616004687
20,000	0.88309703725	1.1309287844
20,000 + 1,000	0.87978583071	1.1280910406
10,000	0.89872629358	1.1631614243
10,000 + 1,000	0.89342260243	1.1527131221

4.3 Web Page for Amazon Mechanical Turk

After 10,000 ratings were picked, we followed the steps given in Section 3.3 to locate the top-250 uncertain items and constructed a web page for the crowdsourcing purposes. Each uncertain item has repeated 4 times averagely and each repetition is assigned to a user randomly, which amounts to the additional 1,000 ratings. The following figure is a snapshot of the constructed web page. In Figure 7, the movie located on the bottom is not rated and is one of the uncertain items. Workers predict the ratings of the movies based on users' preference profiles, which are consisted of movie trailers, movie descriptions and ratings of other four movies. And each worker rates 30 movies.

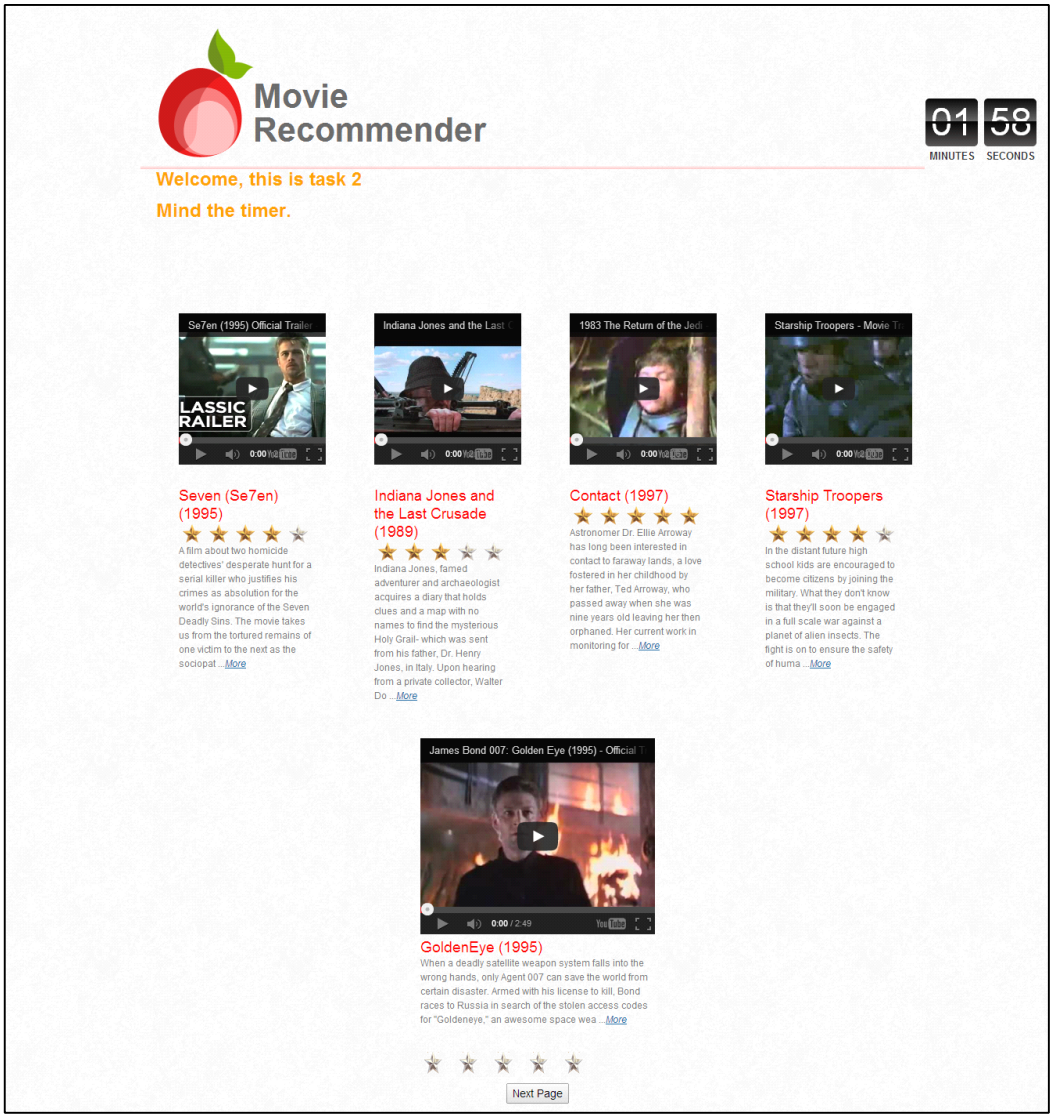

Figure 7 : A snapshot of the web page

The quality control mechanism, which is utilized in the web page, distributes five movies with known ratings into the 30 movies equally. After a worker rates six movies, his/her score is calculated. The perfect score of 30 movies is 1,000 and a bonus is given to the workers if score exceeds 750. The 1,000 predicted ratings are aggregated in the end with equation (12). After 1,000 predicted ratings are gathered and filled in the utility matrix, we can proceed to evaluate the performance by following the left flow in Figure 5. The experimental results are listed in Table 2. The number of workers involved in the experiment is 99. Then, 2,475 ratings, which is

99 workers times 25 unknown items per worker, were collected. Hence, every predicted element in the 1,000 additional ratings were aggregated 2.475 times averagely.

Table 2 : Comparison of performance metrics between the proposed technique (marked in red) and the original training data set

	MAE	RMSE
80,000	0.8453750918	1.0616004687
20,000	0.88309703725	1.1309287844
20,000 + 1,000	0.87978583071	1.1280910406
10,000	0.89872629358	1.1631614243
10,000 + 1,000	0.89342260243	1.1527131221
10,000 + 1,000	0.88992855791	1.1481747352

