Online Collaborative Filtering with Implicit Feedback

Jianwen Yin[1,2], Chenghao Liu[3⋆], Jundong Li[4], BingTian Dai[3], Yun-chen Chen[3], Min Wu[5], and Jianling Sun[1,2⋆]

[1] School of Computer Science and Technology, Zhejiang University, China
[2] Alibaba-Zhejiang University Joint Institute of Frontier Technologies, China
{yinjw,sunjl}@zju.edu.cn
[3] School of Information Systems, Singapore Management University, Singapore
{chliu,btdai,jeanchen}@smu.edu.sg
[4] Computer Science and Engineering, Arizona State University, USA
jundong.li@asu.edu
[5] Data Analytics DepartmentInstitute for Infocomm Research, A-Star, Singapore
wumin@i2r.a-star.edu.sg

Abstract. Studying recommender systems with implicit feedback has become increasingly important. However, most existing works are designed in an offline setting while online recommendation is quite challenging due to the one-class nature of implicit feedback. In this paper, we propose an online collaborative filtering method for implicit feedback. We highlight three critical issues of existing works. First, when positive feedback arrives sequentially, if we treat all the other missing items for this given user as the negative samples, the mis-classified items will incur a large deviation since some items might appear as the positive feedback in the subsequent rounds. Second, the cost of missing a positive feedback should be much higher than that of having a false-positive. Third, the existing works usually assume that a fixed model is given prior to the learning task, which could result in poor performance if the chosen model is inappropriate. To address these issues, we propose a unified framework for Online Collaborative Filtering with Implicit Feedback (OCFIF). Motivated by the regret aversion, we propose a divestiture loss to heal the bias derived from the past mis-classified negative samples. Furthermore, we adopt cost-sensitive learning method to efficiently optimize the implicit MF model without imposing a heuristic weight restriction on missing data. By leveraging meta-learning, we dynamically explore a pool of multiple models to avoid the limitations of a single fixed model so as to remedy the drawback of manual/heuristic model selection. We also analyze the theoretical bounds of the proposed OCFIF method and conduct extensive experiments to evaluate its empirical performance on real-world datasets.

⋆ Chenghao Liu and Jianling Sun are the corresponding authors.

1 Introduction

Recommender systems aim to alleviate information overload by providing personalized suggestions from a superabundant of choices based on the historical behaviour. Among various recommendation algorithms, Collaborative Filtering (CF), an approach that uses known preferences of some users to make predictions to the unknown preferences of other users, has been widely used as one of the core learning techniques in building real-world recommender systems. The prevalent of E-commerce and social media sites generate massive data at an unprecedented rate. More than 10 million transactions are made per day in eBay[6][1] and about half a billion tweets are generated every day [12]. Such data is temporally ordered, high-velocity and time varying. Unfortunately, traditional CF based methods adopt batch machine learning techniques which assume all training data are provided prior to model training. Such assumption makes them unsuitable and non-scalable for real-world applications for the following reasons. First, the user-item interactions usually arrive sequentially and periodically while batch learning model has to be retrained from scratch whenever new samples are received, making the training process extremely expensive. Second, whenever a new user/item is added to the system, batch learning cannot handle such changes immediately without involving an expensive re-training process. Third, it is common that user preferences are likely to change through time, but it is difficult for a batch learning model to capture the changes. Therefore, it is imperative to develop real-time scalable recommendation algorithms.

Recent years have witnessed some emerging studies for online recommendation methods [1, 10, 21]. These methods generally follow the paradigm of Matrix Factorization (MF) model which associates each user and item with a latent vector respectively and assume that the corresponding rating is estimated by the vector inner product. These works formulate the recommendation task as a rating prediction problem which is denoted by explicit feedback. Nevertheless, implicit feedback, such as monitoring clicks, view times, purchases, etc, is much cheaper to obtain than explicit feedback, since it comes with no extra cost for the user and thus is available on a much larger scale. Compared to explicit ratings, implicit feedback is much more challenging due to the natural scarcity of negative feedback (also known as the one-class problem). One popular solution to solve this problem is to select some negative instances from unlabeled entries [2, 14]. However, this adversely decreases the efficacy of the predictive model due to insufficient data coverage. Another solution [8] is to contrast the positive feedback against all the non-observed interactions. However, this strategy significantly increases the computation cost. A state-of-the-art MF method for implicit feedback is the eALS [7], which treats all missing data as the negative feedback but with a heuristic weight. Despite its success in dealing with batch learning setting, it is challenging to develop online recommendation methods with implicit feedback for the following reasons: (i) when positive feedback arrives sequentially, if we treat all the other missing items for this given user as

[6] http://www.webretailer.com/articles/ebay-statistics.asp

the negative feedback, the mis-classified items will incur a large deviation since some items might appear as the positive feedback in the subsequent rounds; (ii) the cost of missing a positive target is much higher than that of having a false-positive [23]; (iii) the existing works usually assume that prior to the learning task, a fixed model is given either by manual selection or via cross validation. This could result in poor performance if the chosen model is inappropriate in a new environment, which happens commonly for real-world applications since user preferences and item attributes dynamically change over time.

To address these issues, we propose a unified framework for Online Collaborative Filtering with Implicit Feedback (OCFIF). First, motivated by the regret aversion [16], we propose a divestiture loss to heal the bias derived from the past mis-classified negative samples. Next, we utilize cost-sensitive learning method [3] to efficiently optimize the implicit MF model without imposing a heuristic weight restriction on missing data. Finally, we leverage meta-learning method [18] to explore a pool of multiple models, which are assigned with weights according to their real-time performance, to remedy the drawback of using a single fixed model by existing methods that often suffer considerably when the single model is inappropriate. In this way, the selection of the optimal model is adaptive and evolving according to the streaming data. By leveraging divestiture loss, cost-sensitive learning and meta-learning, our implicit MF objective function integrates them into a joint formulation. We theoretically analyze the regret bounds of the proposed framework. To validate the efficacy of the proposed method, we conduct extensive experiments by evaluating the proposed algorithms on real-world datasets, showing that our method outperforms the existing state-of-the-art baselines.

