

A Generic Coordinate Descent Framework for Learning from Implicit Feedback

Immanuel Bayer*
University of Konstanz,
Germany
immanuel.bayer@uni-
konstanz.de

Bhargav Kanagal
Google Inc., USA
bhargav@google.com

Xiangnan He*
National University of
Singapore, Singapore
xiangnan@comp.nus.edu.sg

Steffen Rendle
Google Inc., USA
srendle@google.com

ABSTRACT

In recent years, interest in recommender research has shifted from explicit feedback towards implicit feedback data. A diversity of complex models has been proposed for a wide variety of applications. Despite this, learning from implicit feedback is still computationally challenging. So far, most work relies on stochastic gradient descent (SGD) solvers which are easy to derive, but in practice challenging to apply, especially for tasks with many items. For the simple matrix factorization model, an efficient coordinate descent (CD) solver has been previously proposed. However, efficient CD approaches have not been derived for more complex models.

In this paper, we provide a new framework for deriving efficient CD algorithms for complex recommender models. We identify and introduce the property of k -separable models. We show that k -separability is a sufficient property to allow efficient optimization of implicit recommender problems with CD. We illustrate this framework on a variety of state-of-the-art models including factorization machines and Tucker decomposition. To summarize, our work provides the theory and building blocks to derive efficient implicit CD algorithms for complex recommender models.

1. INTRODUCTION

In recent years, the focus of recommender system research has shifted from explicit feedback problems such as rating prediction to implicit feedback problems. Most of the signal that a user provides about her preferences is *implicit*. Examples for implicit feedback are: a user watches a video, clicks on a link, etc. Implicit feedback data is much cheaper to obtain than explicit feedback, because it comes with no extra cost for the user and thus is available on a much larger scale. However, learning a recommender system from implicit feedback is computationally expensive because the observed actions of a user need to be contrasted against *all* the non-observed actions [5, 13].

Stochastic gradient descent (SGD) and coordinate descent (CD) are two widely used algorithms for large scale machine learning. Both algorithms are considered state-of-the-art for learning matrix factorization models from implicit feedback and have been studied extensively. SGD and CD have shown different strengths and weaknesses on various data sets [4,

17, 16, 8, 25, 15, 22, 26]. While SGD is available as a general framework to optimize a broad class of models [13], CD is only available for a few simple models [5, 10]. In fact, it is even unknown if CD can be used to efficiently optimize complex recommender models. Our work closes this gap and identifies a model property called k -separability, that is a sufficient condition to allow efficient learning from implicit feedback. Based on k -separability, we provide a general framework to derive efficient implicit CD solvers.

Our paper is organized as follows: First, we introduce the problem of learning from implicit feedback and show that the number of implicit training examples makes the application of standard algorithms challenging. Next, we provide our general framework for efficient implicit learning with CD. We identify k -separability of a model as a sufficient property to make efficient learning feasible and introduce iCD, a generic learning algorithm for k -separable models. In Section 5, we show how to apply iCD to a diverse set of models, including, matrix factorization (MF), factorization machines (FM) and tensor factorization. This section serves both as solutions to popular models as well as a guide for applying the framework to other complex recommender models.

To summarize, our contributions are:

- We identify a basic property of recommender models that allows efficient CD learning from implicit data.
- We provide iCD, a framework to derive efficient implicit CD algorithms.
- We apply the framework and derive algorithms for MF, MF with side information, FM, PARAFAC and Tucker Decomposition.

2. RELATED WORK

Since several years, matrix factorization (MF) is regarded as the most effective, basic recommender system model. Two optimization strategies dominate the research on MF from implicit feedback data. The first one is Bayesian Personalized Ranking (BPR) [13], a stochastic gradient descent (SGD) framework, that contrasts pairs of consumed to non-consumed items. The second one is coordinate descent (CD) also known as alternating least squares on an elementwise loss over both the consumed and non-consumed items [5]. In terms of the loss formulation, BPR's pairwise classification

*Work done at Google.

		Item I		
Context C		2	3	
			4	
		5		2
		3	1	4
			3	

		Item I			
Context C		1	0	1	0
		0	0	1	0
		0	2	0	1
		1	1	0	2
		0	0	1	0

Figure 1: Left: Explicit rating data, with $S = \{(c_1, i_1, 2), (c_1, i_3, 3), (c_2, i_3, 4), \dots\}$. Right: Implicit data, e.g., watch/ purchase/ click count, $|S_{\text{impl}}| = |C||I|$.

loss is better suited for ranking whereas CD loss is better suited for numerical data. With regard to the optimization task, both techniques face the same challenge of learning over a very large number of training examples. BPR tackles this issue by sampling negative items, but it has been shown that BPR has convergence problems when the number of items is large [7, 12]. It requires more complex, non-uniform, sampling strategies for dealing with this problem [12, 6]. On the other hand, for CD-MF, Hu et al. [5] have derived an efficient algorithm that allows to optimize over the large number of non-consumed items without any cost. This computational trick is exact and does not involve sampling. Many authors have compared both CD-MF and BPR-MF on a variety of datasets and some work reports better quality for BPR-MF [4, 17, 16, 8] whereas for other problems CD-MF works better [8, 25, 15, 22, 26]. This large body of results indicates that the advantages of CD and BPR are orthogonal and both approaches have their merits.

Our discussion so far was focused on learning matrix factorization models from implicit data. Shifting from simple matrix factorization to more complex factorization models has shown large success in many implicit recommendation problems [2, 4, 18, 1, 9, 24]. However, work on complex factorization models relies almost exclusively on SGD optimization using the generic BPR framework. Our work, provides the theory as well as a practical framework for deriving CD learners for such complex models. Like CD for MF, our generic algorithm is able to optimize on all non-consumed items without explicitly iterating over them. To summarize, our paper enables researchers and practitioners to apply CD in their work and gives them a choice between the advantages of BPR and CD.

3. PROBLEM STATEMENT

Let I be a set of items and C a set of contexts. Let S be a set of observed feedback where a tuple $(c, i, y, \alpha) \in S$ indicates that in context c , a score y has been assigned to item i with confidence α . See Figure 1 for an illustration. We use a general notation of context which can include for instance user, time, location, attributes, history, etc. Section 5 and Section 6 show more examples for context.

3.1 Recommender Model

A recommender model $\hat{y} : C \times I \rightarrow \mathbb{R}$ is a function that assigns a score to every context-item pair. The model \hat{y} is parameterized by a set of model parameters Θ . The model \hat{y} is typically used to decide which items to present in a given context.

