The AAAI-17 Workshop on
Crowdsourcing, Deep Learning, and Artificial Intelligence Agents
WS-17-07

Incorporating Collaborative Ranking Algorithm with Weighted Recursive Autoencoder for Item Recommendation

Hanzhang Song, Yunhui Guo, Congfu Xu
Institute of Artificial Intelligence, Zhejiang University, Hangzhou, China
{song_hz,gyhui,xucongfu}@zju.edu.cn

Abstract

Collaborative filtering (CF) with implicit feedback is a successful method for recommending items to users, which does not require a knowledge of the items or users. CF methods can be mainly classified into two categories. One is point-wise regression based and the other is pair-wise ranking based, where the latter one only tries to find out the items that users prefer while ignores the items that users dislike, and usually gives out a better recommended item list. The performance of CF-based methods degrades significantly when the feedback information is sparse. To address the problem, many kinds of auxiliary information have been utilized such as users' reviews on items, items' content and description information, price, brands. In this paper we utilize a weighted recursive autoencoder (RAE) to extract useful features from several heterogeneous auxiliary information and tightly couple the weighted RAE with a pair-wise ranking based CF method. Analysis of the hyperparameters illustrates that auxiliary information from different sources is indeed able to benefit our model. Empirical experiments on six real world datasets show that our method outperforms other state-of-the-art methods.

Introduction

Recommendation techniques have gained notice due to nowadays information overload in online service (Sarwar et al. 2001; Su and Khoshgoftaar 2009; Zhang et al. 2014). The main goal of recommendation system (RS) is to find out the items that users may be interested in from a large repository of items. To provide personalized service, RS should utilize users' feedback which contains two kinds of information, explicit feedback for example ratings on items or implicit feedback such as browse history, clicks and time spent on the websites. Because explicit feedback are always hard to obtain, making recommendation based on implicit feedback is a more valuable task.

Collaborative filtering (CF) based methods using implicit feedback are widely applied in RS for their outstanding performance and simple requirement for feedback. CF based methods can be mainly classified into two categories, point-wise regression algorithms like (Hu, Koren, and Volinsky 2008) and pair-wise ranking preference algorithms like (Rendle et al. 2009) . The task of point-wise regression algorithm is to evaluate the degree that a user likes an item. As a result, point-wise regression algorithm can not only tell which items that users like, but also can predict the items that users dislike, which has less value. Moreover, to predict items that users dislike can constrain the whole model's performance on recommending suitable items for users. Pair-wise ranking algorithm's task is to evaluate which one a user loves more between two items and ignores the degree of love. This relaxes the assumption and pair-wise ranking algorithms usually performs better than point-wise regression algorithm.

When the sparsity of feedback becomes severe, performance of CF methods drops rapidly. Recently, some works have introduced different kinds of auxiliary information about users and/or items, which is also called side information, into the CF framework to improve the performance. The methods utilizing both feedback and content of items/users are called hybrid methods. For the reason of privacy concern, users' detailed profiles are hard to collect. So in most cases, the applied side information is about items such as items' description text, price, brands or the visual image of items. Most of previous works only utilize one kind of the above side information. Further more, hybrid methods can be divided into two sub-classes: loosely coupled methods like (Sevil et al. 2010) and tightly coupled methods like (Wang and Blei 2011). loosely coupled methods process side information once for extracting suitable features. The CF algorithm will not guide the extraction of features, and it is a manual and tedious work to choose out the right features. On the contrary, tightly coupled methods allow CF algorithm to guide the extraction of features, and also take advantage of the features during the CF process.

In this paper, we propose a novel tightly coupled CF based hybrid model: collaborative deep learning with heterologous side information for ranking (CDHR) which incorporates one of the powerful deep learning models, recursive autoencoder (RAE), with CF based pair-wise ranking algorithm to utilize several kinds of heterologous side information at the mean time to improve the performance of recommendation. Because different side information may have different forms, have different characteristics and show items' different aspects, it is intuitive to use RAE to extract features from them. We will explain the reason in following sections. The main contributions of this paper are listed below:

- By utilizing a specific recursive autoencoder (RAE), we extract useful features from the following four kinds of side information: items' description texts, users' review texts on items, items' price and items' brand.