2 Related Work

The proposed work in this paper is mainly related to following two directions: (i) recommendation with implicit feedback; (ii) online recommender systems.

2.1 Recommendation with Implicit Feedback

While early literature on recommendation has largely focused on explicit feedback [9, 15], recent attention is increasingly shifting towards implicit data [7, 8, 23]. We can categorize previous works for implicit feedback into two types: sample-based learning and whole-data based learning. The first type samples negative instances from missing data [2, 14]. The BPR method [14] randomly samples negative instances from missing data, optimizing the model with a pair-wise ranking loss. Later on, [2] improves BPR with a better negative sampler by additionally leveraging view data in E-commerce. By reducing negative examples in training process, these sample-based methods are more efficient in training, but the convergence speed is slower. The second type treats all missing entries as negative instances [6–8]. For example, the WALS [8] method models all missing entries as negative instances, assigning them with a uniform lower weight in point-wise regression learning. Recently, [6] develops efficient learning algorithms for any non-uniform weight on missing data. These whole-data based methods can achieve a higher data coverage, but the training cost is much more expensive. Although the aforementioned batch learning methods achieve relatively accurate prediction performance, they often suffer poor efficiency and scalability issues for the online recommendation tasks.

2.2 Online Recommender Systems

There are a variety of research works studying online recommendation algorithms for explicit feedback. Most of these works focus on how to update the model efficiently in online setting. Inspired by online multi-task learning, OMTCF algorithm [20] treats users as tasks and update the models of multiple users simultaneously. [9] and [11] algorithms consider second-order information to achieve faster convergent rate. Beyond the efficient updating problem, RKMF model [15] focuses on solving the new user/item problem in the stream setting. [1] employs a continuous markov process for each time-varying user/item latent vector to solve the user interest drift problem. Recently, [21] solves new user/item, user interest drift and overload problem in a single framework with probabilistic matrix factorization model. By contrast, there are few studies on implicit online recommendation. [19] develops a fast incremental Matrix Factorization algorithm for recommendation with positive-only feedback. However, modeling only the positive feedback results in biased representations in user profile [8]. To this end, [7] proposes an online implicit matrix factorization method, called eALS, which models all the missing data as negative feedback. Although eALS could update model in online setting, the basic model needs to be trained offline on a large amount of historical data first (compared to the amount of data for online update). Otherwise, the performance of eALS will significantly drop which limits model's flexibility and scalability for real applications.

3 Online Collaborative Filtering with Implicit Feedback

3.1 Problem Formulation

First, we motivate the problem by introducing the formulation of implicit MF model. For a user-item rating matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, m and n denote the number of users and items respectively, Ω denotes the set of user-item pairs that have interactions. In the implicit setting, we define the observation matrix $\mathbf{R}$, where $R_{ij} = 1$ if $(i,j) \in \Omega$, and $R_{ij} = 0$ otherwise. MF maps both users and items into a joint latent feature space of k dimension. Formally, let $\mathbf{U} \in \mathbb{R}^{k \times m}$ be the latent factor corresponding to the users, where the i-th column $\mathbf{u}_i \in \mathbb{R}^k$ is the latent factor for user i. Similarly, let $\mathbf{V} \in \mathbb{R}^{k \times n}$ be the latent factor for the items, where the j-th column $\mathbf{v}_j \in \mathbb{R}^k$ is the latent factor for item j. In this work, we cast implicit MF as an online learning problem. On each round t, an observed matrix entry r_{ij}^t is revealed, where $(i,j) \in \Omega$. The goal of OCFIF is to update $\mathbf{u}_i^t$

and $\mathbf{v}_j^t$ such that $r_{ij}^t \approx (\mathbf{u}_i^t)^\top \mathbf{v}_j^t$. The existing online recommendation methods [9, 11] then alternatively update $\mathbf{u}_i^t$ and $\mathbf{v}_j^t$ while keeping the other one fixed by minimizing the incurred loss $\ell(r_{ij}^t, \hat{r}_{ij}^t)$, where $\hat{r}_{ij}^t = (\mathbf{u}_i^t)^\top \mathbf{v}_j^t$.

However, this learning process is not suitable for online recommendation with implicit feedback for the following reasons: (i) when positive feedback arrives sequentially, if we treat all the other missing items for this given user as the negative feedback, it will significantly increase the time for updating model which hinders it from deploying online. Moreover, the data we treat as negative feedback can appear as positive feedback later, which brings a non-negligible side-effect to the model; (ii) the cost of missing a positive target is much higher than that of having a false-positive; (iii) the existing works usually assume that prior to the learning task, a fixed model is given either by manual selection or via cross validation. This could result in poor performance if the chosen model is inappropriate in a new environment, which is widely observed in real-world applications since user preferences and item attributes dynamically change.

3.2 OCFIF Framework

Due to the one-class problem, the model cannot receive any negative samples for training in online setting. Thus, we follow the conventional assumption that treats all the other missing items for the given user at each round as the negative feedback. To address the above issues, we propose a unified framework for Online Collaborative Filtering with Implicit Feedback (OCFIF) as shown in Figure 1. First, motivated by the regret aversion [16], we propose a divestiture loss to heal the bias derived from the past mis-classified negative samples. Next, we develop a cost-sensitive learning method [3] that efficiently optimizes the implicit MF model without imposing a heuristic weight restriction on missing data. Finally, we utilize meta-learning [18] to explore a pool of multiple models, which are assigned with weights according to their real-time performance, to remedy the drawback of using a single fixed model by existing methods as their performance often degrade considerably when the single model is inappropriate. In this way, the selection of the optimal model is adaptive and evolving according to the streaming data. By leveraging divestiture loss, cost-sensitive learning and meta-learning, the proposed framework integrates them into a joint formulation.

