An Hybrid Approach For Recommender System

Sachin
Department of Information Technology
National Institute of Technology Karnataka
Surathkal, India
sachin113.sb@gmail.com

Sowmya Kamath S
Department of Information Technology
National Institute of Technology Karnataka
Surathkal, India
sowmyakamath@nitk.ac.in

***Abstract*—Collaborative Filtering (CF) is a powerful approach which is widely used in many recommender systems. CF-based methods makes use of ratings which are given to the items by users as only source for learning to do recommendations. However, in many applications, recommendation performance of CF-methods decreases when the ratings are very sparse. One more approach is content based methods (item based filtering), which uses user profile or description of the item to do recommendation. This content based method fails in most of cases as user profiles more secured and it is difficult to get user content.**

To address these above issues, we have came up with a hybrid approach which makes use of both Collaborative Filtering (CF) method and Content based methods to do recommendations. This hybrid method considered user ratings of the items and user profile or item description to do more précised recommendations.

Keywords—Hybrid approach; Recommender system; Bayesian Personalized Ranking; Implicit Feeedback;

I. Introduction

Due to many more choices in most of the online services, recommender systems (RS) plays an vital role in recommending items. To the users, making use of recommender systems allows them to effectively use the information. Besides that, many leading tech giants like Amazon, Netflix, Flickr, etc. are using recommender systems to exclusively target their customers by recommending items or services online. Other examples are video portals like YouTube that recommend movies to customers. Personalization is attractive both for content providers, who can increase sales or views, and for customers, who can find interesting content more easily.

The existing recommender systems can be classified into three categories [6]: content-based methods [19], collaborative based methods [20, 21] and hybrid (content-based +collaborative-based) methods [22, 23, 24].

Content based methods (item based filtering) use user profiles or description of the items to do recommendation. These content based methods fail in most of cases as user profiles more secured and it is difficult to get user content. Collaborative Filtering(CF) is a powerful approach which is widely used in many recommender systems. CF-based methods makes use of ratings which are given by users to the items as the sole source of information for learning to recommend the items. However, in many applications, recommendation performance of CF-methods decreases when the ratings are very sparse.

As we have seen because of privacy concerns, usually it is more difficult to collect user profiles than previous activities. Moreover, CF-based methods do have limitations. The prediction accuracy drops significantly when the user ratings are very sparse in number. However, CF-based methods can't be used for recommending new items which have yet to receive ratings from users.

Recommender systems are an active topic for research. Most recent work is on scenarios where users provide explicit feedback, e.g. in terms of ratings. Nevertheless, in real-world scenarios most feedback is not explicit but implicit. Implicit feedback is tracked automatically, like monitoring users ratings, movie views, etc. Thus it is much easier to collect, because the user has not to express his taste explicitly. In fact implicit feedback is already available in almost any information system – e.g. web servers record any page access in log files.

Unfortunately, very few attempts have been made to develop deep learning models for CF. [28] uses restricted Boltzmann machines instead of the conventional matrix factorization formulation to perform CF and [9] extends this work by incorporating user-user and item-item correlations. Although these methods involve both deep learning and CF, they actually belong to CF-based methods because they do not incorporate content information like CTR, which is crucial for accurate recommendation. [30] uses low-rank matrix factorization in the last weight layer of a deep network to significantly reduce the number of model parameters and speed up training, but it is for classification instead of recommendation tasks. On music recommendation, [27, 29] directly use conventional CNN or deep belief networks (DBN) to assist representation learning for content information, but the deep learning components of their models are deterministic without modeling the noise and hence they are less robust. The models achieve performance boost mainly by loosely coupled methods without exploiting the interaction between content information and ratings. Besides, the CNN is linked directly to the rating matrix, which means the models will perform poorly when the ratings are sparse, as shown in the following experiments.

In this paper, we are focusing on movie recommender system, which is based on hybrid (content and collaborative filtering) approach. The task of hybrid recommender system is to give user specific movie recommendations. Preferences of users about movies are learned from the user's previous ratings and from user profile or similar type of users who watched the movies earlier.