- To guide the extraction of features for recommendation. We tightly couple the RAE with the BPR (Rendle et al. 2009) and design a sampling based learning process.
- In order to balance the importance of the heterogeneous side information and achieve a better performance for recommendation, we improve the RAE by introducing a weight matrix.
- To prove the value of side information for recommendation, we analyze the effect of the weight matrix which controls the importance of the different kinds of side information.

Related Work

In this section, we review some works that are closely related to our work, including point-wise regression and pair-wise ranking CF based methods without using side information, point-wise regression methods using side information and pair-wise ranking based methods only using homologous side information.

One of the most widely used and successful approaches in traditional recommender systems is collaborative filtering. Due to the effectiveness and efficiency in dealing with very large user-item rating matrices, the low-rank matrix factorization (MF) models (Hu, Koren, and Volinsky 2008; Hofmann 2003) receive a great attention. MF framed models' goal is to predict users' ratings on items, which is called point-wise.

Different from point-wise methods, Bayesian Personalized Ranking (BPR) proposed in (Rendle et al. 2009) is one of the most successful CF based pair-wise ranking algorithms. BPR assumes that users prefer observed items than unobserved items and demonstrates that the optimization objective of BPR based methods is to lower the Area Under ROC Curve (AUC), which is more reasonable than the Root Mean Square Error (RMSE) in evaluating the quality of recommended items list.

All CF based recommendation methods will encounter the sparsity problem when the sparsity of feedback is severe. Researchers propose many context-aware methods (Chen et al. 2014; Van den Oord, Dieleman, and Schrauwen 2013; Wang and Blei 2011) which explore information about items to address the problem.

With the development of deep learning, many neural network models show the great power in processing nature language and image, such as the stacked denoising autoencoder (SDAE) (Vincent et al. 2008) and recursive autoencoder (RAE) (Socher et al. 2011a). As a result, it is reasonable to apply neural networks into context-aware recommender system to take advantage of the side information. For example, VBPR (He and McAuley 2016) incorporates CNN with MF for recommending, and CDL (Wang, Wang, and Yeung 2015; Ying et al. 2016) use SDAE to extract features from text for recommending. Moreover, CDR (Ying et al. 2016) combines BPR with SDAE and achieves a better result. However, prior hybrid methods always consider only one kind of homologous side information. In this paper, by utilizing recursive denoising autoencoder, we introduce several heterogeneous side information to help improve the performance.

CDHR Model

In this section, we elucidate our CDHR model. We first state the recursive autoencoder applied in our model, then present our CDHR model, which combines BPR with weighted RAE. Finally, we derive a sampling based learning process to obtain the approximate optimal solution of the model.

Preliminaries

We have the implicit feedback user-item purchase matrix R of size $m \times n$, where m is the number of users, and n is the number of items. $\mathbb{U} = \{u_1, u_2, \ldots, u_m\}$ is the set of users and $\mathbb{V} = \{v_1, v_2, \ldots, v_n\}$ is the set of items. r_{ij} is the element of matrix R, and $r_{ij} = 1$ if user u_i has purchased item v_j, $r_{ij} = 0$ otherwise. Besides purchase matrix, we can also obtain the side information matrix about items. In this paper, we make use of four kinds of side information as follows: the description text of items, the review text of items, the price of items and the brand of items. We use bags-of-words vector to represent text type side information of items, and use numerical tag to represent the brand information. Thus, the side information matrix in this paper $S \in \mathbb{R}^{m \times 4}$ can be defined as $S = [W^d, W^r, P, B]$, where the length of W^d, W^r, P, B is m. And W_j^d, W_j^r are the bags-of-words vector of description text, review text of item v_j, P_j, B_j are the price and numerical tag of item v_j. Let U_i, V_j denote the latent factors with low dimension K of user u_i and v_j. Our objective is to learn the latent factor $U_i (i = 1, 2, \ldots, m)$ and $V_j (j = 1, 2, \ldots, n)$ from implicit feedback matrix R and item side information matrix S for recommending an personalized ranking list for users.