Divestiture Loss To tackle the issue of mis-classified negative samples, we adopt the regret aversion [16] idea to amend the bias. Regret aversion is originated from decision theory, which encourages to anticipate regret when facing a decision and thus incorporate their desire to eliminate or reduce this possibility in their choice. It has been found to influence choices in a variety of important domains including health-related decisions, consumer behavior, and investment decisions. In our setting, we hypothesize that model should attach a higher weight to the positive samples that were mis-classified as the negative samples in the past than the other data. Therefore, we explicitly deal with this cost as the divestiture loss and integrate it into the optimization problem. Formally, we

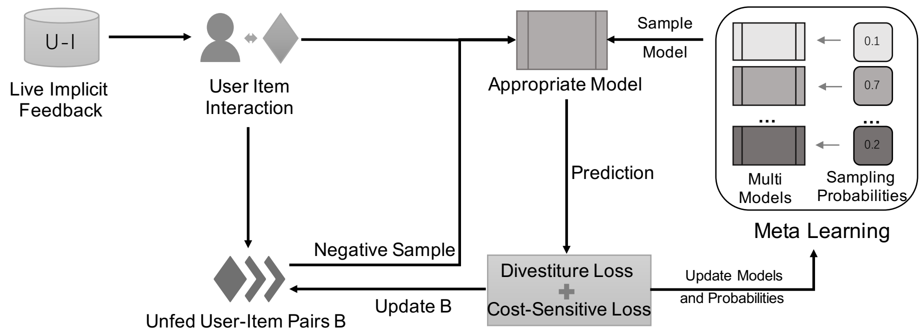


Fig. 1. The OCFIF framework.

denote the set of historical user-item pairs which are treated as negative samples before round t as N_t, then the divestiture loss is formulated as:

$$\xi(r_{ij}^t, \hat{r}_{ij}^t) = (1 + \lambda \mathbb{I}[(i,j) \in N_t])\ell(r_{ij}^t, \hat{r}_{ij}^t), \tag{1}$$

where ℓ could be any convex loss function and we instantiate it as the ϵ-insensitive loss $\ell(r_{ij}^t, \hat{r}_{ij}^t) = max(|r_{ij}^t - \hat{r}_{ij}^t| - \epsilon)$. $\mathbb{I}[\cdot]$ is the indicator function that equals to 1 if the statement holds; and 0 otherwise. $\lambda \geq 0$ is a hyper-parameter that balance the original loss and the extra penalty. If $\lambda = 0$, the extra penalty disappears and the loss function becomes the conventional one. $\mathbb{I}[(i,j) \in N_t]$ indicates whether the user-item pair (i,j) has been mis-classified as the negative sample in the past.

In order to meet high efficiency requirement of online recommendation and reduce the number of mis-classified samples, we denote the set of user-item pairs which have not been fed to the model before round t as B_t, then sample Z negative instances from B_t for model update. A naive sample strategy is to uniformly sample from B_t which assumes each missing entry is negative feedback with equal probability. We follow the assumption in [7, 13] that popular items are more likely to be known by users in general, and thus a miss on a popular item is more likely to be truly negative. In this way, we sample negative instances according to the global item popularity. Let D_t denotes the set of user-item pairs that revealed before round t. The sampling distribution of item j is defined as:

$$p(j) = \frac{\sum_{i'=1}^{m} \mathbb{I}[(i',j) \in D_t])}{\sum_{j'=1}^{n} \sum_{i'=1}^{m} \mathbb{I}[(i',j') \in D_t])}. \tag{2}$$

Cost-Sensitive Learning Although ξ can deal with the issue of mis-classified negative samples, it equally penalizes the mistakes on both positive and negative entries. However, in the implicit feedback scenario, the cost of missing a positive target is much higher than that of having a false-positive. Thus, we assume $r_{ij} = \mathbb{I}[\hat{r}_{ij} \geq q]$, where $q \in [0,1]$ is a threshold, and adopt cost-sensitive learning method with a more appropriate metric, such as the *sum* of weighted *recall* and *specificity*.

$$sum = \mu_p \times recall + \mu_n \times specificity, \tag{3}$$

where $\mu_p + \mu_n = 1$ and $0 \leq \mu_p, \mu_n \leq 1$. In general, the higher the *sum*, the better the performance. Besides, another appropriate metric is to measure the total *cost* of algorithm:

$$cost = c_p \times M_p + c_n \times M_n, \tag{4}$$

where M_p denotes the number of false negatives and M_n denotes the number of false positives. $c_p + c_n = 1$ and $0 \leq c_p, c_n \leq 1$ are the misclassification cost of positive and negative, respectively. In general, the lower the *cost* value, the better the performance.

Lemma 1. *The goal of maximizing the weighted sum in (3) or minimizing the weighted cost in (4) is equivalent to minimizing the following objective:*

$$\sum_{r_{ij}=+1} \rho \mathbb{I}_{(\hat{r}_{ij} \leq q)} + \sum_{r_{ij}=0} \mathbb{I}_{(\hat{r}_{ij} > q)}, \tag{5}$$

where $\rho = \frac{\eta_p T_n}{\eta_n T_p}$ *for the maximization of the weighted sum, and* $\rho = \frac{c_p}{c_n}$ *for the minimization of the weighted cost.* T_p *and* T_n *are the number of positive examples and negative examples, respectively.*

Lemma 1 gives the explicit objective function to optimize, but the indicator function is not convex. To solve this problem, we replace this objective function by its convex surrogate and derive the following two cost-sensitive loss functions:

$$\ell^I(r_{ij}, \hat{r}_{ij}) = \rho \mathbb{I}_{(r_{ij}=1)} \ell(\hat{r}_{ij}, 1) + \mathbb{I}_{(r_{ij}=0)} \ell(\hat{r}_{ij}, 0),$$
$$\ell^{II}(r_{ij}, \hat{r}_{ij}) = \mathbb{I}_{(r_{ij}=1)} \ell(\hat{r}_{ij}, \rho) + \mathbb{I}_{(r_{ij}=0)} \ell(\hat{r}_{ij}, 0).$$

We could find that the slope of $\ell^I(r_{ij}, \hat{r}_{ij})$ changes for specific class, leading to more "aggressive" updating while the required margin of $\ell^I(r_{ij}, \hat{r}_{ij})$ changes for specific class, resulting in more "frequent" updating.