The learning task is to find the values of the model parameters that minimize a loss over the data S , e.g., a squared loss

$$L(\Theta|S) = \sum_{(c,i,y,\alpha) \in S} \alpha (\hat{y}(c, i) - y)^2 + \sum_{\theta \in \Theta} \lambda_\theta \theta^2 \quad (1)$$

where λ_θ is an regularization constant for parameter θ .

3.2 Coordinate Descent Algorithm

Objective (1) can be minimized by coordinate descent (CD). CD iterates through the model parameters and updates one parameter at a time. For a selected parameter $\theta \in \Theta$, CD computes the first L' and second derivative L'' of L with respect to the selected coordinate θ :

$$L'(\theta|S) = 2 \sum_{(c,i,y,\alpha) \in S} \alpha (\hat{y}(c, i) - y) \hat{y}'(c, i) + 2 \lambda_\theta \theta \quad (2)$$

$$L''(\theta|S) = 2 \sum_{(c,i,y,\alpha) \in S} \alpha [(\hat{y}(c, i) - y) \hat{y}''(c, i) + \hat{y}'(c, i)^2] + 2 \lambda_\theta \quad (3)$$

and performs a Newton update step:

$$\theta \leftarrow \theta - \eta \frac{L'(\theta|S)}{L''(\theta|S)} \quad (4)$$

where $\eta \in (0, 1]$ is the step size. For multilinear models, a full step, i.e., $\eta = 1$, can be chosen without risking divergence [11]. All models in Section 5 fall into this category.

Such CD algorithms have been well studied and the runtime complexity is typically linear in the complexity of the training examples and embedding dimension. For MF, [23] shows a complexity of $\mathcal{O}(|S|k)$ and for FM, [11] derives a complexity of $\mathcal{O}(N_Z(X)k)$ where $N_Z(X)$ is the number of non-zero entries in the design matrix X . The linear runtime complexity in the number of training examples makes these algorithms well suited for explicit recommendation settings, however, they become infeasible for implicit problems.

3.3 Learning from Implicit Feedback

In an implicit recommendation problem, the non-consumed items are meaningful and cannot be ignored. For instance, in Figure 1 (right), the data depicts how often each item was consumed in a context in the past. The non-consumed items, i.e., the ones with a count of zero, are useful to learn user preferences. To formalize, the training data S_{impl} of an implicit problem consists of a set S^+ of observed feedback and all the non-consumed tuples S^0

$$S_{\text{impl}} = S^+ \cup S^0, \quad |S_{\text{impl}}| = |C||I| \quad (5)$$

with

$$\forall (c, i, y, \alpha) \in S^0 : y = 0, \alpha = \alpha_0. \quad (6)$$

S^+ contains the observed feedback and is of much smaller scale than S_{impl} , usually $|S^+| \ll |C||I|$.

The implicit learning problem can be stated as minimizing the objective in eq. (1) over the implicit data S_{impl} . While possible in theory, in practice, it is infeasible to apply the learning algorithms of Section 3.2 to this problem due to their linear computational runtime in the size of the training data which is $|S_{\text{impl}}| = |C||I|$ for implicit problems. Our paper shows how to derive efficient CD algorithms for optimizing eq. (1) over implicit data.

4. GENERIC COORDINATE DESCENT ALGORITHM FOR IMPLICIT FEEDBACK

4.1 Implicit Regularizer

As discussed in Section 3.3, the reason why training on implicit data is challenging is the large number of implicit examples S^0 which is typically $|S^0| \in \mathcal{O}(|C||I|)$. Note that S^0 includes all context-item pairs that are **not** in S^+ . We show now that we can rephrase the optimization criterion to sum over **all** context-item pairs. This reformulation is a prerequisite to later allow the decomposition of the loss in Section 4.2. Moreover it allows to study implicit optimization without having to consider S^+ .

LEMMA 1. *Implicit learning can be rephrased as a combination of learning on a small positive set and minimizing the scoring function on **any** context-item pair.*

$$\operatorname{argmin}_{\Theta} L(\Theta|S_{\text{impl}}) = \operatorname{argmin}_{\Theta} \left(L(\Theta|S) + \alpha_0 \underbrace{\sum_{c \in C} \sum_{i \in I} \hat{y}(c, i)^2}_{=: R(\Theta)} \right) \quad (7)$$

where the observed feedback is rescaled

$$S := \left\{ \left(c, i, \frac{\alpha}{\alpha - \alpha_0} y, \alpha - \alpha_0 \right) : (c, i, y, \alpha) \in S^+ \right\}. \quad (8)$$

PROOF. Per definition of the loss (eq. 1) and the implicit training set S_{impl} (eq. 5)

$$\begin{aligned} L(\Theta|S_{\text{impl}}) &= L(\Theta|S^+) + \alpha_0 \sum_{(c, i) \in S^0} \hat{y}(c, i)^2 \\ &= L(\Theta|S^+) - L(\Theta|\{(c, i, 0, \alpha_0) : (c, i, y, \alpha) \in S^+\}) + \alpha_0 R(\Theta) \end{aligned}$$

We further can collapse each pair of examples into a single one. We show this for the pair $(c, i, y, \alpha) \in S$ and its counterpart $(c, i, 0, -\alpha_0)$.

$$\begin{aligned} &L(\Theta|\{(c, i, y, \alpha)\}) + L(\Theta|\{(c, i, 0, -\alpha_0)\}) \\ &= \alpha (\hat{y}(c, i) - y)^2 - \alpha_0 \hat{y}(c, i)^2 \\ &= (\alpha - \alpha_0) \left(\hat{y}(c, i)^2 - 2 \frac{\alpha}{\alpha - \alpha_0} y \hat{y}(c, i) + \frac{\alpha}{\alpha - \alpha_0} y^2 \right) \\ &= (\alpha - \alpha_0) \left(\hat{y}(c, i) - \frac{\alpha}{\alpha - \alpha_0} y \right)^2 + \text{const} \\ &= L \left(\Theta \left| \left\{ \left(c, i, \frac{\alpha}{\alpha - \alpha_0} y, \alpha - \alpha_0 \right) \right\} \right. \right) + \text{const} \end{aligned}$$

The additional constant does not change the optimum for Θ , so rescaling of examples as in eq. (8) preserves the optimum. \square

The lemma allows an interesting interpretation of implicit learning tasks. Implicit problems can be seen as explicit or one-class problems with an additional *implicit regularizer* or bias $R(\Theta)$ for predicting zeros. Compared to a common regularizer such as L2, the implicit regularizer is aware of the model \hat{y} . L2 penalizes non-zero *model parameters* Θ whereas the implicit regularizer penalizes non-zero *predictions* \hat{y} . Consequently, the implicit regularizer is less restrictive than L2 because small predictions can be achieved even with large model parameters.

