Recommendation with Capacity Constraints

Konstantina Christakopoulou
University of Minnesota
christa@cs.umn.edu

Jaya Kawale
Netflix
kawale@cs.umn.edu

Arindam Banerjee
University of Minnesota
banerjee@cs.umn.edu

ABSTRACT

In many recommendation settings, the candidate items for recommendation are associated with a maximum capacity, i.e., number of seats in a Point-of-Interest (POI) or number of item copies in the inventory. However, despite the prevalence of the capacity constraint in the recommendation process, the existing recommendation methods are not designed to optimize for respecting such a constraint. Towards closing this gap, we propose *Recommendation with Capacity Constraints* – a framework that optimizes for both recommendation accuracy and expected item usage that respects the capacity constraints. We show how to apply our method to three state-of-the-art latent factor recommendation models: probabilistic matrix factorization (PMF), bayesian personalized ranking (BPR) for item recommendation, and geographical matrix factorization (GeoMF) for POI recommendation. Our experiments indicate that our framework is effective for providing good recommendations while taking the limited resources into consideration. Interestingly, our methods are shown in some cases to further improve the top-N recommendation quality of the respective unconstrained models.

1 INTRODUCTION

Consider what would happen if a Point-of-Interest (POI) recommendation system suggests to a large number of users to visit the same POI, e.g. the same attraction in a theme park, or the same coffee shop; or what would be the effect of an item recommendation system recommending the same products (e.g. movies, jackets) to the vast majority of customers. It is easy to imagine that in the first case the recommended POI might get overcrowded, resulting in long queues, thus high waiting times. In the second case, the customers might view an 'out of stock' or 'server overload' message. In either scenario, the user experience will be deteriorated.

The above scenarios, although seemingly different, share the following key property: every item candidate for recommendation is associated with a maximum *capacity* – for a POI it could be the number of visitors allowed at the same time or the number of seats or tables; for a product it could be the maximum number of copies that can be purchased/consumed simultaneously.

As recommendation systems become more prevalent and touch more aspects of everyday life, there is an increase in potential applications that require providing recommendations while respecting the capacity constraints for the items. A few interesting examples that can be expressed in this setting are: (1) book recommendation systems employed by libraries, where the books recommended to the borrowers should be on the shelf, (2) route recommendation systems, which aim to suggest the best road for driving while keeping the roads from getting congested, (3) class recommendation systems employed by universities, where every recommended class can accept a limited number of students, (4) viral content recommenders, which serve personalized content in rush watching periods such as Prime Time or the Oscars, and many more.

To the best of our knowledge, none of the state-of-the-art recommendation approaches for item and POI recommendation is designed to respect the capacity constraints. Instead, they are often designed to optimize for rating prediction [32, 34] or personalized ranking accuracy [7, 30, 35], or other metrics such as serendipity [10], novelty [14], and diversity [15].

We propose a novel approach that both optimizes for recommendation accuracy, measured suitably by rating prediction or personalized ranking losses, and penalizes excessive usage of items that surpasses the corresponding capacities. We show how to apply our approach to three state-of-the-art latent factor recommender models: probabilistic matrix factorization (PMF) [32], geographical matrix factorization (GeoMF) [24], and bayesian personalized ranking (BPR) [30]. We introduce the concepts of *user propensity* (i.e., the probability to follow the recommendations) and *item capacity*. Both concepts are key factors for estimating the extent to which the expected usage of the items exceeds the capacities.

Our experimental results in real-world recommendation datasets show that our formulation (i) is suitable for different choices of propensities, capacities, and surrogate loss for the capacity objective, and (ii) allows for recommendations which respect the capacity constraints, while the recommendation quality is not deteriorated by a lot; in fact, in some cases our methods outperform the state-of-the-art in top-N recommendation quality.

Our contributions are four-fold: (1) We formulate the problem of providing recommendations with capacity constraints, and we extend the state-of-the-art in item and POI latent-factor recommendation, to respect the item capacities. (2) We introduce the concept of user propensity to follow the recommendation. (3) We propose a set of strategies for estimating item capacities and user propensities from data with no such information. (4) We show that adding the capacity loss to the recommendation objective can sometimes further improve the top-N recommendation quality, indicating the promise of our approach.

The rest of the paper is organized as follows. We review related work in Section 2. In Section 3 we introduce the concepts of user propensities and item capacities. In Section 4 we devise our methods for recommendation with capacity constraints. We empirically evaluate our method in Section 5, and conclude in Section 6.

Notation. Let N be the total number of items, and M the total number of users. Every user is denoted by an index $i = 1, \ldots, M$

CIKM'17, November 6–10, 2017, Singapore.
 ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: https://doi.org/10.1145/3132847.3133034

and every item by $j = 1, \ldots, N$. Assume that user i has rated a subset of the N items, denoted by $\mathcal{L}_i$, and every item has been rated by the set of users Ra(j).

2 RELATED WORK

Recommendation Models. Matrix factorization (MF) based approaches have become very popular in recommendation systems [18] both for implicit [13, 16] and explicit feedback [32]. They predict a user's rating on an item on the basis of low dimensional latent factors, by optimizing square loss [32, 34] or ranking inspired losses [7, 30, 35]. MF approaches have also been employed in the rapidly grown area of POI recommendation where geo-location is used to further improve recommendation performance [9, 21, 23, 24] – we refer the reader to [39] for a survey of the topic.

Multi Objective Optimization (MOO). Given that we want to maximize for recommendation performance, while the expected usage should respect the capacity constraints, it is natural that our solution would require optimizing multiple objectives. MOO has been very effective for various business motivated purposes, i.e., optimizing for multiple types of feedback [37], for click shaping [4], for homepage relevance [2]. Also, MOO has been employed in recommendation systems for optimizing several criteria [1, 19, 31], such as various ranking metrics [36], diversity [15], novelty [14], time spent along with click-through-rate [5], and more.

Resource constraints. Although we could not trace literature handling the problem of *recommendation with capacity constraints*, there have been works using the notion of constrained resources or budget [12]. In what follows, we present how each of these settings presents unique challenges, and differs from our problem.[1]

Email Volume. One related constraint which has been recently explored by Gupta et al., and posed as a MOO [11], is deciding which emails to send so to minimize the total number of emails, while the number of downstream sessions is maximized, and the number of spam reporting and unsubscribe actions is minimized. While email volume can be seen as a capacity constraint, our work is rather different: In our setting, the end goal is to provide *personalized recommendations* which respect the capacity constraints, whereas in [11] a non-personalized email allocation scheme is proposed.

Budget. The problems of sponsored search [6], advertisement display [20, 27], auction [22, 38] etc., are naturally associated with the notion of a budget. For instance, for ad displays, the algorithm decides which ads to show so that the number of clicks or revenue is maximized, while the advertisers stay in their specified budget. Typically, such problems have been formulated using online matching [27], or bandits [22, 38]. Our setting departs from the above problem in a number of ways: (i) Items have a budget, instead of users having a budget. (ii) We collaboratively allocate items to users in a unified way; in contrast, usually, ad serving systems have separate entities for ad bidding and for serving personalized ads to users.

Course Requirements. In [29] the authors propose novel methods for set recommendations of courses, so that the courses recommended to a student satisfy the specified degree requirements. In our work, instead of focusing on constraints within the list of items, we focus on satisfying the capacity constraint coupling recommendation lists across users as a whole.

Quantity constraints. Zhang et al. proposed to boost the Pareto efficiency of web allocations [41]. While they also provide personalized allocations modeling the web economy as a whole, their focus is rather different from ours; they maximize each user's individual surplus, while we focus on maximizing recommendation performance, as typically captured in recommender systems. Also, they require the product prices as input, which is not applicable in our case. Our work is the first to introduce the novel aspect of the tendency of users to follow the system's recommendations, giving different weights to the various users when estimating item usage, and to provide personalized allocations that also account for geographical influence.

Also, work on queuing theory models [17] is related, if we view our problem of respecting the capacity constraints as minimizing the waiting time of users in queues.

