Efficient Heterogeneous Collaborative Filtering without Negative Sampling for Recommendation

Chong Chen [1], Min Zhang [1], Weizhi Ma [1], Yongfeng Zhang [2], Yiqun Liu[1] and Shaoping Ma [1]
[1] Department of Computer Science and Technology, Institute for Artificial Intelligence,
Beijing National Research Center for Information Science and Technology, Tsinghua University
[2] Department of Computer Science, Rutgers University
cc17@mails.tsinghua.edu.cn, z-m@tsinghua.edu.cn

Abstract

Recent studies on recommendation have largely focused on exploring state-of-the-art neural networks to improve the expressiveness of models, while typically apply the Negative Sampling (NS) strategy for efficient learning. Despite effectiveness, two important issues have not been well-considered in existing methods: 1) NS suffers from dramatic fluctuation, making sampling-based methods difficult to achieve the optimal ranking performance in practical applications; 2) although heterogeneous feedback (e.g., view, click, and purchase) is widespread in many online systems, most existing methods leverage only one primary type of user feedback such as purchase. In this work, we propose a novel non-sampling transfer learning solution, named Efficient Heterogeneous Collaborative Filtering (EHCF) for Top-N recommendation. It can not only model fine-grained user-item relations, but also efficiently learn model parameters from the whole heterogeneous data (including all unlabeled data) with a rather low time complexity. Extensive experiments on three real-world datasets show that EHCF significantly outperforms state-of-the-art recommendation methods in both traditional (single-behavior) and heterogeneous scenarios. Moreover, EHCF shows significant improvements in training efficiency, making it more applicable to real-world large-scale systems. Our implementation has been released [1] to facilitate further developments on efficient whole-data based neural methods.

Introduction

Recommender systems are designed to help deal with the information explosion problem and have been applied to various scenarios (Ricci, Rokach, and Shapira 2011; Chen et al. 2018b). In online information platforms, users can interact with items in various types of behavior. Figure 1 shows an example of heterogeneous user behaviors on E-commerce scenarios. Users can view an item, add an item to shopping cart, or purchase an item, etc. These heterogeneous behaviors provide valuable signals of users preferences, which are helpful for building a fine-grained recommender system (Gao et al. 2019).

[1]https://github.com/chenchongthu/EHCF

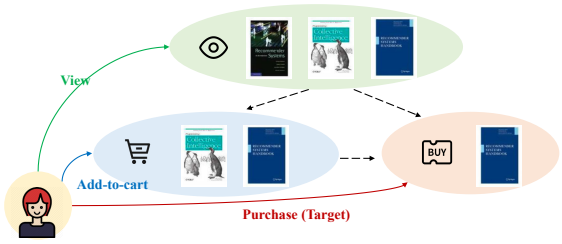

Figure 1: An example of multiple types of user feedback (not limited to these behaviors). Generally there are strong transfer relations among different behaviors (dotted lines).

In recent years, many recommendation models have been proposed and shown to be effective. However, two important issues have not been well-considered in existing models. First, most state-of-the-art methods, especially deep learning models, rely on Negative Sampling (NS) to increase computational efficiency (Chen et al. 2019a; Gao et al. 2019; He et al. 2017). While previous studies (He et al. 2017; Wang et al. 2018; Chen et al. 2019b) have shown that the performance of NS is not robust as it is highly sensitive to the sampling distribution and the number of negative samples. Essentially, sampling is biased, making it difficult to converge to the optimal ranking performance regardless of how many update steps have been taken (Xin et al. 2018). Besides, to leverage heterogeneous user behavior, NS strategy needs to sample a negative instance for every observed interaction (regardless of the behavior type), which produces a very large randomness in total (about K times than single-behavior scenario where K is the number of behavior types). As such, it is more difficult for sampling-based methods to achieve the optimal performance due to the explosive growth of sampling randomness.

Second, in previous work, there is a lack of in-depth investigation of relationships between user behavior types. Specifically, in heterogeneous scenarios, each behavior has its own contexts and there exist strong transfer relations among different behaviors (Gao et al. 2019). An example is shown in Figure 1 by dotted lines, the behaviors may represent the action sequence of a user on an item: view should

happen prior to adding-to-cart and purchase, while purchase may happen after adding-to-cart or view. However, existing heterogeneous recommendation models either focus on extending the Matrix Factorization (MF) methods to perform multiple learning of different behaviors (Tang et al. 2016; Krohn-Grimberghe et al. 2012; Singh and Gordon 2008), or change the negative sampling strategy to enrich the training set from the auxiliary behavioral data (Ding et al. 2018; Loni et al. 2016; Qiu et al. 2018). As a result, these models have largely ignored the translation relationship among different behaviors.

In light of the above limitations, we propose a novel model named Efficient Heterogeneous Collaborative Filtering (EHCF). Particularly, we apply non-sampling optimization for our neural model, which is more effective and stable due to the consideration of all samples in each parameter update (Hu, Koren, and Volinsky 2008; He et al. 2016). To cope with the efficiency challenges caused by non-sampling strategy, we develop an efficient optimization method. Through rigorous mathematical analysis, we resolve computational bottlenecks in optimization by leveraging the sparsity of positive-only data. Moreover, to incorporate the behavior contexts, we link the model prediction of each behavior in a transfer manner. The prediction of a high-level behavior (i.e., purchase) is influenced by transferring the prediction of the low-level behavior (i.e., view). Through this way, the proposed model can capture the underlying contexts of each behavior. The main contributions of this work are as follows:

- We derive an efficient optimization method that solves the challenging problem of learning neural models from the whole data with a controllable time complexity.
- We propose a novel neural model named EHCF for recommendation, which correlates the prediction of each behavior in a transfer way to capture the complicated relations among different behaviors.
- Extensive experiments on three real-world recommendation datasets show that EHCF not only outperforms the state-of-the-art models in both traditional single-behavior and heterogeneous scenarios, but also has a rather fast training process.