4.2 iCD Algorithm for k -separable Models

As shown in eq. (7), implicit learning can be formulated as explicit learning on a small set S with an expensive implicit regularizer R . Learning models over an explicit loss is already well studied [23, 11], so we focus now on the implicit regularizer

$$R(\Theta) = \sum_{c \in C} \sum_{i \in I} \hat{y}(c, i)^2 \quad (9)$$

The general computational complexity is $\mathcal{O}(|C||I|)$.

In this section, we introduce the concept of a k -separable model. We will provide an efficient implicit CD solver for any k -separable model. In Section 5, we show that many common models are k -separable, including matrix factorization, feature-based approaches such as factorization machines, but also higher-order tensor factorization such as PARAFAC or Tucker decomposition. The iCD framework that we derive in this section is not limited to the models described above but can serve as a blueprint for other k -separable models as well.

DEFINITION 1 (k -SEPARABLE). *A model $\hat{y}(c, i)$ is called k -separable iff the model can be rewritten as*

$$\hat{y}(c, i) = \langle \phi(c), \psi(i) \rangle = \sum_{f=1}^k \phi_f(c) \psi_f(i) \quad (10)$$

with functions

$$\phi : C \rightarrow \mathbb{R}^k, \quad \psi : I \rightarrow \mathbb{R}^k \quad (11)$$

where ϕ is parameterized by Θ^C and ψ is parameterized by Θ^I with $\Theta^C \cap \Theta^I = \emptyset$.

LEMMA 2. *The implicit regularizer of any k -separable model can be decomposed to:*

$$R(\Theta) = \sum_{f=1}^k \sum_{f'=1}^k \underbrace{\sum_{c \in C} \phi_f(c) \phi_{f'}(c)}_{=: J_C(f, f')} \underbrace{\sum_{i \in I} \psi_f(i) \psi_{f'}(i)}_{=: J_I(f, f')} \quad (12)$$

PROOF. The lemma follows from inserting the k -separable model (eq. 10) into the implicit regularizer (eq. 9) and rearranging the summations.

$$\begin{aligned} R(\Theta) &= \sum_{c \in C} \sum_{i \in I} \sum_{f=1}^k \phi_f(c) \psi_f(i) \sum_{f'=1}^k \phi_{f'}(c) \psi_{f'}(i) \\ &= \sum_{f=1}^k \sum_{f'=1}^k \left(\sum_{c \in C} \phi_f(c) \phi_{f'}(c) \right) \left(\sum_{i \in I} \psi_f(i) \psi_{f'}(i) \right) \end{aligned}$$

\square

This lemma is key to efficient learning algorithms from implicit data. It shows that the context and item sides can be computed independently, which drops the computational complexity from $\mathcal{O}(|C||I|)$ to $\mathcal{O}((|C| + |I|)k^2)$. Next, we show how this can be used for gradient computation which is required for the update step in CD (see eq. 4).

LEMMA 3. *The implicit regularizer gradients of any k -separable model with respect to any model parameter $\theta \in \Theta^C$ (or analogously $\theta \in \Theta^I$), can be simplified to*

$$R'(\theta) = 2 \sum_{f=1}^k \sum_{f'=1}^k J_I(f, f') \sum_{c \in C} \phi_f(c) \phi_{f'}(c) \quad (13)$$

Algorithm 1 Generic Implicit CD

```

1: procedure ICD-GENERIC( $S, C, I$ )
2:    $\Theta \leftarrow \mathcal{N}(0, \sigma)$ 
3:   repeat
4:     Compute  $\Phi$  and  $\Psi$  if necessary
5:     Compute  $J_I$ 
6:     for  $\theta \in \Theta^C$  do
7:       Compute  $L'(\theta|S), L''(\theta|S)$ 
8:       Compute  $R'(\theta), R''(\theta)$ 
9:        $\theta \leftarrow \theta - \eta \frac{L'(\theta|S) + \alpha_0 R'(\theta)}{L''(\theta|S) + \alpha_0 R''(\theta)}$ 
10:      Update  $\Phi$  if necessary
11:    end for
12:    Apply step 5 to 11 to the items.
13:  until converged
14:  return  $\Theta$ 
15: end procedure

```

$$R''(\theta) = 2 \sum_{f=1}^k \sum_{f'=1}^k J_I(f, f') \sum_{c \in C} [\phi_f(c) \phi_{f'}''(c) + \phi_{f'}'(c) \phi_f'(c)] \quad (14)$$

PROOF. The lemma follows from deriving eq. (12). \square

This lemma shows that computing R' and R'' of any context parameter is independent of $|I|$.

From the analysis follows the recipe to derive an efficient iCD learning algorithm for a model \hat{y} . First, rewrite the model as a dot product of ϕ and ψ . Second, construct the first and second derivative of ϕ and ψ with respect to any model parameter $\theta \in \Theta$. These results allow to compute $R'(\theta)$ and $R''(\theta)$ for any model parameter $\theta \in \Theta$ efficiently. With these gradients for the expensive implicit regularizer, a Newton step can be applied. Algorithm 1 shows a generic iCD algorithm using the ideas of this section.

Most models allow some further optimizations: (i) When the gradients of ϕ or ψ are sparse, some of the summands of eqs. (13, 14) drop. (ii) The model parameters usually have some structure which can be used for traversing the model parameters more systematically. We will show both of these steps in the next section for a variety of models.

5. APPLICATIONS

In this section, we apply iCD to two classes of complex factorization models, namely feature-based factorization models and tensor factorization models. We have chosen these two classes because they are very powerful and frequently used. Moreover each of them has some interesting properties with respect to deriving iCD algorithms. The provided algorithms can be directly applied to many common recommender system tasks. This section also serves as a guide for deriving iCD algorithms in general.

5.1 Matrix Factorization (MF)

We start by applying our framework to matrix factorization (see Figure 2). For MF, the scoring function is

$$\hat{y}(c, i) := \langle \mathbf{w}_c, \mathbf{h}_i \rangle = \sum_{f=1}^k w_{c,f} h_{i,f} \quad (15)$$

with model parameters $\Theta = \{W, H\}$ where $W \in \mathbb{R}^{C \times k}$ and $H \in \mathbb{R}^{I \times k}$.