3 KEY CONCEPTS & PRELIMINARIES

Item Capacities. Every item $j = 1, \ldots, N$ is characterized by a parameter indicating the maximum number of users who can simultaneously use it. We refer to this variable as *capacity* and denote it with $c_j > 0$, resulting in a vector of capacities $\mathbf{c} \in \mathbb{R}_+^N$ for all N items. For POI recommendation, a POI's capacity could be the total number of seats, or number of visitors allowed per time slot. For general item recommendation, an item's capacity could be the maximum number of users that can watch the same movie online without leading to a system crash, or the maximum number of copies of the same item in the inventory.

Capacity is key for recommendation systems, as when many users are directed to the same item, the item will quickly reach its capacity. This will in turn lead to deteriorated user experience, such as long waiting times or out of stock items. This motivates the need for a recommendation system that respects the items' capacities.

User Propensities. Every user i is associated with a variable, indicating the probability that he will follow the system's recommendations. We refer to this variable as *user propensity* and denote it with $p_i \in [0, 1]$, resulting in a vector of propensities $\mathbf{p} \in \mathbb{R}^M$ for all M users. There are many ways propensities can be modeled; one possible definition is:

$$p_i = \frac{\text{\# times user } i \text{ followed the recommendation}}{\text{\# user } i\text{-system interactions}},$$

which uses feedback of the form user follows/ignores the recommendation. Alternative definitions are explored in Section 5.1.

Propensity to follow the recommendations can be an inherent user property: some users tend to listen to the system's recommendations more, compared to others. However, the same user's propensity might vary with time, e.g. the user might go to a certain place for lunch every day at 2pm, so no matter how good the restaurant recommendation is, he will not listen (low propensity); in contrast, he might be experimental during his dinner time (high propensity). Also, factors such as who the user is with or quality of experience with the system can affect the user's propensity. For such cases we plan to include dynamic propensity estimation in the future. We argue that user propensity, which has connections to

[1] An interesting future direction is to treat these quite different problems under a unified framework, such as welfare maximization.

the themes of [25, 33], is a key factor for recommendation systems. Though we use it to compute the expected usage of items, one can also use it to e.g. target the users with low/high propensities.

Here, we briefly review the methods of PMF [32], BPR [30], and GeoMF [24], as we will use them as underlying models for personalized recommendation in Section 4.

PMF. Let $U \in \mathbb{R}^{k \times M}$ be the latent factor corresponding to the users, where the i^{th} column $\mathbf{u}_i \in \mathbb{R}^k$ is the latent factor for user i. Similarly, let $V \in \mathbb{R}^{k \times N}$ be the latent factor for the items, where the j^{th} column $\mathbf{v}_j \in \mathbb{R}^k$ is the latent factor for item j. Then, PMF models the predicted rating of user i on item j as $\hat{r}_{ij} = \mathbf{u}_i^T \mathbf{v}_j$, so that the overall score matrix $R \in \mathbb{R}^{M \times N}$ is of rank $k \ll M, N$, and thus can be approximated by $U^T V$. The objective of PMF [32] is

$$\mathcal{E}_{\text{PMF}}(U, V) = \sum_{i=1}^{M} \sum_{j \in L_i} (r_{ij} - \mathbf{u}_i^T \mathbf{v}_j)^2 + \lambda(\|U\|_F^2 + \|V\|_F^2),$$

where the second term is a L-2 regularization term to prevent overfitting in the training data, with λ denoting the regularization parameter and $\|\cdot\|_F$ the Frobenius norm.

BPR. BPR [30] focuses on correctly ranking item pairs instead of scoring single items. Let $L_{i,+}, L_{i,-}$ denote the set of positively (+1) and negatively rated (-1) items by user i respectively. Maximizing the posterior distribution that a user will prefer the positive items over the negative ones, the MF-based BPR objective is:

$$\mathcal{E}_{\text{BPR}}(U, V) = \sum_{i=1}^{M} \sum_{k \in L_{i,+}} \sum_{j \in L_{i,-}} \log(1 + \exp(-\mathbf{u}_i^T (\mathbf{v}_k - \mathbf{v}_j))) + \lambda(\|U\|_F^2 + \|V\|_F^2.$$

GeoMF. In POI recommendation, the rating matrix R contains the check-in data of M users on N POIs. Because of the natural characterization of POIs in terms of their geographical location (latitude and longitude) and the spatial clustering in human mobility patterns, recommendations can be further improved [39]. This insight has led to the development of GeoMF [24] which jointly models the geographical and MF component. The key idea behind GeoMF, and the related works of [23, 39], is that if a user i has visited a POI j but has not visited the nearby POIs, these nearby POIs are more likely to be disliked by this user compared to far-away non-visited POIs. GeoMF represents the users and items not only by their latent factors U, V, but also by the activity and influence vectors X, Y respectively. By splitting the entire world into L even grids, GeoMF models a user i's *activity area* as a vector $\mathbf{x}_i \in \mathbb{R}^L$, where every entry $x_{i\ell}$ of the vector denotes the possibility that this user will appear in the ℓ-th grid. Similarly, the model represents every POI by a vector $\mathbf{y}_j \in \mathbb{R}^L$, referred to as *influence vector*, where every entry $y_{j\ell}$ indicates the quantity of influence POI j has on the ℓ-th grid. While the activity area vector $\mathbf{x}_i$ of every user i is latent, the influence vectors $\mathbf{y}_j$ for every POI j are given as input to the model and are pre-computed using kernel density estimation [40] as follows. The degree of the influence POI j has on the ℓ-th grid is computed as $y_{j\ell} = \frac{1}{\sigma}\mathcal{K}(\frac{d(j,\ell)}{\sigma})$ where $\mathcal{K}$ is the standard Gaussian distribution, σ is the standard deviation and $d(j, \ell)$ is the Euclidean distance between the POI j and the ℓ-th grid. Although in [24] the activity area vectors are constrained to be non-negative and sparse, in our work we do not make such an assumption. Concretely, GeoMF predicts user i's rating on POI j as $\hat{r}_{ij} = \mathbf{u}_i^T \mathbf{v}_j + \mathbf{x}_i^T \mathbf{y}_j$. GeoMF uses the point-wise square loss of PMF; however, we can also replace this with the BPR loss, giving rise to a method we refer to as GeoBPR.

4 RECOMMENDATION WITH CAPACITY CONSTRAINTS

While PMF, GeoMF, BPR and GeoBPR focus on accurately predicting the best items to be recommended to every user, they do not consider any capacity constraints that might be associated with the items. However, for POI (e.g. theme parks attractions or coffee shops) recommendation, if the system recommends POIs overlapping across the majority of users, the users who follow the recommendation might have to wait in long queues, although there might be perfectly good POIs empty just around the corner. This scenario and more (e.g. viral video recommendation, limited shelf on a virtual store, road congestion) can benefit from a system that gives recommendations while respecting the capacity constraints.

Our goal is that we want good recommendation accuracy, while the items' estimated usage does not exceed the respective capacities. To achieve this, we discuss two alternative approaches.

4.1 Approach 1: Post-Processing

The first simple method we discuss is referred to as *post-process*. This method takes as input the predicted scores $\hat{r}_{ij}$ for every user - item (i, j) pair. Although we use the scores provided by either of the discussed models of PMF, BPR, GeoMF, GeoBPR, any recommendation model can be used in place. The idea of this approach is that for every item j with capacity c_j, we find the c_j users with the top predicted scores, i.e., who are the users who want item j the most, up until it reaches its capacity? To provide top-N recommendations per user, keeping track of which users are allocated to which items, we sort the assigned items based on the predicted scores in descending order. Note that by construction, this approach achieves recommendations that respect the capacity constraints.

Algorithm 1 *post-process*

Require: $\forall j\ c_j$, $\forall (i, j)\ \hat{r}_{ij}$ from unconstrained method

1: For every item j, (i) rank users based on $\hat{r}_{ij}$, (ii) recommend item j only to top c_j users.
2: For every user i, rank the assigned items from Step 1, based on $\hat{r}_{ij}$, and **return** top-N recommendation list.

4.2 Approach 2: Proposed Framework

The second approach we introduce is formalizing the problem of recommendation with capacity constraints as a weighted sum between two objectives: (i) we want to both optimize for recommendation accuracy, while (ii) penalizing when the items' expected usage exceeds the respective capacities. To balance the two objectives we use a trade-off parameter $\alpha \in [0, 1]$. We will refer to our proposed approaches under this framework as *weighted objectives*.

