WeightedSLIM: A Novel Item-Weights Enriched Baseline Recommendation Model

HAIYANG ZHANG[1], XINYI ZENG[2], IVAN GANCHEV[3,4,5]

[1]Department of Computing,
Xi'an Jiaotong, Liverpool University,
Suzhou,
CHINA

[2]College of Artificial Intelligence,
North China University of Science and Technology,
Tangshan,
CHINA

[3]University of Plovdiv "Paisii Hilendarski",
Plovdiv,
BULGARIA

[4]Institute of Mathematics and Informatics, Bulgarian Academy of Sciences (IMI–BAS),
Sofia,
BULGARIA

[5]Telecommunications Research Center (TRC),
University of Limerick,
Limerick,
IRELAND

Corresponding Author

Abstract: - This paper proposes a novel weight-enriched ranking-based baseline model, *WeightedSLIM*, aiming to provide more accurate top-N recommendations from implicit user feedback. The model utilizes *ItemRank* to calculate the ranking score of each item, which is then used as an item weight within the Sparse Linear Model (*SLIM*), while using Bayesian Personalized Ranking (BPR) to optimize the item similarity matrix. Experiments, conducted for performance comparison of the proposed model with existing recommendation models, demonstrate that it can indeed provide better recommendations and can be used as a strong baseline for top-N recommendations.

Key-Words: - ranking-based recommendation, implicit feedback, Bayesian Personalized Ranking (BPR), ItemRank, SLIM, WeightedSLIM.

Received: March 11, 2023. Revised: October 15, 2023. Accepted: December 19, 2023. Published: February 14, 2024.

1 Introduction

In the modern 'Big Data' era, recommendation systems have become essential tools to address a myriad of services on offer to users, e.g., in the context of the vast range of information sources and services readily available, ranging from simple web browsing to sophisticated Internet of Things (IoT) services, by assisting users to discover what they need or receive timely suggestions and recommendations.

Generally, the recommendation problem could be cast as either a rating estimation problem which aims to predict as accurately as possible the rating values for items (or services) unrated by a particular user yet, or as a ranking problem which aims to find top-N ranked items for a particular user that would be of most interest to her/him, and which s/he has not yet ranked. In contexts where explicit item ratings of other users may not be available, ranking prediction can be more important than rating prediction. Most of the existing ranking-based prediction approaches consider items as having equal weights, which is not always the case. Different weights of items could be regarded as a reflection of items' importance, or desirability, to users.

The Sparse Linear Model (*SLIM*) is a state-of-the-art ranking-based collaborative filtering (CF) baseline model, which can make top-N recommendations with high quality and efficiency, [1]. However, some limitations of *SLIM* are that (i) it assumes that all items have the same weight when computing the predictive rank, which is not always the case, and (ii) the similarity matrix is optimized by minimizing the squared error, which is not adaptable for recommendations from implicit feedback. In recommendation scenarios, it is common knowledge that the importance of items varies from one to another due to different popularity or other features of items. Thus, in *SLIM*, the predicted score on an unrated item *i* of user *u* is calculated by the aggregation of items that have been rated by this user. Among these rated items, however, some items should be weighted higher than others.

Based on this observation, this paper proposes a new baseline for item ranking that incorporates item weights, learned from the item correlation graph using *PageRank,* [2], into *SLIM*, and then optimizes the model using Bayesian Personalized Ranking (BPR), [3]. Extensive experiments, conducted on three benchmark datasets, confirm that the proposed model outperforms the existing ranking-based models (which do not use item weights) on several different evaluation metrics.

The reminder of this paper is organized as follows. Section 2 briefly presents the related work in this area. Section 3 explains the background. Section 4 describes the proposed model. Section 5 presents the conducted experiments and analyses the obtained results. Finally, Section 6 concludes the paper.

2 Related Work

2.1 Recommender Systems

A common task of recommender systems is to improve user experience via personalized recommendations based on different sorts of user behavior, [4]. The objective of the recommendation can be treated either as a rating estimation problem which aims to predict the ratings for unrated user–item pairs as accurately as possible, or as a ranking problem which aims to suggest several specific unrated items to a user that are likely to be appealing to her/him, [5]. Essentially, rating prediction and item ranking are two distinct recommendation tasks, and most recommendation approaches are solely designed for either of them, [6]. In the literature, the majority of recommendation techniques, e.g., [6] [7] [8], are focused on the rating prediction task with recommendation performance evaluated by computing the average error between the predicted ratings and the actual ratings, [5]. However, many current works have switched from developing more accurate rating prediction models to ranking-oriented models, since they seem better fitted for the top-N recommendation task, which aims to provide a size-N ranked list of items for users to encourage additional use of items in most recommendation scenarios, [1]. Ranking-oriented models also perform better in recommendation scenarios based only on implicit user feedback, [5], [9]. Another difference is that the evaluation of a top-N ranking model is typically done by calculating ranking-based metrics (i.e., precision and recall) while rating estimation models are evaluated through error-based metrics such as the mean absolute error (MAE) and root mean square error (RMSE).

