A Reinforcement Learning Approach for Collaborative Filtering

Jungkyu Lee[1], Byonghwa Oh[2], Jihoon Yang[2], and Sungyong Park[2]

[1] Cyram Inc, Seoul, Korea
jklee@cyram.com
[2] Sogang University, Seoul, Korea
{mrfive,yangjh,parksy}@sogang.ac.kr

Abstract. In recent years, there has been much interest in recommender systems that provide users with personalized suggestions for products or services. In this paper we presents a reinforcement learning approach for collaborative filtering. First, we formalize the collaborative filtering problem as a Markov Decision Process. Then, we learn the connection between the time sequence of user ratings using Q-learning. Experiments demonstrate the feasibility of our approach and a tight relationship between the past and current ratings.

Keywords: recommender systems, markov decision process, Q-learning, singular value decomposition

1 Introduction

Recommender systems provide users with personalized suggestions for products or services. Recommender systems can be combined with various technologies including autonomous driving and make appropriate suggestions in different spatiotemporal situations. Collaborative Filtering (CF) is one of the most active domain in recommender systems since it avoids using information about the content, but rather it uses historical data (e.g. user ratings). The data of CF can be represented as a large sparse matrix with $(user, item, rating)$ entries, and the goal is to predict the missing entries of users's ratings for items they have not yet considered. Many researchers have developed CF algorithms including Restricted Boltzmann Machine (RBM)[8], K-Nearest Neighbor(KNN)[1] and Singular Value Decomposition (SVD)[5].

This paper presents a new CF algorithm based on reinforcement learning, RLCF. Reinforcement learning has been successful in many applications [3]. However, to the best of our knowledge, there has been no attempt to apply reinforcement learning to CF. The motivation for applying reinforcement learning to CF comes from the *contrast effect*, which is the enhancement or diminishment of a perception and related performance as a result of immediate previous or simultaneous exposure to the stimulus in the same dimension, with a smaller or greater value [10]. The contrast effect can take place in CF. For instance, a user can evaluate the same movie differently depending on what he watched

previously. Thus, the prediction of a user's ratings for an unseen movie can be optimized by taking into account the sequence of movies he has seen. Against this background, we attempt to formulate the contrast effect on CF as a Markov Decision Process (MDP) and apply one of the popular reinforcement learning algorithms, Q-learning [9, 11] to it.

2 Preliminaries

2.1 Problem Definition

Suppose the user-movie dataset consist of ratings (of 1∼5 stars) for N users and M movies. Let $X \in \mathbb{R}^{N \times M}$ be the matrix of the ratings. Let $A \in \mathbb{R}^{N \times M}$ be a matrix including entry $(user, movie, rating) \notin X$ as real (correct) ratings. The performance of a CF algorithm can be measured by the error between the prediction values of the algorithm and A. Let $I \in \{0,1\}^{N \times M}$ is an indicator function such that:

$$I_{ij} = \begin{cases} 1 \text{ if movie } j \text{ is rated by user } i \text{ in } A \\ 0 \text{ if the ratings is missing in } A \end{cases} \tag{1}$$

Let $P \in \mathbb{R}^{N \times M}$ be the matrix of predictions made by the CF algorithm. The goal is to estimate the missing entry of X such that the root mean squared error (RMSE) is minimal. RMSE is defined as follows:

$$RMSE(P, A) = \sqrt{\frac{\sum_{i=1}^{n} \sum_{j=1}^{m} I_{ij}(A_{ij} - P_{ij})^2}{\sum_{i=1}^{n} \sum_{j=1}^{m} I_{ij}}} \tag{2}$$

2.2 Problem Fomulation as MDP

MDP provides a mathematical framework for a sequential decision problem [2, 6], and has been used for a wide range of optimization problems. MDP consists of a tuple of 5 elements $(S, A, \{P_{sa}\}, \gamma, R)$ where

- **State S:** The state S is defined as the ratings of movies. If the range of ratings is integer 1 to K, the size of state $|S|$ is K. $s_t^{(i)} \in S$ refers to the rating of the t-th movie that user i watched.
- **Action A:** As defined above, $s_t^{(i)}$, and $s_{t+1}^{(i)}$ refer to the ratings of t-th movie and $(t+1)$-th that user i watched, respectively. State $s_t^{(i)}$ transitions to $s_{t+1}^{(i)}$ by taking action $a_t^{(i)} \in A$. We represent this process as follows:

$$s_t^{(i)} \xrightarrow{a_t^{(i)}} s_{t+1}^{(i)} \tag{3}$$

  Note that $|A| = |S|$.
- **Transition Probability P_{sa}:** We assume that the MDP for CF is deterministic. That is, if a user takes action $a_t^{(i)}$ at state $s_t^{(i)}$, transition to the next state $s_{t+1}^{(i)}$ is not random. Since $|A| = |S|$ and the MDP is deterministic, we can see that $a_t^{(i)} = s_{t+1}^{(i)}$ in our problem.