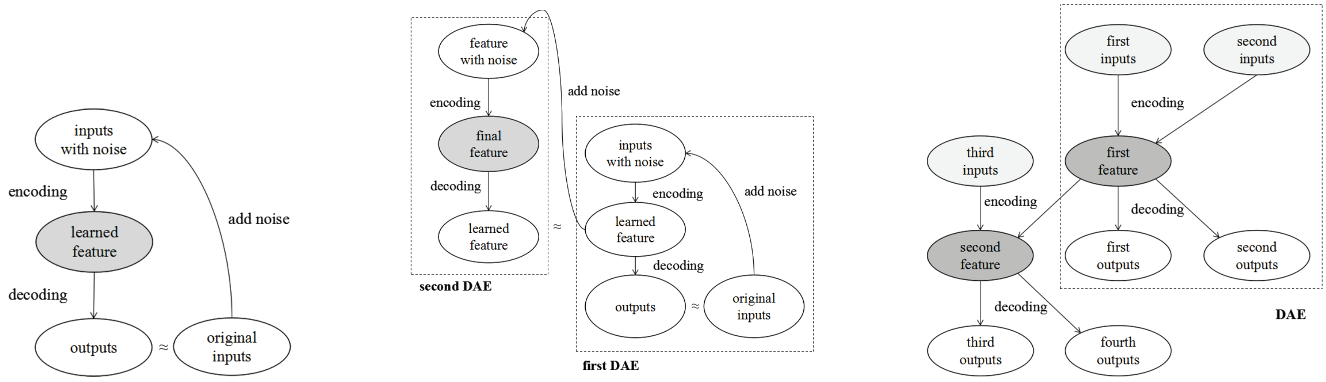
Weighted Recursive Autoencoder

Denoising Autoencoder (DAE) (Vincent et al. 2008) is a neural network that is trained to reconstruct the clean inputs from inputs with added noise. The front half of the DAE can be regarded as an encoder, which is able to extracts some important features that can be decoded into the original inputs by the last half of the DAE. Stacked denoising autoencoder (SDAE) (Vincent et al. 2010) is to stack several DAEs together to get a better performance. Actually DAE is a special SDAE that only has one DAE. Recursive autoencoder (RAE) (Socher et al. 2011b) is similar to SDAE, except that RAE will import one or more new inputs for each DAE. The graphical models of DAE, SDAE and RAE are shown in Figure 1.

The objective of every SDAE in RAE is to minimize the regularized optimization problem as below:

$$\min_{\{W_l\},\{b_l\}} \|X_c - X_L\|_F^2 + \lambda_w \sum_l (\|W\|_F^2 + \|b\|_F^2),$$

where X_c is the clean inputs, X_L is the outputs of SDAE, λ_w is the regularization parameter. W_l and b_l represent the



(a) DAE

(b) SDAE

(c) RAE

Figure 1: The graphical model of DAE, SDAE and RAE

weight matrix and bias vectors of the SDAE, L is the number of layers of the SDAE and $\|\cdot\|_F$ denotes the Frobenius norm.

Suppose that the corrupted input X_o and the clean input X_c are observed variables, similar to (Bengio et al. 2013). SDAE can be generalized as a probabilistic model. The generative process is as follows:

- For each layer l of the SDAE network,

  For each column n of the weight matrix W_l, draw
  $$W_{l,*n} \sim \mathcal{N}(0, \lambda_w^{-1}\mathbf{I}_{K_l}).$$
  Draw the bias vector $b_l \sim \mathcal{N}(0, \lambda_w^{-1}\mathbf{I}_{K_l})$.
  For each row j of X_l, draw
  $$X_{l,j*} \sim \mathcal{N}(\sigma(X_{l-1,j*}W_l + b_l), \lambda_s^{-1}\mathbf{I}_{K_l}),$$
  where $\sigma(\cdot)$ is the sigmoid active function.
- For each item j, draw a clean input
  $$X_{c,j*} \sim \mathcal{N}(X_{L,j*}, \lambda_n^{-1}I_m).$$

Through maximizing a posteriori estimation, the model will degenerate to be the Bayesian formulation of SDAE if λ_s goes to infinity (Strichartz 2003).