No matter whether rating-based or ranking-based, the recommendation models rely on past user feedback, which is either *explicit* (ratings, reviews, etc.) or *implicit* (clicks, browsing history, etc.). In many recommendation scenarios, explicit rating data is sparse or even nonexistent since users are usually reluctant to spend extra time or effort on supplying that information. In such cases, the preferences of users can be approximated by their implicit feedback. Compared to explicit feedback, implicit feedback is closer to the real-industry perception of the problem and potential recommendation solutions, where the feedback data may be collected automatically at a much larger and faster scale with no user efforts needed, [4], [10]. Together with implicit feedback scenarios, many recent works also address the top-N recommendation problem, [3], [11]. This paper aims to address both issues, dealing with top-N item recommendations based on implicit user feedback.

Data-sparsity problem is one of the main limitations in CF-based recommender systems, [12] [13], [14], which ultimately affects the recommendation performance. In the literature, it has been proven that this problem can be partially solved by incorporating additional data sources besides the user–item ratings, [15], [16], [17]. In real-world recommendation situations, however, additional data sources may not always be available but there is still intrinsic information within the user–item rating matrix to utilize for improving the recommendation performance. Concerning this problem, the current paper proposes to exploit item weights from the item correlation graph constituted by the user–item interactions and utilize these weights within the *SLIM* model, [1], with optimization driven by BPR, [3], for incorporating implicit feedback.

2.2 PageRank and ItemRank

PageRank is a very successful technique for ranking nodes in a graph, e.g., for hyperlink analysis in web graphs or user interaction analysis in social networks, [18]. It computes the importance score for each node in the graph according to the graph connectivity. *PageRank* has also been used as a weight-learning method in recommender systems, based on different degrees of correlation between users or items, [19], [20]. Given a graph $G = (V, E)$, where V is the set of nodes and E is the set of links, the ranking score of each node is calculated by *PageRank*, as shown in [21], as follows:

$$PR = \beta \cdot C \cdot PR + (1-\beta) \cdot \frac{1}{|V|} \cdot 1_{|V|} , \quad (1)$$

where C denotes the normalized connectivity matrix for graph G, β denotes the decay factor, PR denotes the vector representing the ranking score of each node, and $1_{|V|}$ is a $|V|$ long vector of ones.

The current paper utilizes *ItemRank,* [21], to calculate the ranking score of items. *ItemRank* is a scoring algorithm based on *PageRank*, which can rank the items according to users' preferences in recommender systems. One advantage of *ItemRank* is that it works with both explicit and implicit feedback since it is based on an item co-occurrence matrix. According to [21], to obtain the items' ranking scores, a directed graph is constructed where nodes represent items and links represent the normalized correlation between items. Suppose $U(i)$ is the set of users who rated item i and $U(i,j)$ is the set of users who rated both item i and item j.

Definition 1. Given an implicit feedback matrix UI, containing m users and n items, each entry in the item co-occurrence matrix CR, computed as $CR = UI^T \times UI$, represents the number of users who showed interest in each pair of items.

CR is a symmetric matrix, i.e., $CR(i,j) = CR(j,i)$. However, in situations where item i received far more ratings than item j, the correlation of item i to item j should also depend on the relation of item i to all other items and thus may differ from the correlation of item j to item i. Therefore, a normalized matrix CM is defined to solve this problem, [19].

Definition 2. CM is an item correlation matrix that records relationships between item pairs, which is a normalized CR, computed as follows:

$$CM_{i,j} = \frac{CR_{i,j}}{\sum_{j \in I} CR_{i,j}} , \quad (2)$$

where I denotes the set of all items in UI. For item i, the sum of its correlation values with the other items is equal to 1, i.e., $\sum_{j \in I} CM_{i,j} = 1$.

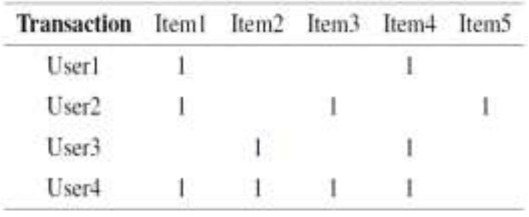
Hence, the ranking score of each item according to *ItemRank*, is calculated as:

$$PR = \beta \cdot CM \cdot PR + (1-\beta) \cdot \frac{1}{|V|} \cdot 1_{|V|} , \quad (3)$$

where PR is an n-dimensional vector representing the ranking score of each item, and $\sum_{i=1}^{n} PR_i = 1$. Thus, each element in PR denotes the importance of an item in relation to all other items. In other words, each element can be utilized as the weight of each item. The rules followed by *ItemRank* for item weighting in a recommendation scenario are that: (i) if item i has incoming links from highly ranked items, then item i will also have a high rank; (ii) an item will 'spread' its rank throughout the correlation graph which is generated from the user–item interactions, with the effects decreasing as it spreads further away, [21].

An *ItemRank*-related example is shown in Table 1, Table 2 and Table 3. Specifically, Table 1 contains a binary matrix of the implicit feedback from four users to five items, Table 2 contains the corresponding item co-occurrence matrix, and Table 3 contains the corresponding correlation matrix, where a row represents the outgoing links for the corresponding item, and a column represents the incoming links.

