Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

DELF: A Dual-Embedding based Deep Latent Factor Model for Recommendation

Weiyu Cheng, Yanyan Shen*, Yanmin Zhu, Linpeng Huang
Department of Computer Science and Engineering
Shanghai Jiao Tong University
{weiyu_cheng, shenyy, yzhu, lphuang}@sjtu.edu.cn

Abstract

Among various recommendation methods, latent factor models are usually considered to be state-of-the-art techniques, which aim to learn user and item embeddings for predicting user-item preferences. When applying latent factor models to recommendation with implicit feedback, the quality of embeddings always suffers from inadequate positive feedback and noisy negative feedback. Inspired by the idea of NSVD that represents users based on their interacted items, this paper proposes a dual-embedding based deep latent factor model named DELF for recommendation with implicit feedback. In addition to learning a single embedding for a user (resp. item), we represent each user (resp. item) with an additional embedding from the perspective of the interacted items (resp. users). We employ an attentive neural method to discriminate the importance of interacted users/items for dual-embedding learning. We further introduce a neural network architecture to incorporate dual embeddings for recommendation. A novel attempt of DELF is to model each user-item interaction with four deep representations that are subtly fused for preference prediction. We conducted extensive experiments on real-world datasets. The results verify the effectiveness of user/item dual embeddings and the superior performance of DELF on item recommendation.

1 Introduction

In the era of information explosion, users are often overwhelmed by numerous choices available online. Recommender systems play a significant role in filtering overloaded information and providing personalized services for users. Modern recommender systems are often based on Collaborative Filtering (CF), the key idea of which is to exploit past user-item interactions for modeling user preferences against items [Sarwar *et al.*, 2001]. Among various CF methods, latent factor models [Aggarwal and Parthasarathy, 2001] are widely used and considered to be the state-of-the-art solutions to recommendation [Aggarwal, 2016]. Latent factor models typically characterize users and items with feature vectors in the same latent space, and estimate each user-item preference based on the corresponding vectors. For example, Matrix Factorization (MF) [Paterek, 2007] directly computes the inner product of user and item vectors as the preference score, while more advanced neural network methods such as Neural Collaborative Filtering (NCF) [He *et al.*, 2017] leverage non-linear function to model the interaction between user and item latent factors, i.e., *embeddings*.

Early literature on latent factor models for recommendation [Koren, 2009; Rendle and Schmidt-Thieme, 2008] has mainly focused on explicit feedback, i.e., the ratings from users expressing their preferences over items. In those works, recommendation is formulated as a rating prediction problem, and the user and item embeddings are updated iteratively by minimizing the residual between the predicted and observed ratings. However, explicit ratings are typically difficult to acquire in many real applications, which inspires recommender systems to exploit more abundant implicit feedback from user behavior history. For example, a simple act of a user browsing a product or clicking a link can reflect the user's endorsement for the item. An important challenge of applying latent factor models to implicit feedback based recommendation is: *how to learn appropriate embeddings for users and items given scarce negative feedback?* Since all the observed interactions are positive implicit feedback, learning user and item embeddings with only positive feedback will result in significant overfitting [Hu *et al.*, 2008; Devooght *et al.*, 2015]. For instance, simply setting all the values in user and item embeddings to be $\frac{1}{\sqrt{K}}$ (K is the embedding dimension) can predict all user-item preferences to be positive and achieve 100% accuracy over the observed entries [Aggarwal, 2016].

To tackle this problem, a simple solution is to consider all the unobserved user-item interactions as negative feedback [Hu *et al.*, 2008]. Nevertheless, since not all the unobserved entries are true negative instances, the solution may hinder useful information from the limited observed interactions and consequently degrade the quality of the learned user and item embeddings [Pan *et al.*, 2008]. Recent works [Pan *et al.*, 2008; Pan and Scholz, 2009; He *et al.*, 2014; 2016; Volkovs and Yu, 2015] has proposed non-uniform weighting strategies and sampling schemes to carefully select negative

*Corresponding author

feedback from unobserved interactions. However, all these methods represent each user or item with a single embedding which is learned progressively with the prepared training instances, and hence the quality of the embeddings still suffers from inadequate positive feedback or noisy negative feedback.

We notice that NSVD [Paterek, 2007] is proposed to parameterize users according to the items that they have rated. In NSVD, a user embedding is determined by the embeddings of all the items interacted with the user, which is not affected by negative feedback and more robust to the number of user interactions. This inspires us to represent users based on the characteristics of their interacted items for recommendation with implicit feedback. To be more specific, in addition to model primitive user embeddings, we propose obtaining additional item-based user embeddings following the idea of NSVD. Furthermore, NSVD simply averages item embeddings to represent users, but different items may contribute differently to user modeling. For instance, an extremely popular item should be less effective to reflect a user's preference since almost everyone likes it. We thus incorporate the attention mechanism [Bahdanau *et al.*, 2014] to discriminate the importance of different interacted items automatically, and the item-based user embeddings can be computed by aggregating item embeddings with non-uniform weights. Likewise, we are able to obtain user-based item embeddings to characterize items from users' perspective. The *dual embeddings* associated with each user and item collaboratively populate their latent representations in a more accurate way, and the advantages of the dual-embedding based recommendation are also verified in our experiments.