II. RELATED WORK

The most popular model for recommender systems is k-nearest neighbor (kNN) collaborative filtering [2]. Traditionally the similarity matrix of kNN is computed by heuristics – e.g. the Pearson correlation – but in recent work [8] the similarity matrix is treated as model parameters and is learned specifically for the task. Recently, matrix factorization (MF) has become very popular in recommender systems both for implicit and explicit feedback. In early work [13] singular value decomposition (SVD) has been proposed to learn the feature matrices. MF models learned by SVD have shown to be very prone to overfitting. Thus regularized learning methods have been proposed. For item prediction Hu et al. [5] and Pan et al. [10] propose a regularized least-square optimization with case weights (WR-MF). The case weights can be used to reduce the impact of negative examples. Hofmann [4] proposes a probabilistic latent semantic model for item recommendation. Schmidt-Thieme [14] converts the problem into a multi-class problem and solves it with a set of binary classifiers. Even though all the work on item prediction discussed above is evaluated on personalized ranking datasets, none of these methods directly optimizes its model parameters for ranking. Instead they optimize to predict if an item is selected by a user or not. In our work we derive an optimization criterion for personalized ranking that is based on pairs of items (i.e. the user-specific order of two items). We will show how state-of-the-art models like MF or adaptive kNN can be optimized with respect to this criterion to provide better ranking quality than with usual learning methods. A detailed discussion of the relationship between our approach and the WRMF approach of Hu et al. [5] and Pan et al. [10] as well as maximum margin matrix factorization [15] can be found in Section 5. In Section 4.1.1, we will also discuss the relations of our optimization criterion to AUC optimization like in [3]. In this paper, we focus on offline learning of the model parameters. Extending the learning method to online learning scenarios – e.g. a new user is added and his history increases from 0 to 1, 2, . . . feedback events – has already been studied for MF for the related task of rating prediction [11]. The same fold-in strategy can be used for BPR.

There is also related work on learning to rank with non-collaborative models. One direction is to model distributions on permutations [7, 6]. Burges et al. [1] optimize a neural network model for ranking using gradient descent. All these approaches learn only one ranking – i.e. they are non-personalized. In contrast to this, our models are collaborative models that learn personalized rankings, i.e. one individual ranking per user. In our evaluation, we show empirically that in typical recommender settings our personalized BPR model outperforms even the theoretical upper bound for non-personalized ranking.

III. IMPLEMENTATION

A. Bayesian Personalized Ranking (BPR)

In this section, we derive a generic method for solving the personalized ranking task. It consists of the general optimization criterion for personalized ranking, BPR-Opt, which will be derived by a Bayesian analysis of the problem using the likelihood function for $p(i >_u j|\Theta)$ and the prior probability for the model parameter $p(\Theta)$. We show the analogies to the ranking statistic AUC (area under the ROC curve) [17]. For learning models with respect to BPR-Opt, we propose the algorithm LearnBPR. Finally, we show how BPROpt and LearnBPR can be applied to two state-ofthe-art recommender algorithms, matrix factorization and adaptive kNN. Optimized with BPR these models are able to generate better rankings than with the usual training methods [17].

B. BPR Optimization Criterion

The Bayesian formulation of finding the correct

personalized ranking for all items $i \in I$ is to maximize the

following posterior probability where Θ represents the parameter vector of an arbitrary model class (e.g. matrix factorization) [17].

$$p(\Theta| >_u) \propto p(>_u |\Theta)\, p(\Theta)$$

Here, $>_u$ is the desired but latent preference structure for user u. All users are presumed to act independently of each other. We also assume the ordering of each pair of items (i, j) for a specific user is independent of the ordering of every other pair. Hence, the above user-specific likelihood function $p(>_u |\Theta)$ can first be rewritten as a product of single densities

and second be combined for all users $u \in U$ [17].

$$\prod_{u \in U} p(>_u |\Theta) = \prod_{(u,i,j) \in U \times I \times I} p(i >_u j|\Theta)^{\delta((u,i,j) \in D_S)} \cdot (1 - p(i >_u j|\Theta))^{\delta((u,j,i) \notin D_S)}$$

where δ is the indicator function:

$$\delta(b) := \begin{cases} 1 & \text{if } b \text{ is true,} \\ 0 & \text{else} \end{cases}$$

Due to the totality and antisymmetry of a sound pairwise ordering scheme the above formula can be simpli- fied to:

$$\prod_{u \in U} p(>_u |\Theta) = \prod_{(u,i,j) \in D_S} p(i >_u j|\Theta)$$

So far it is generally not guaranteed to get a personalized total order. In order to establish this, the already mentioned sound properties (totality, antisymmetry and transitivity) need to be fulfilled. To do so, we define the individual probability that a user really prefers item i to item j as:

$$p(i >_u j|\Theta) := \sigma(\hat{x}_{uij}(\Theta))$$

where σ is the logistic sigmoid:

$$\sigma(x) := \frac{1}{1 + e^{-x}}$$

Here x^uij (Θ) is an arbitrary real-valued function of the model parameter vector Θ which captures the special relationship between user u, item i and item j. In other words, our generic framework delegates the task of modeling the relationship between u, i and j to an underlying model class like matrix factorization or adaptive kNN, which are in charge of estimating xˆuij (Θ). Hence, it becomes feasible to statistically model a personalized total order >u. For convenience, in the following we will skip the argument Θ from ˆxuij . So far, we have only discussed the likelihood function. In order to complete the Bayesian modeling approach of the personalized ranking task, we introduce a general prior density p(Θ) which is a normal distribution with zero mean and variance-covariance matrix ΣΘ [16].