Table 1. A sample table containing an implicit feedback matrix.



Transaction	Item1	Item2	Item3	Item4	Item5
User1	1			1	
User2	1		1		1
User3		1		1	
User4	1	1	1	1	

Table 2. A sample table containing the item co-occurrence matrix corresponding to Table 1

	Item1	Item2	Item3	Item4	Item5
Item1	0	1	2	2	1
Item2	1	0	1	2	0
Item3	2	1	0	1	1
Item4	2	2	1	0	0
Item5	1	0	1	0	0

Table 3. A sample table containing the item correlation matrix corresponding to Table 2

	Item1	Item2	Item3	Item4	Item5
Item1	0	1/6	2/6	2/6	1/6
Item2	1/4	0	1/4	2/4	0
Item3	2/5	1/5	0	1/5	1/5
Item4	2/5	2/5	1/5	0	0
Item5	1/2	0	1/2	0	0

Nearest-neighbor models are widely deployed for CF rating predictions. They rely on the correlation between items (a.k.a. *item*-based models) or users (a.k.a. *user*-based models). They have also been used to solve recommendation problems with implicit feedback data for item ranking. For item recommendation, the model of item-based k-nearest neighbor (KNN) with implicit feedback is given in [3], as:

$$\tilde{r}_{ui} = \sum_{l \in I_u^+ \wedge l \neq i} S_{i,l} , \tag{4}$$

where I_u^+ is the set of items that user u has seen in the past and S is a $n \times n$ symmetric item correlation (item similarity) matrix, which is learned by parameter optimization methods.

Similarity matrix S is not necessarily symmetric, as illustrated by Table 3. So, [1] proposes a similar recommendation model, but without the constraint for the item similarity matrix to be symmetric. Their recommendation model is defined in [1], as:

$$\tilde{r}_{ui} = \sum_{l \in I_u^+ \wedge l \neq i} AS_{i,l} , \tag{5}$$

where AS is an asymmetric $n \times n$ item similarity matrix.

One disadvantage of (5) is that it assumes that similarities between item i and all user-rated items contribute equally to the predictive rating, ignoring the fact that the similarity of item i to an important item may have a higher contribution. For example, suppose that item $l1$ and $l2$ are two items that user u has seen in the past, but item $l1$ is more popular (important) than item $l2$. In both models described above, similarities of item i to items $l1$ and $l2$ possess equal weights when calculating the rating of item i by user u. However, item $l1$ is popular for a reason, which should be considered when estimating the predicted rating. For this, a novel weighted sparse linear model, named *WeightedSLIM*, is proposed in the next section, aiming to provide more accurate top- N recommendations from implicit feedback. The model utilizes *ItemRank*, [21] to calculate the ranking score of each item, which is then used as an item weight within *SLIM* [1], while using BPR, [3], to optimize the item similarity matrix.

3 Background

This paper aims to provide top-N recommendations with the consideration of item weights for implicit feedback recommendation scenarios. More specifically, it proposes to integrate item weights, which are learned from a binary matrix $T \in \mathbb{R}_{m,n}$, containing the transaction records of n items from m users, within the *SLIM* framework to generate more accurate recommendations.

The notations used in the remainder of the paper are summarized in Table 4.

Table 4. Notations

Notation	Description
β	decay factor in PageRank
PR	item weights vector
C	normalized connectivity matrix
$1_{\|V\|}$	vector of 1s
t	index of iterations
r_{ui}	true rating of item i by user u
$\tilde{r}_{ui}$	predicted rating of item i by user u
m	total number of users
n	total number of items
S	symmetric item similarity matrix
AS	asymmetric item similarity matrix
(u, i, j)	triples where i is an item rated by u, and j is unrated
I_u^+	set of items that user u interacted with
γ	learning rate
λ	regularization parameter

The objective of the recommendation task is to recommend unrated items with the highest prediction score to each user. A large number of previous studies focused on predicting rating values for each user as accurately as possible. However, the ranking of the unrated/unobserved items is more important, [22]. For recommendation scenarios where only binary user feedback is available (which is the most frequent case), BPR could be utilized to optimize the parameters of recommendation models, [3]. The assumption behind BPR is that the user prefers a consumed/seen item to an unconsumed/unseen item, aiming to maximize the following posterior probability:

$$p(\Theta|R) \propto p(R|\Theta)p(\Theta)\ , \qquad (6)$$

where R denotes the rating matrix, Θ denotes the parameter vector of a predictive model, and $p(R|\Theta)$ denotes the likelihood of the desired preference structure for all users according to R.

Typically, BPR is based on pairwise comparisons between a small set of positive items and a very large set of negative/unrated items from the users' histories. BPR estimates parameters by minimizing the loss function defined in, [3], as follows:

$$O = -\sum_{u \in U} \sum_{i \in R_u^+, j \in R_u^-} ln\, \sigma\left(\tilde{r}_{ui} - \tilde{r}_{uj}\right) + \lambda \|\Theta\|^2 \quad (7)$$

where $\sigma = 1/(1 + e^{-x})$ denotes the sigmoid function of x, U denotes the set of available users, r_{ui} denote the predicted scores of user u for items i and j, respectively, $j \in R_u^-$ denotes the set of unrated items, and λ is the model-specific regularization term.