From Table 2, we can find out that MAE and RMSE of the proposed technique is very close to the results of same training data set, which means the proposed method possesses the ability to give a pseudo-rating on blank ratings precisely.

5. Conclusions and Future Works

In this paper, the technique is proposed to alleviate the cold start problem. We picked out the most uncertain items as it is often done in active learning, gathered ratings via a built web page, filtered out sloppy workers or spammers with a quality control mechanism. Finally, performance metrics were evaluated and compared. And the proposed technique by the help of

crowdsoucring was demonstrated to be an effective method to relief the consequence caused by the cold start problem.

In order to get bigger improvements in performance evaluation, some sophisticated methods have to be applied, for example, an adoption of both uncertain items and uncertain users. Besides, less than 3 aggregated times for each predicted rating could not magnify the effect of the aggregation of predicted ratings. More involved workers, say 200, which lead to averagely 5 aggregated times are expected to attain the desired results.

6. References

[1] Wikipedia, *Recommender system* [Online]. Available :

http://en.wikipedia.org/wiki/Recommender_system

[2] J. D. Ullman, *Recommendation Systems* [Online]. Available :

http://infolab.stanford.edu/~ullman/mmds/ch9.pdf

[3] L.Candillier, F.Meyer, and M. Boulle, "Comparing State-of-the-Art Collaborative Filtering Systems" in *5th International Conference, MLDM 2007*, Vol.4571, Leipzig, Germany, July 18-20, 2007, pp 548-562

[4] X. Su and T.M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques" in *Advances in Artificial Intelligence*, Vol. 2009, New York, USA, January 2009

[5] J. Lee, M. Sun, and G. Lebanon, *A Comparative Study of Collaborative Filtering Algorithms* [Online]. Available : http://arxiv.org/pdf/1205.3193.pdf

[6] J. Lee et al., "Alleviating the Sparsity in Collaborative Filtering using Crowdsourcing" in CrowdRec 2013: Workshop on Crowdsourcing and Human Computation for Recommender Systems.

[7] J. Zhu et al., "Active Learning With Sampling by Uncertainty and Density for Data Annotations" in IEEE Transactions on Audio, Speech, and Language Processing, Vol. 18, NO. 6, August 2010, pp1323-1331