Meta-learning By introducing cost-sensitive loss and divestiture loss, we can solve the problem of implicit feedback in online recommendation. However, how to decide the value of hyper-parameter like ρ remains an issue. Typically, in a batch learning setting, one can choose hyer-parameters with manual selection or via cross validation prior to the learning task, which is impossible for online learning setting. Moreover, in the real-world online recommender systems, user preferences and item attributes dynamically change. To address this issue, we adopt the meta-learning method [22] to exploit the benefit of multiple implicit MF models. The motivation is that if multiple implicit MF models with a number of hyper-parameters are learned simultaneously, there must exist one setting that is most appropriate to the streaming data. Specifically, take the hyper-parameter ρ as an example, we construct a pool of multiple values of parameter ρ by discretizing $(0,1)$ into S evenly distributed values $\frac{1}{S+1}, ..., \frac{s}{S+1}, ... \frac{S}{S+1}$ and setting ρ_s to $(1-\frac{s}{S+1})/(\frac{s}{S+1})$.

A remaining issue is how to choose appropriate implicit MF model from S candidates for prediction and update them at each round. We apply the Hedge algorithm [4] and randomly select a model according to a distribution $\mathbf{p}_t = (p_t^1, ..., p_t^S)$ such that $\sum_s p_t^s = 1$ and $p_t^s \geq 0$. The sampling probabilities represents the online predictive performance of each model which is defined as

$$p_t^s = \frac{\exp(\gamma M_t^s)}{\sum_{s=1}^S \exp(\gamma M_t^s)}, \; s = 1, ..., S, \tag{6}$$

where $\gamma > 0$ is a temperature hyper-parameter, and M_t^s is an online performance measure on historical data. Here we choose two commonly used performance measure in recommendation with implicit feedback task: F-measure and AUC [14, 23]. However, both of these two performance measures are non-decomposable which makes it significantly challenging to directly optimize them in the online process. Motivated by [22], we propose the following update methods.

Update F-measure. For each entry r_{ij}^t at round t, the model produces an N-size ranking list of items for user i. We denote h_t as the hit result that equals to 1 if item j appears in the ranking list and 0 otherwise. Then the F-measure can be computed by $\mathrm{F@N}_{t+1} = \frac{2\sum_{\tau=1}^t r_{ij}^\tau h_\tau}{\sum_{\tau=1}^t r_{ij}^\tau + \sum_{\tau=1}^t h_\tau}$. However, directly calculating the online F-measure by going through all entries in history is expensive. Therefore, we introduce an incremental calculation method according to [22]. Let $a_t = \sum_{\tau=1}^t r_{ij}^\tau h_\tau$ and $c_t = \sum_{\tau=1}^\tau (r_{ij}^\tau + h_\tau)$, then we can calculate $\mathrm{F@N}_{t+1} = \frac{2a_t}{c_t}$ and update a_t and c_t incrementally by

$$a_{t+1} = \begin{cases} a_t + 1, & \text{if } r_{ij}^{t+1} = 1 \text{ and } h_{t+1} = 1, \\ a_t, & \text{otherwise;} \end{cases}$$

$$c_{t+1} = \begin{cases} c_t + 2, & \text{if } r_{ij}^{t+1} = 1 \text{ and } h_{t+1} = 1, \\ c_t + 1, & \text{if } r_{ij}^{t+1} = 1 \text{ or } h_{t+1} = 1, \\ c_t, & \text{otherwise.} \end{cases}$$

Update AUC. Similar to F-measure case, directly calculating the online AUC is difficult, which requires to compare the present entry to historically received entries. To avoid storing the prediction results of all entries, we introduce two E-length hash tables L_+^t and L_-^t with ranges $(0, 1/E), (1/E, 2/E), ..., ((E-1)/E, 1)$. For $e \in 1, ..., E$, $L_+^t[e]$ and $L_-^t[e]$ store the number of positive entries and negative entries before round t respectively, whose prediction result $\hat{r}_{ij}$ are such that $1/(1+\exp(-\hat{r}_{ij})) \in [(e-1)/E, e/E)$. Then, we can approximately update the online AUC using the two hash tables according to [22]. In particular, if $r_{ij}^{t+1} = 1$, $\mathrm{AUC}_{t+1} = \frac{N_+^t}{N_+^t+1}\mathrm{AUC}_t + \frac{1}{(N_+^t+1)N_-^t}\left(\sum_{k=1}^e L_-^t[k] + \frac{L_-^t[e+1]}{2}\right)$, where e is the largest index such that $e/E \leq 1/(1+\exp(-\hat{r}_{ij}^{t+1}))$; if $r_{ij}^{t+1} = 0$, we have $\mathrm{AUC}_{t+1} = \frac{N_-^t}{N_-^t+1}\mathrm{AUC}_t + \frac{1}{N_+^t(N_-^t+1)}\left(\sum_{k=e+1}^{E-1} L_-^t[k] + \frac{L_+^t[e]}{2}\right)$, where e is the smallest index such that $e/E \geq 1/(1+\exp(-\hat{r}_{ij}^{t+1}))$.

