MPR: Multi-Objective Pairwise Ranking

Rasaq Otunba
Inf. Sci. & Tech. Dept.
George Mason University
Fairfax, Virginia, USA
rotunba@gmu.edu

Raimi A. Rufai
Hybris Marketing
SAP Labs, Inc.
Montreal, Quebec, Canada
raimi.rufai@sap.com

Jessica Lin
Computer Science Dept.
George Mason University
Fairfax, Virginia, USA
jessica@gmu.edu

ABSTRACT

The recommendation challenge can be posed as the problem of predicting either item ratings or item rankings. The latter approach has proven more effective. Pairwise learning-to-rank techniques have been relatively successful. Hence, they are popularly used for learning recommender model parameters such as those in collaborative filtering (CF) models. The model parameters are learned by optimizing close smooth approximations of the non-smooth information retrieval (IR) metrics such as Mean Area Under ROC curve (AUC).

Targeted campaigns are an alternative to item recommendations for increasing conversion. The user ranking task is referred to as audience retrieval. It is used in targeted campaigns to rank push campaign recipients based on their potential to convert. In this work, we consider the task of efficiently learning a ranking model that provides item recommendations and user rankings simultaneously. We adopt pairwise learning for this task. We refer to our approach as *multi-objective pairwise ranking* (MPR).

We describe our approach and use experiments to evaluate its performance.

CCS CONCEPTS

•Information systems →Personalization; •Computing methodologies →Learning to rank;

KEYWORDS

Collaborative filtering; recommendations; audience retrieval

ACM Reference format:
Rasaq Otunba, Raimi A. Rufai, and Jessica Lin. 2017. MPR: Multi-Objective Pairwise Ranking. In *Proceedings of RecSys'17, August 27–31, 2017, Como, Italy.*, 9 pages.
DOI: http://dx.doi.org/10.1145/3109859.3109903

1 INTRODUCTION

Recommendation is a vital part of services rendered by many content providers either as a primary service or as an ancillary one. Search engine providers like Google recommend non-personalized results to search queries as a main service while other institutions such as Amazon recommend similar content to initially searched content as an ancillary service. Personalization refers to recommending content tailored to each user's preferences. Personalization can boost conversion. Most existing personalization systems provide item recommendations. Item recommendation continues to be an attractive research area especially after the Netflix Price a decade ago [15]. Some of the existing methods for item recommendation predict ratings while others predict item rankings. It has been shown that predicting item rankings leads to better item recommendations [1, 28, 29]. The recent proliferation of content on social media platforms has also led to increased interest in tailoring content for consumers. An example is Twitter recently experimenting with sorting tweets by interest rather than the current reverse chronological ordering. Facebook also continues to tweak its news feed ranking algorithm for improved personalization.

Targeted marketing push campaigns can also help increase conversion. An example application is in marketing where products are recommended to potential customers. Customers are ranked based on their potential to engage in product push campaigns. The task of ranking users in this manner is considered ranking from implicit feedback since users typically click or act on the delivered campaign as opposed to rating the campaign. Many techniques have been proposed for item ranking based on implicit feedback [11, 21]. The user ranking problem (aka audience retrieval) in targeted campaigns is gaining more attention under the umbrella of computational advertising [17]. The need for increased conversion is driving the research for more effective campaigns. While item recommendation engines have been successfully used for audience retrieval and even led to increased click-to-open rates (CTOR) of push campaigns [27], techniques specifically designed for audience retrieval have proven more effective [16, 17, 20, 31].

We propose a solution for simultaneous item recommendation and audience retrieval in this work. Our proposed solution avoids training two different systems. In particular, we consider our proposed solution as a framework for combining an item ranking algorithm with an audience retrieval one. We refer to it as a *multi-objective ranking algorithm* (MPR). The combination maintains the same asymptotic time bounds as its constituent ranking algorithms. We adopt the pairwise learning-to-rank approach for implicit feedback in this work because of its success. The remainder of this paper is organized as follows. In Section 2, we summarize related work. In Section 3, we give the necessary background for the rest of the paper. We describe the two algorithms in Sections 4 and 5. We validate our approach with experiments on popular datasets in Section 6 and discuss the results in Sections 7 and 8. Finally, we conclude by summarizing the work and hinting on possible directions of future work in Section 9.

2 RELATED WORK

The inputs to a personalized recommender system include users, items, and feedback. The feedback can be implicit in the form of observed actions such as clicks and views or explicit in the form of ratings. Implicit feedback is often more readily available than explicit feedback. The approaches to recommendations can be knowledge-based, content-based or collaborative. Knowledge-based recommendation takes the context and domain knowledge into consideration. In content-based recommenders, recommendation is based on matches between user profiles and item characteristics. Content filtering enjoys industry success at companies like Pandora by recommending musical content based on user profiles. A challenge to content based recommenders is the lack of information in some instances.

The most prominent of these approaches, collaborative filtering (CF), on the other hand, uses ratings or implicit feedback. In item recommendations, user-based CF considers users' historical actions on items as well as the actions of similar users. Item-based CF however considers historical actions of users on similar items. Content-based recommendation and CF can be combined in a hybrid setting [30]. Matrix factorization (MF)-CF performs better than kNN-CF in general [15]. In this work, we focus on MF-CF for this reason even though our techniques are adaptable to kNN-CF.

Many recommendation algorithms provide item recommendations using a machine learning technique. This often involves learning a model by optimizing it with respect to a rating prediction objective function such as the mean squared error or (a smooth version of) a ranking objective function such as AUC.

Some existing item recommendation techniques have been used for audience retrieval. Robinson [24] addresses the audience retrieval problem using the traditional user-based CF technique. There are other techniques specifically designed for audience retrieval. Sandeep et al. [20] proposed techniques for maximizing campaign conversions by using multiple user actions, including search queries, page views, ad clicks, etc. A conversion for a given campaign is a transaction, initiated by the campaign, that generates revenue. Yandong et al. [17] combined the information about a campaign and a set of *seed users* who converted in a previous campaign to maximize conversions. Bhargav et al. [31] claimed user-campaign feedback is insufficient for audience retrieval due to sparsity. They instead used user-product preferences in a collaborative setting for audience retrieval. Their technique called *focused matrix factorization model* (FMF) is an extension of the popular Bayesian personalized ranking (BPR) [22] algorithm. Haishan et al. [16] attempted to expand an existing targeted audience by identifying new users with similar attributes.