Related Work

Heterogeneous Collaborative Filtering Heterogeneous Collaborative Filtering (CF) or multi-behavior CF (Loni et al. 2016) is an emerging branch in the research community of recommender systems. In previous work, Singh and Gordon (Singh and Gordon 2008) propose Collective Matrix Factorization model (CMF) to simultaneously factorize multiple user-item interactions with sharing item-side embeddings across matrices. CMF is then extended to leverage multiple user behaviors for recommender systems (Krohn-Grimberghe et al. 2012; Zhao et al. 2015). Recently, Gao et al. (Gao et al. 2019) propose a Neural Multi-Task Recommendation (NMTR) model, which combines the recent advances of NCF (He et al. 2017) and the efficacy of Multi-Task Learning (MTL) (Argyriou, Evgeniou, and Pontil 2007) to exploit heterogeneous user behaviors. Another line of research approaches heterogeneous problem from the perspective of learning (Ding et al. 2018; Loni et al. 2016; Qiu et al. 2018). Specifically, Loni et al. (Loni et al. 2016) propose an extension of Bayesian Personalized Ranking (BPR) (Rendle et al. 2009) to adapt the sampling rule from different types of behavior in training. Ding et al. (Ding et al. 2018) develop a margin-based pairwise learning framework when view-data is available. Through the literature review above, it can be found that the existing approaches proposed to handle heterogeneous feedback recommendation tasks are relatively simple. There is a lack of in-depth investigation of relationships between user feedback types, which is one of the main concerns of our EHCF model.

Model Learning from Positive-only Data Learning sparse features from positive-only data is a fundamental task. Generally, there are two strategies to learn from positive-only data: 1) NS strategy (Chen et al. 2019a; He et al. 2017; Rendle et al. 2009) that samples negative instances from unlabeled data; 2) whole-data based strategy (Hu, Koren, and Volinsky 2008; Pan et al. 2008; Liang et al. 2016) that sees all the unlabeled data as negative. Both strategies have pros and cons: NS strategy is more efficient, but may decrease the model's performance; whole-data based strategy leverages the full data with a potentially better coverage, but inefficiency can be an issue. To the best of our knowledge, existing deep learning based recommendation methods almost all rely on NS for efficient model learning. In previous work, some efforts have been devoted to resolving the inefficiency issue of whole-data based strategy. Most of them (e.g., (He et al. 2016; Pilászy, Zibriczky, and Tikk 2010; Chen et al. 2018a)) are based on Alternating Least Squares (ALS) (Hu, Koren, and Volinsky 2008). Unfortunately, ALS based methods are not applicable to neural models which use Gradient Descent (GD) for optimization. Recently, (Xin et al. 2018) and (Yuan et al. 2018) design fast Batch Gradient Descent (BGD) methods to learn from all training examples. However, they only focus on optimizing traditional models. Distinct from previous studies, we derive a novel loss that can be used to learn neural models efficiently from the whole positive and unlabeled data.

Efficient Heterogeneous Collaborative Filtering

In this section, we first formally define the heterogeneous collaborative filtering problem, then introduce our proposed EHCF model in detail.

Problem Formulation

Suppose we have users $\mathbf{U}$ and items $\mathbf{V}$ in the dataset, and we use the index u to denote a user and v to denote an item. Let $\{\mathbf{R}_{(1)}, \mathbf{R}_{(2)}, ..., \mathbf{R}_{(K)}\}$ denote the user-item interaction matrices for all the K types of behaviors, where $\mathbf{R}_{(k)} = [R_{(k)uv}]_{|\mathbf{U}|\times|\mathbf{V}|} \in \{0, 1\}$ indicates whether user u has interacted with item v under behavior k. Generally, heterogeneous collaborative filtering has a target behavior to be optimized, which is denoted as $\mathbf{R}_{(K)}$. An example of the target behavior is the purchase behavior in E-commerce, and other behaviors can include the view, click, adding to cart,

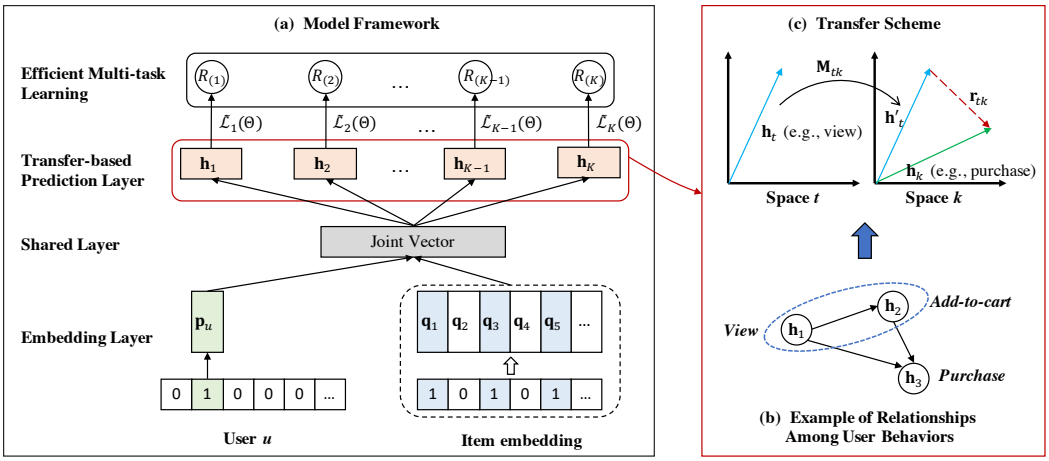

Figure 2: An overview of EHCF model. (a) Illustration of the model framework. (b) An example of the relationships among behaviors, where $\mathbf{h}_1, \mathbf{h}_2, and \mathbf{h}_3$ denotes the prediction functions of behaviors: view, add-to-cart, and purchase, respectively (note that EHCF is not limited to these examples). (c) Illustration of the transfer scheme of two relational behaviors.