To summarize, this paper proposes a Dual-Embedding based deep Latent Factor Model, named DELF, for recommendation with implicit feedback. We introduce an attentive neural method to construct an item-based (resp. user-based) embedding for each user (resp. item), which discriminates the importance of different users and items automatically. We then combine these embeddings with the primary ones to model non-linear user-item interactions using a deep neural network architecture. A novel attempt of DELF is that we employ dual embeddings to learn four kinds of deep interactions for each user-item pair, which enables DELF to generalize two principled CF methods, i.e., NCF and NSVD. To the best of our knowledge, this work is the first neural approach that leverages dual user and item embeddings for recommendation with implicit feedback. We conducted extensive experiments on real-world datasets. The results demonstrate (1) DELF outperforms the state-of-the-art methods in achieving better performance for item recommendation, and (2) exploiting dual embeddings is effective for modeling user-item preferences for recommendation in the presence of noisy negative signals from implicit feedback.

2 Preliminaries

We consider a user-item interaction matrix $\mathbf{R} \in \mathbb{R}^{M \times N}$ from users' implicit feedback, where M and N are the number of users and items, respectively. $R_{ui} = 1$ indicates an interaction between user u and item i, and $R_{ui} = 0$ means no interaction is observed. We denote by $\mathcal{R} = \{(u, i) \mid R_{ui} = 1\}$ the set of all the observed interactions. The problem of recommendation with implicit feedback is to estimate preference scores $\hat{R}_{ui}$ for the unobserved entries in $\mathbf{R}$.

2.1 Neural Collaborative Filtering

Latent factor models such as Matrix Factorization (MF) typically represent each user/item with a real-valued vector of latent features. We denote by $\mathbf{p}_u$ and $\mathbf{p}_i$ the vectors for user u and item i in a joint K-dimensional latent space, respectively. The preference score $\hat{R}_{ui}$ between u and i is computed by the inner product of $\mathbf{p}_u$ and $\mathbf{q}_i$:

$$\hat{R}_{ui} =< \mathbf{p}_u, \mathbf{q}_i >= \mathbf{p}_u^T \mathbf{q}_i \tag{1}$$

The inner product operation linearly aggregates the multiplications of pairwise latent features, which is insufficient to capture complex user-item interactions. Neural Collaborative Filtering (NCF) [He *et al.*, 2017] is thus proposed to learn non-linear interaction function via a multi-layer perceptron (MLP) :

$$\hat{R}_{ui} = \phi_X(...\phi_2(\phi_1(\mathbf{z}_0))...) \tag{2}$$

$$\mathbf{z}_0 = \mathbf{p}_u \oplus \mathbf{q}_i \tag{3}$$

$$\phi_l(\mathbf{z}_{l-1}) = \delta_l(\mathbf{W}_l \mathbf{z}_{l-1} + \mathbf{b}_l), \ \ l \in [1, X] \tag{4}$$

where X is the number of hidden layers in the neural network, $\oplus$ is the concatenation of vectors, $\mathbf{W}_l$, $\mathbf{b}_l$ and δ_l are the weight matrix, bias vector and non-linear activation function for the l-th layer, respectively. Note that the original NCF paper ensembles MLP and MF to obtain the NeuMF model. In this paper, we focus on developing single CF model for recommendation, while the proposed model can be ensembled with other models to achieve better performance.

2.2 NSVD & SVD++

Instead of parameterizing each user explicitly, NSVD [Paterek, 2007] models users based on the items they have rated. Formally, each item is associated with two latent vectors $\mathbf{q}_i$ and $\mathbf{y}_i$. The preference score of user u to item i is estimated as:

$$\hat{R}_{ui} = \mathbf{q}_i^T \frac{\sum_{j \in R(u)} \mathbf{y}_j}{\sqrt{|R(u)|}} + b_u + b_i \tag{5}$$

where $R(u)$ is the set of items rated by user u, b_u and b_i are bias terms. NSVD reduces the redundancy of user factors by representing users with a linear combination of item factors. However, the main issue of NSVD is that two users who have rated the same set of items with entirely different ratings are tied to have the same representation. To address the problem, SVD++ [Koren, 2008] is proposed for recommendation with explicit ratings, which estimates user-item preferences as follows:

$$\hat{R}_{ui} = \mathbf{q}_i^T \left(\mathbf{p}_u + \frac{\sum_{j \in R(u)} \mathbf{y}_j}{\sqrt{|R(u)|}} \right) + b_u + b_i \tag{6}$$

where $\mathbf{p}_u$ is a user latent factor. SVD++ leverages the NSVD-based representation to adjust the user latent factor rather than represent the user. We observe that NSVD-based latent factors are determined by users' rated items, which are useful to avoid false negatives from noisy implicit feedback and more robust than explicitly parameterized factors. In this paper, we propose to learn dual embeddings for both users and items. Instead of simply performing a summation over user/item