3 BACKGROUND

In this section, we define notations and preliminaries that will be used in the rest of the article. We also give an overview of CF recommender systems, objective functions, and the RankNet cost function.

3.1 Notations and Preliminaries

We denote the set of all users by U and the set of all items by I. A user interaction with an item is recorded as 1 to represent implicit feedback. A lack of implicit feedback is not recorded at all, since it might imply either a lack of interest in the item or merely a missing value. In many other systems, unobserved feedback is considered negative. The latter approach causes the trained model to predict unseen feedback as negative even when it could just be missing. The set of all user implicit feedback, $U \times I$, is denoted by S. $S_{training}$ is the training set and S_{test} is the test set. We denote users that prefer item i by U_i^+, users that do not prefer i by U_i^- and the overall ranking of users per item i by U_i. We denote items preferred by user u by I_u^+, items not preferred by user u by I_u^- and the overall ranking of items for user u by I_u. The actual components (users and items) are represented by a matrix $\mathbf{X}$ where x_{ui} is the feedback of user u on item i. The vector $\mathbf{x_u}$ denotes the interactions of user u in $\mathbf{X}$. In MF, $\mathbf{X}$ is factorized into low rank matrices $\mathbf{P}$ for users and $\mathbf{Q}$ for items. Henceforth, we refer to the values of $\mathbf{P}$ and $\mathbf{Q}$ as model parameters. Matrices and vectors are represented in bold font. We denote predicted ranking of items for user u by $\hat{I}_u$ and predicted ranking of users per item i by $\hat{U}_i$. We define $\hat{x}_{ui}$ as a real-valued function that predicts the preference of user u of item i based on model parameters. For item ranking, we denote by $\hat{x}_{uij}$ the predicted relative preference of user u of item i over item j based on model parameters. More formally,

$$\hat{x}_{uij} = \hat{x}_{ui} - \hat{x}_{uj}. \tag{1}$$

Similarly, we denote by $\hat{y}_{kvw}$ the predicted relative preference of user v of item k compared to user w based on model parameters. In formal notation,

$$\hat{y}_{kvw} = \hat{y}_{kv} - \hat{y}_{kw}. \tag{2}$$

3.2 Overview of CF Recommender Systems

Recommender models represent the relationship between input parameters (users, items, and feedback) and the desired output (ratings or implicit feedback). We now review the two CF models: neighborhood-based and latent factor systems. Our proposed technique is applicable to both models. The CF models are differentiated by how $\hat{x}_{ui}$ is derived. Other hybrid variants such as factorized neighborhood [13, 14] are derived from these two CF models. These models could be learned using machine learning algorithms such as stochastic gradient descent (SGD) and alternating least squares (ALS). The SGD method is one of the most popular first-order iterative optimization algorithms. SGD is popularly used in learning-to-rank algorithms.

3.2.1 Latent Factor Models. The two low-rank factor matrices $\mathbf{P}$ and $\mathbf{Q}$ are typically learned using SGD. These matrices are expected to better represent the features of items and users. This is a form of dimensionality reduction. The value $\hat{x}_{ui}$ is typically computed as the dot product of the user latent vector $\mathbf{p_u}$ and the item latent vector $\mathbf{q_i}$.

$$\hat{x}_{ui} = \mathbf{p_u^T} \cdot \mathbf{q_i} \tag{3}$$

Some state-of-the-art techniques compute $\hat{x}_{ui}$ with sophisticated approaches such as deep neural networks. The authors of [9] for example replaced the typical dot product with a deep neural network (multi-layer feed-forward neural network). One wonders if other neural network architectures might yield better results.

3.2.2 Neighborhood Systems. Neighborhood systems can be either *memory-based* or *model-based*. Each of these can be either user-based k-nearest-neighbor (userkNN) or item-based k-nearest-neighbor (item-kNN) [4]. In userkNN, given some k, $\hat{x}_{ui}$ is computed as the weighted sum of the feedback of the k users most similar to user u on item i, where similarity is computed according to some *similarity measure*. In itemkNN, given some k, $\hat{x}_{ui}$ is computed as the weighted sum of the feedback of user u on the k items most similar to item i. ItemKNN is known to outperform [25] userKNN because the item neighborhood is sparse [19]. We describe only itemKNN below, but userKNN is analogous. We define the predicted feedback of user u on item i as:

$$\hat{x}_{ui} = \sum_{j \in C_i \wedge j \neq i} f(x_{uj}), \tag{4}$$

where C_i is the index set of the k items most similar to i and f is a weight function based on the similarity of i to j. Common similarity measures are the Jaccard, Cosine etc. The cosine vector similarity c_{ij} between items i and j is computed by:

$$c_{ij} = \frac{|U_i^+ \cap U_j^+|}{\sqrt{|U_i^+|.|U_j^+|}}. \tag{5}$$

So far, we have described the *memory-based* neighborhood method. In the model-based neighborhood systems, prediction is done on model parameters. An example similarity function used in model-based neighborhood systems is the sparse aggregation coefficients [19]. In such systems, $\hat{x}_{ui}$ is computed as:

$$\hat{x}_{ui} = \sum_{j \in C_i \wedge j \neq i} f(\hat{x}_{uj}), \tag{6}$$

where $\hat{x}_{uj} = \mathbf{p_u}^T.\mathbf{w_j}$ and $\mathbf{w_j}$ is a learned vector coefficient of item j in an n x n matrix $\mathbf{W}$.

3.3 Objective Functions

Learning the parameters of recommender model involves optimizing an objective function, typically including a regularization term to avoid over-fitting. Objective functions can be *point-wise*, *pair-wise* or *list-wise*. In each case, the objective function can be one of a number of loss functions, including the square loss [15], log loss [21], hinge loss [23] etc. We denote the objective function to be optimized by $L(\cdot)$.

3.3.1 Point-wise Objective Function. The goal of a point-wise objective function is to minimize the overall error of predicting the feedback a user u would give an item i. A point-wise objective [15] function can be expressed schematically as:

$$\sum_{u \in U} \sum_{i \in I} L(x_{ui}, \hat{x}_{ui}) \tag{7}$$

where L is some loss function and x_{ui} is the actual feedback of user u on item i.