Table 1: Example of $(user, movie, rating, order)$ data

	movie 1	movie 2	movie 3	movie 4	movie 5
user 1	(2,1st)	missing	(5,2nd)	missing	(2,3th)
user 2	(1,1st)	(1,2nd)	missing	(3,3th)	missing
user 3	(5,1st)	missing	missing	(3,2nd)	(5,3th)
user 4	(1,1st)	missing	missing	missing	(4,2nd)
user 5	(2,1st)	(4,2nd)	(5,3th)	(5,4th)	missing

- **Discount factor γ:** $0 < \gamma < 1$ is called the discount factor.
- **Reward r:** When user i takes action $a_t^{(i)}$ at state $s_t^{(i)}$, user i receives a reward or penalty signals defined as follows:

$$r(s_t^{(i)}, a_t^{(i)}) = s_{t+2}^{(i)} - predictor(i, t) \tag{4}$$

  $predictor(i, t)$ is the prediction that a CF algorithm estimates and $s_{t+2}^{(i)}$ is the next, next state. The rationale behind (4) will be described in Section 3.

2.3 Generating Episodes

As mentioned earlier, $X \in \mathbb{R}^{N \times M}$ consists of $(user, movie, rating)$ paired entries. If we use the CF data with $(user, movie, rating, order)$ paired entries, we can generate the *episodes* as follows:

$$\begin{aligned}
&s_1^{(1)} \xrightarrow{a_1^{(1)}} s_2 \xrightarrow{a_2^{(1)}} s_3 \xrightarrow{a_3^{(1)}} \cdots \xrightarrow{a_{T_1-1}^{(1)}} s_{T_1}^{(1)} \\
&s_1^{(2)} \xrightarrow{a_1^{(2)}} s_2 \xrightarrow{a_2^{(2)}} s_3 \xrightarrow{a_3^{(2)}} \cdots \xrightarrow{a_{T_1-1}^{(2)}} s_{T_1}^{(2)} \\
&\qquad\qquad \vdots \\
&s_1^{(N)} \xrightarrow{a_1^{(N)}} s_2 \xrightarrow{a_2^{(N)}} s_3 \xrightarrow{a_3^{(N)}} \cdots \xrightarrow{a_{T_1-1}^{(N)}} s_{T_1}^{(N)}
\end{aligned}$$

$s_j^{(i)}$ is the rating that user i scores for j-th watched movie. For instance, if the $(user, movie, rating, order)$ entries of user-movie data are like Table 1 (The entry $(1, 1, 2, 1st)$ means that user 1 watched movie 1 first and gave 2 stars), the episodes are generated as follows:

$$\begin{aligned}
&2 \to 5 \to 2 \\
&1 \to 1 \to 3 \\
&5 \to 3 \to 5 \\
&1 \to 4 \\
&2 \to 4 \to 5 \to 5
\end{aligned}$$

3 RLCF

RLCF consists of two phases: *training* and *prediction*. The former is to compute the Q-table of the MDP, and the latter is to predict the rate for a user-movie pair using both the output of a CF algorithm and the value of the Q-table.

3.1 Training

Based on the MDP formulation and preliminaries, we learn the Q function using Q-learning which can be summarized as (see [9, 11] for detailed derivation):

$$Q(s,a) = Q(s,a) + \alpha[r(s,a) + \max_{a'} Q(\delta(s,a), a') - Q(s,a)] \tag{5}$$

where $Q(s,a)$ is the old estimated sum of reward, $r(s,a) + \max_{a'} Q(\delta(s,a), a')$ is new estimated sum of reward after a step forward, and α is the learning rate. Since the number of possible actions in state s is $|S|$, $Q(s,a)$ is a $|S| \times |S|$ table. The episodes generated as described in Section 2.3 are used as training data for Q-learning. Here is the training algorithm:

Algorithm 1 Training Algorithm in RLCF

Input:
α : learning rate
γ : discount factor
T_i : the number of movies user i has rated
Output:
$Q(s,a)$: the estimated sum of reward
Initialize $\forall s \in S, \forall a \in A\ Q(s,a) = 0$;
Convert $X \in \mathbb{R}^{N \times M}$ to training episode set as in Section 2.3;
for each user $i = 1 : N$ **do**
  for each movie $j = 1 : T_i$ **do**
    Calculate the reward: $r(s_j^{(i)}, a_j^{(i)}) = s_{j+2}^{(i)} - \mathit{predictor}(i,j)$;
    Update the Q-function $Q(s,a)$ using eq (5);
  end for
end for