etc. The task of heterogeneous collaborative filtering is to estimate the likelihood $\hat{R}_{(K)uv}$ that a user u will interact with an item v under the target behavior. The items (uninteracted under the target behavior) are ranked in descending order of $\hat{R}_{(K)uv}$ to provide the Top-N item recommendation list.

Model Overview

The overall EHCF model is described in Figure 2. Users and items are first converted to dense vector representations through an embedding layer. Note that distinct from previous work that typically input a user-item pair (u, v), we use a user and all his/her item interactions as inputs in our framework. Besides, instead of applying a sampling-based strategy to optimize the interaction between user and item, we propose an efficient optimization method to learn the model from the whole data without sampling. To make it reasonable under the paradigm of representation learning, we share the embedding layer of users and items for the modeling of all behavior types. Then for each user-item instance (u, v), a mapping function is defined as:

$$\phi\left(\mathbf{p}_u, \mathbf{q}_v\right) = \mathbf{p}_u \odot \mathbf{q}_v \tag{1}$$

where $\mathbf{p}_u \in \mathbb{R}^d$ and $\mathbf{q}_v \in \mathbb{R}^d$ are latent vectors of user u and item v, d denotes the embedding size, and $\odot$ denotes the element-wise product of vectors.

To predict the likelihood of multiple behaviors with the same input, it is essential to learn a separated prediction layer for each behavior. Let $\mathbf{h}_k$ denotes the prediction layer for the k-th behavior, the likelihood that u will perform the k-th behavior on v is estimated by:

$$\hat{R}_{(k)uv} = \mathbf{h}_k^T(\mathbf{p}_u \odot \mathbf{q}_v) = \sum_{i=1}^{d} h_{k,i} p_{u,i} q_{v,i} \tag{2}$$

Transfer-based Multi-Behavior Prediction

As discussed in the introduction, generally there are certain transfer relations among users' behaviors in real life (Figure 1). So the predictive layers for different behaviors should be related to each other, rather than being independent. To capture the transfer features, we enforce that the prediction on a behavior lies in the predictions of its precedent behaviors. Motivated by transfer mechanisms in knowledge representations (Bordes et al. 2013), we assume that the relationships between two behaviors can also be portrayed as translations in the representation space. Specifically, the transfer scheme of two relational behaviors (from $\mathbf{h}_t$ to $\mathbf{h}_k$) is defined as:

$$f_{\mathbf{h}_t \to \mathbf{h}_k} = \mathbf{h}_t \mathbf{M}_{tk} + \mathbf{r}_{tk} \tag{3}$$

where $\mathbf{M}_{tk} \in \mathbb{R}^{d \times d}$ is a transfer matrix which projects $\mathbf{h}_t$ from the t-th behavior space to the k-th behavior space. $\mathbf{r}_{tk} \in \mathbb{R}^d$ is the bias vector. A graphical illustration of the transfer scheme is shown in Figure 2(c). Based on that, the prediction layer of the k-th behavior is calculated as:

$$\mathbf{h}_k = \sum_t (f_{\mathbf{h}_t \to \mathbf{h}_k}) = \sum_t (\mathbf{h}_t \mathbf{M}_{tk} + \mathbf{r}_{tk}) \tag{4}$$

where t denotes the precedent behaviors of the k-th behavior. The initial behavior which has no precedent relation (e.g., view) is randomly initialized.

Our transfer-based prediction is not only able to incorporate heterogeneous user behaviors, but also particularly useful for predicting the preference of inactive users that have few data on the target behavior. Typically, the data of low-level behaviors (e.g., view) is easier to collect and has a larger volume than the target behavior (e.g., purchase).

Efficient Optimization without Sampling

We introduce the efficient optimization method without sampling in this section, which is the basis for learning our model from the whole heterogeneous feedback data. For positive-only data, the observed interactions are rather limited, and non-observed examples are of a much larger scale. To learn model parameters, Hu and Volinsky (Hu, Koren, and Volinsky 2008) introduce a weighted regression loss,

which associates a confidence to each prediction in the implicit data matrix. Following this idea, for a batch of users $\mathbf{B}$ and the whole item set $\mathbf{V}$, the loss of a single behavior matrix $\mathbf{R}_{(k)}$ is:

$$\begin{aligned}\mathcal{L}_k(\Theta) &= \sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}} c^k_{uv}(R_{(k)uv} - \hat{R}_{(k)uv})^2 \\ &= \sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}} c^k_{uv}(R^2_{(k)uv} - 2R_{(k)uv}\hat{R}_{(k)uv} + \hat{R}^2_{(k)uv})\end{aligned} \tag{5}$$

where c^k_{uv} denotes the weight of entry $R_{(k)uv}$. As can be seen, the time complexity of computing this loss is $O(|\mathbf{B}||\mathbf{V}|d)$, which means the straightforward way to calculate gradients is generally unaffordable in practice.

