Expert Systems With Applications 64 (2016) 467–475

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Large-scale recommender system with compact latent factor model

Chien-Liang Liu[a,*], Xuan-Wei Wu[b]

[a] Department of Industrial Engineering and Management, National Chiao Tung University, 1001 University Road, Hsinchu 300, Taiwan ROC
[b] Department of Computer Science, National Chiao Tung University, 1001 University Road, Hsinchu 300, Taiwan ROC

ARTICLE INFO

Article history:
Received 6 March 2016
Revised 20 July 2016
Accepted 2 August 2016
Available online 3 August 2016

Keywords:
Recommender system
Latent factor model
Collaborative filtering
Content-based

ABSTRACT

This work devises a factorization model called compact latent factor model, in which we propose a compact representation to consider query, user and item in the model. The blend of information retrieval and collaborative filtering is a typical setting in many applications. The proposed model can incorporate various features into the model, and this work demonstrates that the proposed model can incorporate context-aware and content-based features to handle context-aware recommendation and cold-start problems, respectively. Besides recommendation accuracy, a critical problem concerning the computational cost emerges in practical situations. To tackle this problem, this work uses a buffer update scheme to allow the proposed model to process data incrementally, and provide a means to use historical data instances. Meanwhile, we use stochastic gradient descent algorithm along with sampling technique to optimize ranking loss, giving a competitive performance while considering scalability and deployment issues. The experimental results indicate that the proposed algorithm outperforms other alternatives on four datasets.

1. Introduction

The last decade has witnessed the great success of recommender systems, which can significantly help users find relevant and interesting items in the information era. There has been much work done both in the industry and academia on developing new approaches to recommender systems. The Netflix Prize was one of the key events that energized research in recommender systems, and inspired many researchers to devise novel recommendation algorithms during the past few years (Agarwal & Chen, 2011; Bobadilla, Ortega, Hernando, & GutiéRrez, 2013; Cho, Kim, & Kim, 2002; Colombo-Mendoza, Valencia-García, Rodríguez-González, Alor-Hernández, & Samper-Zapater, 2015; Gogna & Majumdar, 2015; Liu & Wu, 2016; Yang, Cheng, & Dia, 2008; Zheng & Li, 2011). Besides recommendation accuracy, a critical problem concerning the computational cost emerges in practical situations. Building large-scale real-world recommender systems need to consider other practical issues such as scalability or deployment (Amatriain, 2012). However, most academic recommender systems are based on off-line methods that require costly recalculations over all the observed data when non-registered users or new rating scores are added (Yoshii, Goto, Komatani, Ogata, & Okuno, 2008).

In a standard recommendation setting, one is given a user × item matrix, indicating the known relevance of the given item to a given user, but most elements of the matrix are unknown. Koren, Bell, and Volinsky (2009) demonstrated that using low rank matrix factorization is superior to classic nearest-neighbor techniques for product recommendations. Compared to most of the previous studies on recommendation, this work focuses on *collaborative retrieval* tasks (Weston, Wang, Weiss, & Berenzweig, 2012), which is a joint problem of recommending items to a user with respect to a given query, namely the given data is query × user × item tensor. The blend of information retrieval and recommendation is a typical setting in many applications. For example, when users shop an item on the e-commerce Web sites, the recommender systems should recommend items relevant to the browsing one. The browsing item can be viewed as a query, and the recommender systems should consider the relevance between users and items in providing personalized recommendations.

Traditional recommender systems use explicit information such as rating to learn a recommendation model, and root mean squared error (RMSE) is probably the most popular metric used in evaluating accuracy of predicted ratings. However, rating information is unavailable in most e-commerce Web sites, but millions of registered members and recommendation logs are generated continuously in a very high fluctuating data rates. Therefore, this work

[*] Corresponding author.
E-mail addresses: clliu@mail.nctu.edu.tw, jacky168@gmail.com (C.-L. Liu), XWWu@itri.org.tw (X.-W. Wu).

http://dx.doi.org/10.1016/j.eswa.2016.08.009
0957-4174/© 2016 Elsevier Ltd. All rights reserved.

focuses on implicit information to recommend items. Meanwhile, this work focuses on training to optimize retrieval for the top *N* items by learning a ranking function in which the top *N* items are of particular interest to the user. Thus, the goal of the proposed objective function is to maximize area under the ROC curve (AUC), but maximizing AUC requires to optimize the pairwise loss between two instances from different classes, which is in contrast to the on-line learning setting. This work uses a buffer update approach and sampling techniques to address this problem. The proposed compact latent factor model along with these techniques can process recommendation logs incrementally and efficiently.

The contributions of this work are listed as follows. First, this work devises a recommendation algorithm with a new scoring function to consider query, user, and item in the latent factor space. The proposed model provides the flexibility to consider collaborative filtering and content-based features simultaneously to attain a higher recommendation accuracy and tackle cold-start problem. Second, this work focuses on large-scale recommendation systems, and focuses on the top *N* items that are of particular interest to the user. We use stochastic gradient descent to optimize the proposed objective function, and use a buffer update algorithm and sampling techniques to let the model adapt to new observations continuously. In the experiments, we compare the proposed algorithm with several algorithms on four data sets. The experimental results indicate that the proposed algorithm outperforms other methods. Third, the proposed model can incorporate various features into the model, and we demonstrate that the proposed model can incorporate context-aware and content-based features to deal with context-aware recommendation and cold-start problems, respectively. The experiments on context-aware data set show that the proposed algorithm can be extended to context-aware recommendation algorithm, and the experiments on cold-start problems indicate that the proposed method can benefit from additional information and perform well in the two cold-start situations.