3.3.2 Pair-wise objective function. The goal a pair-wise objective function is to minimize the overall error of predicting the feedback of a user u that prefers item i compared to another item j with (missing implicit feedback). This is particularly suitable for predicting rankings rather than ratings. An example of this is the work of Rendle et al. [21] for item recommendations.

$$\sum_{u \in U} \sum_{i \in I_u^+} \sum_{j \in I_u^-} L(x_{uij}, \hat{x}_{uij}) \tag{8}$$

3.3.3 List-wise objective function. The goal of a list-wise objective function is to minimize the overall error of predicting the ranking of items in I for each user u in U. This is less common than pair-wise and point-wise approaches. An example of this for item recommendations is proposed by Shi et al. [26] and Huang et al. [12].

$$\sum_{u \in U} L(I_u, \hat{I}_u) \tag{9}$$

3.4 RankNet

The RankNet probabilistic cross-entropy cost function is a pairwise ranking objective function widely used in many machine learning algorithms [3]. The cross-entropy between two entities can be intuitively described as a measure of uncertainty in a system with respect to two entities. The goal of minimizing the cross-entropy cost function is to reduce the amount of uncertainty of the model in ranking two items. The cross-entropy cost function is known to improve the performance of machine learning algorithms e.g. DNN [5]. Hence, we adopt the RankNet function in our techniques. The RankNet cost function can also be described as a measure of the prediction error from swapping two correctly ranked items. The RankNet cost function combines the pairwise cross-entropy cost and the logistic sigmoid of the difference of the item relevance scores. Given a user u and two items i and j, the RankNet loss L_{uij} is written as:

$$L_{uij} = -\overline{P}_{uij} \ln P_{uij} - (1 - \overline{P}_{uij}) \ln\bigl(1 - P_{uij}\bigr), \tag{10}$$

where P_{uij} is defined as

$$\frac{1}{1 + e^{-\hat{x}_{uij}}}, \tag{11}$$

i.e. the probability that user u prefers an item i to item j is modeled by the logistic sigmoid function. Given a user u and a pair of items i and j, without loss of generality, let us assume that u always prefers item i to j. The probability of this event is denoted by $\overline{P}_{uij} = 1$. By substituting $\overline{P}_{uij}$ and P_{uij} in Equation (10), we obtain

$$L_{uij} = \ln\Bigl(1 + e^{-\hat{x}_{uij}}\Bigr). \tag{12}$$

The derivative of L_{uij} with respect to $\hat{x}_{uij}$ is:

$$\frac{\partial L_{uij}}{\partial \hat{x}_{uij}} = \frac{-1}{1 + e^{\hat{x}_{uij}}}. \tag{13}$$

4 AUDIENCE RETRIEVAL

We give a pairwise ranking algorithm for audience retrieval here. It is straightforward to derive an item ranking equivalent. We describe our optimization criterion which we refer to as AR-OPT for easy reference and its derivative. We also explore the analogy of AR-OPT to the AUC metric.

4.1 AR-OPT Optimization Criterion

We define the likelihood of estimating the model parameters **P** and **Q** given that a user v would prefer an item k more strongly than another user w. Formally, this likelihood is equivalent to the probability, $\Pr(y_{kvw}|\mathbf{P},\mathbf{Q})$. In terms of the cross entropy cost function, $\mathcal{L}(\mathbf{P},\mathbf{Q}|y_{kvw})$, the likelihood function can be expressed as

$$\begin{aligned} \mathcal{L}(\mathbf{P},\mathbf{Q}|y_{kvw}) &= \Pr(y_{kvw}|\mathbf{P},\mathbf{Q}) && (14)\\ &= e^{-L_{kvw}}. && (15) \end{aligned}$$

Since maximizing the likelihood estimate (MLE) is equivalent to minimizing the negative log-likelihood , we can define our audience retrieval optimization criterion, AR-OPT, in terms of the negative log-likelihood function as

$$\begin{aligned} \textit{AR-OPT} &= -\ln \prod_{k\in I, v\in U_k^+, w\in U_k^-} e^{-L_{kvw}} && (16)\\ &= \prod_{k\in I, v\in U_k^+, w\in U_k^-} L_{kvw} && (17)\\ &= \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} \ln\left(1+e^{-y_{kvw}}\right) && (18) \end{aligned}$$

$$(\mathbf{P}_{MLE},\mathbf{Q}_{MLE}) = \arg\min_{\mathbf{P},\mathbf{Q}} \textit{AR-OPT} \quad (19)$$

$$(\mathbf{P}_{MAP},\mathbf{Q}_{MAP}) = \arg\min_{\mathbf{P},\mathbf{Q}} \left\{\textit{AR-OPT}\cdot \Pr(\mathbf{P},\mathbf{Q})\right\} \quad (20)$$

where MAP is the maximum a posteriori estimate derived from Bayesian inference which takes priors of **P** and **Q** into account. $\Pr(\mathbf{P},\mathbf{Q})$ is the prior distribution before optimization. Since in this work, we assumed $\Pr(\mathbf{P},\mathbf{Q})$ to be uniform, the two solutions in Equation (19) and Equation (20) coincide. We further express *AR-OPT* as:

$$\begin{aligned} \textit{AR-OPT} = \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} &\left\{ \ln\left(1+e^{-\hat{y}_{kvw}}\right) \right. \\ &\left. + \left\{\frac{\lambda_P}{2}||\mathbf{p_v}||_F^2 + \frac{\lambda_P}{2}||\mathbf{p_w}||_F^2 + \frac{\lambda_Q}{2}||\mathbf{q_k}||_F^2\right\}\right\} \end{aligned} \quad (21)$$

where λ is the regularization hyper-parameter introduced to avoid overfitting. The partial derivatives of AR-OPT are:

$$\frac{\partial}{\partial \mathbf{q_k}}\textit{AR-OPT} = \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} \frac{\partial L_{kvw}}{\partial \hat{y}_{kvw}}\cdot\frac{\partial}{\partial \mathbf{q_k}}\hat{y}_{kvw} + \lambda_Q \mathbf{q_k} \quad (22)$$

$$\frac{\partial}{\partial \mathbf{p_v}}\textit{AR-OPT} = \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} \frac{\partial L_{kvw}}{\partial \hat{y}_{kvw}}\cdot\frac{\partial}{\partial \mathbf{p_v}}\hat{y}_{kvw} + \lambda_P \mathbf{p_v} \quad (23)$$

$$\frac{\partial}{\partial \mathbf{p_w}}\textit{AR-OPT} = \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} \frac{\partial L_{kvw}}{\partial \hat{y}_{kvw}}\cdot\frac{\partial}{\partial \mathbf{p_w}}\hat{y}_{kvw} + \lambda_P \mathbf{p_w} \quad (24)$$