4 Proposed Model: *WeightedSLIM*

Most of the current ranking-based CF models treat items as having equal weights. However, it is common sense that some items are more important than others, which can be represented by assigning bigger weights to them. This idea is incorporated into the proposed ranking-based CF model, *WeightedSLIM*, which takes into account the item weights as follows:

$$\tilde{r}_{ui} = \frac{\sum_{l \in I_u^+ \wedge l \neq i} AS_{i,l}\, PR(l)}{\sum_{l \in I_u^+ \wedge l \neq i} PR(l)}\ , \qquad (8)$$

where *PR*(*l*) denotes the weight of item *l*.

Example, illustrating the advantages of the proposed model, is the following one. Suppose that, in the past, user u has rated two items, $l1$ and $l2$. For determining the rank of two unrated items i and j, their predictive rating should be computed first. Suppose that $AS(i, l1) = 0.9$, $AS(i, l2) = 0.3$; and $AS(j, l1) = 0.1$, $AS(j, l2) = 0.9$. According to (5), $\tilde{r}_{ui} = 1.200$ and $\tilde{r}_{uj} = 1.000$. Thus, item i should be ranked higher than item j.

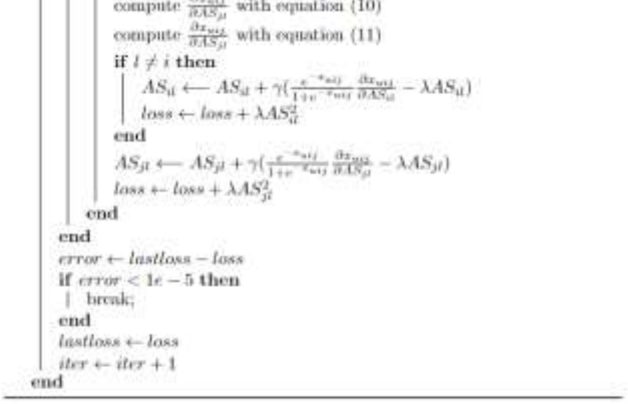
Assume now that item $l2$ is known as much more popular than item $l1$, with $PR(l2) = 0.7$ and $PR(l1) = 0.1$. Now, according to (8), $\tilde{r}_{ui} = 0.375$ and $\tilde{r}_{uj} = 0.800$. This indicates that item j should be ranked higher than item i, which makes more sense since item j is more similar to the popular items, and thus should have a higher chance for recommendation.

In the proposed *WeightedSLIM* model, the parameter for estimation is the similarity matrix AS. The BPR optimization [3] and the stochastic gradient descent (SGD), [23], were used to estimate it. Let $x_{uij} = \tilde{r}_{ui} - \tilde{r}_{uj}$ for each triple (u, i, j), $i \in R_u^+$, $j \in R_u^-$ (denoting that user u prefers item i over item j, [3]). Then, the gradient of x_{uij} with respect to AS can be calculated as:

$$\frac{\partial x_{uij}}{\partial AS_{il}} = \frac{PR(l)}{\sum_{l \in I_u^+ \wedge l \neq i} PR(l)} \qquad (9)$$

$$\frac{\partial x_{uij}}{\partial AS_{jl}} = \frac{PR(l)}{\sum_{l \in I_u^+ \wedge l \neq j} PR(l)}\ . \qquad (10)$$

The pseudocode for learning the proposed *WeightedSLIM* model is shown in Figure 1, where the input arguments contain the implicit feedback matrix R, the regularization term λ, the learning rate γ, the number of dimensions d, and the maximum number of iterations *maxIter*. The output is the learned item similarity matrix *AS*.



```
Algorithm 1: Learning the Weighted linear Model
1.35
Input  : UI (binary rating matrix), $\lambda, \gamma$ (learning rate), $d$, $maxIter$,
Output: Learned item similarity matrix $AS$, $iter \leftarrow 0$, $lastloss \leftarrow 0$
Compute the weights $PR$ of all items with equation (4)
Initialize the similarity matrix $S$
Generate triples $D_s = \{d((u, i, j)|i \in R_u^+, j \in R_u^-)\}$
while $iter < maxIter$ do
    $loss \leftarrow 0$
    foreach $d \in D_s$ do
        foreach $l \in I_u^+$ do
            $x_{uij} = \tilde{r}_{ui} - \tilde{r}_{uj}$
            $loss \leftarrow loss - \ln \sigma(x_{uij})$
            compute $\frac{\partial x_{uij}}{\partial AS_{jl}}$ with equation (10)
            compute $\frac{\partial x_{uij}}{\partial AS_{jl}}$ with equation (11)
            if $l \neq i$ then
                $AS_{il} \longleftarrow AS_{il} + \gamma(\frac{e^{-x_{uij}}}{1+e^{-x_{uij}}} \frac{\partial x_{uij}}{\partial AS_{il}} - \lambda AS_{il})$
                $loss \leftarrow loss + \lambda AS_{il}^2$
            end
            $AS_{jl} \longleftarrow AS_{jl} + \gamma(\frac{e^{-x_{uij}}}{1+e^{-x_{uij}}} \frac{\partial x_{uij}}{\partial AS_{jl}} - \lambda AS_{jl})$
            $loss \leftarrow loss + \lambda AS_{jl}^2$
        end
    end
    $error \leftarrow lastloss - loss$
    if $error < 1e-5$ then
        break;
    end
    $lastloss \leftarrow loss$
    $iter \leftarrow iter + 1$
end
```