While note that in positive-only data, the non-observed instances are usually set to a label of $R_{(k)uv} = 0$ (He et al. 2016; Wang et al. 2018; Yuan et al. 2018). So the non-observed $R_{(k)uv}$ can be eliminated to simplify the equation. Also, the loss of non-observed data can be expressed by the residual between the loss of all data and that of positive data. We have the following derivation:

$$\begin{aligned}\tilde{\mathcal{L}}_k(\Theta) &= -2\sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}^{k+}} c^{k+}_{uv}R_{(k)uv}\hat{R}_{(k)uv} + \sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}} c^k_{uv}\hat{R}^2_{(k)uv} \\ &= \overbrace{\sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}^{k+}}\left((c^{k+}_{uv} - c^{k-}_{uv})\hat{R}^2_{(k)uv} - 2c^{k+}_{uv}R_{(k)uv}\hat{R}_{(k)uv}\right)}^{\mathcal{L}^P_k(\Theta)} \\ &\quad + \overbrace{\sum_{u\in\mathbf{B}}\sum_{v\in\mathbf{V}} c^{k-}_{uv}\hat{R}^2_{(k)uv}}^{\mathcal{L}^A_k(\Theta)}\end{aligned} \tag{6}$$

where the Θ-invariant constant value has be eliminated, $\mathcal{L}^P_k(\Theta)$ denotes the loss for positive data, and $\mathcal{L}^A_k(\Theta)$ denotes the loss for all data. Thus, $\mathcal{L}_k(\Theta)$ can be seen as a combination of the loss of positive data and the loss of all data. And the loss of unlabeled data has been eliminated. The new computational bottleneck lies in $\mathcal{L}^A_k(\Theta)$ now.

Recall the prediction of $\hat{R}_{(k)uv}$ (Eq.(2)), based on a decouple manipulation for the inner product operation, the summation operator and elements in $\mathbf{p}_u$ & $\mathbf{q}_v$ can be rearranged:

$$\begin{aligned}\hat{R}^2_{(k)uv} &= \sum_{i=1}^{d} h_{k,i}p_{u,i}q_{v,i}\sum_{j=1}^{d} h_{k,j}p_{u,j}q_{v,j} \\ &= \sum_{i=1}^{d}\sum_{j=1}^{d}(h_{k,i}h_{k,j})(p_{u,i}p_{u,j})(q_{v,i}q_{v,j})\end{aligned} \tag{7}$$

By substituting Eq.(7) in $\mathcal{L}^A_k(\Theta)$, there emerges a nice structure: usually c^{k-}_{uv} is a uniform (Hu, Koren, and Volinsky 2008) or item-dependent (Liang et al. 2016) parameter c^{k-}_v, so the interaction between $p_{u,i}$ and $q_{v,i}$ can be properly separated. Then, the optimization of $\sum_{v\in\mathbf{V}} c^{k-}_v q_{v,i}q_{v,j}$ and $\sum_{u\in\mathbf{B}} p_{u,i}p_{u,j}$ are independent from each other, and we can achieve a significant speed-up by precomputing the two terms. The final efficient whole-data based loss for a single k-th behavior is as follows:

$$\begin{aligned}\tilde{\mathcal{L}}_k(\Theta) &= \mathcal{L}^P_k(\Theta) \\ &+ \sum_{i=1}^{d}\sum_{j=1}^{d}\left((h_{k,i}h_{k,j})\left(\sum_{u\in\mathbf{B}} p_{u,i}p_{u,j}\right)\left(\sum_{v\in\mathbf{V}} c^{k-}_v q_{v,i}q_{v,j}\right)\right)\end{aligned} \tag{8}$$

The rearrangement of nested sums in $\mathcal{L}^A_k(\Theta)$ is the key transformation that allows the fast optimization. The computing complexity of $\mathcal{L}^A_k(\Theta)$ is reduced from $O(|\mathbf{B}||\mathbf{V}|d)$ to $O((|\mathbf{B}|+|\mathbf{V}|)d^2)$. Note that in this section we present a user-based batch optimization loss, the item-based loss can also be derived in the same way.

Multi-Task Learning

Multi-task learning (MTL) is a paradigm that performs joint training on the models of different but correlated tasks, so as to obtain a better model for each task (Argyriou, Evgeniou, and Pontil 2007). We propose a MTL objective function defined as follows:

$$\mathcal{L}(\Theta) = \sum_{k=1}^{K}\lambda_k\tilde{\mathcal{L}}_k(\Theta) \tag{9}$$

where K is the number of types of users' behavior, λ_k is added to control the influence of the k-th behavior on the joint training, which is a hyper-parameter to be specified for different datasets. We additionally enforce that $\sum_{k=1}^{K}\lambda_k = 1$ to facilitate the tuning of these hyper-parameters.

Our efficient optimization method can be naturally implemented in modern machine learning tools like Tensorflow and PyTorch. To optimize the objective function, we use mini-batch Adagrad (Duchi, Hazan, and Singer 2011) as the optimizer. Its main advantage is that the learning rate can be self-adaptive during the training phase, which eases the pain of choosing a proper learning rate. Dropout is an effective solution to prevent neural networks from overfitting (Srivastava et al. 2014), which is also adopted in our model.