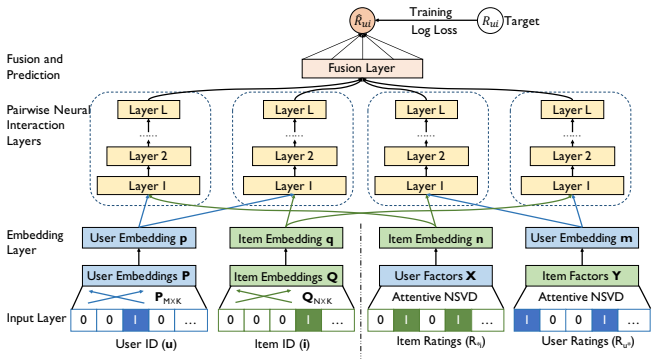


Figure 1: Dual-Embedding based Deep Latent Factor Model

dual embeddings, we employ neural networks to model deep representations for users/items as well as complex user-item interactions.

3 DELF

In this section, we present our DELF model that combines the strengths of dual embeddings and deep neural networks for recommendation.

3.1 Model

DELF is generally based on the conventional framework for latent factor models, which assumes that $\hat{R}_{ui}$ can be generated by the underlying user and item latent factors, i.e.,

$$\hat{R}_{ui} = f(u, i|\Theta) \tag{7}$$

where Θ denotes latent factors of u and i, and f denotes the interaction function. Figure 1 illustrates the design of Θ and f in DELF, the details of which are provided as follows.

Input Layer. The input layer of DELF consists of feature vectors for user u and item i. Single-embedding based latent factor models simply associate u and i with their one-hot representations $\mathbf{u}$ and $\mathbf{i}$. In addition to the one-hot vectors, DELF also incorporates the binary interaction vectors $\mathbf{R}_{u*}$ and $\mathbf{R}_{*i}$ from the observed interactions for u and i, respectively. To this end, we obtain two kinds of feature vectors for both u and i from the input layer.

Embedding Layer. The embedding layer projects each feature vector from the input layer into a dense vector representation. The primitive feature vector embeddings (i.e., $\mathbf{u}$ and $\mathbf{i}$) can be obtained by referring to the embedding matrix as follows.

$$\mathbf{p}_u = \mathbf{P}^T \mathbf{u} \tag{8}$$

where $\mathbf{P} \in \mathbb{R}^{M \times K}$ denotes the user embedding matrix, and K is the dimension of user embeddings. Similarly, $\mathbf{q}_i$ can be obtained from the item embedding matrix $\mathbf{Q}$.

As for the interaction vectors $\mathbf{R}_{u*}$ and $\mathbf{R}_{*i}$, we employ an attentive method to learn the corresponding embedding representations. Recall that NSVD models each user as a linear combination of the interacted item factors. Formally, items are associated with another group of latent factors $\mathbf{y}_i$, and the *item-based user embedding* is calculated by $\frac{\sum_{i \in R(u)} \mathbf{y}_i}{\sqrt{|R(u)|}}$, where $R(u)$ denotes the collection of all the positive entries in $\mathbf{R}_{u*}$. NSVD averages the factors of rated items to represent a user. However, different items can reflect user preference in different degrees. Therefore, we employ the attention mechanism [Bahdanau *et al.*, 2014] to discriminate the importance of the interacted items automatically, as defined below:

$$\mathbf{m}_u = \sum_{i \in R(u)} \alpha_i \mathbf{y}_i \tag{9}$$

where $\mathbf{m}_u$ is the item-based user embedding, and α_i is the attention score for item i rated by user u. Here we parameterize the attention score for item i by:

$$\mathbf{h}_i = \tanh(\mathbf{W}_a \mathbf{y}_i + \mathbf{b}_a) \tag{10}$$

$$\alpha_i = \frac{\exp(\mathbf{h}_i^T \mathbf{h}_a)}{\sum_{i \in R(u)} \exp(\mathbf{h}_i^T \mathbf{h}_a)} \tag{11}$$