p(Θ) ~ N(0, ΣΘ)

In the following, to reduce the number of unknown hyper parameters we set ΣΘ = λΘI. Now we can formulate the maximum posterior estimator to derive our generic optimization criterion for personalized ranking BPR-Opt [16].

$$\begin{aligned} \text{BPR-OPT} &:= \ln p(\Theta| >_u) \\ &= \ln p(>_u |\Theta)\, p(\Theta) \\ &= \ln \prod_{(u,i,j) \in D_S} \sigma(\hat{x}_{uij})\, p(\Theta) \\ &= \sum_{(u,i,j) \in D_S} \ln \sigma(\hat{x}_{uij}) + \ln p(\Theta) \\ &= \sum_{(u,i,j) \in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta ||\Theta||^2 \end{aligned}$$

where λΘ are model specific regularization parameters.

C. *BPR Learning Algorithm*

In the last section we have derived an optimization criterion for personalized ranking. As the criterion is differentiable, gradient descent based algorithms are an obvious choice for maximization. But as we will see, standard gradient descent is not the right choice for our problem [16]. To solve this issue we propose LearnBPR, a stochastic gradient-descent algorithm based on bootstrap sampling of training triples (see figure 4). First of all the gradient of BPR-Opt with respect to the model parameters is [16]:

$$\begin{aligned} \frac{\partial \text{BPR-OPT}}{\partial \Theta} &= \sum_{(u,i,j) \in D_S} \frac{\partial}{\partial \Theta} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta \frac{\partial}{\partial \Theta} ||\Theta||^2 \\ &\propto \sum_{(u,i,j) \in D_S} \frac{-e^{-\hat{x}_{uij}}}{1 + e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_\Theta \Theta \end{aligned}$$

1: **procedure** LEARNBPR(D_S, Θ)
2: initialize Θ
3: **repeat**
4: draw (u, i, j) from D_S
5: $\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{uij}}}{1+e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} + \lambda_\Theta \cdot \Theta \right)$
6: **until** convergence
7: **return** $\hat{\Theta}$
8: **end procedure**

Figure 4: Optimizing models for BPR with bootstrapping based stochastic gradient descent. With learning rate α and regularization λΘ [17].

The other popular approach is stochastic gradient descent. In

this case for each triple (u, i, j) ∈ DS an update is performed

[16].

$$\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{uij}}}{1 + e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} + \lambda_\Theta \Theta \right)$$

In general this is a good approach for our skew problem but the order in which the training pairs are traversed is crucial. A typical approach that traverses the data item-wise or user-wise will lead to poor convergence as there are so many consecutive updates on the same user-item pair – i.e. for one

user-item pair (u, i) there are many j with (u, i, j) ∈ DS. To

solve this issue we suggest to use a stochastic gradient descent algorithm that chooses the triples randomly (uniformly distributed). With this approach the chances to pick the same user-item combination [16].

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable in consecutive update steps is small. We suggest to use a bootstrap sampling approach with replacement because stopping can be performed at any step. Abandoning the idea of full cycles through the data is especially useful in our case as the number of examples is very large and for convergence often a fraction of a full cycle is sufficient. We choose the number of single steps in our evaluation linearly depending on the number of observed positive feedback S [16].

D. LightFM

LightFM is a Python implementation of a number of popular recommendation algorithms for both implicit and explicit feedback, including efficient implementation of BPR and WARP ranking losses. It's easy to use, fast (via multithreaded model estimation), and produces high quality results.

It also makes it possible to incorporate both item and user metadata into the traditional matrix factorization algorithms. It represents each user and item as the sum of the latent representations of their features, thus allowing recommendations to generalize to new items (via item features) and to new users (via user features) [17].

IV. DATASETS

The dataset we have used is MovieLens 100k dataset. MovieLens data sets were collected by the GroupLens Research Project at the University of Minnesota [18].