3.2 Prediction

The prediction $p_j^{(i)}$ of the entry $x_{ij} \in X$ that user i gives for movie j can be calculated by as follows:

$$p_j^{(i)} = \mathit{predictor}(i,j) + Q(s_{j-2}^i, a_{j-2}^i) \tag{6}$$

In eq (4), the reason why we used the *next next* state $s_{t+2}^{(i)}$, not the *next* state $s_{t+1}^{(i)}$, in the reward function $r(s_t^{(i)}, a_t^{(i)})$ is related to eq (6). If we used the next

Table 2: The RMSE of RLCF

Algorithm	**Base**	**RLCF**	improvement	γ	α
MM	0.9381	0.9109	0.0272	0.5	0.000003
SVD	0.8014	0.7970	0.0044	0.5	0.000006
SVD++	0.8000	0.7954	0.0046	0.65	0.000006

state $s_{t+1}^{(i)}$, prediction $p_j^{(i)}$ must be defined as

$$p_j^{(i)} = predictor(i,j) + Q(s_{j-1}^i, a_{j-1}^i) \tag{7}$$

However, it is impossible to know $Q(s_{j-1}^i, a_{j-1}^i) = Q(s_{j-1}^i, s_j^i)$ when we try to predict $p_j^{(i)}$. This is because the parameter s_j^i is exactly the rating we want to predict.

4 Experiment

4.1 Data set

We adopted MovieLens data set which contains 10,000,054 ratings applied to 10,681 movies by 71,567 users of the online movie recommender service [7]. Users who rated at least 20 movies were chosen and their ratings were sorted chronologically. 20% the most recent movie rating of the entire data is used for testing, and the remaining 80% is used for training.

4.2 Base Predictor

As a base predictor for CF, we adopted SVD and SVD++ which is an improvement over standard SVD by incorporating implicit feedback, implemented by stochastic gradient descent. (See [4] for detailed descriptions on the algorithm.) In addition, we added a simple movie means (MM) algorithm that outputs the mean of movie ratings.

4.3 Results

The state S is defined with ratings in [0.5, 5.0] with 0.5 intervals, making $|S| = 10$ and $Q(s,a)$ is a 10×10 table. We compared the RMSE of RLCF combined with base predictors with that of pure base predictors. We trained SVD, SVD++ predictors with 10 latent features. There are two parameters in RLCF to be tuned: the learning rate α and the discount factor γ of Q-learning. Tables 2 shows experimental results for each predictor with various parameter settings. As expected, SVD++ produced the best performance while MM produced the worst. RLCF improved the performance regardless of the base predictor. That is, MM, SVD, and SVD++ with RLCF predictor reduced RMSE by 0.0246, 0.0044, 0.0046 respectively than its pure predictor. This verifies our idea of incorporating the contrast effect via reinforcement learning.

5 Conclusion

We presented a new reinforcement learning based algorithm, RLCF, for CF. We first formalize the CF as MDP to model the effect that the sequences of a users' previous ratings influence current ratings. Then, the effect is learned and recorded in a Q-table via Q-learning. The experimental results show that the order of past ratings affect the current ratings significantly, and is thus useful to elicit more accurate predictions.

There are some interesting directions for future work. When we formalize the CF as MDP, states can be defined by other factors as well as ratings. For example, the genre, season, and diverse tags can be defined as states in MDP. If the movie genre is used, RLCF can discover the effect that influences ratings for action movies when a user watches an action movie after having watched a comedy.

In addition, RLCF can be combined with other emerging technologies to provide relevant information to the user. For instance, it can be implemented in autonomous vehicles to make suggestions based on the driver's spatiotemporal context.

Acknowledgments

This work was supported by the Sogang University Research Grant of 2012 (201214004.01).

References

1. Bell, R., Koren, Y.: Improved neighborhood-based collaborative filtering. In: Proc. KDD-Cup and Workshop at the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Citeseer (2007)
2. Bellman, R.: A Markovian decision process. (1957)
3. Kaelbling, L., Littman, M., Moore, A.: Reinforcement learning: A survey. Arxiv preprint cs/9605103 (1996)
4. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 426–434. ACM (2008)
5. Paterek, A.: Improving regularized singular value decomposition for collaborative filtering. In: Proceedings of KDD Cup and Workshop. vol. 2007. Citeseer (2007)
6. Puterman, M.: Markov decision processes. Handbooks in Operations Research and Management Science 2, 331–434 (1990)
7. Riedl, J., Konstan, J.: Movielens dataset (1998)
8. Salakhutdinov, R., Mnih, A., Hinton, G.: Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th international conference on Machine learning. p. 798. ACM (2007)
9. Sutton, R., Barto, A.: Reinforcement learning: An introduction. The MIT press (1998)
10. Thurstone, L.: A law of comparative judgment. Psychological review 34(4), 273–286 (1927)
11. Watkins, C., Dayan, P.: Q-learning. Machine learning 8(3), 279–292 (1992)