Now we can summarize our proposed framework in Algorithm 1. We could use any online optimization methods [17] to solve the objective in (1). A default choice is online gradient descent (OGD) [17] due to its simplicity and popularity. At each round t, we alternatively update $\mathbf{u}_i^t$ and $\mathbf{v}_j^t$ while keeping the other matrix fixed, the update rules are:

$$\mathbf{u}_i^{t+1} = \mathbf{u}_i^t - \eta_t \nabla_{\mathbf{u}_i} \xi(r_{ij}^t, \hat{r}_{ij}^t), \tag{7}$$

$$\mathbf{v}_j^{t+1} = \mathbf{v}_j^t - \eta_t \nabla_{\mathbf{v}_j} \xi(r_{ij}^t, \hat{r}_{ij}^t), \tag{8}$$

where η_t is the learning rate.

Algorithm 1 The OCFIF Framework

```
Input: the number of models S
Randomly initialize U_s, V_s for s = 1, 2, ..., S, p_1 = (1/S, 1/S, ..., 1/S);
for t = 1, 2, ..., T do
   Receive an observed entry r_{i_t j_t};
   Sample negative item set Z from B_t \ {(i_t, j_t)};
   for all r_ij ∈ {{r_{i_t j_t}} ∪ {r_{i'j'} = 0 | i' = i, j' ∈ Z}} do
      Sampling a model s according to the distribution in (6);
      Compute prediction r̂_ij and loss ξ;
      for s = 1, 2, ..., S do
         Update u_{s,i}, v_{s,j} with Eq. (7), (8);
         Update the performance M_s;
      end for
      Update p_{t+1} with Eq. (6);
   end for
   Update B_{t+1} ← B_t \ {(i_t, j_t) ∪ Z}, N_{t+1} ← N_t ∪ Z and D_{t+1} ← D_t ∪ (i_t, j_t);
end for
```

3.3 Theoretical Analysis

We now analyze the theoretical performance of the OCFIF framework in terms of online regret bound analysis. To ease our discussion, we simplify some notations in our analysis as in Table 1:

We denote by $\mathcal{S}$ the set of indexes that correspond to the trials when a loss happens, $\mathcal{S} = \{t|\xi_t(\mathbf{w}_t) > 0\}$. Similarly, we denote by $\mathcal{S}_p = \{t|\xi_t(\mathbf{w}_t) > 0$ and $y_t = 1\}$, $\mathcal{S}_n = \{t|\ell_t(\mathbf{w}_t) > 0$ and $y_t = 0\}$, $S_p = |\mathcal{S}_p|$, $S_n = |\mathcal{S}_n|$.

Theorem 1. *Let $(\mathbf{x}_1, y_1), ..., (\mathbf{x}_T, y_T)$ be a sequence of input-target pairs, where $\mathbf{x}_t \in \mathbb{R}^k$ and $G = \max_t \|\mathbf{x}_t\|^2$, $y_t \in \{0, 1\}$. Let $\mathbf{w}_1, ..., \mathbf{w}_T$ be a sequence of vectors obtained by the proposed algorithm. Then for any $\mathbf{w} \in \mathbb{R}^k$, by setting*

Table 1. Simplification of notations.

Notations		Meaning
$\mathbf{v}_j^t$ or $\mathbf{u}_i^t$	$\mathbf{x}_t$	input
$\mathbf{u}_i^t$ or $\mathbf{v}_j^t$	$\mathbf{w}_t$	current status
$\mathbf{u}_i^{t+1}$ or $\mathbf{v}_j^{t+1}$	$\mathbf{w}_{t+1}$	solution
$\mathbf{u}$ or $\mathbf{v}$	$\mathbf{w}$	variable
r_{ij}^t	y_t	target

$\eta = \frac{\|\mathbf{w}\|}{\sqrt{G[(\rho+\rho\lambda)^2 S_p + S_n]}}$ for ξ^I, $\eta = \frac{\|\mathbf{w}\|}{\sqrt{G[(1+\lambda)^2 S_p + S_n]}}$ for ξ^{II}, we then have the bounds of the proposed algorithms:

$$\sum_{t=1}^{T} \xi_t^I(\mathbf{w}_t) - \sum_{t=1}^{T} \xi_t^I(\mathbf{w}) \leq \|\mathbf{w}\| \sqrt{G[(\rho + \rho\lambda)^2 S_p + S_n]},$$
$$\sum_{t=1}^{T} \xi_t^{II}(\mathbf{w}_t) - \sum_{t=1}^{T} \xi_t^{II}(\mathbf{w}) \leq \|\mathbf{w}\| \sqrt{G[(1 + \lambda)^2 S_p + S_n]}.$$

Proof. Relying on the definition of OGD, we have

$$\begin{aligned} \|\mathbf{w}_{t+1} - \mathbf{w}\| &= \|\mathbf{w}_t - \eta \nabla \xi_t(\mathbf{w}_t) - \mathbf{w}\|^2 \\ &= \|\mathbf{w}_t - \mathbf{w}\|^2 + \eta^2 \|\nabla \xi_t(\mathbf{w}_t)\|^2 - 2\eta \nabla \xi_t(\mathbf{w}_t)(\mathbf{w}_t - \mathbf{w}). \end{aligned}$$

For the convexity of the loss function: $\xi_t(\mathbf{w}_t) - \xi_t(\mathbf{w}) \leq \nabla \xi_t(\mathbf{w}_t)(\mathbf{w}_t - \mathbf{w})$), we have the following:

$$\xi_t(\mathbf{w}_t) - \xi_t(\mathbf{w}) \leq \frac{\|\mathbf{w}_t - \mathbf{w}\|^2 - \|\mathbf{w}_{t+1} - \mathbf{w}\|^2}{2\eta} + \frac{\eta}{2} \|\nabla \xi_t(\mathbf{w}_t)\|^2.$$

Summing over $t = 1, ..., T$, gives

$$\sum_{t=1}^{T} (\xi_t(\mathbf{w}_t) - \xi_t(\mathbf{w})) \leq \frac{\|\mathbf{w}\|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\nabla \xi_t(\mathbf{w}_t)\|^2.$$