Personalized Scoring. To provide personalized recommendations, we follow the representation used by PMF/BPR for item recommendation, and the one used by GeoMF/GeoBPR in POI recommendation.

Specifically, the predicted score of user i on item j is:

$$\hat{r}_{ij} = \begin{cases} \mathbf{u}_i^T \mathbf{v}_j \text{ for item recommendation} \\ \mathbf{u}_i^T \mathbf{v}_j + \mathbf{x}_i^T \mathbf{y}_j \text{ for POI recommendation} \end{cases} \quad (1)$$

Importantly, any model estimating whether a user i will buy/visit item/POI j can be used to replace Equation (1), thus leading to a family of recommendation with capacity constraints algorithms.

Expected Usage. We define the expected usage of an item j as the expected number of users who have been recommended item j and will follow the recommendation: $\sum_{i=1}^{M} p_i \hat{r}_{ij}$. If $\hat{r}_{ij}$ was either 1 or 0, the $\sum_{i=1}^{M} \hat{r}_{ij}$ term would indicate the total number of users who have been recommended item j. For ease of optimization, we do not threshold $\hat{r}_{ij}$ to be either 0 or 1. Instead, we constrain $\hat{r}_{ij}$ in the range of $[0, 1]$ by using the sigmoid function, $\sigma(\cdot) = \frac{1}{1+\exp(-(\cdot))}$:

$$\mathbf{E}[\text{usage}(j)] = \sum_{i=1}^{M} p_i \sigma(\hat{r}_{ij}) \,, \quad (2)$$

which is the weighted combination of the estimated ratings for the users, using as weights the corresponding user propensities.

Capacity Loss. Since we wish to penalize the model for giving recommendations which result in the expected usage of the items exceeding the corresponding capacities, we want to minimize the average capacity loss, given by:

$$\frac{1}{N} \sum_{j=1}^{N} \mathbf{1}[c_j \leq \mathbf{E}[\text{usage}(j)]] \,. \quad (3)$$

Given that the use of the indicator function $\mathbf{1}[\cdot]$ is not suitable for optimization purposes, we will use a surrogate for the indicator function. Considering the difference

$$\Delta(c_j, \mathbf{E}[\text{usage}(j)]) = c_j - \mathbf{E}[\text{usage}(j)], \quad (4)$$

we use the logistic loss of the difference as the surrogate:

$$\ell(\Delta(c_j, \mathbf{E}[\text{usage}(j)])) = \log(1 + \exp(-\Delta(c_j, \mathbf{E}[\text{usage}(j)])) \,, \quad (5)$$

noting that it forms a convex upper bound to the indicator function. Alternative surrogate losses to consider are:

$$\ell(\Delta(c_j, \mathbf{E}[\text{usage}(j)])) = \begin{cases} \exp(-\Delta) \;\; \text{(Exponential loss)} \\ \max(-\Delta, 0) \;\; \text{(Hinge Loss)} \end{cases} \quad (6)$$

Although the square loss $(-\Delta)^2$ is a convex surrogate of the indicator as well, it is not a suitable surrogate for the capacity loss, as it penalizes both positive and negative differences, whereas we want to penalize only when the expected usage *exceeds* the capacity.

Overall Objective. Depending on which loss to minimize in order to capture recommendation quality, i.e., rating prediction loss or pairwise ranking loss, we devise the following four methods: `Cap-PMF`, `Cap-BPR` for item recommendation, and `Cap-GeoMF`, `Cap-GeoBPR` for POI recommendation.

Putting everything together, using the logistic loss as the surrogate loss, our method minimizes the following objective for item recommendation (`Cap-PMF`):

$$\mathcal{E}_{\texttt{Cap-PMF}}(U, V) = (1-\alpha) \cdot \sum_{i=1}^{M} \sum_{j \in L_i} (r_{ij} - \mathbf{u}_i^T \mathbf{v}_j)^2 \quad (7)$$
$$+ \alpha \cdot \frac{1}{N} \sum_{j=1}^{N} \log \left(1 + \exp \left(\sum_{i=1}^{M} p_i \sigma(\mathbf{u}_i^T \mathbf{v}_j) - c_j \right) \right) + \lambda(\|U\|_F^2 + \|V\|_F^2) \,,$$

while for POI recommendation our method `Cap-GeoMF` minimizes:

$$\mathcal{E}_{\texttt{Cap-GeoMF}}(U, V, X) = (1-\alpha) \cdot \sum_{i=1}^{M} \sum_{j \in L_i} (r_{ij} - \mathbf{u}_i^T \mathbf{v}_j - \mathbf{x}_i^T \mathbf{y}_j)^2, \quad (8)$$
$$+ \alpha \cdot \frac{1}{N} \sum_{j=1}^{N} \log \left(1 + \exp \left(\sum_{i=1}^{M} p_i (\sigma(\mathbf{u}_i^T \mathbf{v}_j + \mathbf{x}_i^T \mathbf{y}_j)) - c_j \right) \right)$$
$$+ \lambda(\|U\|_F^2 + \|V\|_F^2 + \|X\|_F^2) \,.$$

In the above formulation, α is a fixed, user-chosen parameter in $[0, 1]$ that handles the trade-off between the prediction loss and the *capacity loss*. When α is equal to 0, our model reduces to `PMF` (or `GeoMF`). When α is 1, we are not interested in good prediction accuracy – our only concern is to ensure that every item's expected usage will not exceed its fixed capacity.

If we replace the first objective in (7) (or (8)) with the one of BPR to optimize for ranking accuracy, we obtain `Cap-BPR` (or `Cap-GeoBPR`). For `Cap-BPR` we optimize for:

$$\mathcal{E}_{\texttt{Cap-BPR}}(U, V) = (1-\alpha) \cdot \sum_{i=1}^{M} \sum_{k \in L_{i,+}} \sum_{j \in L_{i,-}} \log(1 + \exp(-\mathbf{u}_i^T (\mathbf{v}_k - \mathbf{v}_j))) \quad (9)$$
$$+ \alpha \cdot \frac{1}{N} \sum_{j=1}^{N} \log \left(1 + \exp \left(\sum_{i=1}^{M} p_i \sigma(\mathbf{u}_i^T \mathbf{v}_j) - c_j \right) \right) + \lambda(\|U\|_F^2 + \|V\|_F^2) \,,$$

Learning the model. Following, due to space constraints, we give the gradient updates only for `Cap-PMF` and `Cap-GeoMF`. The optimization for (7) or (8) is done by alternating minimization. The latent factor updates are done using gradient descent, and for iteration $t+1$ they are:

$$\forall i = 1, \ldots, M, \; \mathbf{u}_i^{t+1} \leftarrow \mathbf{u}_i^t - \eta \nabla_{\mathbf{u}_i} \mathcal{E}_{\texttt{capMF}}(U^t, V^t, X^t)$$
$$\forall j = 1, \ldots, N, \; \mathbf{v}_j^{t+1} \leftarrow \mathbf{v}_j^t - \eta \nabla_{\mathbf{v}_j} \mathcal{E}_{\texttt{capMF}}(U^{t+1}, V^t, X^t)$$
$$\forall i = 1, \ldots, M, \; \mathbf{x}_i^{t+1} \leftarrow \mathbf{x}_i^t - \eta \nabla_{\mathbf{x}_i} \mathcal{E}_{\texttt{Cap-GeoMF}}(U^{t+1}, V^{t+1}, X^t)$$

where the last gradient update is valid only for `Cap-GeoMF`. We denote with $\mathcal{E}_{\texttt{capMF}}$ either $\mathcal{E}_{\texttt{Cap-GeoMF}}$ or $\mathcal{E}_{\texttt{Cap-PMF}}$. The gradients can be obtained by a direct application of the chain rule. From Equations (1), (2), (4) and (5), we get the gradients