The rest of this paper is organized as follows. Section 2 presents related surveys of recommender systems. Section 3 introduces the problem specification and the proposed compact latent factor model. Section 4 summarizes the results of several experiments. Next, Section 5 discusses and analyzes the experimental results. Conclusions are finally drawn in Section 6.

2. Related work

Collaborative filtering (CF) recommender systems attracted much of attention in the past decade, resulting in significant progress and being adopted by many commercial systems, including Amazon (Linden, Smith, & York, 2003), Netflix and iTunes. Various CF methods have been devised in the past decades (Bobadilla et al., 2013), and matrix factorization is one of the most popular collaborative filtering methods. The winning team of the Netflix Prize competition used more than 100 different predictor sets, and the two underlying algorithms with the best performance in the ensemble are matrix factorization (MF) (Koren et al., 2009) and restricted Boltzmann machines (RBM) (Salakhutdinov, Mnih, & Hinton, 2007). The idea behind MF is to use low rank matrix factorization technique to factorize the rating matrix into two matrices.

In a practical setting, most entries of the rating matrix are missing, and the number of entries is enormous. Therefore, appropriate and efficient optimization algorithms are required to perform matrix factorization. Koren et al. (2009) proposed to use stochastic gradient descent (SGD) and alternating least square (ALS) for learning the parameters of matrix factorization. The SGD is one of the most popular algorithms for matrix factorization, but it is difficult to be parallelized for handling web-scale problems. Zhuang, Chin, Juan, and Lin (2013) developed a fast parallel SGD method called FPSGD for shared memory systems. Luo, Xia, and Zhu (2012) investigated the training process of MF algorithm to propose an incremental learning algorithm to update the trained parameters according to the new data. Near-neighbor search is a common task in recommender systems, but performing exact search in real-time is not scalable on large-scale data sets. Recently, approximate nearest neighbor searching algorithms have become increasingly important, especially in high dimensional data sets (Indyk & Motwani, 1998; Krauthgamer & Lee, 2004; Kushilevitz, Ostrovsky, & Rabani, 1998). Locality-sensitive hashing (LSH) (Indyk & Motwani, 1998) is one of the most promising approximate algorithms, and has been used in recommender systems (Das, Datar, Garg, & Rajaram, 2007). The MF involves the inner product of vectors, and Shrivastava and Li (2014) focused on Maximum Inner Product Search (MIPS) problems to propose a hashing algorithm called Asymmetric LSH (ALSH), which can be used in recommender systems. Karatzoglou, Amatriain, Baltrunas, and Oliver (2010) focused on context-aware problems, and proposed to use tensor factorization in the model, which involves three matrices and a tensor.

Latent factor models become popular in recent years, since they can model interactions between variables and obtain good scalability and high predictive accuracy even in problems with huge sparsity (Rendle, 2010). The majority of matrix factorization models share common patterns, inspiring Chen et al. (2012) to propose a framework called feature-based matrix factorization, which can incorporate various features into the model. Agarwal and Chen (2009) devised a regression-based latent factor model (RLFM) to improve prediction for old user-item dyads by simultaneously incorporating features and past interactions. The RLFM induces a stochastic process on the dyadic space with kernel given by a polynomial function of features. Rendle combined the advantages of support vector machines (SVM) with factorization models to propose an algorithm called factorization machines (FM) (Rendle, 2010; Rendle, Gantner, Freudenthaler, & Schmidt-Thieme, 2011). Compared to SVM, FM considers all interactions between variables with factorized parameters, so it can handle the problems with huge sparsity where SVMs fail. For the data with strong relational patterns, the feature vector of a data instance can become very large, making learning and prediction slow or even infeasible. Rendle (2013) scaled FM to relational data by exploiting repeating patterns in the feature vectors.

The idea behind FM is the latent factor model, which is the same as the proposed method. However, several differences exist between the two methods. First, the proposed compact latent factor model along with the buffer update technique become an online learning approach, allowing the model to process data incrementally and adapting the model to the subsequent data instances. In contrast, FM only focuses on factorization models. Second, the proposed method is a ranking with tensor factorization, while FMs focuses on interactions between features. Third, we introduce a latent matrix to represent user latent information, while FM uses latent vectors to represent all features.

Weston et al. (2012) defined a task called latent collaborative retrieval to consider the input data containing query × user × item. To focus on the top of the ranked list of returned items, the latent collaborative retrieval weights the pairwise violations depending on their positions in the ranked list. However, computing the exact rank is a computationally intensive task, Weston, Bengio, and Hamel (2011) approximated the loss by using weighted approximate-rank pairwise (WARP) loss. The proposed method and latent collaborative retrieval both focus on collaborative retrieval task, but several differences between the two methods. First, scoring functions in the two methods are different. Besides the interaction among query, user and item, the latent collaborative retrieval considers to use the relevance score between user and item as a bias term of the scoring function, while the proposed method