MOR-LinUCB: A Multi-Objective and Context-Aware Ranking Approach

Nirandika Wanigasekara[1], Yuxuan Liang[1], Joseph J. Williams[2], Ye Liu[1] and David S. Rosenblum[1]

[1] National University of Singapore
{nirandiw, liangyx, liuye, david}@comp.nus.edu.sg
[2]University of Toronto
williams@cs.toronto.edu

***Abstract*—Understanding users' search intents on the web can be enhanced by contextual information about users and web resources. Providing users with relevant search results requires balancing multiple objectives, such as users' explicitly stated preferences for novelty, versus similarity to what they have liked in the past. We consider a novel problem in which an algorithm needs to learn a user's unknown utility function in different contexts over the multiple reward objectives for online ranking. We call this problem the Multi-Objective Contextual Ranked Bandit (MOCR-B) problem. To solve the MOCR-B problem, we present a novel algorithm, Multi-Objective Contextual Ranked-UCB (MOR-LinUCB), which uses generates a ranked list of resources, by using context information about users and resources to maximize a user's specific utility function over multiple reward objectives. We use Neural Network Embeddings to efficiently model user-resource context information for the MOR-LinUCB algorithm. We evaluate MOR-LinUCB empirically, using synthetic data and real-world data from TripAdvisor. Our results reveal that the ranked lists generated by MOR-LinUCB lead to better click-through rates, compared to approaches that do not learn the utility function over multiple objectives.**

***Index Terms*—multi-objective ranking, feature embedding, neural networks, contextual-bandits**

I. Introduction

There are many settings on the Web where we want to provide users with ranked lists of resources to choose from, such as suggesting hotels on TripAdvisor, jobs on LinkedIn, products to buy on Amazon, or news articles to read on Yahoo. This paper focuses on the problem of actively choosing ranked lists of resources to present to users, and dynamically using users' interactions (i.e., user clicks, user ratings, user feedback) to decide which resources to present in the future. Such problems often rely critically on using *contextual information* about the users and resources [1]–[3]. Most of existing approaches often use context information to optimize a single objective function, such as the click-through-rate (CTR) or query relevancy (QR) [2], [4]. However, in many real applications, there are multiple objectives and a user may reflect his/her preferred trade-off between multiple objectives in different contexts [5], [6]. Take the hotel recommendation as an example, a user's rating for hotels covers over multiple objectives (e.g., location rating, value for money, cleanliness rating, business service rating), which reflects their needs in different context. Hence, contextual information about the users and resources (hotels) must therefore be modeled across multiple objectives. The users' preferred trade-off across these multiple objectives can then be used in different contexts to optimize some higher level objective that is a proxy for actual behavior such as purchasing the hotel or applying for the job in the above examples.

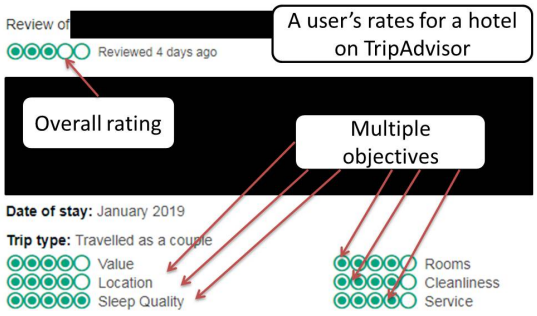

Fig. 1: A user's overall rating and ratings in different objectives for a hotel in TripAdvisor

To elaborate, consider an online ranking algorithm that aims to learn which hotels to suggest to users in TripAdvisor. The algorithm can access contextual information about the user, such as previous reviews given by the user for hotels, number of helpful reviews by the user, location, users previous ratings for hotels, or the user's current activity (e.g., relaxing, jogging). The algorithm can also access contextual information about the hotels being suggested (e.g., location, services provided, and stars-level). When suggesting hotels for a user to stay, the algorithm can optimize *multiple objectives* (e.g., location rating, value for money, cleanliness rating, business service rating) such that users can find hotels that best suit for their needs in different context.