Fig. 1: The algorithm for learning the proposed *WeightedSLIM* model

5 Experiments and Results

5.1 Datasets

The performance of the proposed *WeightedSLIM* model was evaluated on three different public datasets, namely FilmTrust, [24], MovieLens-100k (ML-100k) [25], and MovieLens-1M (ML-1M) [26], with characteristics presented in Table 5, where density denotes the ratio of the number of observed ratings over the total number of entries in the user–item matrix.

Table 5. The statistics (initial) of the three public datasets used in the experiments

Features	FilmTrust	ML-100k	ML-1M
#Users	1508	943	6040
#Items	2071	1682	3952
#Ratings	35497	100000	1000209
Density	1.14%	6.31%	4.19%

Since FilmTrust is a very sparse dataset, every user–movie pair having a rating is regarded as an observed interaction. However, for ML-100k and ML-1M, only ratings with values equal to 5 were kept as part of the observed positive feedback. The statistics of the three datasets after pre-processing are shown in Table 6. In each dataset, 80% of the rating values were randomly taken as a training set and the rest – as a test set. This process was repeated ten times to minimize the bias.

Table 6. The statistics (after pre-processing) of the three public datasets used in the experiments

Features	FilmTrust	ML-100k	ML-1M
#Users	1508	**928**	**6014**
#Items	2071	**1172**	**3232**
#Transactions	35497	**21201**	**226310**
#Density	1.14%	**1.95%**	**1.16%**

5.2 Evaluation Metrics

To evaluate the recommendation performance of the proposed *WeightedSLIM* model in comparison to existing ranking-based models, the standard evaluation metrics used for numeric prediction (which is the case with rating-based models), such as MAE and RMSE, are not suitable. Instead, the well-studied top-N ranking metrics, used for information retrieval evaluation, were employed, namely the precision, recall, F1-measure, normalized discounted cumulated gain (NDCG), mean reciprocal rank (MRR), and area under the Receiver Operating Characteristic (ROC) curve (AUC), to the ranked lists found by the compared models. These metrics are briefly presented below.

- *Prec@k*: Precision indicates how many items are relevant among all recommended items. For a given user u, the precision value of a ranked recommendation list at position k is defined as follows:
$$Prec@k = \frac{1}{k}\sum_{i=1}^{k}\delta(x_u(i) \in I_u^{test}), \quad (11)$$
where $x_u(i)$ denotes the predicted ranking list for user u and I_u^{test} denotes the set of preferred items by user u in the test set; $\delta(y) = 1$ if y is true; otherwise $\delta(y) = 0$.
- *Rec@k*: Recall represents the number of recommended items among all relevant items. For a given user u, the recall at position k of a ranked recommendation list is defined as follows:
$$Rec@k = \frac{1}{|I_u^{test}|}\sum_{i=1}^{k}\delta(x_u(i) \in I_u^{test}), \quad (12)$$
where $|I_u^{test}|$ denotes the number of preferred items by user u in the test set.
- *F1@k*: The F1-measure is the harmonic mean of precision and recall, defined as follows:
$$F1@k = \frac{2\times Prec@k \times Rec@k}{Prec@k + Rec@k}. \quad (13)$$
- *NDCG@k*: NDCG measures the quality of a recommendation model based on a weighted sum of the degree of relevancy of the ranked items (the graded relevance of the ranked items), [27]. If an item is more relevant, it contributes more to the recommendation quality, by adjusting its relative position in the ranking list. The *NDCG* value at position k of a ranked item list for a given user u is defined as follows:
$$NDCG@k = \frac{1}{Z_u}\sum_{i=1}^{k}\frac{2^{\delta(x_u(i)\in I_u^{test})}-1}{log(i+1)}, \quad (14)$$
where $Z_u = \sum_{i=1}^{k}\frac{1}{log(i+1)}$ denotes the normalization term.
- *MRR*: *MRR* measures the inverse of the position in the ranking of the first recommended item which also appears in the test set, [11]. The *MRR* value is calculated as:
$$MRR = \frac{1}{|U^{test}|}\sum_{u\in U^{test}}\frac{1}{min_{i\in I_u^{test}} p_u(i)}, \quad (15)$$
where U^{test} denotes the set of users in the test dataset and $p_u(i)$ denotes the position of item i recommended to user u.
- *AUC*: *AUC* is a popular criterion to measure

the quality of classification in information retrieval research field, [28]. The AUC is computed as:

$$AUC = \frac{1}{|U^{test}|}\sum_{u \in U^{test}} \frac{1}{|E(u)|}\sum_{i,j \in E(u)} \delta(\tilde{r}_{ui} > \tilde{r}_{uj}) \text{ , (16)}$$

where $E(u) = (i,j)|(u,i) \in S^{test} \Lambda (u,j) \notin (S^{test} \cup S^{train}$ denote the evaluation pairs per user, [3], with S^{train} and S^{test} being the training and test sets, respectively.