$$\nabla_{\Delta} \ell(\Delta(c_j, \mathbf{E}[\text{usage}(j)])) = -\sigma(-\Delta(c_j, \mathbf{E}[\text{usage}(j)])) \quad (10)$$
$$\nabla_{\mathbf{u}_i} \Delta(c_j, \mathbf{E}[\text{usage}(j)]) = -p_i \mathbf{v}_j \sigma(\hat{r}_{ij}) \sigma(-\hat{r}_{ij}) \quad (11)$$
$$\nabla_{\mathbf{v}_j} \Delta(c_j, \mathbf{E}[\text{usage}(j)]) = -\sum_i p_i \mathbf{u}_i \sigma(\hat{r}_{ij}) \sigma(-\hat{r}_{ij}) \quad (12)$$
$$\nabla_{\mathbf{x}_i} \Delta(c_j, \mathbf{E}[\text{usage}(j)]) = -p_i \mathbf{y}_j \sigma(\hat{r}_{ij}) \sigma(-\hat{r}_{ij}) \,, \quad (13)$$

where for (11), (12), (13) we used the property $\nabla_x \sigma(x) = \sigma(x) \cdot \sigma(-x)$. Using (10)-(13), we obtain the gradient of the latent parameters as follows. The gradient of the objective, w.r.t $\mathbf{u}_i$ is

$$\nabla_{\mathbf{u}_i} \mathcal{E}_{\texttt{capMF}} = -(1-\alpha) \sum_{j \in L_i} 2(r_{ij} - \mathbf{u}_i^T \mathbf{v}_j) \mathbf{v}_j + 2\lambda \mathbf{u}_i +$$
$$+ \frac{\alpha}{N} \sum_{j=1}^{N} \sigma(-\Delta(c_j, \mathbf{E}[\text{usage}(j)])) \cdot p_i \mathbf{v}_j \sigma(\hat{r}_{ij}) \sigma(-\hat{r}_{ij}) \,.$$

Similarly, the gradient of the objective, w.r.t $\mathbf{v}_j$ is

$$\nabla_{\mathbf{v}_j}\mathcal{E}_{\texttt{capMF}} = -(1-\alpha)\sum_{i\in \text{Ra}(j)} 2(r_{ij}-\mathbf{u}_i^T\mathbf{v}_j)\mathbf{u}_i + 2\lambda\mathbf{v}_j + \\ + \frac{\alpha}{N}\sigma(-\Delta(c_j,\mathbf{E}[\text{usage}(j)]))\cdot\sum_{i=1}^{M} p_i\mathbf{u}_i\sigma(\hat{r}_{ij})\sigma(-\hat{r}_{ij})\ .$$

For cap-GeoMF, the gradient of the objective, w.r.t $\mathbf{x}_i$ is

$$\nabla_{\mathbf{x}_i}\mathcal{E}_{\texttt{cap-GeoMF}} = -(1-\alpha)\sum_{j\in L_i} 2(r_{ij}-\mathbf{u}_i^T\mathbf{v}_j-\mathbf{x}_i^T\mathbf{y}_j)\mathbf{y}_j \\ + 2\lambda\mathbf{x}_i + \frac{\alpha}{N}\sum_{j=1}^{N}\sigma(-\Delta(c_j,\mathbf{E}[\text{usage}(j)]))\cdot p_i\mathbf{y}_j\sigma(\hat{r}_{ij})\sigma(-\hat{r}_{ij})\ .$$

We have demonstrated here our method in the batch setting. In the future, we will consider the online or bandit setting [3, 26].

5 EXPERIMENTAL RESULTS

We present empirical results to address the following questions:

(1) What is the interplay of rating prediction and capacity loss for Cap-PMF, Cap-GeoMF as we vary the trade-off α (Section 5.2.1)? Similarly for pairwise ranking loss versus capacity loss for Cap-BPR, Cap-GeoBPR (Section 5.2.2)?
(2) How do the proposed approaches Cap-PMF, Cap-GeoMF, Cap-BPR, Cap-GeoBPR compare with their state-of-the-art unconstrained counterparts? (Section 5.3, Section 5.4)
(3) How do *post-process* and *weighted objectives* solutions compare in terms of top-N recommendation? (Section 5.4)
(4) How robust is our framework to different choices, i.e., surrogate loss for capacity loss (Section 5.5.1), propensity choices (Section 5.5.2), capacity choices (Section 5.5.3), implicit vs. explicit data (Section 5.5.4)?

5.1 Experimental Setting

Data. For item recommendation, we considered two public real-world datasets containing users' ratings on movies: Movielens 100K and Movielens 1M[2]. For POI recommendation, we used two public real-world datasets containing user check-ins in POIs: Gowalla and Foursquare[3]. For the Foursquare and Gowalla datasets, we removed the users and POIs with less than or equal to 10 ratings. The users, items and ratings statistics of the data are found in Table 1.

Foursquare and Gowalla contain implicit check-in data (only ratings of 1 and 0 are present). A rating of 1 denotes that the user has visited the POI while a 0 denotes that the user has not visited it; because the user potentially does not like the POI or does not know about it. Movielens 100K and Movielens 1M contain ratings in a multi-relevance scale of 1 to 5 stars. We experimented with two feedback setups for the Movielens datasets: (i) two-scale explicit feedback (+1, -1), using 4 as the threshold of liking, and (ii) implicit feedback, marking every non-zero rating as a 1, and keeping the rest as 0.

Evaluation Setup. We repeated all experiments five times by drawing new train/test splits in each round. We randomly selected half of the ratings for each user in the training set and moved the rest of the observations to the test set. This scheme was chosen to simulate the real recommendation setup, where there exist users with a few ratings as well as users with many ratings.

Dataset	# Users	# Items	# Ratings
Movielens 100K	943	1,682	100,000
Movielens 1M	6,040	3,706	1,000,209
Foursquare	2,025	2,759	85,988
Gowalla	7,104	8,707	195,722

Table 1: Dataset Statistics

For the implicit feedback datasets, as only positive observations are available, we introduced negative observations (disliked items) in the training set as follows. For every user with N_i^{train} positive observations (+1), we sampled N_i^{train} items as negatives (-1) from the set of items marked as 0 both for train and test data. This is common practice to avoid skewed predictions resulting from training on only positive observations, and to reduce the computational overhead of having all unrated items as negative observations. (Several negative sampling strategies are discussed in [28].)

Capacities & Propensities. The solution to the *recommendation with capacity constraints* problem relies on the availability of (i) users' propensities (how likely they are to follow the recommendations), and (ii) item/POI capacities (how many people are allowed to simultaneously use/occupy an item/place). As the information of capacities and propensities is not given in the considered data, and to the best of our knowledge to any of the publicly available recommendation datasets, we considered different simple ways of estimating them based on usage data.

For capacities, we analyzed the three exhaustive cases: (i) capacities analogous to usage, i.e., such a setting is inspired by the supply-demand law: the more users ask for an item, the more copies of the item will be in the market, (ii) capacities inversely proportional to usage, i.e., capturing when items with low capacities are often in high demand, and (iii) irrespective of usage. In particular, we instantiated the above concepts with the following:

(1)'actual': $\forall j,\ c_j$ = # users who have rated item j.[4]
(2)'binning': Transform actual capacities to the bins: $[0,20]\rightarrow 5$, $[21,100]\rightarrow 50$, $[101,\text{max capacity}]\rightarrow 150$.
(3)'uniform-k': Set all item capacities to a value k, e.g. 10.
(4)'linear max': Spread the items' capacities in [0, maximum actual capacity] using a linear function.
(5)'linear mean'. Same as 'linear max', but spread in the range [0, 2*mean of actual capacities].
(6)'reverse binning': Map the actual capacities to the following bins : $[0,20]\rightarrow 150$, $[21,100]\rightarrow 50$, $[101,\text{max capacity}]\rightarrow 5$.

Note that (1) -(2) are analogous to usage, (3) - (5) are irrespective of usage and (6) is inversely proportional to usage.