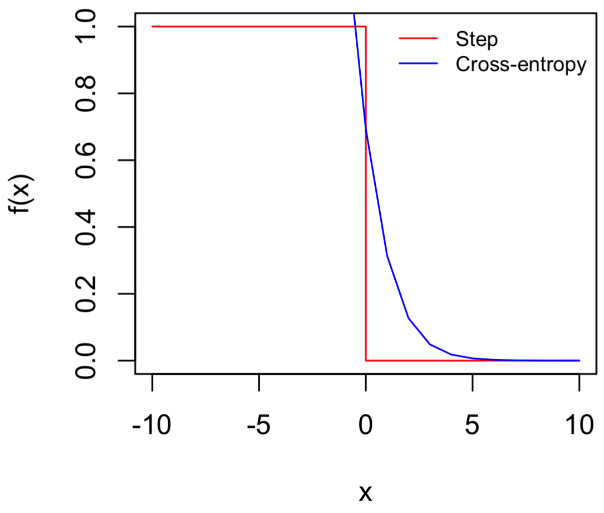

Figure 1: The RankNet cross entropy cost function and the Heaviside step function

4.1.1 Analogy to AUC. Pairwise learning algorithms including the ones presented in this work ultimately aim to maximize the number of correctly ranked pairs. The receiver operating characteristic (ROC) curve depicts the number of correctly ranked pairs more closely. AUC is sometimes criticized for not penalizing incorrect rankings. Nonetheless, it is still widely considered to be a good measurement of click-through rate [6]. Mean AUC over all items and all users is defined as:

$$AUC = \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} z_k * \delta(\hat{y}_{kvw}) \quad (25)$$

where the above expression implicitly defines AUC *gain* and z_k is the normalization constant:

$$z_k = \frac{1}{|I||U_k^+||U_k^-|} \quad (26)$$

and $\delta(x)$ is the Heaviside function:

$$\delta(\hat{y}_{kvw}) = \begin{cases} 1, & \text{if } \hat{y}_{kvw} > 0\\ 0, & \text{otherwise} \end{cases} \quad (27)$$

We can express AUC loss as:

$$\begin{aligned} AUC_{loss} &= 1 - \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} z_k * \delta(\hat{y}_{kvw}) && (28)\\ &= \sum_{k\in I}\sum_{v\in U_k^+}\sum_{w\in U_k^-} z_k * (1-\delta(\hat{y}_{kvw})) && (29) \end{aligned}$$

Equation (18) has a similar structure to Equation (29) if we ignore the normalization constant z_k in the latter. We observe that minimizing the RankNet cost function is equivalent to minimizing the AUC loss i.e. $1-\delta(\hat{y}_{kvw})$.

$$1-\delta(\hat{y}_{kvw}) = \begin{cases} 0, & \text{if } \hat{y}_{kvw} > 0\\ 1, & \text{otherwise} \end{cases} \quad (30)$$

Figure 1 shows the similarity in the shapes of the above step function and the RankNet cross-entropy cost function.

4.2 Audience Retrieval Learning Algorithm

Our audience retrieval algorithm (AR) uses SGD with the optimization criterion introduced above to learn the model parameters **P** and **Q**. The task of pairwise ranking is to predict $\mathbf{p_k^T q_v} - \mathbf{p_k^T q_w}$ for user v on preferred item k and another user w with no preference for item k. So, we have two models to update: **P** and **Q**. The vectors $\mathbf{p_k}$ and $\mathbf{q_v}$ are row feature vectors of the item and user latent factor matrices respectively. For an f-ranked matrix **P** (and **Q**), the vector $\mathbf{p_k}$ (and $\mathbf{q_u}$) is f-dimensional. The partial derivatives of $\hat{y}_{kvw}$ are given as:

$$\frac{\partial}{\partial \mathbf{q_k}} \hat{y}_{kvw} = \mathbf{p_v} - \mathbf{p_w} \tag{31}$$

$$\frac{\partial}{\partial \mathbf{p_v}} \hat{y}_{kvw} = \mathbf{q_k} \tag{32}$$

$$\frac{\partial}{\partial \mathbf{p_w}} \hat{y}_{kvw} = -\mathbf{q_k} \tag{33}$$

Note that the partial derivatives given in Equations (31) to (33) direct the path that SGD takes to arrive at the optimal solution. The pseudocode of AR is given in Algorithm 1.

Algorithm 1 AR(I, U)

```
1: Output: Optimized matrices P and Q
2: initialize learning rate $\eta$, P and Q
3: repeat
4:     draw $k, v, w$ from $I, U_k^+, U_k^-$ uniformly
5:     $\mathbf{q_k} \leftarrow \mathbf{q_k} - \eta * \frac{\partial}{\partial \mathbf{q_k}} AR\text{-}OPT$
       $\mathbf{p_v} \leftarrow \mathbf{p_v} - \eta * \frac{\partial}{\partial \mathbf{p_v}} AR\text{-}OPT$
       $\mathbf{p_w} \leftarrow \mathbf{p_w} - \eta * \frac{\partial}{\partial \mathbf{p_w}} AR\text{-}OPT$
6: until convergence or maximum number of iterations
7: return P, Q
```

Note that the updates to the model parameters in Line 5 are performed concurrently and take a step in the stochastic descent towards the optimal solution. In Line 4 of Algorithm 1, we draw k from I such that neither U_k^+ nor U_k^- is empty. This takes time $O(t|I|)$, where $t \in (0, 1)$ is the proportion of new items not currently preferred by anyone. The retrieval of the two sets U_k^+ and U_k^- takes $O(|U|)$. The time complexity of AR is thus $O(s(t|I| + |U|)) = O(s(|I|+|U|))$ where s is the number of times the loop in Algorithm 1 is executed.