Discussion

We first discuss the time complexity of our model, in Eq.(8), updating a batch of users for the k-th behavior takes $O((|\mathbf{B}| + |\mathbf{V}|)d^2 + |\mathcal{R}^k_{\mathbf{B}}|d)$ time, where $\mathcal{R}^k_{\mathbf{B}}$ denotes positive item interactions of this batch of users under the k-th behavior. When updating the whole model through multi-task learning for all the K types of behavior, one batch takes $O(K(|\mathbf{B}| + |\mathbf{V}|)d^2 + \sum_{k=1}^{K}|\mathcal{R}^k_{\mathbf{B}}|d)$ time (the time overhead of behavior transferring is rather small and can be ignored). For the original regression loss (Eq.(5)), it takes $O(K|\mathbf{B}||\mathbf{V}|d)$ time. Since $|\mathcal{R}^k_{\mathbf{B}}| \ll |\mathbf{B}||\mathbf{V}|$ and $d \ll |\mathbf{B}|$ in practice, the computional complexity of our model is reduced by several magnitudes. Moreover, since no approximation is introduced during the derivation process, the optimization results are exactly the same as the original whole-data based regression loss. It is also noteworthy that our EHCF model is applicable to traditional single-behavior recommendation by optimizing the target behavior loss only.

Table 1: Statistical details of the evaluation datasets.

Dataset	#User	#Item	#View	#Add-to-cart	#Purchase
Movielens	6,940	3,706	–	–	1,000,209
Beibei	21,716	7,977	2,412,586	642,622	304,576
Taobao	48,749	39,493	1,548,126	193,747	259,747

As fast whole-data based learning is a challenging problem, the designed efficient optimization method is still preliminary. It is now limited to learn models with linear prediction layer, because the rearrange operation in Eq.(7) requires the prediction of $\hat{R}_{(k)uv}$ to be linear. We leave the extension of the method as future work.

Experiments and Results

Experimental Settings

Datasets We conduct extensive experiments on three real-world recommendation system datasets. 1) ***Movielens-1M*** [2] (Harper and Konstan 2016) is a single-behavior dataset, 2) ***Beibei*** [3] (Gao et al. 2019), and 3) ***Taobao*** [4] (Zhu et al. 2018) contain heterogeneous user behaviors (view, add-to-cart, and purchase). The datasets were constructed following previous work (Ding et al. 2018; Gao et al. 2019). First, we merge the duplicated user-item interactions by keeping the earliest one. Second, we filter out users and items with less than 5 purchase interactions. After that, the last purchase records of users are used as test data, the second last records are used as validation data, and the remaining records are used for training. The statistical details of these datasets are summarized in Table 1.

Baselines We compare the performance of our **EHCF** model with the various recommendation methods, which can be divided into two groups based on whether it models single-behavior or heterogeneous data. The compared single-behavior methods include:

- **BPR** (Rendle et al. 2009), a widely used pairwise learning method for item recommendation.
- **ExpoMF** (Liang et al. 2016), a whole-data based MF method which treats all missing interactions as negative and weighs them by item popularity.
- **NCF** (He et al. 2017), a state-of-the-art deep learning method which combines MF with a multilayer perceptron (MLP) model for item ranking.

The second group that can leverage heterogeneous data are as follows:

- **CMF** (Zhao et al. 2015), it decomposes the data matrices of multiple behavior types simultaneously.
- **MC-BPR** (Loni et al. 2016), it adapts the negative sampling rule in BPR for heterogeneous data.
- **NMTR** (Gao et al. 2019), a state-of-the-art heterogeneous method, which combines the recent advances of NCF modeling and the efficacy of multi-task learning.

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

[3]Provided by the authors of (Gao et al. 2019)

[4]https://tianchi.aliyun.com/dataset/dataDetail?dataId=649

Evaluation Methodology We apply the widely used leave-one-out technique (Gao et al. 2019; Rendle et al. 2009; Wang et al. 2018) and then adopt two popular metrics, HR (Hit Ratio) and NDCG (Normalized Discounted Cumulative Gain), to judge the performance of the ranking list. HR is a recall-based metric, measuring whether the testing item is in the Top-N list, while NDCG is position-sensitive, which assigns higher score to hits at higher positions. Note that for a user, our evaluation protocol ranks ***all unobserved items*** in the training set. Through this way, the obtained results are more persuasive than ranking a random subset of negative times only (Gao et al. 2019; Dacrema, Cremonesi, and Jannach 2019).

Parameter settings We search for the optimal parameters on validation data and evaluate the model on test data. For the parameters of baseline models, we refer to their original papers and follow their tuning strategies. After the tuning process, the batch size is set to 512, the size of the latent factor dimension d is set to 64. The learning rate is set to 0.05, and the dropout ratio ρ is set to 0.8 for Movielens, and 0.5 for Beibei & Taobao to prevent overfitting. We set the negative sampling ratio as 4 for sampling-based methods, an empirical value that shows good performance.

Performance Comparison

The results of the comparison of different methods on three datasets are shown in Table 2. We investigate the Top-N performance with N setting to [10, 50, 100, 200]. From the results, the following observations can be made:

1) The methods using heterogeneous feedback generally outperform methods that only making use of purchase behavior, which shows the complementarity of users heterogeneous feedback. 2) The methods using whole-data based learning strategies generally perform better than sampling-based methods. For example, in Table 2, the performances of ExpoMF are better than BPR; and our EHCF outperforms all the baselines. This is consistent with previous work (Yuan et al. 2018; Xin et al. 2018), which indicates that regardless of what sampler is utilized or how many updates are taken, sampling is still a biased approach. 3) Our EHCF significantly outperforms the state-of-the-art CF methods in both traditional (single-behavior) and heterogeneous scenarios. In particular, compared to NMTR – a recently proposed heterogeneous deep learning model, EHCF exhibits remarkable average improvements of 47.5% on Beibei dataset and 57.1% on Taobao dataset.