The generalized bayesian model of RAE is similar to SDAE, except that it will import one or more new inputs for each DAE. Moreover, the dimension of the previous learned feature vectors and the new input vectors may vary widely. For example, in this paper the dimension of feature learned from bags-of-words vector is usually larger than ten, while the dimension of the price is only one. This will indeed reduce the importance of the price feature, hence we introduce a weight matrix M_w to control the effect of the inputs. Thus the objective of each DAE in RAE is modified to:

$$\min_{\{W_l\},\{b_l\}} \|(X_c - X_L)M_w\|_F^2 + \lambda_w \sum_l (\|W\|_F^2 + \|b\|_F^2),$$

where M_w is a diagonal matrix, each element at the diagonal controls the effect of the corresponding input, M_w is an identity matrix in original DAE. It is notable that X_c in the RAE concludes the clean inputs and all the learned feature vectors that used as inputs in the next DAE.

Compared with SDAE, RAE can change the noise's level and type for every new inputs and can adjust the sequence of inputs to control the importance of the inputs. These characteristics make RAE intuitive to extracts features from heterogeneous side information.

Combine RAE with BPR

The training data of BPR is the triple set $D_s = \{(u_i, v_j, v_k)\}$. Each triple denotes that user u_i prefers item v_j than item v_k. Let $p_{ijk} = \sigma(\delta_{ijk})(i = 1, 2, \ldots, m; j, k = 1, 2, \ldots, n;)$ be the probability of each triple's occurrence, where $\sigma(\cdot)$ is the logistic sigmoid function. To maximize the posterior probability of BPR is equally to optimize the following objective function:

$$\max_{(i,j,k)\in D_s} \sum_{(i,j,k)} \ln \sigma(\delta_{ijk}) - \lambda_u \|U\|_F^2 - \lambda_v \|V\|_F^2,$$

where $\delta_{ijk} = U_i^T V_j - U_i^T V_k$, $\lambda_u \|U\|_F^2$ and $\lambda_v \|V\|_F^2$ are regularization terms that are used to avoid over-fitting.

Like (Wang, Wang, and Yeung 2015), maximizing the posterior probability is equivalent to maximizing the joint log-likelihood of $U, V, \{X_l\}, X_c, \{W_l\}, \{b_l\}$, and R given $\lambda_u, \lambda_v, \lambda w, \lambda s$, and λ_n. And we also set λ_s to infinity. Combining RAE with BPR, we get the following likelihood function:

$$\begin{aligned} L = &\sum_{(i,j,k)\in D_s} \ln \sigma(U_i^T V j - U_i^T V_k) - \frac{\lambda_u}{2}\|U\|_F^2 \\ &- \frac{\lambda_w}{2} \sum_l (\|W_l\|_F^2 + \|b_l\|_F^2) \\ &- \frac{\lambda_v}{2} \sum_{j=1}^{n} \|V_j - X_{f,j*}^T\|_F^2 \\ &- \frac{\lambda_n}{2} \sum_{j=1}^{n} \|(X_{L,j*} - X_{c,j*})M_w\|_F^2. \end{aligned} \tag{1}$$

From (1), BPR and RAE are connected by the term $\|V_j - X_{f,j*}^T\|_F^2$. When λ_v is set to a large value, CDHR actually treats $X_{f,j*}^T$ as V_j. When λ_v is set to 0, CDHR is separated to two parts.

In this paper, we have four kinds of inputs for the RAE, and the graphical model of CDHR is shown in Figure 2. To show the structure of the RAE, we zoom in the the dashed box on the left, and show the details on the right.

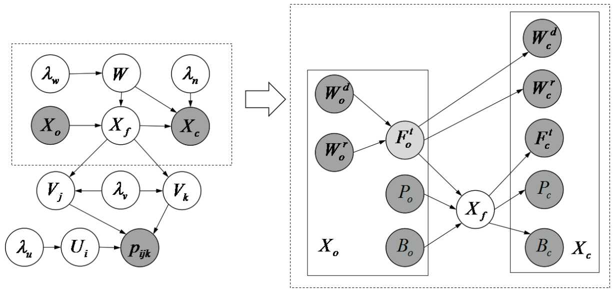


Figure 2: The graphical model of CDHR.

Parameter Learning

We optimize the objective function using coordinate ascent by alternatively optimizing latent factors U, V, weight matrix W and bias vector b. For the reason that the triples in D_s is too many, it is not feasible to update U and V in the BPR part using full gradient over all training data. We apply bootstap sampling in each iteration to update the parameters by a stochastic gradient descent algorithm.