5 MULTI-OBJECTIVE PAIRWISE RANKING

We propose MPR as a combination of AR and its item pairwise ranking equivalent. We define the objective function *MPR-OPT* as the weighted sum of the item ranking objective function *IR-OPT* and the audience retrieval objective function *AR-OPT*. This approach is sometimes called *linear scalarization* [18].

$$MPR\text{-}OPT = \alpha AR\text{-}OPT + (1-\alpha) IR\text{-}OPT \tag{34}$$

where α is a scalarization hyperparameter for trading off between the constituent objective functions. It is obvious from the above equation that:

$$\frac{\partial}{\partial \mathbf{q_k}} MPR\text{-}OPT = \alpha \frac{\partial}{\partial \mathbf{q_k}} AR\text{-}OPT, \tag{35}$$

$$\frac{\partial}{\partial \mathbf{p_v}} MPR\text{-}OPT = \alpha \frac{\partial}{\partial \mathbf{p_v}} AR\text{-}OPT, \tag{36}$$

$$\frac{\partial}{\partial \mathbf{p_w}} MPR\text{-}OPT = \alpha \frac{\partial}{\partial \mathbf{p_w}} AR\text{-}OPT, \tag{37}$$

$$\frac{\partial}{\partial \mathbf{p_u}} MPR\text{-}OPT = (1-\alpha) \frac{\partial}{\partial \mathbf{p_u}} IR\text{-}OPT, \tag{38}$$

$$\frac{\partial}{\partial \mathbf{q_i}} MPR\text{-}OPT = (1-\alpha) \frac{\partial}{\partial \mathbf{q_i}} IR\text{-}OPT, \tag{39}$$

$$\frac{\partial}{\partial \mathbf{q_j}} MPR\text{-}OPT = (1-\alpha) \frac{\partial}{\partial \mathbf{q_j}} IR\text{-}OPT. \tag{40}$$

We defined *AR-OPT* and its partial derivatives in Equations (21) to (24). We define IR-OPT and its derivatives as

$$\begin{aligned} IR\text{-}OPT = \sum_{u \in U} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \Big\{ \ln\Big(1 + e^{-\hat{x}_{uij}}\Big) \\ + \Big\{ \frac{\lambda_P}{2} ||\mathbf{p_u}||_F^2 + \frac{\lambda_Q}{2} ||\mathbf{q_i}||_F^2 + \frac{\lambda_Q}{2} ||\mathbf{q_j}||_F^2 \Big\} \Big\}. \end{aligned} \tag{41}$$

The partial derivatives of IR-OPT are defined as

$$\frac{\partial IR\text{-}OPT}{\partial \mathbf{p_u}} = \sum_{u \in U} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \frac{\partial L_{uij}}{\partial \hat{x}_{uij}} \cdot \frac{\partial}{\partial \mathbf{q_k}} \hat{x}_{uij} + \lambda_P \mathbf{p_u} \tag{42}$$

$$\frac{\partial IR\text{-}OPT}{\partial \mathbf{q_i}} = \sum_{u \in U} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \frac{\partial L_{uij}}{\partial \hat{x}_{uij}} \cdot \frac{\partial}{\partial \mathbf{q_i}} \hat{x}_{uij} + \lambda_Q \mathbf{q_i} \tag{43}$$

$$\frac{\partial IR\text{-}OPT}{\partial \mathbf{q_j}} = \sum_{u \in U} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \frac{\partial L_{uij}}{\partial \hat{x}_{uij}} \cdot \frac{\partial}{\partial \mathbf{q_j}} \hat{x}_{uij} + \lambda_Q \mathbf{q_j} \tag{44}$$

Algorithm 2 MPR(I, U)

```
1: Output: Optimized matrices P and Q
2: initialize $\alpha$, $\eta$, P and Q
3: repeat
4:     draw $u, i, j$ from $U, I_u^+, I_u^-$ uniformly
5:     $\mathbf{p_u} \leftarrow \mathbf{p_u} - \eta * \frac{\partial}{\partial \mathbf{p_u}} MPR\text{-}OPT$
       $\mathbf{q_i} \leftarrow \mathbf{q_i} - \eta * \frac{\partial}{\partial \mathbf{q_i}} MPR\text{-}OPT$
       $\mathbf{q_j} \leftarrow \mathbf{q_j} - \eta * \frac{\partial}{\partial \mathbf{q_j}} MPR\text{-}OPT$
6:     draw $k, v, w$ from $I, U_k^+, U_k^-$ uniformly
7:     $\mathbf{q_k} \leftarrow \mathbf{q_k} - \eta * \frac{\partial}{\partial \mathbf{q_k}} MPR\text{-}OPT$
       $\mathbf{p_v} \leftarrow \mathbf{p_v} - \eta * \frac{\partial}{\partial \mathbf{p_v}} MPR\text{-}OPT$
       $\mathbf{p_w} \leftarrow \mathbf{p_w} - \eta * \frac{\partial}{\partial \mathbf{p_w}} MPR\text{-}OPT$
8: until convergence or maximum number of iterations
9: return P, Q
```

Algorithm 2 gives the pseudocode of the MPR algorithm. Each set of updates of the model parameters is done concurrently. Note that sampling in Line 6 is independent of that in Line 4. The sampling of items and users in Line 4 and Line 6 are done similarly to Line 4 of Algorithm 1 .

Let s be the number of times the loop in the Algorithm 2 is executed. The time complexity of MPR is $O(s(t|I|+|U|+r|U|+|I|)) =$

$O(s(|I| + |U|))$, where s and t are as defined for AR and $r \in (0, 1)$ is the proportion of new users. The three algorithms AR, BPR, and MPR have the same worst-case time and space bounds. However, the constant embedded in the time complexity of MPR might be slightly higher due to the extra updates to the model parameters in each iteration.

We have already provided the partial derivatives of $\hat{y}_{kvw}$ in Equations (31) to (33). The partial derivatives of $\hat{x}_{uij}$ are given below:

$$\frac{\partial}{\partial \mathbf{p_u}} \hat{x}_{uij} = \mathbf{q_i} - \mathbf{q_j} \tag{45}$$

$$\frac{\partial}{\partial \mathbf{q_i}} \hat{x}_{uij} = \mathbf{p_u} \tag{46}$$

$$\frac{\partial}{\partial \mathbf{q_j}} \hat{x}_{uij} = -\mathbf{p_u} \tag{47}$$

6 EXPERIMENTS

Our goal is to determine if MPR is able to compete on both tasks of item ranking and audience retrieval, while maintaining the same complexity as the other ranking algorithms.