Algorithm 2 Implicit CD for MF

```

1: procedure ICD-MF( $S, C, I$ )
2:    $W, H \leftarrow \mathcal{N}(0, \sigma)$ 
3:   repeat
4:     for  $f^* \in \{1, \dots, k\}$  do
5:       for  $f \in \{1, \dots, k\}$  do
6:         Compute  $J_I(f^*, f)$ 
7:       end for
8:     for  $c^* \in C$  do
9:       Compute  $L'(w_{c^*, f^*}|S), L''(w_{c^*, f^*}|S)$ 
10:      Compute  $R'(w_{c^*, f^*}), R''(w_{c^*, f^*})$ 
11:       $w_{c^*, f^*} \leftarrow w_{c^*, f^*} - \frac{L'(w_{c^*, f^*}|S) + \alpha R'(w_{c^*, f^*})}{L''(w_{c^*, f^*}|S) + \alpha R''(w_{c^*, f^*})}$ 
12:    end for
13:    Apply step 5 to 12 to the items.
14:  end for
15:  until converged
16:  return  $W, H$ 
17: end procedure

```

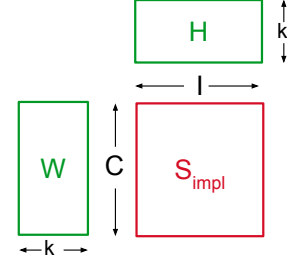


Figure 2: Matrix factorization: Each context c is associated with an embedding \mathbf{w}_c and every item i with an embedding \mathbf{h}_i . The model parameters $W \in \mathbb{R}^{C \times k}, H \in \mathbb{R}^{I \times k}$ are learned to approximate the data S_{impl} with the dot product of $\langle \mathbf{w}_c, \mathbf{h}_i \rangle$.

A MF model is trivially k -separable with

$$\phi_f(c) = w_{c,f}, \quad \psi_f(i) = h_{i,f}. \quad (16)$$

Furthermore, the gradients are sparse

$$\frac{\partial \phi_f(c)}{\partial w_{c^*, f^*}} = \begin{cases} 1, & \text{if } c = c^* \wedge f = f^* \\ 0, & \text{else} \end{cases} \quad (17)$$

and all second derivatives are 0. Thus, the regularizer derivatives simplify to

$$R'(w_{c^*, f^*}) = 2 \sum_{f=1}^k J_I(f, f^*) w_{c^*, f} \quad (18)$$

$$R''(w_{c^*, f^*}) = 2 J_I(f^*, f^*) \quad (19)$$

The derivation is symmetric for the item side.

As MF associates each model parameter with an embedding dimension f , we can traverse the parameters one dimension at a time. A full step $\eta = 1$ can be taken because MF is bilinear. Algorithm 2 shows the full procedure.

The computation of $J_I(f^*, \cdot)$ is trivially in $\mathcal{O}(|I|k)$. Gradient computation of the implicit regularizer is $\mathcal{O}(k)$ per parameter and for the explicit part $\mathcal{O}(|S|)$ for all parameters. Overall, the algorithm has a complexity of $\mathcal{O}((|I| + |C|)k^2 + |S|k)$ per iteration.

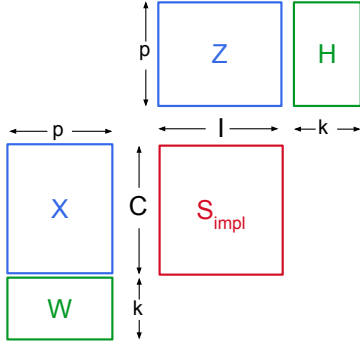


Figure 3: Matrix factorization with side information: In addition to the data S_{impl} , a feature vector $\mathbf{x}_c \in \mathbb{R}^p$ is given for every context $c \in C$ and a feature vector $\mathbf{z}_i \in \mathbb{R}^p$ for each item $i \in I$. Each of the context features $l \in \{1, \dots, p\}$ is assigned a k -dimensional embedding vector $\mathbf{w}_l \in \mathbb{R}^k$ and similarly $\mathbf{h}_l \in \mathbb{R}^k$ for each item feature. The model parameters $W \in \mathbb{R}^{p \times k}$, $H \in \mathbb{R}^{p \times k}$ are learned to approximate the data S_{impl} with a $XW(ZH)^t$.

5.2 Feature-Based Factorization Models

One of the most powerful extension of MF is feature based modeling for the context and item. Feature-based factorization models are strictly more powerful than MF and have shown large improvements in many applications (e.g. [2, 11]). For instance, the cold-start problem is commonly solved by replacing or complementing user and item ids with user and item attributes [2]. Another example is context-aware recommendation, where the context is represented by several variables, e.g. location or time in addition to the user id. Also sequential models can be represented by feature based modeling [6].

Learning general feature-based models on implicit feedback was restricted to BPR so far. This is the first work that provides an implicit CD algorithm for this important model class.

To formalize the problem, assume each $c \in C$ is represented by a feature vector $\mathbf{x}_c \in \mathbb{R}^p$ and each $i \in I$ by a feature vector $\mathbf{z}_i \in \mathbb{R}^p$. See Figure 3 for an illustration.

5.2.1 MF with Side Information (MFSI)

We start with a feature based extension of matrix factorization similar to [2]:

$$\hat{y}(c, i) = \mathbf{x}_c W (\mathbf{z}_i H)^t = \sum_{f=1}^k \left(\sum_{l=1}^p x_{c,l} w_{l,f} \right) \left(\sum_{l=1}^p z_{i,l} h_{l,f} \right) \quad (20)$$

with $\Theta = \{W, H\}$. MFSI is k -separable using

$$\phi_f(c) = \sum_{l=1}^p x_{c,l} w_{l,f}, \quad \psi_f(i) = \sum_{l=1}^p z_{i,l} h_{l,f} \quad (21)$$

and the gradients are sparse

$$\frac{\partial \phi_f(c)}{\partial w_{l^*, f^*}} = \begin{cases} x_{c, l^*}, & \text{if } f = f^* \\ 0, & \text{else} \end{cases} \quad (22)$$