For all the evaluation metrics introduced above, a higher value indicates a better quality of the model.

5.3 Performance Comparison

In the conducted experiments, the proposed *WeightedSLIM* model was compared to the following existing ranking-based recommendation models:

- *PopRank*: A baseline ranking-based recommendation model, which makes top-N recommendations for each user based on item popularity, [29];
- *SLIM*: A sparse linear ranking-based model, which computes the prediction score for a new item based on an aggregation of other items, aiming to generate high-quality recommendations fast, [1];
- *SLIMbpr*: A *SLIM* model, but with optimization driven by BPR [3].

For a fair comparison, the same parameter settings were adopted in the experiments for both *SLIMbpr* and *WeightedSLIM*, which were implemented using the same algorithmic framework (c.f., Figure 1). Suppose $R(u)$ is the number of items that user u has rated. $R(u) \times 10$ triples $(u,i,j) : i \in R_u^+, j \in R_u^-$ were randomly generated for each user u for all datasets, where i is an observed item by user u, and j is an unobserved item. The number of iterations was set to 30. For regularization parameter λ, several values were tried ($\lambda = \{0.001, 0.01, 0.1\}$) and the one that gave the best AUC result was chosen, namely 0.01 for all datasets. The learning rate γ was set to 0.05. The recommendation performance comparison results of *WeightedSLIM* and the three other existing models on five evaluation metrics are shown in Table 7, Table 8 and Table 9.

Table 7. Recommendation performance comparison of *WeightedSLIM* with other ranking-based recommendation models (on FilmTrust)

Model	Pre@10	Rec@10	F1@10	NDCG@10	MRR	AUC
PopRank	0.3490	0.6318	0.4496	0.5362	0.6145	**0.9601**
SLIM	0.1628	0.3545	0.2231	0.3144	0.5109	0.9013
SLIMbpr	0.3401	0.6074	0.4360	0.5283	0.6125	0.9496
WeightedSLIM	**0.3533**	**0.6388**	**0.4550**	**0.5583**	**0.6457**	0.9422

Table 8. Recommendation performance comparison of *WeightedSLIM* with other ranking-based recommendation models (on MovieLens-100k)

Model	Pre@10	Rec@10	F1@10	NDCG@10	MRR	AUC
PopRank	0.0672	0.1361	0.0898	0.1124	0.2163	0.8478
SLIM	0.0107	0.0189	0.0137	0.0134	0.0442	0.7775
SLIMbpr	0.0810	0.1788	0.1049	0.1722	0.2525	0.8211
WeightedSLIM	**0.0891**	**0.2023**	**0.1237**	**0.1578**	**0.2839**	**0.8729**

Table 9. Recommendation performance comparison of *WeightedSLIM* with other ranking-based recommendation models (on MovieLens-1M)

Model	Pre@10	Rec@10	F1@10	NDCG@10	MRR	AUC
PopRank	0.0768	0.1173	0.0928	0.0997	0.2204	0.8926
SLIM	0.0207	0.0292	0.0242	0.0262	0.0840	0.8393
SLIMbpr	0.0960	0.1502	0.1171	0.1332	0.2858	0.8933
WeightedSLIM	**0.1021**	**0.1639**	**0.1258**	**0.1442**	**0.3016**	**0.9130**

Based on these results, one can make the following observations:

- *WeightedSLIM* outperforms all other models, used in this comparison, on all evaluation metrics on the two MovieLens datasets, which clearly proves its effectiveness. On the FilmTrust dataset, *WeightedSLIM* generally achieves the best result on all evaluation metrics, except for AUC, where *PopRank* takes the first place. This implies that the the-state-of-the-art models may not always get better performance than other well-performing models, even the simple baselines.
- *WeightedSLIM* generally achieves overall better performance than *SLIMbpr*, which indicates that the consideration of item weights in a sparse linear ranking-based model can help improve the recommendation performance. Figure 2 shows the experimental results for precision vs. recall for different values of N ($N \in \{5, 10, 15, 20\}$) across the datasets, where N denotes the size of the ranked list. This further demonstrates the effectiveness of the item weights impact.