where $\mathbf{W}_a$, $\mathbf{b}_a$ denote the weight matrix and bias vector respectively, and $\mathbf{h}_a$ is a context vector. That is, we first feed the item factor $\mathbf{y}_i$ to a one-layer MLP that produces $\mathbf{h}_i$ as a latent representation for $\mathbf{y}_i$. We then measure the importance of item i based on the similarity between $\mathbf{h}_i$ and a context vector $\mathbf{h}_a$ and derive a normalized importance weight α_i via a softmax function. Likewise, we can represent the item with user factors as follows:

$$\mathbf{n}_i = \sum_{u \in R(i)} \alpha_u \mathbf{x}_u \tag{12}$$

where we aggregate a set of user factors $\mathbf{x}$ to parameterize the *user-based item embedding* $\mathbf{n}_i$, and α_u can be computed in a similar way as in Equation (10) and (11). Note that given additional content information, more complex attention mechanisms [Chen *et al.*, 2017] can be incorporated into DELF, which are orthogonal to our approach.

Pairwise Neural Interaction Layers. We feed the dual embeddings into the pairwise neural interaction layers to model feature interactions between u and i. Instead of using a single network structure, we model interactions for the two kinds of user/item embeddings separately, and obtain four deep representations for different embedding interactions. Formally,

$$\mathbf{h}^j = \phi_L^j(...\phi_2^j(\phi_1^j(\mathbf{z}_0[j]))...) \tag{13}$$

$$\phi_l^j = \delta_l^j(\mathbf{W}_l^j \mathbf{z}_{l-1}^j + \mathbf{b}_l^j), \ \ l \in [1, L] \tag{14}$$

$$\mathbf{z}_0 = [\mathbf{p}_u \oplus \mathbf{n}_i, \mathbf{p}_u \oplus \mathbf{q}_i, \mathbf{m}_u \oplus \mathbf{n}_i, \mathbf{m}_u \oplus \mathbf{q}_i] \tag{15}$$

where $j \in \{1, 2, 3, 4\}$; $\mathbf{h}^j$ is the deep representation of embedding interaction learned by the j-th feedforward neural network; ϕ_l^j is the l-th layer in network j; W_l^j, b_l^j and δ_l^j denote the weight matrix, bias vector and activation function of layer l in network j, respectively; $\mathbf{z}_0$ includes pairwise concatenations of user and item dual embeddings. An insight of DELF is that the primitive and additional embeddings should be of varying importance to the final preference score under different circumstances. For example, for users with few interactions, $\mathbf{p}_u$ can be trivial by learning from very limited true positive instances. Modeling embedding interactions separately avoids two kinds of embeddings from affecting each other and hence may benefit the prediction result. In DELF, we choose Rectifier ReLU as the activation function by default if not otherwise specified, which is proven to be non-saturated and yields good performance in deep networks [Glorot *et al.*, 2011]. As for the network structure, we

follow the setting proposed by [He *et al.*, 2017] and employ a tower structure for each network, where higher layers have smaller number of neurons.

Fusion and Prediction. The fusion layer is above the pairwise neural interaction layers, which combines four deep representations of embedding interactions into a single one. We propose two fusion schemes: MLP and an empirical scheme. For MLP, the combined feature after the fusion layer is formulated as:

$$\mathbf{h}_f = \delta_f(\mathbf{W}_f \mathbf{z}_f + \mathbf{b}_f) \tag{16}$$

$$\mathbf{z}_f = \mathbf{h}^1 \oplus \mathbf{h}^2 \oplus \mathbf{h}^3 \oplus \mathbf{h}^4 \tag{17}$$

where $\mathbf{W}_f$, $\mathbf{b}_f$, δ_f are the weight matrix, bias vector and activation function, respectively; $\mathbf{z}_f$ is the concatenation of four latent interaction representations. We dub this model "DELF-MLP". The empirical scheme follows our observation that primitive embeddings $\mathbf{p}_u$ and $\mathbf{q}_i$ should be less expressive with fewer ratings but yield good performance with enough true instances. Hence, we empirically assign non-uniform weights to four deep representations. Formally, for user u and item i, we have:

$$\alpha = \min(\frac{\sqrt{R(u)}}{\lambda_u}, 1), \quad \beta = \min(\frac{\sqrt{R(i)}}{\lambda_i}, 1) \tag{18}$$

$$\mathbf{h}_f = \alpha(1-\beta)\mathbf{h}^1 + \alpha\beta\mathbf{h}^2 + (1-\alpha)(1-\beta)\mathbf{h}^3 + (1-\alpha)\beta\mathbf{h}^4 \tag{19}$$

where λ_u and λ_i are hyper-parameters to be tuned via the validation set. The empirical scheme assigns weights to different networks according to the number of interactions from u and i, and the unified representation $\mathbf{h}_f$ is computed by the summation of $\mathbf{h}^j$. We dub this model "DELF-EF".