Discussion on the Remarkable Improvements Although sampling has been widely adopted in previous work, it is still reasonable to argue that sampling is not suitable for learning from heterogeneous behavior data. To generate a training instance, sampling-based methods (e.g., MC-BPR, NMTR) need to sample a negative instance for every observed interaction (regardless of the behavior type). This produces a very large randomness in total (K times than single-behavior scenario), and can ignore many important instances. Different from previous work, the parameters in our model are jointly optimized on the whole data. This explains why our EHCF outperforms EHCF-sin by 79.4%

Table 2: Performance of different models on three datasets. * and ** denotes the statistical significance for $p < 0.05$ and $p < 0.01$, respectively, compared to the best baseline.

Movielens-1M		HR@10	HR@50	HR@100	HR@200	NDCG@10	NDCG@50	NDCG@100	NDCG@200
Single	**BPR**	0.0822	0.2637	0.4048	0.5710	0.0418	0.0757	0.0986	0.1217
	ExpoMF	0.0892	0.2808	0.4253	0.5864	0.0457	0.0764	0.1053	0.1309
	NCF	0.0936	0.2902	0.4316	0.6023	0.0481	0.0837	0.1097	0.1324
	EHCF-Sin	**0.1025****	**0.3104****	**0.4597****	**0.6177****	**0.0508****	**0.0934****	**0.1169****	**0.1392****
Beibei		**HR@10**	**HR@50**	**HR@100**	**HR@200**	**NDCG@10**	**NDCG@50**	**NDCG@100**	**NDCG@200**
Single	**BPR**	0.0437	0.1246	0.2192	0.3057	0.0213	0.0407	0.0539	0.0689
	ExpoMF	0.0452	0.1465	0.2246	0.3282	0.0227	0.0426	0.0553	0.0723
	NCF	0.0441	0.1562	0.2343	0.3583	0.0225	0.0445	0.0584	0.0757
	EHCF-Sin	**0.0464****	**0.1637****	**0.2586****	**0.3743****	**0.0247****	**0.0484****	**0.0639****	**0.0799****
Heterogeneous	**CMF**	0.0482	0.1582	0.2843	0.4288	0.0251	0.0462	0.0661	0.0852
	MC-BPR	0.0504	0.1743	0.2755	0.3862	0.0254	0.0503	0.0653	0.0796
	NMTR	0.0524	0.2047	0.3189	0.4735	0.0285	0.0609	0.0764	0.0968
	EHCF	**0.0608****	**0.3316****	**0.4312****	**0.5460****	**0.0325****	**0.1213****	**0.1374****	**0.1535****
Taobao		**HR@10**	**HR@50**	**HR@100**	**HR@200**	**NDCG@10**	**NDCG@50**	**NDCG@100**	**NDCG@200**
Single	**BPR**	0.0376	0.0708	0.0871	0.1035	0.0227	0.0269	0.0305	0.0329
	ExpoMF	0.0386	0.0713	0.0911	0.1068	0.0238	0.0270	0.0302	0.0334
	NCF	0.0391	0.0728	0.0897	0.1072	0.0233	0.0281	0.0321	0.0345
	EHCF-Sin	**0.0398***	**0.0743****	**0.0936****	**0.1141****	**0.0244***	**0.0298****	**0.0339****	**0.0372****
Heterogeneous	**CMF**	0.0483	0.0774	0.1185	0.1563	0.0252	0.0293	0.0357	0.0379
	MC-BPR	0.0547	0.0791	0.1264	0.1597	0.0263	0.0297	0.0361	0.0397
	NMTR	0.0585	0.0942	0.1368	0.1868	0.0278	0.0334	0.0394	0.0537
	EHCF	**0.0717****	**0.1618****	**0.2211****	**0.2921****	**0.0403****	**0.0594****	**0.0690****	**0.0789****

Table 3: Performance of variants of EHCF on Taobao dataset.

	Taobao	HR@10	HR@50	HR@100	HR@200	NDCG@10	NDCG@50	NDCG@100	NDCG@200
Data Ablation	**Purchase**	0.0398	0.0743	0.0936	0.1141	0.0244	0.0298	0.0339	0.0372
	Purchase&Add-to-cart	0.0612	0.1187	0.1569	0.2008	0.0297	0.0465	0.0527	0.0589
	Purchase&View	0.0673	0.1438	0.2061	0.2824	0.0384	0.0493	0.0593	0.0700
Model Ablation	**Without Transferring**	0.0633	0.1112	0.1525	0.2088	0.0312	0.0389	0.0457	0.0535
	Without MTL	0.0611	0.1171	0.1552	0.2003	0.0304	0.0448	0.0509	0.0573
Full Model	**EHCF**	**0.0717****	**0.1618****	**0.2211****	**0.2921****	**0.0403****	**0.0594****	**0.0690****	**0.0789****

on Beibei and 108.8% on Taobao, while the state-of-the-art NMTR only outperforms NCF by 34.6% and 46.9% on the two datasets (consistent with (Gao et al. 2019)). Existing sampling-based heterogeneous methods have not fully leveraged the auxiliary behavior data, which is also one of the major points of this work.

Ablation Study

To understand the effectiveness of auxiliary behavior data, transfer based prediction, and multi-task learning strategy, we conduct experiments with several variants of EHCF. Note that without transferring, the prediction layer of each behavior (h_k) is randomly initialized and independent. Without MTL, the prediction of each behavior is trained alternately within an epoch. The results on Taobao dataset are recorded in Table 3 and the results on Beibei dataset are similar.

As shown in the table, both adding view data and cart data to our model lead to improvements, which verify the effectiveness of auxiliary behaviors for user preference modeling. The remarkable improvements of our methods also show the necessary of applying non-sampling strategy for learning from heterogeneous feedback. Besides, variants without transferring and without MTL both perform worse than the full EHCF model, which verifies the effectiveness of the proposed transfer-based prediction layer and multi-task training component.