6.1 Evaluation Metrics

Mean AUC was described earlier. We describe the other metrics used for evaluations here from an audience retrieval perspective. We leave out the description of metrics for item ranking because they are trivial to derive. Evaluations are done on both audience retrieval and item ranking. All Metrics are computed for top 10 except where stated e.g. P@5 is precision for top 5.

6.1.1 Recall. Recall at truncation level k, $R@k$ is defined as:

$$R@k = \sum_{k=1}^{r} \frac{rel_k}{r} \tag{48}$$

where r is the total number of relevant users and rel_k is the binary relevance label of the k-th user.

6.1.2 Mean Average Precision. MAP is a measure of the precision of users at every rank. MAP over all users for all items is evaluated as:

$$MAP = \frac{1}{|S_{test}|} \sum_{i \in S_{test}} \frac{1}{n} \sum_{i=1}^{n} rel_i \sum_{k=1}^{i} \frac{rel_k}{k} \tag{49}$$

where $\sum_{k=1}^{i} \frac{rel_k}{k}$ is the precision at $k, P@k$.

6.1.3 Mean Normalized Discounted Cumulative Gain. NDCG is a measure of the overall relevance of users based on rank. Correctly ranked users at the top are highly regarded and vice versa. NDCG over all users for all items is evaluated as:

$$NDCG = \frac{1}{|S_{test}|} \sum_{i \in S_{test}} \frac{DCG}{IDCG} \tag{50}$$

where $IDCG$ is the ideal ranking DCG and

$$DCG@K = \sum_{i=1}^{K} \frac{2^{rel_i} - 1}{ln(i+1)} \tag{51}$$

where rel_i is the relevance label of the user ranked i-th.

Table 1: Datasets

Dataset	#Users	#Items	#Ratings
Yahoo	7,642	11,915	211,231
MovieLens	6,040	3,706	1,000,209
Eachmovie	72,916	1,628	2,811,983

6.1.4 Mean Reciprocal Rank. MRR is a measure of how high the topmost relevant user is ranked. MRR over all users for all items is evaluated as:

$$MRR = \frac{1}{|S_{test}|} \sum_{i \in S_{test}} \frac{1}{rank_i} \tag{52}$$

where $rank_i$ is the rank of the highest ranked relevant user.

6.2 Comparison Baselines

We compare MPR with AR and three other algorithms. The first is a simple non-personalized baseline algorithm that ranks users and items by popularity called MostPop i.e. ranking is done by item popularity for item ranking and ranking is done by user popularity for audience retrieval. We consider another item ranking algorithm referred to as BPR [21] to check the effectiveness of a model optimized for item ranking on item ranking and audience retrieval. The last baseline is the state-of-the-art algorithm referred to as element-wise Alternating Least Squares (eALS) [10], which optimizes for ratings prediction.

6.3 Experimental Repeatability

We perform the experiments with a publicly available recommendation engines framework called LibRec[1] [7]. LibRec contains implementations of many state-of-the-art recommendation algorithms. We implemented our contributions in LibRec and the artifacts (software, datasets, and manuscripts) for our experiments in this work are publicly available at [2]. We evaluate the proposed methods against the baselines with 5-fold Leave-One-Out-Cross-Validation. We experimented with different values of model hyperparameters such as the learning rate and regularization parameters till we obtained the best performance for all algorithms. We set α to 0.5 for equally weighted sum for our experiments.

6.4 Datasets

All three datasets used for the experiments are available in the public domain. A summary of these datasets can be found in Table 1. We further describe them below. All the datasets contain explicit ratings but we set all positive ratings to 1 as the only valid class for implicit feedback. Implicit feedback in this work is used for both item ranking and audience retrieval. All the datasets contain user identifiers, item identifiers, and ratings.

6.4.1 Yahoo! Movies User Ratings Dataset. This dataset was retrieved from the Yahoo! Webscope Program released for validation of recommender systems. It contains the Yahoo! Movies community's preferences for different movies. The movies are rated on a scale from A+ to F. All users in this datasets have rated at least 10

[1]https://www.librec.net
[2]https://goo.gl/HhqQHd

Table 2: Yahoo Results on Item Ranking

Metrics	AR	MPR	BPR	MostPop	eALS
P@5	0.089	**0.170**	0.144	0.127	**0.177**
P@10	0.076	**0.133**	0.120	0.095	**0.136**
R@5	0.100	0.209	0.167	0.165	**0.223**
R@10	0.169	0.317	0.277	0.247	**0.329**
AUC	0.891	**0.965**	**0.965**	0.950	0.936
MAP	0.104	**0.210**	0.174	0.164	**0.220**
NDCG	0.320	**0.443**	0.410	0.388	**0.444**
MRR	0.213	**0.380**	0.322	0.319	**0.389**
#Wins	0	3.0	0.5	0	**4.5**

Table 3: Yahoo Results on Audience Retrieval

Metrics	AR	MPR	BPR	MostPop	eALS
P@5	**0.101**	**0.101**	0.064	0.065	0.082
P@10	**0.075**	**0.076**	0.049	0.045	0.061
R@5	**0.155**	**0.162**	0.076	0.100	0.104
R@10	**0.212**	**0.218**	0.104	0.129	0.140
AUC	**0.908**	**0.914**	0.869	0.795	0.773
MAP	**0.135**	**0.142**	0.077	0.071	0.093
NDCG	**0.326**	**0.334**	0.256	0.248	0.270
MRR	**0.252**	**0.256**	0.163	0.166	0.202
#Wins	**4**	**4**	0	0	0

movies and all movies are rated by at least one user. We refer to this dataset simply as Yahoo.

6.4.2 Movielens Dataset. MovieLens dataset [8] is a publicly available dataset used for research in different fields such as education, research, and industry. The dataset was obtained from the MovieLens website (http://movielens.org), a website run by the GroupLens Research lab at the University of Minnesota. MovieLens runs a free personalized movie recommendation engine based on CF for recommending movies to users. We denote this dataset by MovieLens for easy reference.