At last, the output $\mathbf{h}_f$ of the fusion layer is transformed to the final prediction score:

$$\hat{R}_{ui} = \delta_p(\mathbf{W}_p \mathbf{h}_f + b_f) \tag{20}$$

where $\mathbf{W}_p$, b_f are the weight matrix and bias term, respectively; δ_p is the sigmoid function as we expect the prediction score to be in the range of $[0, 1]$.

It is worthy noticing that both NCF and NSVD can be interpreted as special cases of our DELF framework. By setting $\mathbf{z}_f = \mathbf{h}^2$ in MLP scheme, or $\alpha = \beta = 1$ in empirical scheme, we obtain exactly the same model as NCF. By setting $\mathbf{z}_f = \mathbf{h}^4$ or $\alpha = 0, \beta = 1$ empirically, we can recover NSVD thanks to the universal approximation of neural networks to the inner product function [Hornik *et al.*, 1989].

3.2 Learning

Both point-wise and pair-wise objective functions are widely used in recommender systems. In this work, we employ point-wise objective function for simplicity and leave the other one as future work. Due to the one-class nature of implicit feedback, we follow [He *et al.*, 2017] to use the binary cross-entropy loss, which is defined as:

$$L_{\log} = -\sum_{(u,i)\in\mathcal{R}\bigcup\mathcal{R}^-} R_{ui}\log(\hat{R}_{ui}) + (1-R_{ui})\log(1-\hat{R}_{ui}) \tag{21}$$

where $\mathcal{R}\bigcup\mathcal{R}^-$ includes all true positive instances and sampled negative instances. To optimize the objective function, we adopt Adam, a variant of Stochastic Gradient Descent (SGD) that dynamically tunes the learning rate during training process and leads to faster convergence [Kingma and Ba, 2014]. Due to the non-convexity of the objective function, gradient-based optimization methods are easily trapped in local optimal solutions. For the experiments, we pre-train DELF with MLP to initialize primitive user and item embeddings in DELF. This empirically leads to faster convergence and better performance.

Dataset	Interaction#	User#	Item#	Sparsity
Movielens	1,000,209	6,040	3,706	95.53%
Amazon	75,932	1,835	41,488	99.90%

Table 1: Statistics of the Datasets

4 Experiments

In this section, we conduct experiments with the aim of answering the following research questions:

RQ1 How does our approach perform compared with the state-of-the-art recommendation methods?
RQ2 Are the key components in DELF (i.e., attentive module, pairwise neural interaction layers) useful for improving recommendation results?
RQ3 How dose the performance of DELF vary with different values of the hyper-parameters?

4.1 Experimental Settings

• **Datasets.** We conducted experiments using two public datasets: Movielens 1M[1] and Amazon Music[2]. We transformed both datasets to implicit feedback, where each entry is marked as 0 or 1 denoting whether the user has rated the item. Following previous work [He *et al.*, 2017], we filtered the datasets to retain users with at least 20 interactions. The statistics of the two datasets are summarized in Table 1.

• **Evaluation Protocol.** To evaluate the recommendation performance, we employed the widely used leave-one-out evaluation [Rendle *et al.*, 2009; He *et al.*, 2017; Bayer *et al.*, 2017]. We held-out the latest interaction of each user as the test set, and collected the second latest interactions as the validation set. The remaining data were used for training. Since it is time-consuming to rank all items for each user during evaluation, we followed the common strategy to randomly sample 100 items that are not interacted with the user. Each test item is ranked among the 100 items to generate a ranked list. We used Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) [He *et al.*, 2015] as metrics. The ranked list is truncated at 10 for both metrics.

• **Compared Methods.** We compared our methods, DELF-MLP and DELF-EF, with the following methods:

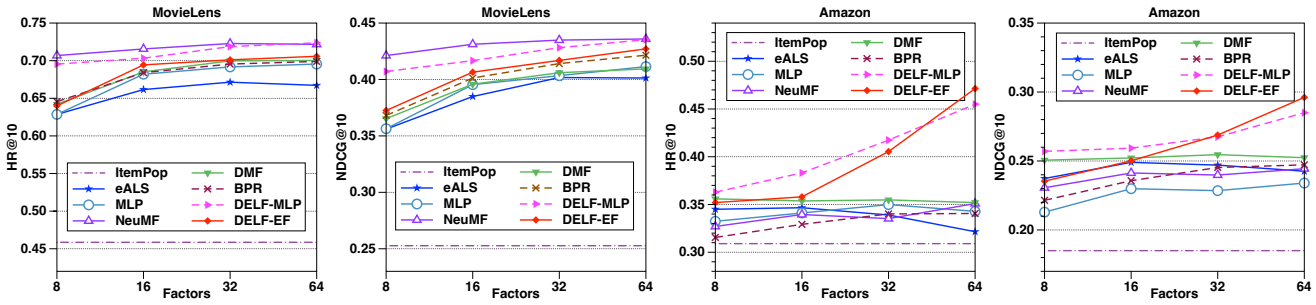
- **ItemPop**. This method simply ranks items by the number of interactions, which is a non-personalized method.