Efficiency Analysis

We conduct further experiments to explore the training efficiencies of our EHCF and two state-of-the-art neural recommendation methods: NCF and NMTR. In our experiments, the neural models are all trained on a single NVIDIA GeForce GTX TITAN X GPU. The runtime results are shown in Table 4. From the table, first, the training time cost of EHCF is much less than that using original regression loss (Eq.(4)), which verifies that the derived loss can be learned more efficiently. Second, the training of EHCF is much faster on both single-behavior data and heterogeneous data compared to NCF and NMTR, respectively. In real-world systems, the cost of training time is an important factor to be considered. Our EHCF model shows significant advantage in training efficiency, which makes it more practical to be applied in real life. We also investigate the training process of NCF, NMTR, and our EHCF. Figure 3 shows the

Table 4: Comparisons of runtime (second/minute [s/m]). "S", "I", and "T" represent the training time for a single iteration, the number of iterations to converge, and the total training time, respectively.

Model	*Movielens-1M*			*Beibei*			*Taobao*		
	S	**I**	**T**	**S**	**I**	**T**	**S**	**I**	**T**
NCF	91s	100	152m	62s	100	104m	115s	100	192m
EHCF-Sin	**4.5s**	100	**8m**	**3.2s**	100	**6m**	**6s**	100	**10m**
NMTR	–	–	–	165s	200	550m	180s	200	600m
EHCF-Original	–	–	–	62s	200	207m	192s	200	640m
EHCF	–	–	–	**7s**	200	**24m**	**16s**	200	**54m**

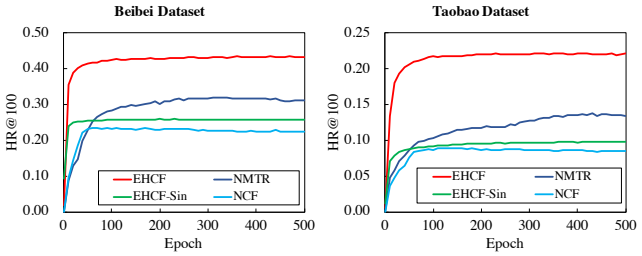

Figure 3: Performance curves of NCF (best single-behavior baseline), NMTR (best heterogeneous baseline), EHCF-Sin, and EHCF.

prediction accuracy of the models with respect to different training epochs. Due to the space limitation, we only show the results on HR@100 metrics. For other metrics, the observations are similar. From the figure, we can see that EHCF converges much faster on both single-behavior data and heterogeneous data than NCF and NMTR.

Handling Data Sparsity Issue

Data sparsity is a big challenge in recommendation (Volkovs, Yu, and Poutanen 2017), and heterogeneous collaborative filtering provides a solution. Thus, we further study how EHCF model performs for the users with few records of target behavior. The results are shown in Figure 4. From the results, we can see that EHCF consistently outperforms other methods by a big margin. Particularly in the first user group with only 5-8 purchase records, our EHCF still keeps a good HR@100 performance of 0.4460 on Beibei dataset and 0.2217 on Taobao dataset, which outperforms the best baseline NMTR by 70.6% and 62.5%, respectively. Since EHCF learns all types of behaviors in a reasonable way, it can achieve a good performance for users with sparse interactions.

Impact of Parameters

To understand how hyper-parameters influence the performance of EHCF model, we test the impact of coefficient λ_k in the joint loss function of MTL since it is a key parameter of our method. There are three behavior types for Beibei and Taobao, which means there are three loss coefficients λ_1, λ_2, and λ_3. As $\lambda_1 + \lambda_2 + \lambda_3 = 1$, when λ_1 and λ_2 are given, the value of λ_3 is determined. We tune the three coefficients in $[0, 1/6, 2/6, 3/6, 4/6, 5/6, 1]$ and plot the results

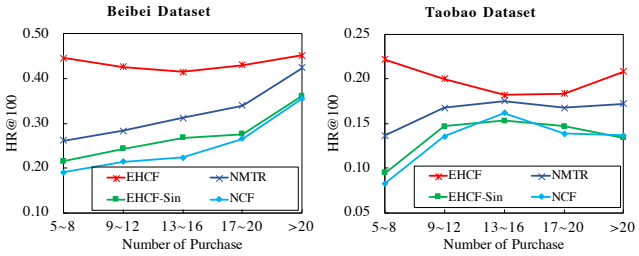

Figure 4: Performances of NCF, NMTR, EHCF-Sin, and EHCF on users with different number of purchase records.

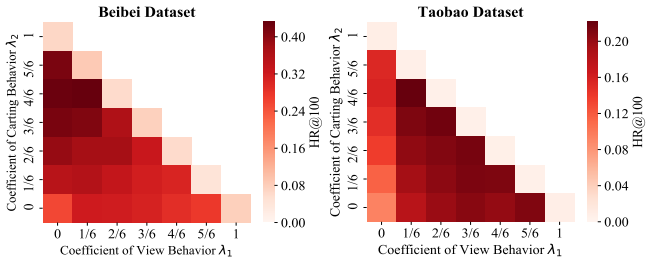

Figure 5: Performance of EHCF with different loss coefficient.

of HR@100 in Figure 5 where darker blocks means better performance. In the figure, outermost blocks are rather shallow since they represent a zero λ_3, which is the coefficient of purchase behavior. On Beibei dataset, a relative large coefficient of carting behavior outperforms that of view behavior, while on Taobao dataset, a relative large λ_1 leads to better performance. The reason may be the size difference of auxiliary behavioral data in two datasets.