Due to sparse gradients of ϕ and ψ , the first and second

Algorithm 3 Implicit CD for MF with Side Information

```

1: procedure ICD-MFSIDE( $S, C, I$ )
2:    $W, H \leftarrow \mathcal{N}(0, \sigma)$ 
3:   repeat
4:     Compute  $\Phi$  and  $\Psi$ 
5:     for  $f^* \in \{1, \dots, k\}$  do
6:       for  $f \in \{1, \dots, k\}$  do
7:         Compute  $J_I(f^*, f)$ 
8:       end for
9:       for  $l^* \in \{1, \dots, p\}$  do
10:        Compute  $L'(w_{l^*, f^*} | S), L''(w_{l^*, f^*} | S)$ 
11:        Compute  $R'(w_{l^*, f^*}), R''(w_{l^*, f^*})$ 
12:         $w_{l^*, f^*} \leftarrow w_{l^*, f^*} - \frac{L'(w_{l^*, f^*} | S) + \alpha R'(w_{l^*, f^*})}{L''(w_{l^*, f^*} | S) + \alpha R''(w_{l^*, f^*})}$ 
13:        Update  $\Phi$ 
14:      end for
15:      Apply step 6 to 14 to the items.
16:    end for
17:  until converged
18: end procedure

```

regularizer derivatives simplify to:

$$R'(w_{l^*, f^*}) = 2 \sum_{f=1}^k J_I(f, f^*) \sum_{c \in C} x_{c, l^*} \phi_f(c) \quad (23)$$

$$R''(w_{l^*, f^*}) = 2 J_I(f^*, f^*) \sum_{c \in C} x_{c, l^*}^2 \quad (24)$$

Note that the sums over the context variable depend only on context where $x_{c, l^*} \neq 0$, so with a sparse iterator, the computation is $\mathcal{O}(k N_Z(X))$ for optimizing all of the context variables in a given embedding layer f^* .

This computation assumes that Φ and Ψ are given. Obviously, while optimizing W , Ψ does not change and while optimizing H , Φ does not change. However, while optimizing W , Φ changes but can be kept in sync with changes in W by updating:

$$\phi_{f^*}(c) \leftarrow \phi_{f^*}(c) + x_{c, l^*} (w_{l^*, f^*}^{\text{new}} - w_{l^*, f^*}^{\text{old}}) \quad (25)$$

The item side can be derived analogously. The total runtime of Algorithm 3 for one epoch over all variables is $\mathcal{O}(k^2 (N_Z(X) + N_Z(Z)))$ for the implicit regularizer.

5.2.2 Factorization Machines

The Factorization Machine (FM) model [11] is a more complex factorized model that includes biases and interactions between all variables. In general, the FM for a feature vector $\mathbf{x} \in \mathbb{R}^p$ is defined as

$$\hat{y}(\mathbf{x}) = b + \sum_{l=1}^p x_l \tilde{w}_l + \sum_{l=1}^p \sum_{l' > l} \langle \mathbf{w}_l, \mathbf{w}_{l'} \rangle x_l x_{l'} \quad (26)$$

where b is a global bias parameter, $\tilde{\mathbf{w}}$ are feature biases and W are the embeddings. In our case, for a context-item pair (c, i) , we set the input feature vector \mathbf{x} as the concatenation of the context and item feature vectors: $\mathbf{x} := (\mathbf{x}_c, \mathbf{z}_i)$.

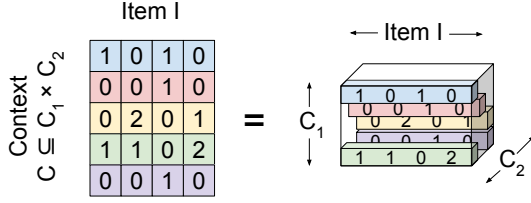


Figure 4: Example for tensor factorization data: two categorical variables on the context side, $C \subseteq C_1 \times C_2$. This data can be interpreted as a 3-mode tensor with missing values.

The FM model is $(k+2)$ -separable, with

$$\phi_f(c) = \sum_{l=1}^p x_{c,l} w_{l,f}, \quad \psi_f(i) = \sum_{l=1}^p z_{i,l} h_{l,f}, \quad (27)$$

$$\phi_{k+1}(c) = b + \sum_{l=1}^p x_{c,l} \tilde{w}_l + \sum_{l=1}^p \sum_{l' > l} \langle \mathbf{w}_l, \mathbf{w}_{l'} \rangle x_{c,l} x_{c,l'}, \quad (28)$$

$$\psi_{k+1}(i) = 1, \quad (29)$$

$$\phi_{k+2}(c) = 1, \quad (30)$$

$$\psi_{k+2}(i) = \sum_{l=1}^p z_{i,l} \tilde{h}_l + \sum_{l=1}^p \sum_{l' > l} \langle \mathbf{h}_l, \mathbf{h}_{l'} \rangle z_{i,l} z_{i,l'}. \quad (31)$$

where for the context, ϕ is parameterized by $\tilde{\mathbf{w}} \in \mathbb{R}^p$ for the linear part and $W \in \mathbb{R}^{p \times k}$ for the factors. And analogously for items, ψ is parameterized by $\tilde{\mathbf{h}} \in \mathbb{R}^p$ for the linear part and $H \in \mathbb{R}^{p \times k}$ for the factors.

The gradients are sparse:

$$\frac{\partial \phi_f(c)}{\partial \tilde{w}_{l^*}} = \begin{cases} x_{c,l^*}, & \text{if } f = k+1 \\ 0, & \text{else} \end{cases} \quad (32)$$

$$\frac{\partial \phi_f(c)}{\partial w_{l^*,f^*}} = \begin{cases} x_{c,l^*}, & \text{if } f = f^* \\ x_{c,l^*} (\phi_{f^*}(c) - x_{c,l^*} w_{l^*,f^*}), & \text{if } f = k+1 \\ 0, & \text{else} \end{cases} \quad (33)$$

Similar to MFSI, due to the sparsity in gradients, one of the nested loops drops for the first regularizer derivative R' and both nested sums drop for the second regularizer derivative R'' . Consequently, the flow and runtime analysis for FM is the same as for MFSI.

5.3 Tensor Factorization

Tensor factorization generalizes matrix factorization and deals with problems that involve more than two categorical variables. For instance, in personalized recommendation of tags for bookmarks [20], the *context* consists of two variables, the user C_1 and the bookmark C_2 , and the item I corresponds to the tag. For personalized web search [19], the context consists of the user C_1 and the query C_2 and the item I to the web page. The data can be seen as a three mode tensor over C_1 , C_2 and I . Figure 4 shows an example of how observations over context $C \subseteq C_1 \times C_2$ and items I translate to a tensor. A tensor factorization model tries to approximate the tensor with a low rank decomposition (see Figure 5). Although tensor factorization models are multilinear, we show that they fit well into our framework.