- **eALS** [He *et al.*, 2016]. This is a MF method that treats all unobserved interactions as negative instances and assigns non-uniform weights to them based on item popularity.

- **BPR** [Rendle *et al.*, 2009]. This method optimizes the MF model of Equation 1 with a pairwise ranking loss, which is tailored to learn from implicit feedback.

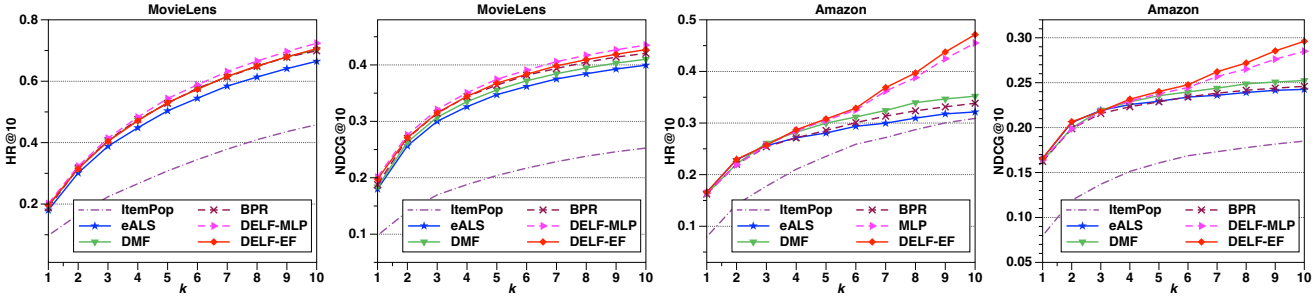
[1] https://grouplens.org/datasets/movielens/1m/

[2] http://jmcauley.ucsd.edu/data/amazon/



(a) Movielens — HR@10 (b) Movielens — NDCG@10 (c) Amazon — HR@10 (d) Amazon — NDCG@10

Figure 2: Performance of HR@10 and NDCG@10 w.r.t. the number of predictive factors on the two datasets



(a) Movielens — HR@k (b) Movielens — NDCG@k (c) Amazon — HR@k (d) Amazon — NDCG@k

Figure 3: Evaluation of Top-k item recommendation where k ranges from 1 to 10 on the two datasets

- **MLP** [He *et al.*, 2017]. It uses MLP to learn the non-linear interaction function of embeddings from data.

- **NeuMF** [He *et al.*, 2017]. This is a state-of-the-art latent factor model. It pre-trains MLP and MF separately, and then ensembles both models to predict the final preference score.

- **DMF** [Xue *et al.*, 2017]. This is a state-of-the-art MF model for item recommendation. It directly maps the rating vectors of users and items into a common low-dimensional space with non-linear projections.

• **Parameter Settings.** We implemented our proposed methods based on Tensorflow. We tuned all the hyper-parameters on the validation set. We sampled four negative instances per positive instance. We used the batch size of 256 and the learning rate of 0.001. The size of the last hidden layer is termed as predictive factors [He *et al.*, 2017] and we evaluated the factors in $\{8, 16, 32, 64\}$. We employed three hidden layers for each feedforward network. For example, if the size of predictive factors is 8, the pairwise interaction layers follow $32 \rightarrow 16 \rightarrow 8$, and the size of the fused representation is 8.

4.2 Performance Comparison (RQ1)

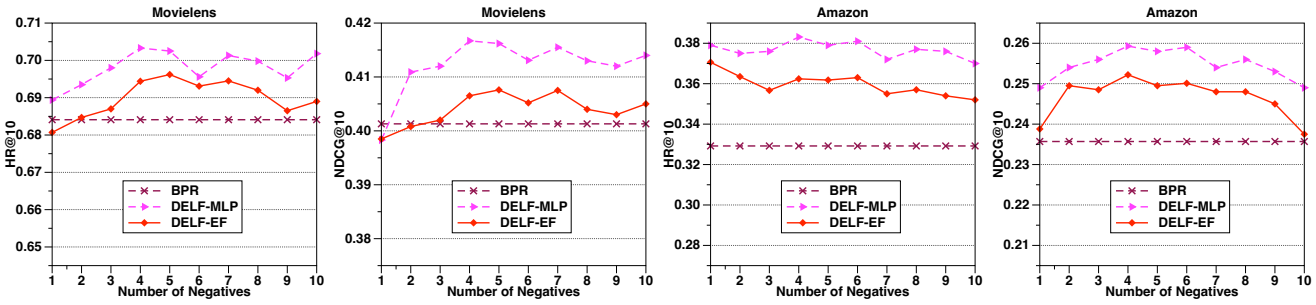
Figure 2 shows the performance of HR@10 and NDCG@10 with respect to the size of predictive factors. For BPR and eALS, the number of predictive factors is equal to the dimension of latent factors.