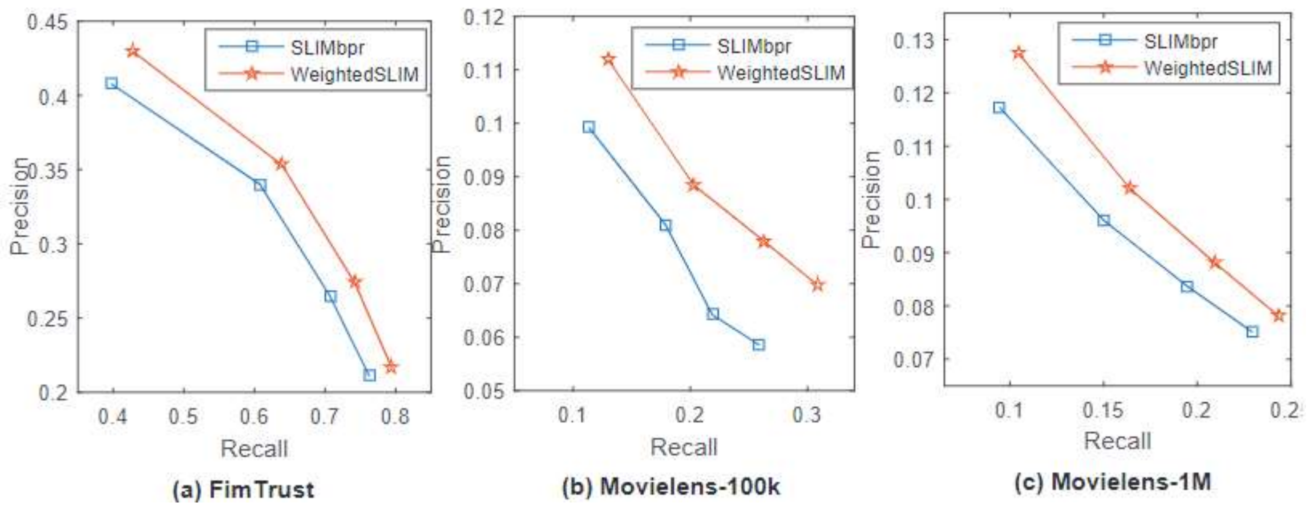


Fig. 2: Precision vs. recall for BPR-based models across the datasets

- The BPR-based models (*SLIMbpr* and *WeightedSLIM*) perform much better than *SLIM*, which demonstrates the power of pairwise preference learning algorithms, [3], for top-N recommendations from implicit user feedback.
- *PopRank* beats *SLIMbpr* on all evaluation metrics on FilmTrust. This may be because FilmTrust is a relatively sparse dataset and that in the experiments every observed rating was treated as positive feedback. However, low rating values (e.g., ‘1’ and ‘2’) cannot reflect users’ preferences and the assumption of BPR (i.e., that users prefer observed items over the unobserved ones).

5.4 Computational Complexity Analysis

The time complexity of the model training process consists of two stages – item weighting and model learning. In the first stage, the main time is spent on item correlation matrix computation and item weights learning with *ItemRank*. Let the number of observed ratings be $|K|$. Then, in the worst case, the computational complexity for correlation matrix computation is $O(|K|n)$ using Sparse Basic Linear Algebra Subprograms (BLAS), [30], and for item weights learning it is $O(tn^2)$, where t is the number of iterations and n is the number of items. In the model training stage, the total time complexity of learning the model parameters is $O(tm|I^+|)$, where t is the number of iterations, m is the number of users, and $|I^+|$ is the average number of items that user likes. Thus, the worst-case total time complexity of training process is $O(|K|n + tn^2 + tm|I^+|)$.

For the parameter learning stage, the time complexity of *WeightedSLIM* is the same as for *SLIMbpr*. Thus, the complexity gain is in the item weights learning. In [21], it has been demonstrated that *ItemRank* scales very well with the increase of the number of users, and the computation is very efficient with fewer number of iterations.

6 Conclusion

This paper has proposed a novel ranking-based model, *WeightedSLIM*, which aims to provide users with high-quality top-*N* recommendations based on implicit feedback. *WeightedSLIM* is able to take different item weights into consideration when computing the predicted score for each unobserved user–item pair, whereby the item weights are calculated using *ItemRank*, and optimization is driven by Bayesian Personalized Ranking (BPR). Experiments conducted on three benchmark datasets, using six ranking-based evaluation metrics for performance comparison with one baseline and two state-of-the-art ranking-based recommendation models, demonstrate the effectiveness of *WeightedSLIM*.

References:

[1] X. Ning and G. Karypis, "SLIM: Sparse Linear Methods for Top-N Recommender Systems," in *2011 IEEE 11th International Conference on Data Mining*, 11-14 Dec. 2011 2011, pp. 497-506, doi: 10.1109/ICDM.2011.134.

[2] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," in *The Web Conference*, 1999.

[3] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian

personalized ranking from implicit feedback," in *25th Conference on Uncertainty in Artificial Intelligence*, Montreal, Quebec, Canada, 2009.

[4] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," in *2008 Eighth IEEE International Conference on Data Mining*, 15-19 Dec. 2008 2008, pp. 263-272, doi: 10.1109/ICDM.2008.22.

[5] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *4th ACM Conference on Recommender Systems - RecSys '10*, Barcelona, Spain, 2010, doi: 10.1145/1864708.1864721.

[6] G. B. Guo, J. Zhang, and N. Yorke-Smith, "A Novel Recommendation Model Regularized with User Trust and Item Ratings,", *IEEE Transactions on Knowledge and Data Engineering,* vol. 28, no. 7, pp. 1607-1620, Jul 1 2016, doi: 10.1109/Tkde.2016.2528249.