This data set consists of:

- 100,000 ratings (1-5) from 943 users on 1682 movies.
- Each user has rated at least 20 movies.
- Simple demographic info for the users (age, gender, occupation, zip) [18]

The data was collected through the MovieLens web site (movielens.umn.edu) during the seven-month period from September 19th, 1997 through April 22nd, 1998. This data has been cleaned up users who had less than 20 ratings or did not have complete demographic information were removed from this data set. We have fitted implicit feedback model on the MovieLens 100k dataset [18].

Here are brief descriptions of the data.

- ml-data.tar.gz -- Compressed tar file. To rebuild the u data files do this:
  gunzip ml-data.tar.gz
  tar xvf ml-data.tar
  mku.sh [18]

- u.data -- The full u data set, 100000 ratings by 943 users on 1682 items.

Each user has rated at least 20 movies. Users and items are numbered consecutively from 1. The data is randomly ordered. This is a tab separated list of user id | item id | rating | timestamp. The time stamps are unix seconds since 1/1/1970 UTC.

- u.genre -- A list of the genres.
- u.user -- Demographic information about the users;

this is a tab separated list of user id | age | gender |occupation | zip code The user ids are the ones used in the u.data data set [18].

V. RESULTS

The result of our hybrid based approach as shown following:

```
User 2
     Already Watched Movies :
        Return of the Jedi (1983)
        Event Horizon (1997)
        Schindler's List (1993)
    New Recommended Movies:
        L.A. Confidential (1997)
        Titanic (1997)
        Apt Pupil (1998)
User 20
     Already Watched Movies :
        Toy Story (1995)
        Twelve Monkeys (1995)
        Dead Man Walking (1995)
    New Recommended Movies:
        Trainspotting (1996)
        Scream (1996)
        Twelve Monkeys (1995)
User 40
     Already Watched Movies :
        Toy Story (1995)
        Star Wars (1977)
        Pulp Fiction (1994)
    New Recommended Movies:
        Raiders of the Lost Ark (1981)
        Silence of the Lambs, The (1991)
        Dr. Strangelove or: How I Learned
```

Fig 4: Already watched movies and newly recommended movies for different users.

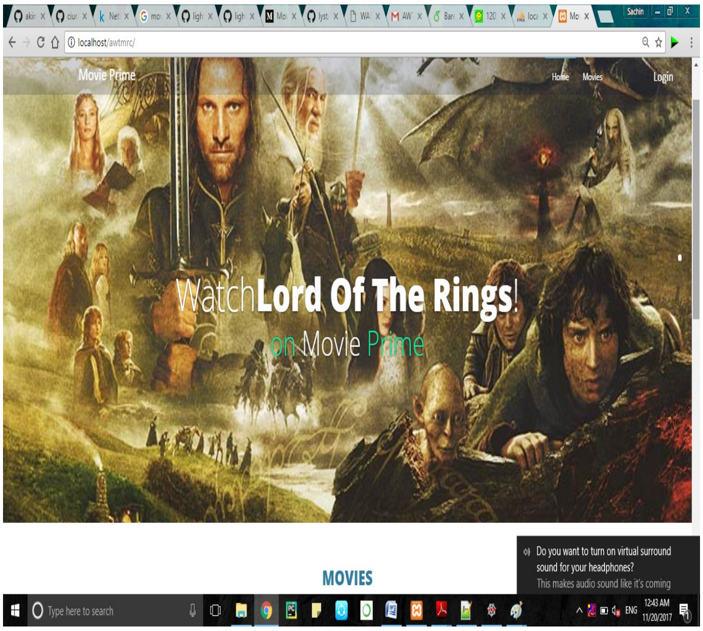

Fig 5: Home page of movie recommender system.

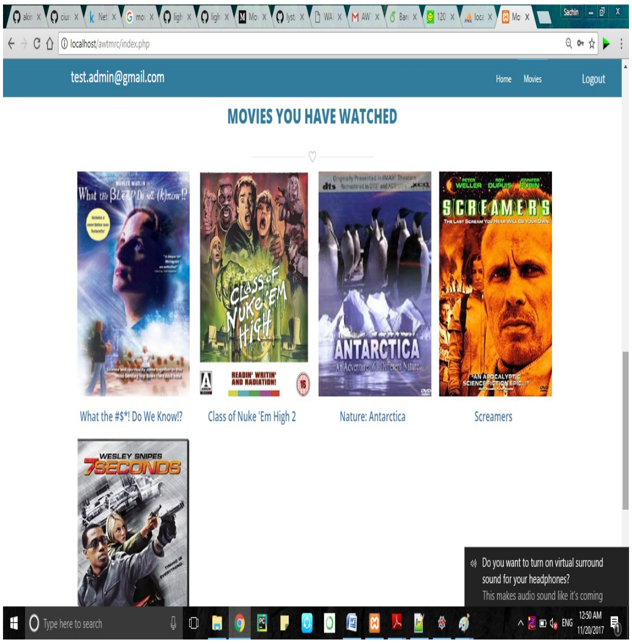

Fig 6: Movies user has already watched.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have demonstrated that performance of movie recommender system can be increased significantly by using hybrid (content-based and collaborative filtering) method, *Bayesian Personalized Ranking (BPR)* algorithm.