Given the weight matrix W and bias vector b, we update U and V by:

$$\begin{aligned}
U_i &= U_i + \alpha\Big(\frac{e^{-\delta_{ijk}}}{1+e^{-\delta_{ijk}}} \cdot (V_j - V_k) - \lambda_u U_i\Big) \\
V_j &= V_j + \alpha\Big(\frac{e^{-\delta_{ijk}}}{1+e^{-\delta_{ijk}}} \cdot U_i - \lambda_v (V_j - X_{f,j*}^T)\Big) \\
V_k &= V_k + \alpha\Big(\frac{-e^{-\delta_{ijk}}}{1+e^{-\delta_{ijk}}} \cdot U_i - \lambda_v (V_j - X_{f,j*}^T)\Big)
\end{aligned} \tag{2}$$

where α is the learning rate.

Given U and V, weight matrix W and bias vector b can be updated by back-propagation learning algorithm. The gradients of the likelihood with respect to W_l and b_l are as follows:

$$\begin{aligned}
\nabla_{W_l} L =& -\lambda_w W_l - \lambda_v \sum_j \nabla_{W_l} X_{f,j*}^T (X_{f,j*} - V_j) \\
& - \lambda_n \sum_j \nabla_{W_l} X_{L,j*} (X_{L,j*} - X_{c,j*}) M_w \\
\nabla_{b_l} L =& -\lambda_w b_l - \lambda_v \sum_j \nabla_{b_l} X_{f,j*}^T (X_{f,j*} - V_j) \\
& - \lambda_n \sum_j \nabla_{b_l} X_{L,j*} (X_{L,j*} - X_{c,j*}) M_w
\end{aligned} \tag{3}$$

Experiment

In this section, we compare our approach with the most related other state-of-the-art algorithms on six real-world datasets, and demonstrate that our approach has a better performance.

Dataset	users	items	feedback	sparsity
Beauty	11,448	27,017	154,559	99.95%
Books	94,375	10,002	298,248	99.97%
Cell Phones & Accessories	8,643	24,767	100,756	99.95%
Office Products	2,849	8,530	43,793	99.82%
Health & Personal Care	14,210	33,005	218,373	99.95%
Sports & Outdoors	17,278	37,342	211,351	99.97%

Table 1: General statistics of datasets

Datasets

In our experiments, we use six real-world datasets from Amazon website. All the datasets are publicly available. [1] The general statistics of the datasets are shown in Table 1. We have filtered out the users who have less than five feedback in the datasets. We consider a user likes an item if he/she has rated the item and vice versa.

For each item, we collect the description text, review text, brand tag and price information. All review text about one item are viewed as a whole which is represented by a bags-of-words vector in this paper. We follow the same procedure as that in (Wang and Blei 2011) to preprocess the text information. After removing stop words, the top 3,000 discriminative words according to the tf-idf values are chosen to construct the bags-of-words vectors.

We use numerical tags $(1, 2, \ldots, N_t)$ to represent the brand information, where N_t is the total number of brands. We map brand tag and price to the $(0, 1)$ scopes to be consistent with the bags-of-words vectors. As a result, the weight matrix M_w undertakes the responsibility to control the different importance of various side information.

Competitors

We compare our approach CDHR with three state-of-art recommendation algorithms for implicit feedback as follows:

- **BPR**:Bayesian Personalized Ranking (Rendle et al. 2009) is a pair-wise ranking algorithm for recommending as mentioned in previous section.
- **CDL**:Collaborative Deep Learning (Wang, Wang, and Yeung 2015) is a point-wise regression algorithm which incorporates MF with SDAE. By using SDAE to extract useful features from text form side information of items, CDL achieves a better performance.
- **CDR**:Collaborative Deep Ranking (Ying et al. 2016) revises CDL by utilizing BPR instead of MF for collaborative filtering. Benefit from the pair-wise ranking algorithm BPR, CDR outperforms CDL significantly.