[7] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer,* vol. 42, no. 8, pp. 30-37, 2009, doi: 10.1109/mc.2009.263.

[8] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '08*, Las Vegas, Nevada, USA, 2008, doi: 10.1145/1401890.1401944.

[9] M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola, "COFIRANK Maximum Margin Matrix Factorization for collaborative ranking," in *20th International Conference on Neural Information Processing Systems*, Vancouver, British Columbia, Canada, 2007.

[10] W. Pan, H. Zhong, C. Xu, and Z. Ming, "Adaptive Bayesian personalized ranking for heterogeneous implicit feedbacks," *Knowledge-Based Systems,* vol. 73, pp. 173-180, 2015, doi: 10.1016/j.knosys.2014.09.013.

[11] 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 *6th ACM Conference on Recommender Systems - RecSys '12*, Dublin, Ireland, 2012, doi: 10.1145/2365952.2365981.

[12] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, no. 6, pp. 734-749, 2005, doi: 10.1109/tkde.2005.99.

[13] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems,* vol. 46, pp. 109-132, 2013, doi: 10.1016/j.knosys.2013.03.012.

[14] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," *Advances in Artificial Intelligence,* vol. 2009, pp. 1-19, 2009, doi: 10.1155/2009/421425.

[15] K. K. Fletcher, "A Method for Dealing with Data Sparsity and Cold-Start Limitations in Service Recommendation Using Personalized Preferences," in *2017 IEEE International Conference on Cognitive Computing (ICCC),* pp. 72-79, 2017.

[16] G. Guo, J. Zhang, and D. Thalmann, "Merging trust in collaborative filtering to alleviate data sparsity and cold start," *Knowledge-Based Systems,* vol. 57, pp. 57-68, 2014, doi: 10.1016/j.knosys.2013.12.007.

[17] P. Melville, R. J. Mooney, and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," in *AAAI/IAAI*, 2002.

[18] D. Schall, "Expertise ranking using activity and contextual link measures," *Data & Knowledge Engineering,* vol. 71, no. 1, pp. 92-113, 2012, doi: 10.1016/j.datak.2011.08.001.

[19] M. Gao, Z. Wu, and F. Jiang, "Userrank for item-based collaborative filtering recommendation," *Information Processing Letters,* vol. 111, no. 9, pp. 440-446, 2011, doi: 10.1016/j.ipl.2011.02.003.

[20] D. Menezes, A. Lacerda, L. Silva, A. Veloso, and N. Ziviani, "Weighted slope one predictors revisited," in *22nd International Conference on World Wide Web*, Rio de Janeiro, Brazil, 2013, doi: 10.1145/2487788.2488093.

[21] M. Gori and A. Pucci, "ItemRank: a random-walk based scoring algorithm for recommender engines," in *20th International Joint Conference on Artificial Intelligence*, Hyderabad, India, 2007.

[22] J.-F. Pessiot, T.-V. Truong, N. Usunier, M.-R. Amini, and P. Gallinari, "Learning to Rank for Collaborative Filtering," in *International Conference on Enterprise Information Systems*, 2007.

[23] M. A. Zinkevich, M. Weimer, A. Smola, and L. Li, "Parallelized stochastic gradient descent," in *23rd International Conference on*

Neural Information Processing Systems, Vancouver, British Columbia, Canada, 2010.

[24] J. Golbeck and J. Hendler, "FilmTrust: movie recommendations using trust in web-based social networks," in *2006 3rd IEEE Consumer Communications and Networking Conference - CCNC 2006,* 8-10 Jan. 2006, vol. 1, pp. 282-286, doi: 10.1109/CCNC.2006.1593032.

[25] MovieLens 100K Dataset, [Online]. http://grouplens.org/datasets/movielens/100k/ (Accessed Date: January 17, 2024).

[26] MovieLens 1M Dataset, [Online]. https://grouplens.org/datasets/movielens/1m/ (Accessed Date: January 17, 2024).

[27] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of IR techniques," *ACM Transactions on Information Systems,* vol. 20, no. 4, pp. 422-446, 2002, doi: 10.1145/582415.582418.

[28] G. Schröder, "Setting Goals and Choosing Metrics for Recommender System Evaluations," 2011.

[29] T. D. Noia, V. C. Ostuni, P. Tomeo, and E. D. Sciascio, "SPrank," *ACM Transactions on Intelligent Systems and Technology,* vol. 8, no. 1, pp. 1-34, 2016, doi: 10.1145/2899005.

[30] I. S. Duff, M. A. Heroux, and R. Pozo, "An overview of the sparse basic linear algebra subprograms," *ACM Transactions on Mathematical Software,* vol. 28, no. 2, pp. 239-267, 2002, doi: 10.1145/567806.567810.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed to the presented research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

This publication has emanated from research conducted with the financial support of the Bulgarian National Science Fund (BNSF) under the Grant No. KP-06-IP-CHINA/1 (КП-06-ИП-КИТАЙ/1).

Conflict of Interest

The authors have no conflicts of interest to declare.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)