Additionally, we want to highlight, that existing tensor factorization learning algorithms [19, 20, 10] require that the

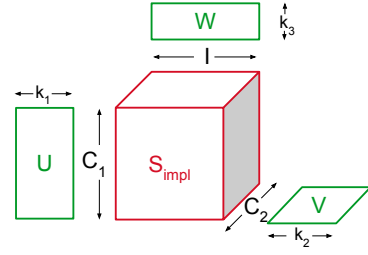


Figure 5: A tensor factorization model, decomposes a given tensor into one matrix per mode, here $U \in \mathbb{R}^{C_1 \times k_1}$ for C_1 , $V \in \mathbb{R}^{C_2 \times k_2}$ for C_2 and $W \in \mathbb{R}^{C_3 \times k_3}$ for I .

tensor data is dense, i.e., the empty parts in the tensor in Figure 4 are filled with zeros. This would imply that context combinations that never have been observed, are used for training as well, i.e., $C = C_1 \times C_2$. In some applications, this might not make sense, for instance if C_1 encodes a device type and C_2 encodes an operating system version. Our iCD framework works for both sparse and dense context. We will point out the differences when necessary.

5.3.1 Parallel Factor Analysis (PARAFAC)

We first discuss the Parallel Factor Analysis (PARAFAC) [3] model which is a 3-mode extension of matrix factorization.

$$\hat{y}(c_1, c_2, i) := \sum_{f=1}^k u_{c_1,f} v_{c_2,f} w_{i,f} \quad (34)$$

with $\Theta = \{U, V, W\}$ where $U \in \mathbb{R}^{C_1 \times k}$, $V \in \mathbb{R}^{C_2 \times k}$ and $W \in \mathbb{R}^{I \times k}$. PARAFAC is k -separable with

$$\phi_f(c_1, c_2) = u_{c_1,f} v_{c_2,f}, \quad \psi_f(i) = w_{i,f} \quad (35)$$

Again, gradients are sparse:

$$\frac{\partial \phi_f(c_1, c_2)}{\partial u_{c_1^*,f^*}} = \begin{cases} v_{c_2,f}, & \text{if } c_1 = c_1^* \wedge f = f^* \\ 0, & \text{else} \end{cases} \quad (36)$$

and the loss derivatives simplify to

$$R'(u_{c_1^*,f^*}) = 2 \sum_{f=1}^k J_I(f, f^*) u_{c_1^*,f} \sum_{c_2: (c_1^*, c_2) \in C} v_{c_2,f} v_{c_2,f^*} \quad (37)$$

$$R''(u_{c_1^*,f^*}) = 2 J_I(f^*, f^*) \sum_{c_2: (c_1^*, c_2) \in C} v_{c_2,f^*} v_{c_2,f^*} \quad (38)$$

The item side is equivalent to matrix factorization.

If the context is dense and includes all possible combinations of context variables, i.e., if $C = C_1 \times C_2$, then the computation of $J_C(f, f')$, can be decomposed to:

$$J_C(f, f') = \underbrace{\sum_{c_1 \in C} u_{c_1,f} u_{c_1,f'}}_{=: J_{C_1}(f, f')} \underbrace{\sum_{c_2 \in C} v_{c_2,f} v_{c_2,f'}}_{=: J_{C_2}(f, f')} \quad (39)$$

This means, the computation is in $\mathcal{O}(|C_1| + |C_2|)$ instead of $\mathcal{O}(|C_1| |C_2|)$. On the other hand if C is sparse and contains only the subset of the observed context combinations, i.e., $C \subset C_1 \times C_2$, then there is no need for decomposing this sum. The same applies to the loss derivatives of eqs. (37,38): Again, if all possible context is modeled, then $\{c_2 : (c_1^*, c_2) \in C\} = C_2$ and thus $J_{C_2}(f, f')$ can replace the sum over C_2 .

The overall runtime for PARAFAC’s implicit regularizer is $\mathcal{O}((|C| + |I|)k^2)$ for sparse context and $\mathcal{O}((|C_1| + |C_2| + |I|)k^2)$ for dense context. The traversal over model parameters can be arranged as in the MF algorithm.

5.3.2 Tucker Decomposition

Tucker Decomposition (TD) [21] is a generalization of PARAFAC which computes all interactions between the factor matrices. The strength of each interaction is given by a core tensor B . For our running example with two context variables c_1, c_2 and one item variable i , TD is defined as

$$\hat{y}(c_1, c_2, i) = \sum_{f_1=1}^{k_1} \sum_{f_2=1}^{k_2} \sum_{f_3=1}^{k_3} b_{f_1, f_2, f_3} u_{c_1, f_1} v_{c_2, f_2} w_{i, f_3} \quad (40)$$

with $\Theta = \{B, U, V, W\}$ where $B \in \mathbb{R}^{k_1 \times k_2 \times k_3}$ is the core tensor and $U \in \mathbb{R}^{|C_1| \times k_1}$, $V \in \mathbb{R}^{|C_2| \times k_2}$ and $W \in \mathbb{R}^{|I| \times k_3}$. TD is much more computationally expensive than PARAFAC, requiring $\mathcal{O}(k_1 k_2 k_3)$ operations just for evaluating the model on one data point.

Even though Tucker decomposition contains nested sums, it is k_3 -separable with

$$\phi_f(c_1, c_2) = \sum_{f_1=1}^{k_1} \sum_{f_2=1}^{k_2} b_{f_1, f_2, f} u_{c_1, f_1} v_{c_2, f_2}, \quad \psi_f(i) = w_{i, f}$$

The derivatives of these functions are:

$$\frac{\partial \phi_f(c_1, c_2)}{\partial u_{c_1, f_1^*}} = \begin{cases} \sum_{f_2=1}^{k_2} b_{f_1^*, f_2, f} v_{c_2, f_2}, & \text{if } c_1 = c_1^* \\ 0, & \text{else} \end{cases} \quad (41)$$

$$\frac{\partial \phi_f(c_1, c_2)}{\partial v_{c_2, f_2^*}} = \begin{cases} \sum_{f_1=1}^{k_1} b_{f_1, f_2^*, f} u_{c_1, f_1}, & \text{if } c_2 = c_2^* \\ 0, & \text{else} \end{cases} \quad (42)$$

$$\frac{\partial \phi_f(c_1, c_2)}{\partial b_{f_1^*, f_2^*, f_3^*}} = \begin{cases} u_{c_1, f_1^*} v_{c_2, f_2^*}, & \text{if } f = f_3^* \\ 0, & \text{else} \end{cases} \quad (43)$$

$$\frac{\partial \psi_f(i)}{\partial w_{i^*, f_3^*}} = \begin{cases} 1, & \text{if } f = f_3^* \wedge i = i^* \\ 0, & \text{else} \end{cases} \quad (44)$$

Unlike all the other models we have presented so far, the gradients for ϕ are non-zero for any factor index $f \in \{1, \dots, k\}$. Consequently, the nested loops over factors of the loss gradient (eq. 13) cannot be improved further. However, for ψ , which is sparse, the same optimization as in the other models can be applied.