[1] http://snap.stanford.edu/data/web-Amazon-links.html

Dataset	Evaluation	BPR	CDL	CDR	CHDR	Dataset	Evaluation	BPR	CDL	CDR	CDHR
Beauty	precision	0.0143 29.4%	0.0122 51.6%	0.0172 7.6%	**0.0185**	Office Products	precision	0.0101 35.6%	0.0086 59.3%	0.0126 8.7%	**0.0137**
	recall	0.0868 6.8%	0.0547 69.5%	0.0893 3.8%	**0.0927**		recall	0.0562 18.9%	0.0412 62.1%	0.0602 11.0%	**0.0668**
	nDCG	0.0170 10.0%	0.0131 43.8%	0.0182 2.7%	**0.0187**		nDCG	0.0116 33.6%	0.0090 72.2%	0.0137 13.1%	**0.0155**
Books	precision	0.0055 30.9%	0.0039 84.7%	0.0065 10.8%	**0.0072**	Health & Personal Care	precision	0.0115 40.0%	0.0084 91.7%	0.0154 4.5%	**0.0161**
	recall	0.0809 5.2%	0.0548 55.3%	0.0819 3.9%	**0.0851**		recall	0.0639 50.7%	0.0554 73.8%	0.0857 12.4%	**0.0963**
	nDCG	0.0068 26.5%	0.0049 75.5%	0.0083 3.6%	**0.0086**		nDCG	0.0138 31.9%	0.0098 85.7%	0.0167 9.0%	**0.0182**
Cell Phones & Accessories	precision	0.0112 28.6%	0.0088 63.6%	0.0130 10.8%	**0.0144**	Sports & Outdoors	precision	0.0082 53.7%	0.0067 88.1%	0.0115 9.6%	**0.0126**
	recall	0.0805 25.8%	0.0642 57.8%	0.0894 13.3%	**0.1013**		recall	0.0556 26.4%	0.0439 60.1%	0.0621 13.2%	**0.0703**
	nDCG	0.0136 30.1%	0.0113 56.6%	0.0161 9.9%	**0.0177**		nDCG	0.0097 40.2%	0.0076 78.9%	0.0123 10.6%	**0.0136**

Table 2: Performance comparison of BPR, CDL, CDR, and CHDR

For CDL and CDR, we combine the four kinds of feature vectors in this paper as the side information vectors in the above models.

Parameter Setting

We ramdomly split the whole dataset into three parts at the ratio of 8:1:1 for training, cross validation, testing correspondingly. Each approach is repeated five times on every dataset, and the average performance is reported. The grid search is applied to find optimal hyperparameters for each approach.

For BPR, we set $\lambda_u = \lambda_v = 0.1$ and sample $100 \times N_{training}$ triples for training in every iteration, where $N_{training}$ is the number of feedback in training set. For CDL and CDR, we both apply a 2-layer SDAE with the architecture 3000-200-K-200-3000. A salt-and-pepper noise is applied, and the noise rate is 0.3. Besides, we also use a dropout rate of 0.1 for all the autoencoders in this paper to achieve adaptive regularization. After grip search, we set $\lambda_w = 0.0001$ and $\lambda_n = 0.01$ for CDL, and $\lambda_w = 0.0001$ and $\lambda_n = 0.1$ for CDR. For CDL, CDR and CDHR, λ_u and λ_v vary in different datasets, and are decided by the validation process.

We set $\lambda_w = 0.0001$ and $\lambda_n = 0.01$ for CDHR. And the architecture of RAE in CDHR shown in Figure 2 is stated as follows:

- First, we use a 6000-800-100-800-6000 SDAE to learn the middle feature vectors, which takes the corrupted bags-of-words vectors of description text and review text as inputs. A salt-and-pepper noise with 0.3 noise rate is utilized to generate the corrupted inputs.
- Second, we apply a 102-K-102 weighted DAE to learn the final feature, which takes the middle feature vectors, price and brand as input. A Gaussian noise with deviation of 0.001 is added to the price input, while we do not add noise to the brand input.
- It is obvious that we should enhance the weight of price and brand input to avoid the RAE just ignore the two inputs to achieve a less loss output. After several experiments, we set the weight parameters to 1,50,10 for middle feature, price and brand correspondingly.