When we adopt ξ^I, it is easy to see that $\|\nabla \xi_t(\mathbf{w}_t)\| \leq \sqrt{G}$ if $t \in \mathcal{S}_n$, $\|\nabla \xi_t(\mathbf{w}_t)\| \leq \rho(1 + \lambda)\sqrt{G}$ if $t \in \mathcal{S}_p$ and $\|\nabla \xi_t(\mathbf{w}_t)\| = 0$ otherwise. Thus, we can obtain the bound $\frac{\|\mathbf{w}\|^2}{2\eta} + \frac{\eta G[(\rho+\rho\lambda)^2 S_p + S_n]}{2}$, by setting $\eta = \frac{\|\mathbf{w}\|}{\sqrt{G[(\rho+\rho\lambda)^2 S_p + S_n]}}$. Similarly, when we adopt ξ^{II}, $\|\nabla \xi_t(\mathbf{w}_t)\| \leq \sqrt{G}$ if $t \in \mathcal{S}_n$, $\|\nabla \xi_t(\mathbf{w}_t)\| \leq (1 + \lambda)\sqrt{G}$ if $t \in \mathcal{S}_p$ and $\|\nabla \xi_t(\mathbf{w}_t)\| = 0$ otherwise. Thus, we can obtain the bound $\frac{\|\mathbf{w}\|^2}{2\eta} + \frac{\eta G[(1+\lambda)^2 S_p + S_n]}{2}$ by setting $\eta = \frac{\|\mathbf{w}\|}{\sqrt{G[(1+\lambda)^2 S_p + S_n]}}$.

Because $S_p + S_n \leq T$, we get a regret bound $\sqrt{T}$. From this theorem, our framework is guaranteed to converge to obtain the optimal average loss with respect to the online learning setting with divestiture loss and cost-sensitive loss.

According to the theory of the Hedge algorithm [4], we can show that the OCFIF framework can achieve an optimal upper bound of regret by $\sqrt{T \ln S/2}$ with S models after T iterations. This implies that it can asymptotically approach the most appropriate paramenter setting and ensure the per-round regret vanishes over time in a sub-linear rate. The details of the proof can be found in [4].

4 Experiments

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

RQ1: Does our proposed OCFIF framework outperform the state-of-the-art online implicit collaborative filtering methods?

RQ2: How do our sampling strategies perform? Which negative sampling strategy is better?

RQ3: How sensitive is our framework to hyper parameters?

In what follows, we first present the experimental settings, followed by answering the above three research questions.

4.1 Experimental Setting

Dataset We experimented with three publicly accessible datasets: MovieLens[7], Yelp[8] and Pinterest[9]. The characteristics of the three datasets are summarized in Table 2.

- MovieLens. This movie rating dataset has been widely used in recommendation task. We used the version containing one million ratings, where each user has at least 20 ratings. We transformed it into implicit data, where each entry is marked as 0 or 1 indicating whether the user has rated the item.

- Yelp. This is the Yelp Challenge data of user ratings on businesses. We use the filtered subset created by [7].

- Pinterest. This implicit feedback data is constructed by [5] for evaluating content-based image recommendation.The original data is very large but highly sparse. We filtered the dataset in the same way as for the MovieLens data that retained only users with at least 20 interactions (pins). This results in a subset of the data that contains 55,187 users and 1,500,809 interactions.

Evaluation Metrics For experimental setup, each dataset is randomly divided into two parts: 80% for training and 20% for test. We repeat such a random permutation 10 times for each dataset and compute the average results of each algorithm over the 10 runs. For the metrics, The accuracy of a recommendation model is measured by two widely-used metrics, namely AUC and F-measure@N.

[7] http://grouplens.org/datasets/movielens/
[8] https://www.yelp.com/dataset/challenge
[9] https://sites.google.com/site/xueatalphabeta/academic-projects

AUC measures whether the items which are observed were held out during learning are ranked higher than unobserved items. F-measure@N is the weighted harmonic mean of precision and recall. In our experiments, we set $N = 20$. Basically, the higher these measures, the better the performance. For each metric, We report the score averaged by all the users.

Baselines We compare three variants of the proposed OCFIF framework with the state-of-the-art algorithms for online recommendation tasks with implicit feedback as follows:

- OBPR. BPR [14] is a sample-based method that optimizes the pair-wise ranking between the positive and negative samples via SGD. We propose an Online BPR by online incremental learning [15]. We use a fixed learning rate, varying it and reporting the best performance.

- ISGD [19]. This is a incremental matrix factorization method for positive-only feedback. It learns by incremental SGD, which is acceptable for streaming data.

- NN-APA [9]. A second order online collaborative filtering algorithm.

- OCFIF. The proposed OCFIF framework. To examine the effectiveness of the framework on different components, we run two variants of OCFIF, OCFIF-C and OCFIF-CD. OCFIF-C only adopts cost-sensitive loss, and OCFIF-CD consider both cost-sensitive loss and divestiture loss.

For parameter settings, we adopt the same parameter tuning schemes for all the compared algorithm to enable fair comparisons. We perform grid search to choose the best parameters for each algorithm on the training set. For MF methods, the number of latent factors is tuned from $\{10, 15, ..., 50\}$. For OCFIF, we search the ranges of values for negative sample size from $\{1, 5, 10, ..., 50\}$, cost sensitive parameter $\rho = \frac{c_p}{c_n}$ tuned with c_p from $\{0.5, 0.55, ..., 0.95\}$ with a stepsize of 0.5, and extra loss parameter $\lambda \in \{0.1, 0.2, 0.5, 1, 2\}$. We set the number of models $S = 10$ and adopts uniform sampling strategy.

Table 2. Statistics of the evaluation datasets.