Conclusions

In this paper we introduce a novel end-to-end model named EHCF for recommendation with heterogeneous user feedback. The proposed EHCF has two key characteristics: 1) a newly designed optimization method is used for efficient whole-data based model learning; 2) the prediction of each behavior is correlated in a transfer way to capture the complicated relations among different behaviors. Extensive experiments on three real-world datasets show that EHCF not only outperforms the state-of-the-art recommendation models by a big margin, but also has a rather fast training process. This work complements the mainstream sampling-based neural models for recommendation with implicit feedback, opening up a new avenue of research possibilities for neural recommendation models. The designed efficient whole-data based strategy has the potential to benefit many tasks where only positive data is observed. Future work includes exploring our EHCF model in other related tasks such as network embedding and multi-label classification. We will also try to extend our optimization method to make it applicable to learn non-linear models.

Acknowledgements

This work is supported by Natural Science Foundation of China (Grant No. 61672311, 61532011) and the National Key Research and Development Program of China (2018YFC0831900). The work is also partially supported by National Science Foundation (IIS-1910154). Any opinions, findings and conclusions expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

References

Argyriou, A.; Evgeniou, T.; and Pontil, M. 2007. Multi-task feature learning. In *Proceedings of NeuIPS*, 41–48.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of NeuIPS*, 2787–2795.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2018a. Missing data modeling with user activity and item popularity in recommendation. In *Proceedings of AIRS*, 113–125.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2018b. Neural attentional rating regression with review-level explanations. In *Proceedings of WWW*, 1583–1592.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2019a. Social attentional memory network: Modeling aspect-and friend-level differences in recommendation. In *Proceedings of WSDM*.

Chen, C.; Zhang, M.; Wang, C.; Ma, W.; Li, M.; Liu, Y.; and Ma, S. 2019b. An efficient adaptive transfer neural network for social-aware recommendation. In *Proceedings of SIGIR*, 225–234.

Dacrema, M. F.; Cremonesi, P.; and Jannach, D. 2019. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. *arXiv preprint arXiv:1907.06902*.

Ding, J.; Yu, G.; He, X.; Quan, Y.; Li, Y.; Chua, T.-S.; Jin, D.; and Yu, J. 2018. Improving implicit recommender systems with view data. In *Proceedings of IJCAI*, 3343–3349.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12(Jul):2121–2159.

Gao, C.; He, X.; Gan, D.; Chen, X.; Feng, F.; Li, Y.; Chua, T.-S.; and Jin, D. 2019. Neural multi-task recommendation from multi-behavior data. In *Proceedings of ICDE*.

Harper, F. M., and Konstan, J. A. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5(4):19.

He, X.; Zhang, H.; Kan, M.-Y.; and Chua, T.-S. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of SIGIR*, 549–558.

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural collaborative filtering. In *Proceedings of WWW*, 173–182.

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of ICDM*, 263–272.

Krohn-Grimberghe, A.; Drumond, L.; Freudenthaler, C.; and Schmidt-Thieme, L. 2012. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of WSDM*, 173–182.

Liang, D.; Charlin, L.; McInerney, J.; and Blei, D. M. 2016. Modeling user exposure in recommendation. In *Proceedings of WWW*, 951–961.

Loni, B.; Pagano, R.; Larson, M.; and Hanjalic, A. 2016. Bayesian personalized ranking with multi-channel user feedback. In *Proceedings of RecSys*, 361–364.

Pan, R.; Zhou, Y.; Cao, B.; Liu, N. N.; Lukose, R.; Scholz, M.; and Yang, Q. 2008. One-class collaborative filtering. In *2008 Eighth IEEE International Conference on Data Mining*, 502–511.

Pilászy, I.; Zibriczky, D.; and Tikk, D. 2010. Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proceedings of Recsys*, 71–78.

Qiu, H.; Liu, Y.; Guo, G.; Sun, Z.; Zhang, J.; and Nguyen, H. T. 2018. Bprh: Bayesian personalized ranking for heterogeneous implicit feedback. *Information Sciences* 453:80–98.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of UAI*, 452–461.

Ricci, F.; Rokach, L.; and Shapira, B. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*. 1–35.

Singh, A. P., and Gordon, G. J. 2008. Relational learning via collective matrix factorization. In *Proceedings of SIGKDD*, 650–658.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

Tang, L.; Long, B.; Chen, B.-C.; and Agarwal, D. 2016. An empirical study on recommendation with multiple types of feedback. In *Proceedings of SIGKDD*, 283–292.

Volkovs, M.; Yu, G.; and Poutanen, T. 2017. Dropoutnet: Addressing cold start in recommender systems. In *Proceedings of NeuIPS*, 4957–4966.

Wang, M.; Gong, M.; Zheng, X.; and Zhang, K. 2018. Modeling dynamic missingness of implicit feedback for recommendation. In *Proceedings of NeuIPS*, 6669–6678.

Xin, X.; Yuan, F.; He, X.; and Jose, J. M. 2018. Batch is not heavy: Learning word representations from all samples. In *Proceedings of ACL*, 1853–1862.

Yuan, F.; Xin, X.; He, X.; Guo, G.; Zhang, W.; Tat-Seng, C.; and Jose, J. M. 2018. fbgd: Learning embeddings from positive unlabeled data with bgd.

Zhao, Z.; Cheng, Z.; Hong, L.; and Chi, E. H. 2015. Improving user topic interest profiles by behavior factorization. In *Proceedings of WWW*, 1406–1416.

Zhu, H.; Li, X.; Zhang, P.; Li, G.; He, J.; Li, H.; and Gai, K. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of SIGKDD*, 1079–1088.