Like for PARAFAC, if C is dense, i.e., $C = C_1 \times C_2$, we can precompute intermediate matrices for C_1 and C_2 and the computation of $J_C(f, f')$ simplifies to

$$\sum_{f_1=1}^{k_1} \sum_{f_1'=1}^{k_1} \sum_{f_2=1}^{k_2} \sum_{f_2'=1}^{k_2} b_{f_1, f_2, f} b_{f_1', f_2', f'} J_{C_1}(f_1, f_1') J_{C_2}(f_2, f_2')$$

If C is sparse, there is no need for this optimization and we can use a straightforward computation of J_C . The overall runtime complexities are $\mathcal{O}(k_1^2 k_2^2 k_3^2 (|C_1| + |C_2| + |I|))$ for dense context and $\mathcal{O}(k_1^2 k_2^2 k_3^2 (|C| + |I|))$ for sparse context.

6. EXPERIMENTS

The main objective of the experiments is to illustrate the generality of the iCD framework. We show how iCD can be applied to a variety of recommender problems that cannot be solved with MF alone. For MF models, efficient coordinate

descent algorithms (CD) have been previously proposed [5] and its performance compared against gradient descent algorithms such as BPR [13]. Both approaches are considered state-of-the-art and while CD outperforms BPR on certain datasets [8, 25, 15, 22, 26], BPR has been shown to work better on others [4, 17, 16, 8]. The purpose of our experiments is not to compare BPR and CD on yet another dataset, but rather to demonstrate the versatility of the iCD framework and illustrate how it can serve as a building block for future research on complex recommender models. As with MF, it is likely that both iCD and BPR will show strengths in different applications.

6.1 Experimental Setup

We evaluate on a dataset of 200,000 users interacting with YouTube. Our subset contains $|I| = 68,000$ videos. The dataset also contains side information about age, country, gender and device info. We apply iCD to three popular recommendation problems – Cold-Start, Offline Recommendation, and Instant Recommendation (see Section 6.2). We compare the following algorithms:

- **Popularity**: a static recommender that returns the most popular videos.
- **Coview**: returns based on the previously watched video, the most commonly chosen next video.
- **iCD-MF**: user-item matrix factorization using iCD for optimization, similar to [5].
- **iCD-FM**: a factorization machine with varying features for the context (Section 5.2). We report results for different feature choices.

We measure the recall and NDCG for the top 100 returned videos. Note that we report relative improvements over the **Popularity** recommender. All hyperparameters are tuned on a separate tuning holdout set.

6.2 Results

6.2.1 Cold-Start Recommendation

In the Cold-Start recommendation [2] scenario, we assume that a user interacts with the recommender system for the first time. To simulate this scenario, we select a random subset of users and hold out all their events for evaluation purposes; we train on the remaining users.

The common approach for dealing with cold-start is to represent a user by side information [2]. Here, we use the feature-based FM model (iCD-FM) with the user’s age, gender, country and device info as context features. Figure 7 shows that attribute-aware FM achieves a 2x improvement over the baselines. As expected, neither MF nor Coview can do any better than most-popular recommendation.

6.2.2 Offline Recommendation

In the *Offline Recommendation* scenario, we hold out the last feedback of each user and use all the previous feedback for training. This is the most commonly used protocol to evaluate the performance of a recommender algorithm. We experiment with multiple FM models: (1) iCD-FM A: an FM with user attributes, (2) iCD-FM P: a sequential FM that only uses the previously watched video (similar to FPMC [14] or Coview) and (3) iCD-FM A+P+U: an FM

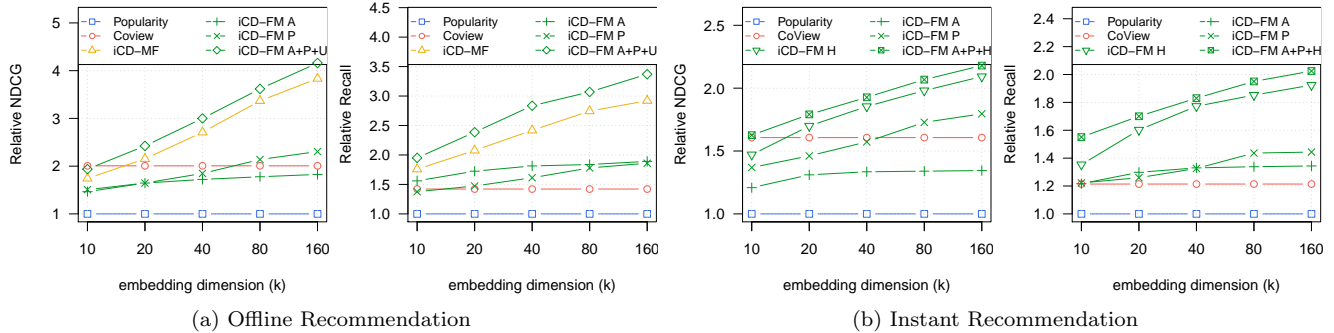


Figure 6: Different variants of context features are used in the iCD-FM models: A = gender, age, country and device, P = the previously watched video, H = all videos watched so far, U = user id.

that uses all signals: attributes, previously watched video and user id (similar to FPMC [14] with user attributes). As shown in Figure 6a, the complex FM model with all features achieves the best quality, illustrating the flexibility of feature engineering with iCD.

6.2.3 Instant Recommendation

In large-scale industrial applications, online training is often not feasible due to complex serving stacks. Commonly, models are periodically trained offline (e.g., every day or week) and applied on a stream of user interactions. When the model is queried to generate recommendations for a user, all feedback until the current time is taken into account for prediction. We simulate this setting by choosing a global cutoff time where all the events before the cutoff are used for training and all the remaining ones for evaluation.

In such settings, models relying on user ids, such as MF, cannot capture recent feedback. Instead, describing a user by the sequence of previously watched videos allows for instant personalization. Such a model can be configured using a feature-based FM model (Section 5.2) and we experiment with four configurations (1) iCD-FM A : FM using user attributes, (2) iCD-FM P : a sequential FM based on the previously watched video, (3) iCD-FM H : a FM based on all previously watched videos, (4) iCD-FM $A+P+H$: an FM combining all signals. As expected, the complex FM model with all features achieves the best quality. Again, we would like to note the generality of the iCD framework, which enables flexible feature engineering.