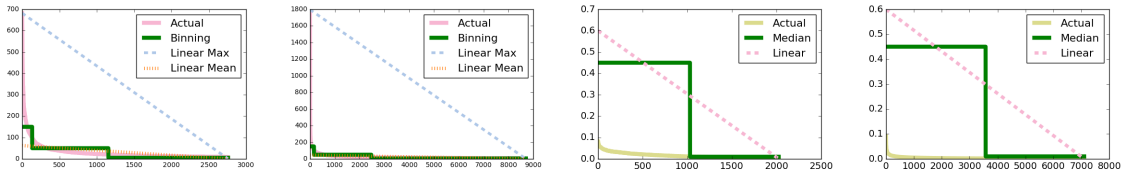
Figures 1(a), (b) show all items' capacity scores sorted decreasingly for various capacity choices, for the POI datasets.

For user propensities, we considered the following cases: (i) user propensities are analogous to system usage by the user , and (ii) user propensities are irrespective of usage (i.e., capturing that propensity is an inherent user property). In particular, we considered: (1) 'actual': $p_i = \frac{\text{\# observed ratings for user } i}{\text{Total \# items}} = \frac{|L_i|}{N}$ where $|\cdot|$ denotes a set's cardinality.[5]

[2] http://www.grouplens.org
[3] http://www.ntu.edu.sg/home/gaocong/data /poidata.zip

[4] An alternative would be # users who have liked the item in explicit data.

[5] An alternative could be: $p_i = \frac{\text{\# liked items for user } i}{\text{\# observed ratings for user } i} = \frac{|L_i^+|}{|L_i|}$.

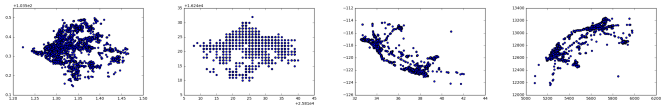


(a) Foursquare, capacities (b) Gowalla, capacities (c) Foursquare, propensities (d) Gowalla, propensities

Figure 1: (a) - (b) Item capacities, (c) - (d) User Propensities, sorted in decreasing order.

(2) 'median': Set propensities ≥ the median of the actual propensities to 0.45, and propensities < median to 0.01. This illustrates two distinct groups of users; those who tend to listen to the system's recommendations, and those who do not.
(3) 'linear': Spread propensities in $[0, 0.6]$ using a linear function.
Note that (1)-(2) are analogous to user usage, while (3) is irrespective of usage. Figures 1(c), (d) show users' propensity scores sorted decreasingly for the various propensity choices for the POI data.

More intricate capacity and propensity estimation models can be considered as input to our framework as well.



(a) Foursquare, lat./long. coordinates (b) Foursquare, tile coordinates (c) Gowalla, lat./long. coordinates (d) Gowalla, tile coordinates

Figure 2: Location information Scatter plots.

Location. The POI datasets Foursquare and Gowalla contain apart from the check-in observations, location information about where every POI lies in terms of geographical latitude and longitude coordinates. In Figures 2(a), (c) we show scatter plots of the latitudes (x axis) and longitudes (y axis) for the POIs of the non-subsampled datasets of Foursquare and Gowalla respectively. Using the Mercator projection, and fixing the ground resolution to level of detail set to 15, we transformed the latitude/longitude coordinates of every POI to one of the 2^{15} tiles where the entire Earth can be divided into[6]. We show the tiles x-y coordinates of the various POIs for Foursquare in Figure 2(b) and for Gowalla in Figure 2(d). We set L, i.e., the dimension of the influence vector, to the number of unique tiles found. For Foursquare L is 290, whereas for Gowalla L is 4320. Considering the set of all L unique tiles and representing every POI as a pair of (tileX, tileY) coordinates, we pre-computed the influence matrix $Y \in \mathbf{R}^{L \times N}$ using Kernel Density Estimation [40], following the procedure briefly described in Section 3.

Parameter Setting. We stopped the training of the algorithm either when the improvement in the value of the optimization objective in the training set was smaller than 10^{-5} or after 3,000 iterations. Similarly to [21], we set the rank of the latent factors k to 10. We used for the learning rate of gradient descent the Adagrad rule [8], which for the latent factor of the user i, $\mathbf{u}_i$ at iteration t is: $\eta_t = 1/\sqrt{\sum_{\tau=0}^{t-1} \nabla^2_{\mathbf{u}_{i,\tau}}}$. We fixed the regularization parameter λ to 10^{-5}. We chose these parameters because we found they were sufficient for stable model performance. In practice, to further improve the performance, we can tune the parameters of rank and regularization in a validation set.

[6]https://msdn.microsoft.com/en-us/library/bb259689.aspx

Evaluation Metrics. For the *weighted objectives* solutions of `CapMF`, `Cap-(Geo)BPR`, we report metrics relevant to the losses they optimize, to study the interplay of the two objectives. To compare methods overall, we use top-N recommendation accuracy as measured by *AP@k*. Particularly, we report test-set performance in terms of:

(1) For `Cap-PMF` and `Cap-GeoMF`, we report *Root Mean Square Error (RMSE)* which measures test set rating prediction accuracy:

$$\text{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\frac{1}{|L_{i,\text{test}}|}\sum_{j \in L_{i,\text{test}}}(\hat{r}_{ij} - r_{ij})^2}\,. \tag{14}$$

$L_{i,\text{test}}$ denotes the set of observed ratings for user i in the test set.
(2) For `Cap-BPR` and `Cap-GeoBPR` we report *0/1 Pairwise Loss* which measures the average number of incorrectly ordered pairs (-1 ranked above +1):

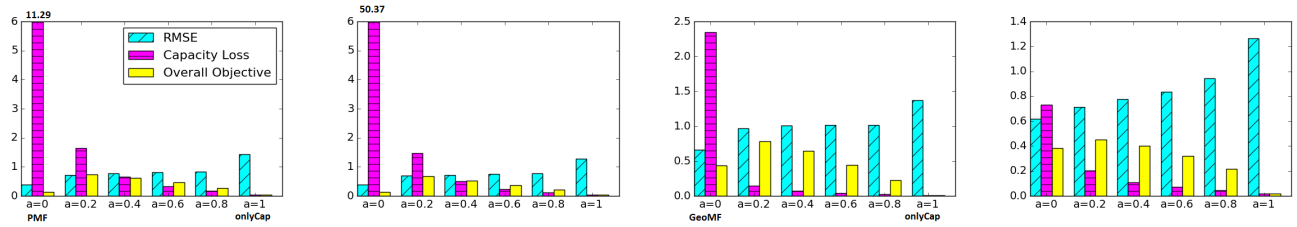
$$\text{0/1 Pair. Loss} = \frac{1}{M}\sum_{i=1}^{M}\frac{1}{|L^-_{i,\text{test}}| \cdot |L^+_{i,\text{test}}|}\sum_{j \in L^-_{i,\text{test}}}\sum_{k \in L^+_{i,\text{test}}}\mathbf{1}[\hat{r}_{ij} \geq \hat{r}_{ik}], \tag{15}$$

where $L^+_{i,\text{test}}$ and $L^-_{i,\text{test}}$ denote the set of positively and negatively rated items by user i in the test set respectively.
(3) *Capacity Loss*, which measures on average the extent to which the recommendations lead to violating the capacity constraints:

$$\text{Capacity Loss} = \frac{1}{N}\sum_{j=1}^{N}\ell\left(c_j - \sum_{i=1}^{M} p_i \sigma(\hat{r}_{ij})\right)\,. \tag{16}$$

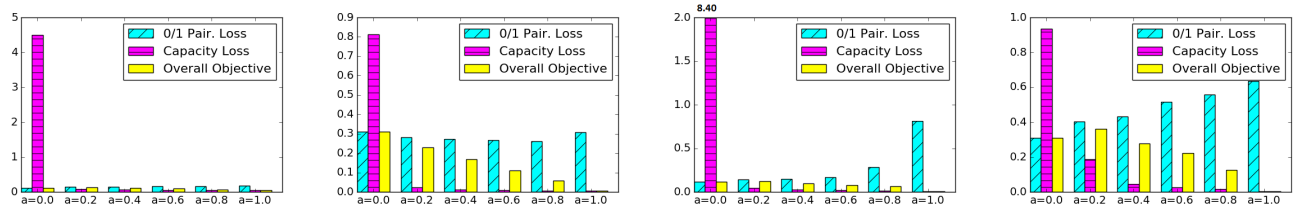
For metrics (1), (2), (3) values closer to 0 are better.
(4) *Overall Objective.* Given that our methods optimize the two objectives of rating prediction (`CapMF`)/ranking (`Cap-BPR`) loss and capacity loss, we also report: $(1-\alpha)\text{RMSE}^2 + \alpha$Capacity Loss for `CapMF` (or $(1-\alpha)$0/1 Pairwise Loss + αCapacity Loss for `Cap-BPR`).
(5) *Mean Average Precision @k*, for top-k recommendation quality. AP@k is the average of Precisions computed at each relevant position (+1) in the top k items of the user's ranked list. Precision@k (P@k) is the fraction of relevant items out of the top k items.

$$AP@k = \sum_{r=1}^{k}\frac{P@r \cdot rel(r)}{\min(k, \#\text{ relevant items})}, \tag{17}$$



(a) Movielens 100K (b) Movielens 1M (c) Foursquare (d) Gowalla

Figure 3: Effect of capacity trade-off parameter α in range $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$ on CapMF's performance in test RMSE, Capacity Loss and Overall Objective. As expected, the higher the α, the higher the RMSE and the lower the Capacity Loss.