First, we can see that our DELF methods achieve the best overall performance on both datasets. For Movielens, DELF-MLP shows similar performance to NeuMF, and the best results of HR and NDCG outperform BPR and DMF with a relative improvement of 3.9% and 5.3%, respectively. Note that NeuMF is an ensemble method that fuses MF and MLP, while all the other models including DELF are single CF models. Achieving similar performance to NeuMF demonstrates the competitive performance of our approach. For Amazon, DELF outperforms other methods by a larger margin (on average, the relative improvement of DELF-EF and DELF-MLP over DMF is 8.0% and 10.1%, respectively). This may be caused by the higher sparsity of Amazon dataset than Movielens. Since our method utilizes dual-embeddings to represent users/items, they are more robust to the number of user/item interactions and can better handle the sparsity problem. Besides, DELF-MLP generally performs better than DELF-EF, while DELF-EF is able to get good results on Amazon. This shows that empirically assigning non-uniform weights to embedding interactions based on the number of user/item ratings is effective on the sparse dataset.

Among baseline methods, NeuMF outperforms other models on Movielens, and DMF performs best on Amazon. NeuMF explicitly learns user and item embeddings and yields good performance when more positive instances are provided. DMF implicitly obtains embeddings by learning a non-linear projection from rating vectors to latent representations, and can be less sensitive to the sparsity of datasets. Our approach integrates the advantage of both methods, which reserves primitive embeddings and utilizes interaction vectors to populate user/item representations, thus achieving good performance in both datasets. Moreover, DELF outperforms MLP significantly on both datasets, which further illustrates the effectiveness of dual embeddings for recommendation.

4.3 Effects of Key Components (RQ2)

We compare two variants of DELF with the proposed DELF-MLP and DELF-EF. DELF-ni removes the pairwise interaction layers and employs a unified network architecture. For



(a) Movielens — HR@10 (b) Movielens — NDCG@10 (c) Amazon — HR@10 (d) Amazon — NDCG@10

Figure 4: Performance of DELF methods w.r.t. the number of negative samples per positive instance (factors=16). We also show the performance of BPR, which uses pairwise ranking loss (i.e., only one negative instance is paired with a positive instance for learning)

Factors	DELF-MLP	DELF-EF	DELF-ni	DELF-na
Movielens				
8	**0.695**	0.640	0.672	0.680
16	**0.703**	0.694	0.693	0.688
32	**0.719**	0.701	0.704	0.705
64	**0.724**	0.706	0.713	0.709
Amazon				
8	**0.363**	0.352	0.356	0.355
16	**0.383**	0.358	0.362	0.374
32	**0.417**	0.405	0.390	0.409
64	0.455	**0.471**	0.423	0.442

Table 2: HR@10 of different variants of DELF

Factors	DELF-MLP	DELF-EF	DELF-ni	DELF-na
Movielens				
8	**0.407**	0.373	0.389	0.398
16	**0.417**	0.407	0.409	0.407
32	**0.428**	0.417	0.416	0.419
64	**0.435**	0.424	0.426	0.426
Amazon				
8	**0.257**	0.235	0.252	0.250
16	**0.259**	0.250	0.247	0.249
32	0.268	**0.269**	0.259	0.258
64	0.285	**0.296**	0.264	0.277

Table 3: NDCG@10 of different variants of DELF

DELF-ni, we set the number of layers and the number of neurons in each layer to be the same as those in DELF-MLP for a fair comparison. We also evaluate the effects of attention mechanism with DELF-na, where we simply average item (resp. user) factors to get item-based user (resp. user-based item) embeddings, based on the MLP scheme.

The results are provided in Table 2 and 3. We observe that our proposed methods achieve the best performance on both datasets with respect to different numbers of factors. Specifically, the best results of our DELF methods outperform DELF-ni with a relative improvement of 4.4%, demonstrating the advantage of pairwise embedding interactions instead of using a unified network structure. DELF-MLP outperforms DELF-na with a relative improvement of 2.1%, indicating that employing non-uniform weights on items/users improves NSVD to obtain a better latent representation.