In future, we can have more customized UI and we can use Tensorflow package for high end performance.

References

[1] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In ICML '05: Proceedings of the 22nd international conference on Machine learning, pages 89–96, New York, NY, USA, 2005. ACM Press.

[2] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. ACM Transactions on Information Systems. Springer-Verlag, 22/1, 2004.

[3] A. Herschtal and B. Raskutti. Optimising area under the roc curve using gradient descent. In ICML '04: Proceedings of the twenty-first international conference on Machine learning, page 49, New York, NY, USA, 2004. ACM.

[4] T. Hofmann. Latent semantic models for collaborative filtering. ACM Trans. Inf. Syst., 22(1):89– 115, 2004.

[5] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In IEEE International Conference on Data Mining (ICDM 2008), pages 263–272, 2008.

[6] J. Huang, C. Guestrin, and L. Guibas. Efficient inference for distributions on permutations. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems 20, pages 697–704, Cambridge, MA, 2008. MIT Press.

[7] R. Kondor, A. Howard, and T. Jebara. Multiobject tracking with representations of the symmetric group. In Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, San Juan, Puerto Rico, March 2007.

[8] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 426–434, New York, NY, USA, 2008. ACM.

[9] B. Marlin. Modeling user rating profiles for collaborative filtering. In S. Thrun, L. Saul, and B. Sch¨olkopf, editors, Advances in Neural Information Processing Systems 16, Cambridge, MA, 2004. MIT Press.

[10] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. M. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In IEEE International Conference on Data Mining (ICDM 2008), pages 502–511, 2008.

[11] S. Rendle and L. Schmidt-Thieme. Onlineupdating regularized kernel matrix factorization models for large-scale recommender systems. In RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008.

[12] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In ICML '05: Proceedings of the 22nd international conference on Machine learning, pages 713–719, New York, NY, USA, 2005. ACM.

[13] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental singular value decomposition algorithms for highly scalable

recommender systems. In Proceedings of the 5th International Conference in Computers and Information Technology, 2002.

[14] L. Schmidt-Thieme. Compound classification models for recommender systems. In IEEE International Conference on Data Mining (ICDM 2005), pages 378–385, 2005.

[15] M. Weimer, A. Karatzoglou, and A. Smola. Improving maximum margin matrix factorization. Machine Learning, 72(3):263–276, 2008.

[16] Rendle, Steffen, et al. "BPR: Bayesian personalized ranking from implicit feedback." Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. AUAI Press,2009.

[17] https://github.com/lyst/lightfm

[18] https://grouplens.org/datasets/movielens/

[19] J. Bobadilla, F. Ortega, A. Hernando, and A. Guti_errez. Recommender systems survey.Knowledge Based Systems, 46:109 132, 2013.

[20] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In UAI, pages 452 461, 2009.

[21] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In NIPS, pages 1257 1264, 2007.

[22] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In KDD, pages 19 28, 2009.

[23] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu.Personalized recommendation via cross-domain triadic factorization. In WWW, pages 595 606, 2013.

[24] W.-J. Li, D.-Y. Yeung, and Z. Zhang. Generalized latent factor models for social network analysis. In IJCAI, pages 1705{1710, 2011.

[25] J. Bobadilla, F. Ortega, A.Hernando, and A. Guti_errez. Recommender systems survey. Knowledge Based Systems, 46:109{132, 2013.

[26] T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In ICASSP, pages 6655{6659, 2013.

[27] R. Salakhutdinov and G. E. Hinton. Deep Boltzmann machines. In AISTATS, pages 448{455, 2009.

[28] R. Salakhutdinov and G. E. Hinton. Semantic hashing. J. Approx. Reasoning, 50(7):969{978, 2009.

[29] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In NIPS, pages 1257{1264, 2007.

[30] R. Salakhutdinov, A. Mnih, and G. E. Hinton.

Restricted boltzmann machines for collaborative _ltering. In ICML, pages 791{798, 2007.