(a) Movielens 100K (b) Foursquare (c) Movielens 100K (d) Foursquare

Figure 4: Effect of capacity trade-off parameter α in range $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$ on CapBPR's performance in test 0/1 Pairwise Loss, Capacity Loss and Overall Objective. (a), (b): Using 'Actual' Capacity definition. We observe that the 0/1 Pairwise Loss does not increase much as we increase α. (See why in Figure 5.) (c), (d): Using 'Reverse Binning' Capacity definition. We observe the expected trade-off between ranking loss and capacity loss.

where $rel(r)$ is 1 if the item in position r is relevant, and 0 otherwise. After computing Average Precision per user, we average the results over all users. Values closer to 1 are better.

Methods Compared. We compare our methods with the baselines of: PMF [32], BPR [30], GeoMF [24], GeoBPR, and onlyCap, i.e., the baseline of setting α to 1. Also, we compare our proposed *weighted objectives* methods with the *post-process* methods.

5.2 Validation: Interplay between Objectives

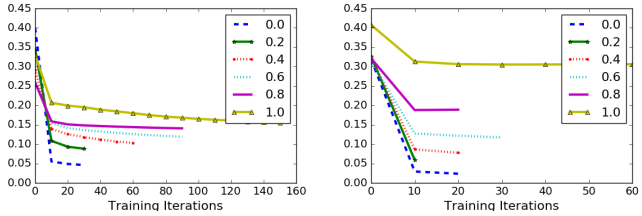
In the first set of experiments, our goal is to validate that indeed using the trade-off parameter α, one can effectively trade-off between recommendation performance and capacity loss. Recall that for CapMF (Cap-PMF, Cap-GeoMF), α captures the trade-off among the objective of rating prediction for whether a user will purchase (visit) an item (POI), and the objective of respecting the items' capacities. Also, recall that for Cap-BPR, Cap-GeoBPR, α is used to tune the interplay between 0/1 Pairwise Loss, and Capacity Loss.

We expect that as α approaches 0, the algorithm will have better predictive performance but will be violating more the capacity constraints. In contrast, as α approaches 1, we expect the algorithm to respect more the capacity constraints at the cost of worse predictive ability. To illustrate this, we vary the trade-off parameter in $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$. We set the surrogate loss for the capacity objective to the logistic loss, and use the 'actual' capacity and propensity definitions.

5.2.1 Rating Prediction vs. Capacity Loss. In Figure 3, we report the test set metrics of RMSE, Capacity Loss and Overall objective for all four datasets. The results show that indeed the more we increase the trade-off parameter, the smaller the capacity loss and the higher the RMSE. This validates that α can be used to specify to what extent the rating prediction accuracy is more/less important compared to the capacity loss for the considered application domain.

5.2.2 Ranking vs. Capacity Loss. Considering as first objective the pairwise ranking objective, in Figures 4(a), (b), we show the results of Cap-BPR/Cap-GeoBPR for Movielens 100K and Foursquare, as we vary the trade-off parameter in the range of [0, 1]. Interestingly, we observe that while as expected, the capacity loss decreases with the increase of α, the 0/1 pairwise loss does not change much. Similar trends hold for the rest of the datasets (not shown). We found that the reason why this happens is that when the capacities are analogous to usage (here we used the 'actual' capacity definition), even if α is set to 1, i.e., not optimizing at all for ranking accuracy, the 0/1 pairwise training loss still decreases (Figure 5). This shows that when capacities are analogous to usage, the capacity loss is related to the pairwise loss; namely, the capacity loss can help reconstruct the correct pairwise accuracy given only the item capacities and user propensities, with no other access to user-item ratings. Thus, the aggregate statistics of 'actual' user propensities and 'actual' item capacities act as sufficient statistics.

To see the expected trade-off between the objectives of capacity and ranking loss, we report in Figures 4(c), (d) Cap-BPR's results for item capacities set inversely proportional to usage, using the 'reverse binning' capacity definition. Indeed, in this case, the fraction of incorrectly ordered pairs increases with the increase in α.



(a) Movielens 100K (b) Foursquare

Figure 5: Training Pairwise 0/1 Loss of (a) Cap-BPR, (b) Cap-GeoBPR for 'actual' capacities, varying trade-off $\alpha \in [0, 1]$.

5.3 Comparison with Unconstrained Methods

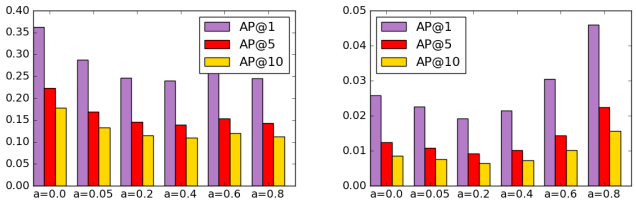
For the next experiments we set α to 0.2, as we found it can lead to a reasonable trade-off among the objectives. In practice, α should be tuned in a validation set, to meet the desired trade-off of capacity loss vs. recommendation performance per application domain.

5.3.1 Comparison with PMF/GeoMF in RMSE/Capacity Loss. Based on Figure 3, considering the left end of the spectrum for the value of the trade-off parameter, i.e., $\alpha = 0$, we can compare our algorithm CapMF with the unconstrained methods PMF/GeoMF. We observe that: (i) For item recommendation, Cap-PMF significantly improves over PMF's Capacity Loss performance of 11.29 to 1.65. This happens though at the cost of deteriorated performance in terms of RMSE from .38 to .71. Similar is the trend for Movielens 1M. (ii) For POI recommendation, we observe that for Foursquare, Cap-GeoMF improves over GeoMF's Capacity Loss performance of 2.35 to .15, at the cost of RMSE which increases from .66 to .97. Similar trends hold for Gowalla.

5.3.2 Comparison with BPR/Geo-BPR in Pairwise Loss/Capacity Loss. From Figures 4(a), (b), we can compare our method Cap-BPR with BPR and GeoBPR respectively. We see that for Movielens 100K, Cap-BPR achieves Capacity Loss of 0.08 compared to 4.51 of BPR, while it results in 0/1 Pairwise Loss of 0.14 compared to 0.12 (higher values are worse). For Foursquare, Cap-GeoBPR achieves 0.02 Capacity Loss compared to 0.81 of GeoBPR, and also achieves a better 0/1 Pairwise Loss of 0.28 compared to 0.31 (the reason why was explained in Section 5.2.2). Similar trends hold for the other datasets.

5.3.3 Comparison with onlyCap. Focusing now on the other end of the spectrum of the trade-off parameter, i.e., $\alpha = 1$, we compare CapMF with onlyCap. We can see from Figure 3(a) that for Movielens 100K, onlyCap improves Capacity Loss from 1.65 to .04, but results in RMSE from .71 to 1.43. Also, onlyCap results in a worse 0/1 pairwise loss of 0.17, compared to 0.14 of Cap-BPR, as seen from Figure 4(a).