6.4.3 EachMovie Dataset. This dataset has been used in many research works. It was obtained from the EachMovie collaborative filtering service created for experimental purposes as part of a research project at the Digital Equipment Corporation (DEC) Systems Research Center. We refer to this dataset as EachMovie for convenience.

7 RESULTS

We describe our observations from the results in Tables 2 to 7. The winning algorithm per metric is emboldened in each row of all tables. We assume a margin of error of 0.005, hence the winning algorithm has to be greater than the next winner by at least a margin of 0.01. #Wins represents the number of wins per algorithm. In the case of a tie between k algorithms, each algorithm is assigned a Wins of $1/k$. We split observations on the results per task below.

Table 4: MovieLens Results on Item Ranking

Metrics	AR	MPR	BPR	MostPop	eALS
P@5	0.140	**0.396**	**0.386**	0.211	**0.393**
P@10	0.128	**0.345**	**0.338**	0.182	**0.343**
R@5	0.024	**0.100**	**0.095**	0.041	**0.098**
R@10	0.044	**0.163**	**0.157**	0.067	**0.161**
AUC	0.831	**0.937**	**0.935**	0.858	0.927
MAP	0.079	**0.240**	**0.230**	0.102	**0.241**
NDCG	0.427	**0.603**	**0.594**	0.459	**0.602**
MRR	0.283	**0.626**	0.612	0.391	**0.625**
#Wins	0	**3**	2.5	0	2.5

Table 5: MovieLens Results on Audience Retrieval

Metrics	AR	MPR	BPR	MostPop	eALS
P@5	**0.364**	**0.359**	0.157	0.202	**0.362**
P@10	**0.321**	**0.316**	0.138	0.177	**0.318**
R@5	**0.061**	**0.066**	0.021	0.033	**0.059**
R@10	**0.099**	**0.104**	0.035	0.050	**0.095**
AUC	**0.917**	**0.920**	0.804	0.815	0.907
MAP	**0.181**	**0.188**	0.069	0.085	**0.183**
NDCG	**0.553**	**0.559**	0.424	0.448	**0.552**
MRR	**0.574**	**0.577**	0.320	0.406	**0.572**
#Wins	**2.8**	**2.8**	0	0	2.3

Table 6: EachMovie Results on Item Ranking

Metrics	AR	MPR	BPR	MostPop	eALS
P@5	0.230	**0.405**	0.392	0.227	0.369
P@10	0.183	**0.321**	**0.313**	0.172	0.301
R@5	0.227	**0.362**	0.348	0.202	0.343
R@10	0.317	**0.506**	0.491	0.265	0.487
AUC	0.900	**0.980**	**0.980**	0.944	0.963
MAP	0.278	**0.445**	0.427	0.259	0.419
NDCG	0.531	**0.682**	0.668	0.522	0.657
MRR	0.444	**0.688**	0.666	0.457	0.632
#Wins	0	7	1	0	0

Table 7: EachMovie Results on Audience Retrieval

Metrics	AR	MPR	BPR	MostPop	eALS
P@5	0.480	**0.564**	0.159	0.317	0.525
P@10	0.421	**0.483**	0.160	0.235	0.438
R@5	**0.085**	**0.093**	0.005	0.048	0.078
R@10	**0.119**	**0.128**	0.012	0.057	0.104
AUC	**0.972**	**0.973**	0.905	0.897	0.963
MAP	0.223	**0.255**	0.079	0.097	0.222
NDCG	0.655	**0.683**	0.480	0.524	0.650
MRR	0.716	**0.807**	0.274	0.671	0.782
#Wins	1.5	**6.5**	0	0	0

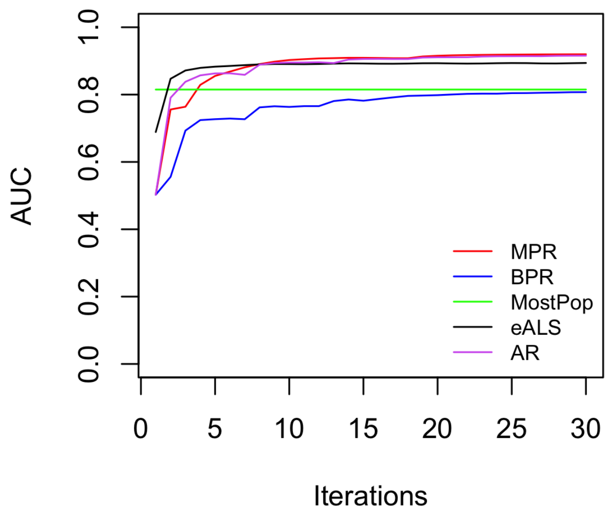

Figure 2: Convergence of all algorithms to the optimum AUC on the MovieLens dataset for audience retrieval

7.1 Item Ranking

eALS, followed closely by MPR, outperforms the other algorithms on Yahoo on all metrics except AUC where BPR competes more with MPR. BPR comes in third place on Yahoo. MPR wins on MovieLens, followed closely by BPR and eALS on all metrics. MPR clearly outperforms the other techniques on EachMovie, while BPR comes in second. MostPop and AR perform the worst on all datasets.

7.2 Audience Retrieval

AR ties with MPR on Yahoo. AR ties with MPR on MovieLens with eALS coming closely after them. MPR is the clear winner on EachMovie. AR consistently scores high on AUC across all the datasets. BPR and MostPop performed the worst across datasets.

7.3 Significance test

Our null hypothesis is that MPR is no better than any of the other algorithms. We used the Wilcoxon signed rank test to compare all algorithms with MPR with a p-value = 0.01. We account for the correlation of the metrics used by controlling for false discovery rate (FDR) [2] with $q^* = 0.01$. All algorithms in comparison with MPR have p-value < 0.00001, so we are able to reject the null hypothesis even with the control for FDR.

8 DISCUSSION

The results in Section 7 show that MPR performs the best overall on both tasks. It clearly outperforms all the other algorithms on EachMovie, the largest of the datasets. The experiments also show that a simple equally weighted sum can be an effective multi-objective optimization technique. MPR competes well on the metrics where it does not outperform the other algorithms. We initially set out to check if MPR would compete favorably with the item ranking algorithms on the item ranking task and also with the audience retrieval algorithms on the audience retrieval task. However, MPR exceeded that expectation by performing better on item ranking and audience retrieval than the respective item ranking and audience retrieval algorithms on most of the datasets. We believe MPR performs so well on both tasks because it is optimized for both tasks. For either task, MPR is able to draw on extra latent information that other algorithms could not.