4.4 Hyper-parameter Investigation (RQ3)

We first report the results on Top-k recommended lists where k ranges from 1 to 10 in Figure 3. We keep the size of predictive factors to be 64. To make the figure content clearer, we only include the results of four representative baselines. Results on Movielens show that DELF-MLP keeps consistent improvements over other methods for all the values of k (on average, the relative improvement of DELF-MLP over DMF is 4.2%). For Amazon, DELF methods outperform baselines with significant improvements for k larger than 3. Among baseline models, DMF performs best while BPR slightly performs better than eALS, and all of them outperforms ItemPop significantly. This demonstrates the necessity of modeling characteristics of users and items, instead of just recommending popular items.

We illustrate the impact of negative sampling ratio for DELF methods in Figure 4. It can be clearly seen that employing more than one negative instance is beneficial to recommendation performance. For Movielens, a sampling ratio larger than 4 helps DELF methods to get the better results. For Amazon, the optimal negative sampling ratio is between 3 and 8. The results are consistent with previous works [Xue *et al.*, 2017; He *et al.*, 2017].

5 Conclusion and Future Work

In this paper, we propose a novel deep latent factor model with dual embeddings for recommendation. In addition to the primary user and item embeddings, we employ an attentive neural method to obtain additional embeddings for users and items based on their interaction vectors from implicit feedback. We introduce a neural network architecture to learn deep representations for pairwise interactions among dual user/item embeddings and subtly fuse the latent interactions to predict the preference score. Extensive experiments on two real-world datasets demonstrate the superior performance of our proposed model compared with the state-of-the-art methods.

In the future, we plan to extend DELF to incorporate auxiliary information. Auxiliary information such as social relations, user review and knowledge base can be utilized to characterize users/items from different perspectives. Besides, negative instances can be assigned with non-uniform weights for optimization. Thus we plan to study the effects of different weighting schemes on our method.

Acknowledgments

This work was supported in part by 973 Program (No. 2014CB340303), NSFC (No. 61772341, 61472254, 61572324, 61170238, 61602297 and 61472241) and the Shanghai Municipal Commission of Economy and Informatization (No. 201701052). This work was also supported by the Program for Changjiang Young Scholars in University of China, the Program for China Top Young Talents, and the Program for Shanghai Top Young Talents.

References

[Aggarwal and Parthasarathy, 2001] Charu C. Aggarwal and Srinivasan Parthasarathy. Mining massively incomplete data sets by conceptual reconstruction. In *SIGKDD*, pages 227–232, 2001.

[Aggarwal, 2016] Charu C. Aggarwal. *Recommender Systems - The Textbook*. Springer, 2016.

[Bahdanau *et al.*, 2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[Bayer *et al.*, 2017] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. A generic coordinate descent framework for learning from implicit feedback. In *WWW*, pages 1341–1350, 2017.

[Chen *et al.*, 2017] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *SIGIR*, pages 335–344. ACM, 2017.

[Devooght *et al.*, 2015] Robin Devooght, Nicolas Kourtellis, and Amin Mantrach. Dynamic matrix factorization with priors on unknown values. In *SIGKDD*, pages 189–198. ACM, 2015.

[Glorot *et al.*, 2011] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, pages 315–323, 2011.

[He *et al.*, 2014] Xiangnan He, Ming Gao, Min-Yen Kan, Yiqun Liu, and Kazunari Sugiyama. Predicting the popularity of web 2.0 items based on user comments. In *SIGIR*, pages 233–242. ACM, 2014.

[He *et al.*, 2015] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. Trirank: Review-aware explainable recommendation by modeling aspects. In *CIKM*, pages 1661–1670, 2015.

[He *et al.*, 2016] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*, pages 549–558. ACM, 2016.

[He *et al.*, 2017] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017.

[Hornik *et al.*, 1989] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[Hu *et al.*, 2008] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272, 2008.

[Kingma and Ba, 2014] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Koren, 2008] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*, pages 426–434. ACM, 2008.

[Koren, 2009] Yehuda Koren. Collaborative filtering with temporal dynamics. In *SIGKDD*, pages 447–456, 2009.

[Pan and Scholz, 2009] Rong Pan and Martin Scholz. Mind the gaps: weighting the unknown in large-scale one-class collaborative filtering. In *SIGKDD*, pages 667–676, 2009.

[Pan *et al.*, 2008] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *ICDM*, pages 502–511, 2008.

[Paterek, 2007] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[Rendle and Schmidt-Thieme, 2008] Steffen Rendle and Lars Schmidt-Thieme. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *Recsys*, pages 251–258, 2008.

[Rendle *et al.*, 2009] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[Sarwar *et al.*, 2001] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295. ACM, 2001.

[Volkovs and Yu, 2015] Maksims Volkovs and Guang Wei Yu. Effective latent models for binary feedback in recommender systems. In *SIGIR*, pages 313–322, 2015.

[Xue *et al.*, 2017] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. Deep matrix factorization models for recommender systems. In *IJCAI*, pages 3203–3209, 2017.