6.3 Computational Costs

As stated in Section 3.3, any conventional CD solver, e.g. [11], could solve the implicit feedback problem. Now, we substantiate that this is infeasible because of the large number of implicit examples. Figure 8 compares the computational cost for learning an FM with a conventional CD to the costs of iCD on our dataset with 70k items. We use three different context features from Figure 6. The plot shows relative costs to iCD-FM P . For all three context choices, conventional CD shows four orders of magnitude higher computational costs than iCD. The empirical measured runtime for iCD was in the order of minutes; consequently, CD's four order of magnitude increase in runtime translates to weeks of training for each iteration. Clearly, using a conventional CD solver to optimize the implicit loss directly is infeasible.

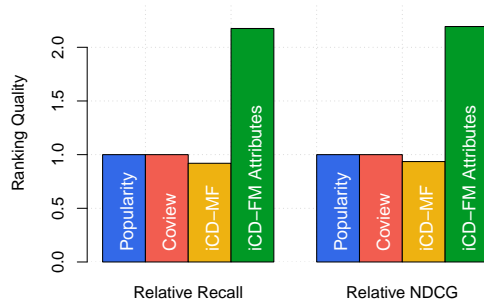


Figure 7: Cold Start Recommendation

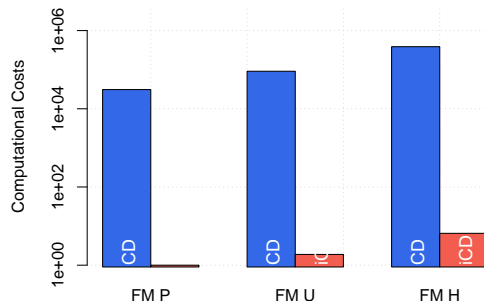


Figure 8: Training costs in log-scale of conventional CD (left, blue) versus iCD (right, red) on our implicit dataset.

7. CONCLUSION

In this work, we have presented a general, efficient framework for learning recommender system models from implicit feedback. First, we have shown that learning from implicit feedback can be reformulated as optimizing a cheap explicit loss and an expensive implicit regularizer. Then we have introduced the concept of k -separable models. We have shown that the implicit regularizer of any k -separable model can be computed efficiently without iterating over all context-item pairs. Finally, we have shown that many popular recommender models are k -separable, including matrix factorization, factorization machines and tensor factorization. Moreover, we have provided efficient learning algorithms for these models based on our framework. Our framework is not limited to the models discussed in the paper but designed to serve as a general blueprint for deriving learning algorithms for recommender systems.

8. REFERENCES

- [1] C. Cheng, H. Yang, M. R. Lyu, and I. King. Where You Like to Go Next: Successive Point-of-Interest Recommendation. In *IJCAI*, volume 13, pages 2605–2611, 2013.
- [2] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *2010 IEEE International Conference on Data Mining*, pages 176–185. IEEE, 2010.
- [3] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an “explanatory” multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16(1):84, 1970.
- [4] R. He and J. McAuley. VBPR: Visual bayesian personalized ranking from implicit feedback. In D. Schuurmans and M. P. Wellman, editors, *AAAI*, pages 144–150. AAAI Press, 2016.
- [5] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM ’08, pages 263–272, 2008.
- [6] B. Kanagal, A. Ahmed, S. Pandey, V. Josifovski, J. Yuan, and L. Garcia-Pueyo. Supercharging recommender systems using taxonomies for learning user purchase behavior. *Proc. VLDB Endow.*, 5(10):956–967, June 2012.
- [7] B. McFee, T. Bertin-Mahieux, D. P. Ellis, and G. R. Lanckriet. The million song dataset challenge. In *Proceedings of the 21st International Conference on World Wide Web*, WWW ’12 Companion, pages 909–916, New York, NY, USA, 2012. ACM.
- [8] X. Ning and G. Karypis. Slim: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th International Conference on Data Mining*, pages 497–506. IEEE, 2011.
- [9] W. Pan and L. Chen. GBPR: Group Preference Based Bayesian Personalized Ranking for One-Class Collaborative Filtering. In *IJCAI*, volume 13, pages 2691–2697, 2013.
- [10] I. Pilászy, D. Zibriczky, and D. Tikk. Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 71–78. ACM, 2010.
- [11] S. Rendle. Factorization machines with libfm. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, may 2012.
- [12] S. Rendle and C. Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM ’14, pages 273–282, New York, NY, USA, 2014. ACM.
- [13] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI ’09, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press.
- [14] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW ’10, pages 811–820. ACM, 2010.
- [15] S. Sedhain, A. K. Menon, S. Sanner, and D. Braziunas. On the effectiveness of linear models for one-class collaborative filtering. In *Proceedings of the 30th Conference on Artificial Intelligence (AAAI-16)*, 2016.
- [16] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, A. Hanjalic, and N. Oliver. TFMAP: optimizing MAP for top-n context-aware recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 155–164. ACM, 2012.
- [17] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic. CLIMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 139–146. ACM, 2012.
- [18] E. Shmueli, A. Kagian, Y. Koren, and R. Lempel. Care to comment?: recommendations for commenting on news stories. In *Proceedings of the 21st international conference on World Wide Web*, pages 429–438. ACM, 2012.
- [19] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: A novel approach to personalized web search. In *Proceedings of the 14th International Conference on World Wide Web*, WWW ’05, pages 382–390, New York, NY, USA, 2005. ACM.
- [20] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. *IEEE Trans. on Knowl. and Data Eng.*, 22(2):179–192, Feb. 2010.
- [21] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31:279–311, 1966.
- [22] M. Volkovs and G. W. Yu. Effective latent models for binary feedback in recommender systems. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 313–322. ACM, 2015.
- [23] H.-F. Yu, C.-J. Hsieh, S. Si, and I. Dhillon. Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In *Proceedings of the 12th International Conference on Data Mining*, ICDM ’12, pages 765–774, 2012.
- [24] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norrick, and J. Han. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM ’14, pages 283–292. ACM, 2014.
- [25] T. Zhao, J. McAuley, and I. King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM ’14, pages 261–270, New York, NY, USA, 2014. ACM.
- [26] T. Zhao, J. McAuley, and I. King. Improving latent factor models via personalized feature projection for one class recommendation. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 821–830. ACM, 2015.