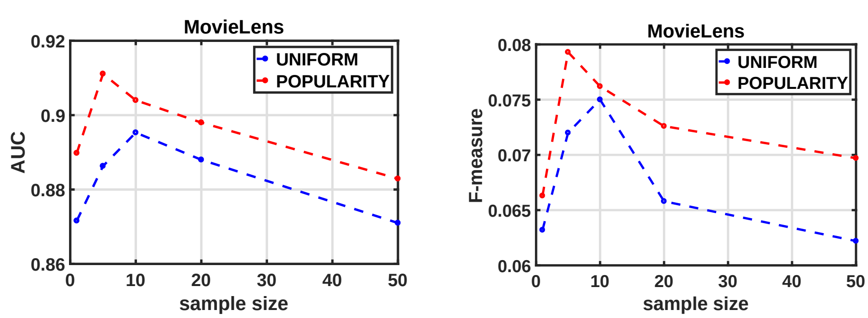
Dataset	#Interaction	#Item	#User
MovieLens	1000209	3706	6040
Yelp	731671	25815	25677
Pinterest	1500809	9916	55187

4.2 Performance Comparison (RQ1)

Table 3 summarizes the comparison results in terms of AUC and F-measure, from which we can draw several observations. First, it is clear to see that the proposed OCFIF framework and its variants significantly outperform OBPR, ISGD and NN-APA on all the datasets. This encouraging results validate the effectiveness of utilizing cost-sensitive loss. Furthermore, by examining the two variants of the proposed framework, we found that the OCFIF-CD with both divestiture loss and cost-sensitive loss significantly outperforms the OCFIF-C with only cost-sensitive loss. The reason is that divestiture loss could heal the bias derived from the past mis-classified negative samples. By further comparing OCFIF with its variants, we found that OCFIF is able to achieve the best performance. This result highlights the importance of meta learning which dynamically explores a pool of multiple models to avoid the limitations of a single fixed model. Interestingly, we can observe that NN-APA outperform OBRP and ISGD in most cases, which is consistent with [9] that exploiting second-order information can improve the performance for explicit feedback. This could be a potential direction for OCFIF.

Table 3. Comparison of different algorithms in terms of AUC and F-measure for recommendation task.

Algorithm	Metrics	Movielens	Yelp	Pinterest
OBPR	AUC	0.8433±0.0012	0.7754±0.0004	0.6996±0.0023
	F-measure	0.0487±0.0004	0.0105±0.0004	0.0210±0.0003
ISGD	AUC	0.8578±0.0009	0.8165±0.0018	0.8620±0.0018
	F-measure	0.0618±0.0008	0.0150±0.0003	0.0231±0.0005
NN-APA	AUC	0.8600±0.0003	0.8427±0.0012	0.8810±0.0008
	F-measure	0.0581±0.0008	0.0155±0.0001	0.0246±0.0004
OCFIF-C	AUC	0.8953±0.0010	0.8621±0.0016	0.8901±0.0022
	F-measure	0.0750±0.0007	0.0162±0.0003	0.0259±0.0007
OCFIF-CD	AUC	0.9043±0.0008	0.9105±0.0011	0.9012±0.0016
	F-measure	0.0788±0.0009	0.0176±0.0005	0.0263±0.0006
OCFIF	AUC	**0.9105±0.0006**	**0.9126±0.0012**	**0.9312±0.0013**
	F-measure	**0.0809±0.0004**	**0.0186±0.0005**	**0.0275±0.0002**



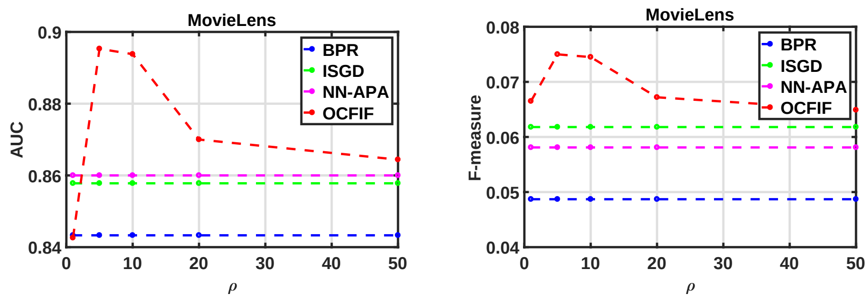
(a) AUC

(b) F-measure

Fig. 2. Evaluation of different sample strategies: AUC (a) and F-measure (b).

4.3 Sampling Strategies Comparison (RQ2)

We compared OCFIF under different sampling strategies: uniform sampling and popularity based sampling. In particular, we set the sampling size gradually increasing from 0 to 50, and report the performance under different sampling strategies. Figure 2(a) shows the performance evaluated by AUC and figure 2(b) shows the performance evaluated by F-measure. We can observe that popularity based sampling strategy is able to achieve the better performance than uniform sampling strategy. Moreover, the results show the same trend that better performance is obtained by increasing the sampling size. However, when sampling size is set too large, the performance suffers. The reason is that the likelihood of positive entries in negative sampling set will increase rapidly when sampling size is too large, which results in severely negative side-effect on the model.



(a) AUC (b) F-measure

Fig. 3. Impact of cost sensitive parameter ρ.

4.4 Evaluation of Parameter Sensitivity (RQ3)

For the proposed OCFIF framework, there are two key parameters: cost sensitive paramenter ρ and divestiture loss paramenter λ. Figure 3 and Figure 4 show the results of parameter sensitive evaluations using different values of ρ and λ.

First of all, by examining the influence of cost sensitive parameter ρ, we found that the performance of framework is gradually improved with the increase of ρ. This indicates the effectiveness of our cost sensitive learning. Moreover, when ρ is larger than 10, the performance starts to drop significantly. This reveals the drawback of over-updating of positive entries.

Second, by examining the influence of divestiture loss parameter λ, we found that the better performance is achieved by balancing between the impact of cost-sensitive loss and divestiture loss, while either a large or a small value of λ will adversely degrade the performance. This is primarily because that too large divestiture loss can cause excessive correction, then reducing the accuracy of model.