On the AUC metric, we make a few observations across datasets. MPR consistently performs well even when it is outperformed on the other metrics. BPR consistently performs well on the task of item ranking, while AR performs consistently well on the task of audience retrieval. This can be explained by the optimization of a smooth approximation to AUC in BPR, MPR, and AR. MPR minimizes AUC loss for all tasks. BPR maximizes AUC gain for item ranking while AR maximizes AUC gain for audience retrieval. As expected, BPR performs well on item ranking because it is optimized for it, but does poorly on audience retrieval since it is not optimized for it. Similarly, AR performs well on audience retrieval because it is optimized for it but does poorly on item ranking as it not optimized for it.

eALS performs very well on item ranking for the Yahoo dataset and decently overall. This can be explained by the fact that eALS is a rating prediction technique and it is not optimized for any metric in particular nor for either task. We are not too surprised by the lackluster performance of MostPop as it is a naive approach with no personalization. However, MostPop performed better than BPR on audience retrieval in most instances. MostPop also performed better than AR on item ranking in some instances.

In order to demonstrate the convergence of algorithms, we include Figure 2 here to show how each algorithm converges with respect to AUC on the MovieLens dataset. MostPop has no convergence rate since there is no training at all. eALS converges quickly but MPR and AR continue to converge to a higher AUC. The faster convergence of eALS is not surprising as this was already demonstrated by its creators in [10].

9 CONCLUSION

We have presented a multi-objective pairwise ranking (MPR) algorithm that performs simultaneous item ranking and audience retrieval without compromising efficiency. The experiments on different popular datasets and metrics show the effectiveness of MPR.

We noted that MPR's margin of victory on the EachMovie dataset is so large that we would like to explore the reasons for it on this dataset in a future work. We would like to investigate the performance of other multi-objective optimization algorithms such as multi-gradient descent algorithm (MGDA) and the *Lagrangian* multiplier method. We also plan to extend our study to more datasets. Although we use the audience retrieval algorithm described earlier and its item ranking equivalent in this work for demonstration, MPR is an extensible framework that can accommodate the combination of different item ranking and audience retrieval techniques. It would be interesting to explore different weights for the item ranking objective function and audience retrieval techniques in future work. We believe MPR is also adaptable to online processing and parallelization similar to its constituent ranking algorithms. We would like to perform experiments to show these.

REFERENCES

[1] S. Balakrishnan and S. Chopra. Collaborative ranking. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 143–152, New York, NY, USA, 2012. ACM.

[2] Y. Benjamini and Y. Hochberg. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.

[3] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM.

[4] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, Jan. 2004.

[5] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[6] T. Graepel, J. Q. n. Candela, T. Borchert, and R. Herbrich. Web-Scale Bayesian Click-Through rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine. In *Proceedings of the 27th International Conference on Machine Learning*, pages 13–20, Haifa, 2010.

[7] G. Guo, J. Zhang, Z. Sun, and N. Yorke-Smith. Librec: A java library for recommender systems. In *Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd Conference on User Modeling, Adaptation, and Personalization (UMAP 2015), Dublin, Ireland, June 29 - July 3, 2015.*, 2015.

[8] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, Dec. 2015.

[9] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, New York, NY, USA, 2017. ACM.

[10] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. 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*, SIGIR '16, pages 549–558, New York, NY, USA, 2016. ACM.

[11] 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, Washington, DC, USA, 2008. IEEE Computer Society.

[12] S. Huang, S. Wang, T.-Y. Liu, J. Ma, Z. Chen, and J. Veijalainen. Listwise collaborative filtering. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 343–352, New York, NY, USA, 2015. ACM.

[13] S. Kabbur, X. Ning, and G. Karypis. Fism: Factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 659–667, New York, NY, USA, 2013. ACM.

[14] Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 426–434, New York, NY, USA, 2008. ACM.

[15] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.

[16] H. Liu, D. Pardoe, K. Liu, M. Thakur, F. Cao, and C. Li. Audience expansion for online social network advertising. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 165–174, New York, NY, USA, 2016. ACM.

[17] Y. Liu, S. Pandey, D. Agarwal, and V. Josifovski. Finding the right consumer: Optimizing for conversion in display advertising campaigns. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 473–482, New York, NY, USA, 2012. ACM.

[18] R. T. Marler and J. S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41(6):853–862, 2010.

[19] 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, Dec 2011.

[20] S. Pandey, M. Aly, A. Bagherjeiran, A. Hatch, P. Ciccolo, A. Ratnaparkhi, and M. Zinkevich. Learning to target: What works for behavioral targeting. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 1805–1814, New York, NY, USA, 2011. ACM.

[21] 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.

[22] S. Rendle and L. Schmidt-Thieme. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 251–258, New York, NY, USA, 2008. ACM.

[23] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 713–719, New York, NY, USA, 2005. ACM.

[24] G. Robinson. Automated collaborative filtering in world wide web advertising, July 24 1997. WO Patent App. PCT/US1996/020,429.

[25] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM.

[26] Y. Shi, M. Larson, and A. Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 269–272, New York, NY, USA, 2010. ACM.

[27] J. van Rijn and J. Braams. National client email report, May 2015.

[28] M. Volkovs and R. S. Zemel. Collaborative ranking with 17 parameters. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2294–2302. Curran Associates, Inc., 2012.

[29] M. Weimer, A. Karatzoglou, Q. V. Le, and A. J. Smola. Cofi rank - maximum margin matrix factorization for collaborative ranking. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1593–1600, Cambridge, MA, 2007. MIT Press.

[30] S. Xu, S. Wu, and L. Wang. Personalized semantic ranking for collaborative recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 971–974, New York, NY, USA, 2015. ACM.

[31] J. Yuan, B. Kanagal, V. Josifovski, L. Garcia-Pueyo, A. Ahmed, and S. Pandey. Focused matrix factorization for audience selection in display advertising. In *Proceedings of the 2013 IEEE International Conference on Data Engineering (ICDE 2013)*, ICDE '13, pages 386–397, Washington, DC, USA, 2013. IEEE Computer Society.