5.3.4 Top-N Recommendations. Here, we evaluate the top-N recommendation quality using AP@top as our metric, varying top N in $\{1, 5, 10\}$. From Figure 6 we see that for CapMF, and 'actual' capacities and propensities: for (a) Movielens 1M, Cap-PMF has lower AP compared to PMF, while for (b) Foursquare, Cap-GeoMF achieves higher AP compared to GeoMF. We observe that as α increases, the top-N recommendation performance potentially improves, indicating that emphasizing more the capacity loss can further make the quality of the recommendations better. This aligns with our finding that the capacity loss can relate to the pairwise ranking loss, using the sufficient statistics of capacity and propensity. Also, as expected, while we increase the top, AP@top decreases.



(a) Movielens 1M (b) Foursquare

Figure 6: Average Precision (AP)@$\{1, 5, 10\}$ of CapMF.

5.4 Comparison between Proposed Approaches

In this set of experiments, we compare our *weighted objective* methods (Section 4.2) with the *post-process* methods (Section 4.1). Recall that the *post-process* methods, by design, do not violate the capacity constraints. For the purposes of the novel setting of recommendation with capacity constraints, we propose a new metric:

$$\text{Weighted AP @ top (WAP@top)} = \frac{\sum_{i=1}^{M} p_i \text{AP}_i@top}{\sum_{i=1}^{M} p_i}, \qquad (18)$$

where AP_i is user i's Average Precision. WAP@top measures top-N recommendation quality taking into account the users' propensities, by using propensities as weights in the weighted average of users' average precisions. We report in Table 2 the comparison of the *post-process* methods, using the best performing tuned α, with our proposed *weighted objectives* methods, and the respective unconstrained methods. For this experiment, we use Foursquare and explicit Movielens 100K, and consider as underlying models GeoMF/GeoBPR and PMF/BPR respectively.

Based on Table 2, we make the following overall observations:

- For square loss, when capacities are proportional to usage ('actual', 'binning'), Cap-PMF outperforms PostMF, and even more interestingly outperforms PMF for $top = 50$. However, when the capacities are inversely proportional to usage, or are uniform, our proposed weighted objectives solution outperforms PostMF, but cannot outperform the unconstrained method. We can also see that for any choice of capacity, for geographically-aware methods, Cap-GeoMF outperforms both Post-GeoMF and GeoMF.
- For ranking loss, when capacities are proportional to usage, similar to square loss, Cap-BPR outperforms post-process, and can even outperform BPR. When capacities are uniform, Cap-BPR outperforms Post-BPR, and for item recommendation can even improve upon the unconstrained method. In contrast, for capacities inversely proportional to usage, Cap-BPR, Cap-GeoBPR tends to be outperformed by the other methods.

Similar trends can be observed for non-weighted average precision, and for the other datasets, as well. Overall, these results highlight that our weighted objectives methods largely outperform the *post-process* solution.

We can also draw the further conclusion that out of the alternative definitions considered for capacity, the definitions which are analogous to usage, namely 'actual' and 'binning', are the ones which lead to the best top-N recommendation results. In particular,

		Square Loss						Pairwise Ranking Loss					
	Capacity Def.	WAP@10			WAP@50			WAP@10			WAP@50		
		MF	PostMF	CapMF	MF	PostMF	CapMF	BPR	PostBPR	Cap-BPR	BPR	PostBPR	Cap-BPR
ML 100K	Actual	0.152	**0.153***	0.138	0.086	0.087	**0.127***	0.115	0.117	**0.209***	0.069	0.07	**0.126***
	Binning	**0.152**	0.15*	0.136	**0.086**	0.077	**0.086***	0.115	0.112	**0.209***	0.069	0.064	**0.136***
	Reverse	**0.152**	0.045	0.072*	**0.086**	0.013	0.031*	**0.115**	0.04*	0.02	**0.069**	0.012*	0.01
	Uniform	**0.152**	0.055	0.135*	**0.086**	0.014	0.062*	0.115	0.055	**0.137***	0.069	0.013	**0.078***
Foursquare	Actual	0.016	0.016	**0.041***	0.011	0.011	**0.027***	0.056	0.049	**0.084***	0.033	0.023	**0.048***
	Binning	0.016	0.016	**0.023***	0.011	0.009	**0.014***	0.056	0.04	**0.084***	0.033	0.017	**0.047***
	Reverse	0.016	0.011	**0.039***	0.011	0.005	**0.015***	**0.056**	0.016*	0.013	**0.033**	0.005	0.007*
	Uniform	0.016	0.011	**0.047***	0.011	0.004	**0.024***	**0.056**	0.016	0.039*	**0.033**	0.005	0.023*

Table 2: Comparison of *weighted objectives* methods (denoted with `Cap-`) with *post-process* methods (denoted with `Post`), in terms of Weighted Average Precision@*top*, *top* = {10, 50}, for various capacity settings. Bold letters denote the best among the unconstrained, post-process, and `Cap-` methods. Asterisk (*) shows which capacity constrained method, i.e., `Post-` or `Cap-`, is better. Overall, the `Cap-` methods largely outperform the respective `Post-` methods. Interestingly, in some cases, `Cap-` methods outperform the unconstrained methods, showing the value of our method to even improve the recommendation quality.

under these settings the *weighted objectives* approach surpasses the *post-process* solution, and also is better than the unconstrained methods. This shows that these capacity settings can be realistic, and are worth incorporating in the optimization objective of the recommendation algorithms, to further improve top-N performance.

5.5 Sensitivity Analysis

5.5.1 Effect of Surrogate Loss for Capacity Term. Figure 7 compares the effect of different surrogate losses for the capacity objective on `CapMF`'s performance. For this experiment, we consider rating prediction as the first objective, we set α to 0.2, and use the 'actual' propensity and capacity definitions. We observe that for Movielens 100K and Movielens 1M (omitted), logistic and hinge loss result in similar performance, while exponential loss results in the highest RMSE and the smallest capacity loss. For Foursquare and Gowalla (omitted), hinge loss obtains the smallest capacity loss and RMSE, making it the best option for the POI datasets. This happens since the exponential loss assigns very high penalties for large negative differences. Also, exponential and logistic losses assign small penalties for small negative and also positive values, while hinge does not penalize when expected usage is within the capacity constraints.

5.5.2 Effect of User Propensities. In Figure 8 we study the effect of different user propensities as input to our algorithm (i.e., 'actual', 'median' and 'linear') on `CapMF`'s performance in the Foursquare dataset. For this experiment, we set α to 0.2, the surrogate loss to the logistic loss, the capacities to 'actual' and the first objective to rating prediction. We see that for 'median' and 'linear' choices of propensity, the values of both RMSE and Capacity Loss are higher compared to those obtained for the 'actual' propensity choice. This happens because the 'median' and 'linear' options generally result in higher values of user propensities (Figure 1(c)). This leads to higher expected usage values, making it more likely to have the capacity constraints violated.

5.5.3 Effect of Item Capacities. Here we study the effect of different item capacities, namely 'actual', 'linear mean', 'linear max' and 'binning', as input to `CapMF`. The setting is the same as the one described in 5.5.2, except for the propensity definition which is set to 'actual'. Figure 9 compares `CapMF`'s performance for the four choices of capacity for Gowalla – similar trends hold for the other datasets. We can see that the choice of 'binning' results in the highest RMSE and Capacity Loss, whereas the other choices result in almost identical performance. We explain this result as based on Figure 1(b) the 'binning' definition typically results in the smallest items' capacities, which means that it is more likely that the recommendations will violate the capacity constraints. Also, when setting all items' capacities uniformly to e.g. the mean actual capacity (not shown), the capacity constraints are directly satisfied.

5.5.4 Implicit vs. Explicit Feedback. Finally, we report in Figure 10 the results of `Cap-PMF` in terms of the competing objectives for the original explicit Movielens datasets. This allows us to explore how sensitive our framework is to implicit versus explicit data. Thus, we can compare Figure 10(a) with Figure 3(a) for Movielens 100K, and Figure 10(b) with Figure 3(b) for Movielens 1M. We can see that although the particular values of the objectives are different, the trends are similar. Similar trends were also found for `Cap-BPR`.