Evaluation Metrics and Performance

We apply *Precision*, *Recall* and *nDCG* to evaluate the performances of models. Given the recommended items list, *Precision* and *Recall* are defined as:

$$\begin{aligned} Precision@M &= \frac{\text{\# items the user likes in the list}}{M} \\ Recall@M &= \frac{\text{\# items the user likes in the list}}{\text{\# total items the user likes}} \end{aligned} \tag{4}$$

where M is the length of the recommend list.

nDCG is widely used to evaluate the quality of a ranked list, which is defined as below:

$$\begin{aligned} nDCG &= \frac{DCG}{IDCG} \\ DCG &= \sum_{i=1}^{M} \frac{2^{r(i)} - 1}{\log(1+i)} \end{aligned} \tag{5}$$

where $r(i)$ is the relevance level of items. In this paper, $r(i) = 1$ if user likes the item, and $r(i) = 0$ otherwise. *IDCG* is the max value of DCG.

Because of space limitation, We only show the results with $M = 20$ and dimension $K = 50$ in Table 2. Bold cell in

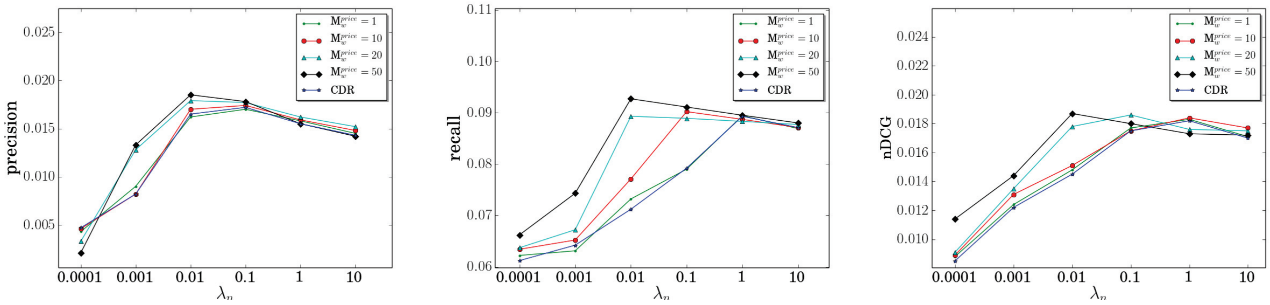


Figure 3: The impact of λ_n and M_w^{price} on dataset Beauty

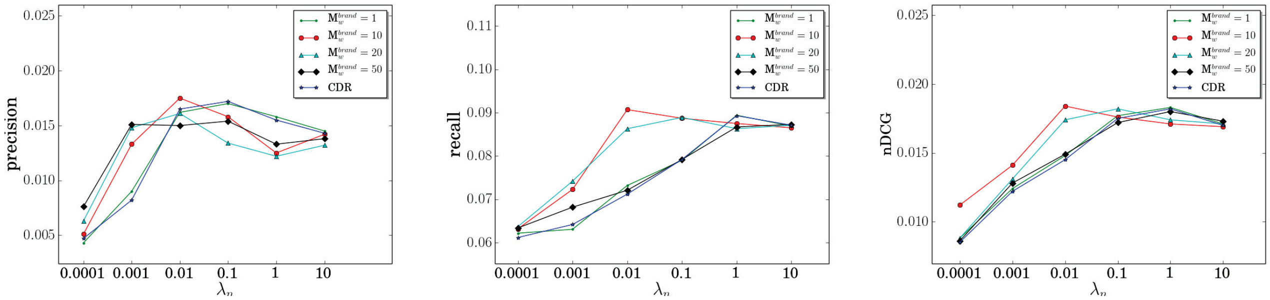


Figure 4: The impact of λ_n and M_w^{brand} on dataset Beauty

the table is the best results compared with other approaches and the percentage improvement of CHDR compared with other methods is also shown in the cells. Our approach keeps outperforming others when M and K change. As we can see in Table 2, CDL performs poorly because it is a point-wise regression algorithm. Although CDL mines useful features from side information, it is still worse than the basic pair-wise ranking algorithm BPR. CDR and CDHR are better than BPR for introducing extra valuable side information. To focus on the comparison of CDR and CDHR, we can see that our approach CDHR outperforms CDR by a margin of 7.6%-10.8% on precision, 3.8%-13.3% on recall and 2.7%-13.1% on nDCG. The reason is that CDHR utilizes various side information, especially the price and brand information, to extract more valuable feature vectors of items for learning V.