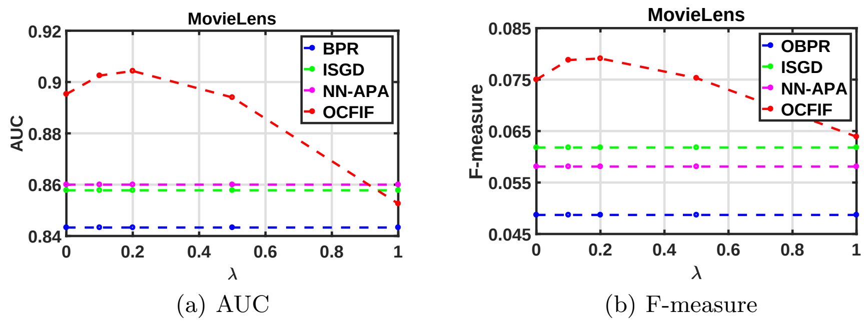


Fig. 4. Impact of divestiture loss parameter λ.

5 CONCLUSION

In this work, we propose a unified framework for online collaborative filtering with implicit feedback. Specifically, motivated by the regret aversion, we propose a divestiture loss to heal the bias derived from the past mis-classified negative samples. Furthermore, we adopt cost-sensitive learning method to efficiently optimize the implicit MF model without imposing a heuristic weight restriction on missing data. By leveraging meta-learning, we dynamically explore a pool of multiple models to avoid the limitations of a single fixed model so as to remedy the drawback of manual/heuristic model selection. We also analyze the theoretical bounds of the proposed OCFIF method, conduct extensive experiments and ablation studies, and achieve state-of-the-arts results on real-world datasets for online recommendation with implicit feedback task.

References

1. Chang, S., Zhang, Y., Tang, J., Yin, D., Chang, Y., Hasegawa-Johnson, M.A., Huang, T.S.: Streaming recommender systems. In: Proceedings of the 26th International Conference on World Wide Web. pp. 381–389. International World Wide Web Conferences Steering Committee (2017)
2. Ding, J., Feng, F., He, X., Yu, G., Li, Y., Jin, D.: An improved sampler for bayesian personalized ranking by leveraging view data. In: Companion of the The Web Conference 2018 on The Web Conference 2018. pp. 13–14. International World Wide Web Conferences Steering Committee (2018)
3. Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., Herrera, F.: Cost-sensitive learning. In: Learning from Imbalanced Data Sets, pp. 63–78. Springer (2018)
4. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences 55(1), 119–139 (1997)
5. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web. pp. 173–182. International World Wide Web Conferences Steering Committee (2017)

6. He, X., Tang, J., Du, X., Hong, R., Ren, T., Chua, T.S.: Fast matrix factorization with non-uniform weights on missing data. arXiv preprint arXiv:1811.04411 (2018)
7. He, X., Zhang, H., Kan, M.Y., Chua, T.S.: Fast matrix factorization for online recommendation with implicit feedback. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. pp. 549–558. ACM (2016)
8. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. pp. 263–272. Ieee (2008)
9. Liu, C., Hoi, S.C., Zhao, P., Sun, J., Lim, E.P.: Online adaptive passive-aggressive methods for non-negative matrix factorization and its applications. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. pp. 1161–1170. ACM (2016)
10. Liu, C., Jin, T., Hoi, S.C., Zhao, P., Sun, J.: Collaborative topic regression for online recommender systems: an online and bayesian approach. Machine Learning 106(5), 651–670 (2017)
11. Lu, J., Hoi, S., Wang, J.: Second order online collaborative filtering. In: Asian Conference on Machine Learning. pp. 325–340 (2013)
12. Mathioudakis, M., Koudas, N.: Twittermonitor: trend detection over the twitter stream. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. pp. 1155–1158. ACM (2010)
13. Rendle, S., Freudenthaler, C.: Improving pairwise learning for item recommendation from implicit feedback. In: Proceedings of the 7th ACM international conference on Web search and data mining. pp. 273–282. ACM (2014)
14. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence. pp. 452–461. AUAI Press (2009)
15. Rendle, S., Schmidt-Thieme, L.: Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In: Proceedings of the 2008 ACM conference on Recommender systems. pp. 251–258. ACM (2008)
16. Seiler, M., Seiler, V., Traub, S., Harrison, D.: Regret aversion and false reference points in residential real estate. Journal of Real Estate Research 30(4), 461–474 (2008)
17. Shalev-Shwartz, S., et al.: Online learning and online convex optimization. Foundations and Trends® in Machine Learning 4(2), 107–194 (2012)
18. Vanschoren, J.: Meta-learning: A survey. arXiv preprint arXiv:1810.03548 (2018)
19. Vinagre, J., Jorge, A.M., Gama, J.: Fast incremental matrix factorization for recommendation with positive-only feedback. In: International Conference on User Modeling, Adaptation, and Personalization. pp. 459–470. Springer (2014)
20. Wang, J., Hoi, S.C., Zhao, P., Liu, Z.Y.: Online multi-task collaborative filtering for on-the-fly recommender systems. In: Proceedings of the 7th ACM conference on Recommender systems. pp. 237–244. ACM (2013)
21. Wang, W., Yin, H., Huang, Z., Wang, Q., Du, X., Nguyen, Q.V.H.: Streaming ranking based recommender systems. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 525–534. ACM (2018)
22. Yan, Y., Yang, T., Yang, Y., Chen, J.: A framework of online learning with imbalanced streaming data. (2017)
23. Yang, P., Zhao, P., Liu, Y., Gao, X.: Robust cost-sensitive learning for recommendation with implicit feedback. In: Proceedings of the 2018 SIAM International Conference on Data Mining. pp. 621–629. SIAM (2018)