6 CONCLUSIONS

We have presented a novel approach for providing recommendations that satisfy capacity constraints. We have demonstrated how this generic approach can be applied to three state-of-the-art latent factor models, PMF [32], GeoMF [24], and BPR [30], so that the items' expected usage respects the corresponding capacities. Our experimentation has given light into how our methods perform under different settings of item capacities and user propensites. We have shown that our framework effectively provides recommendations which respect the capacity constraints, without deteriorating the recommendation performance by a lot; interestingly, in some cases our methods can improve the top-N recommendation quality compared to the respective unconstrained methods.

Acknowledgments. The work was supported in part by NSF grants IIS-1447566, IIS-1422557, CCF-1451986, CNS-1314560, IIS-0953274, IIS- 1029711,

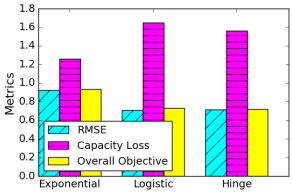


Figure 7: Movielens 100K. Effect of surrogate loss for capacity term.

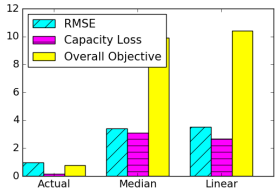


Figure 8: Foursquare. Effect of user propensities on `CapMF`.

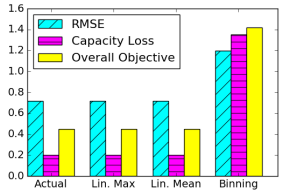
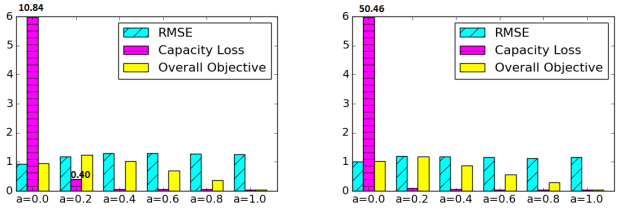


Figure 9: Gowalla. Effect of item capacities on `CapMF`.



(a) Movielens 100K (b) Movielens 1M

Figure 10: `Cap-PMF` on explicit feedback data. The results shown compared to Figures 3(a), (b) indicate that the trends for implicit and explicit feedback data are similar.

NASA grant NNX12AQ39A, and a gift from Adobe Research. We also acknowledge technical support from the University of Minnesota Supercomputing Institute. KC would like to thank Evangelia Christakopoulou for the valuable comments.

REFERENCES

[1] Gediminas Adomavicius and YoungOk Kwon. 2007. New recommendation techniques for multicriteria rating systems. *IEEE Intelligent Systems* 22, 3 (2007).

[2] Deepak Agarwal, Shaunak Chatterjee, Yang Yang, and Liang Zhang. 2015. Constrained optimization for homepage relevance. In *WWW*. ACM, 375–384.

[3] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. 2009. Explore/exploit schemes for web content optimization. In *ICDM*. IEEE, 1–10.

[4] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2011. Click shaping to optimize multiple objectives. In *KDD*. ACM, 132–140.

[5] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2012. Personalized click shaping through lagrangian duality for online recommendation. In *SIGIR*. ACM, 485–494.

[6] Kareem Amin, Michael Kearns, Peter Key, and Anton Schwaighofer. 2012. Budget optimization for sponsored search: Censored learning in MDPs. *arXiv preprint arXiv:1210.4847* (2012).

[7] Konstantina Christakopoulou and Arindam Banerjee. 2015. Collaborative Ranking with a Push at the Top. In *WWW*. ACM, 205–215.

[8] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR* 12, Jul (2011), 2121–2159.

[9] Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2013. Exploring temporal effects for location recommendation on location-based social networks. In *RecSys*. ACM, 93–100.

[10] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. 2010. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *RecSys*. ACM, 257–260.

[11] Rupesh Gupta, Guanfeng Liang, Hsiao-Ping Tseng, Ravi Kiran Holur Vijay, Xiaoyu Chen, and Rómer Rosales. 2016. Email Volume Optimization at LinkedIn. In *KDD*. ACM, 97–106.

[12] Ralf Herbrich. 2016. Learning Sparse Models at Scale. In *KDD*. ACM, 407.

[13] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *ICDM*. IEEE, 263–272.

[14] Neil Hurley and Mi Zhang. 2011. Novelty and diversity in top-n recommendation–analysis and evaluation. *ACM Transactions on Internet Technology (TOIT)* 10, 4 (2011), 14.

[15] Tamas Jambor and Jun Wang. 2010. Optimizing multiple objectives in collaborative filtering. In *RecSys*. ACM, 55–62.

[16] Christopher C Johnson. 2014. Logistic matrix factorization for implicit feedback data. In *NIPS 2014 Workshop on Distributed Machine Learning and Matrix Computations*.

[17] Ioannis Karatzas and Steven Shreve. 2012. *Brownian motion and stochastic calculus*. Vol. 113. Springer Science & Business Media.

[18] Yehuda Koren, Robert Bell, Chris Volinsky, et al. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[19] Kleanthi Lakiotaki, Nikolaos F Matsatsinis, and Alexis Tsoukias. 2011. Multicriteria user modeling in recommender systems. *IEEE Intelligent Systems* 26, 2 (2011), 64–76.

[20] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2014. *Mining of massive datasets*. Cambridge University Press.

[21] Huayu Li, Yong Ge, and Hengshu Zhu. 2016. Point-of-Interest Recommendations: Learning Potential Check-ins from Friends. In *KDD*. ACM, 975–984.

[22] Haifang Li and Yingce Xia. 2017. Infinitely Many-Armed Bandits with Budget Constraints. In *AAAI*. 2182–2188.

[23] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-GeoFM: a ranking based geographical factorization method for point of interest recommendation. In *SIGIR*. ACM, 433–442.

[24] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD*. ACM, 831–840.

[25] Dawen Liang, Laurent Charlin, James McInerney, and David M Blei. 2016. Modeling user exposure in recommendation. In *WWW*. ACM, 951–961.

[26] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. 2010. Online learning for matrix factorization and sparse coding. *JMLR* 11, Jan (2010), 19–60.

[27] Aranyak Mehta et al. 2013. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science* 8, 4 (2013), 265–368.

[28] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *ICDM*. IEEE, 502–511.

[29] Aditya Parameswaran, Petros Venetis, and Hector Garcia-Molina. 2011. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Transactions on Information Systems (TOIS)* 29, 4 (2011), 20.

[30] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. AUAI Press, 452–461.

[31] Marco Tulio Ribeiro, Anisio Lacerda, Adriano Veloso, and Nivio Ziviani. 2012. Pareto-efficient hybridization for multi-objective recommender systems. In *RecSys*. ACM, 19–26.

[32] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic matrix factorization. In *NIPS*, Vol. 20. 1257–1264.

[33] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. *arXiv preprint arXiv:1602.05352* (2016).

[34] Hanhuai Shan and Arindam Banerjee. 2010. Generalized probabilistic matrix factorizations for collaborative filtering. In *ICDM*. IEEE, 1025–1030.

[35] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. 2012. CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *RecSys*. ACM, 139–146.

[36] Krysta M Svore, Maksims N Volkovs, and Christopher JC Burges. 2011. Learning to rank with multiple objective functions. In *WWW*. ACM, 367–376.

[37] Liang Tang, Bo Long, Bee-Chung Chen, and Deepak Agarwal. 2016. An Empirical Study on Recommendation with Multiple Types of Feedback. In *KDD*. ACM, 283–292.

[38] Baosheng Yu, Meng Fang, and Dacheng Tao. 2016. Linear Submodular Bandits with a Knapsack Constraint. In *AAAI*. 1380–1386.

[39] Yonghong Yu and Xingguo Chen. 2015. A Survey of Point-of-Interest Recommendation in Location-Based Social Networks. In *Workshops at AAAI*.

[40] Jia-Dong Zhang and Chi-Yin Chow. 2013. iGSLR: personalized geo-social location recommendation: a kernel density estimation approach. In *SIGSPATIAL*. ACM, 334–343.

[41] Yongfeng Zhang, Qi Zhao, Yi Zhang, Daniel Friedman, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Economic recommendation with surplus maximization. In *WWW*. ACM, 73–83.