Impact of M_w and λ_n

In this section, we illustrate the influence of side information from different sources on improving performance of recommendation. With other hyperparameters remaining unchanged, we show the impact of M_w and λ_n in Figure 3 and Figure 4. The size of M_w is 3×3, with the first diagonal element M_w^{middle} controlling the impact of middle feature vector, the second diagonal element M_w^{price} controls the impact of price side information and the third diagonal element M_w^{brand} controls the impact of brand side information. We set the value of λ_n to [0.001,0.01,0.1,1,10] separately and keep $M_w^{middle} = 1$ then vary M_w^{price} with $M_w^{brand} = 1$ or vice versa to see whether the import of new side information will benefit the performance. Because of space limitation, we only show the results of dataset Beauty. Given a fixed λ_v , the smaller λ_n is the more RAE domains the learning process of V. And M_w controls the effect of different inputs of RAE on the final learned feature vector.

As we can see in Figure 3 and Figure 4, when both M_w^{price} and M_w^{brand} approximate to 1, the RAE will degrade to SDAE and the performance of CDHR and CDR are similar. When the hyperparameter M_w^{price} and M_w^{brand} are tuned to values, CDHR becomes better than CDR. When λ_n is large, CDHR degenerates to two separate bayesian models which are loosely coupled. As a result, the performance of CDHR and CDR degrade significantly and CDHR becomes insensitive to both price and brand information.

Conclusion

In this paper, we propose the CDHR model which incorporates BPR with RAE in order to take advantage of several heterogeneous side information for item recommendation. We construct a specific RAE architecture in terms of the attributes of side information. The feature vectors extracted by the RAE can also be applied in other tasks besides recommendation. Empirical studies on six real-world datasets illustrate that our approach CDHR outperforms other competitors. Moreover, analysis on the impact of hyperparameters λ_n and M_w demonstrates that price and brand indeed have useful value.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (NSFC) No. 61672449, 61272303.

References

Bengio, Y.; Yao, L.; Alain, G.; and Vincent, P. 2013. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, 899–907. Curran Associates, Inc.

Chen, C.; Zheng, X.; Wang, Y.; Hong, F.; Lin, Z.; et al. 2014. Context-aware collaborative topic regression with social matrix factorization for recommender systems. In *AAAI*, volume 14, 9–15.

He, R., and McAuley, J. 2016. VBPR: Visual bayesian personalized ranking from implicit feedback. In *30th AAAI Conference on Artificial Intelligence*. AAAI.

Hofmann, T. 2003. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 259–266. ACM.

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *2008 8th IEEE International Conference on Data Mining*, 263–272. IEEE.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 452–461. AUAI Press.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, 285–295. ACM.

Sevil, S. G.; Kucuktunc, O.; Duygulu, P.; and Can, F. 2010. Automatic tag expansion using visual similarity for photo sharing websites. *Multimedia Tools and Applications* 49(1):81–99.

Socher, R.; Huang, E. H.; Pennin, J.; Manning, C. D.; and Ng, A. Y. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, 801–809. Curran Associates, Inc.

Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; and Manning, C. D. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 151–161. Association for Computational Linguistics.

Strichartz, R. S. 2003. *A guide to distribution theory and Fourier transforms*. World Scientific.

Su, X., and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009:4.

Van den Oord, A.; Dieleman, S.; and Schrauwen, B. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*, 2643–2651.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P.-A. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine learning*, 1096–1103. ACM.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; and Manzagol, P.-A. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research* 11(Dec):3371–3408.

Wang, C., and Blei, D. M. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 448–456. ACM.

Wang, H.; Wang, N.; and Yeung, D.-Y. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1235–1244. ACM.

Ying, H.; Chen, L.; Xiong, Y.; and Wu, J. 2016. Collaborative deep ranking: a hybrid pair-wise recommendation algorithm with implicit feedback. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 555–567. Springer.

Zhang, W.; Sun, H.; Liu, X.; and Guo, X. 2014. Temporal qos-aware web service recommendation via non-negative tensor factorization. In *Proceedings of the 23rd International Conference on World Wide Web*, 585–596. ACM.