In this setting it is generally difficult to model the temporal changes solely based on the hotels' context information. We need to explore the unknown by collecting feedback from the users in real time to evaluate the suitability of hotels in different user contexts. As a result, we need an online sequential decision making process that addresses the learning challenges that arise from the presence of multiple objectives rewards in-addition to utility rewards.

However, devising an online ranking with multiple objectives in different contexts is a challenging task due to the following reasons. First, it is unknown apriori how context information is related to the user's rating in each objective in an online learning setting. Take the hotel recommendation in TripAdvisor as example, as there are several ratings for the hotel and we don't know explicitly the relationship between each specific rating (e.g., Service rating) with users contextual information (e.g., user location, user's previous ratings for hotels, or users current activity). Thus, how to discover the correlation between each objective and the user context is challenge task. Second, it is unknown what the user's preferences are over the multiple objectives for ranking. Each user may value each objective differently. For example, whether a user will stay in a hotel or rate it high could be influenced by the extent to which the user values the hotel's cleanliness versus the location versus the business service. Third, as users contextual information is always high-dimensional and unstructured, hence, how to encode those contextual information effectively is a non-trival task.

To tackle above challenges, in this paper, we present a novel framework to solve this problem with Multi-Armed Bandits (MABs) [7] and neural network embeddings [8]–[10]. In particular, the framework learns to model a user's selection/rate of an item over the *user's weighting for each objectives*. The algorithm combines multiple capabilities: (1) online learning that trades off exploration and exploitation in suggesting resources; (2) discovery and modeling of how contextual features of users and resources are mapped to rewards in each objective using neural network embeddings; and (3) discovery and modeling of how competing rewards in multiple objectives map to the user's actual behavior, (4) encoding the high-dimensional and unstructured contextual data into an uniformed representation.

More specifically, we formulate the above context-aware, multi-objective, ranking problem as a novel MAB problem that considers multiple-objective reward vectors, contextual information and ranking. We call this problem the *Multi-Objective Contextual Ranked Bandit* (MOCR-B) problem. As a solution to the MOCR-B problem, we present a novel algorithm, Multi-Objective Ranked-LinUCB (MOR-LinUCB), which minimizes the number of overall poor rankings and maximizes the long-term rewards in each objective, conditioned on the context of the user and resources.

To maximize the long-term reward in each objective, MOR-LinUCB first models the expected payoff of each objective of an arm as a linear function of an l-dimensional context feature vector, and it learns the goodness of the match between the user's context and an arm's expected multiple-objective reward. Given the estimated multiple-objective rewards, MOR-LinUCB then learns an unknown utility function over the multiple objectives for each user based on the user's click behavior and ranks the web resources to minimize the total number of poor rankings. To the best of our knowledge, our work is the first to present a bandit solution that combines context-awareness, multi-objective optimization, and ranking.

In summary, the contributions in our paper are as follows.

1) We present a novel problem formulation to handle context information, user preferences and multiple objective reward functions in online ranking called Multi-Objective Contextual Ranked Bandits (MOCR-B).
2) We present a novel algorithm, named MOR-LinUCB, to solve the MOCR-B problem. MOR-LinUCB models context information in each objective and directly learns the utility over the multiple objectives for each user to generate ranked lists that minimize search abandonment.
3) We evaluate our framework on both synthetic data and real-world data from TripAdvisor demonstrating the effectiveness of our approach.

II. Problem Definition

In this section, we define the Multiple-Objective Contextual Ranked Bandit (MOCR-B) problem, where the goal is to rank resources to maximize user satisfaction based on multiple objectives. In the MOCR-B problem setting we have users interacting with a system over several trials. An algorithm solving the MOCR-B problem must learn to provide users with a list of resources (also called arms), that are ranked according to their utility for the user, where the utility function is a weighted combination of multiple observed rewards in different objectives, and each observed reward is a function of contextual features of the user *and* the resource.

Formally, as in a contextual bandit, an algorithm solving the MOCR-B problem must run over a sequence of trials indexed by $t = 1, \ldots, T$. On each trial t, the algorithm observes a user u_t and presents a ranked list $\tilde{\mathcal{A}}_t$ of resources to the user:

$$\tilde{\mathcal{A}}_t = \{a_k^t | 1 \leq k \leq K\} \tag{1}$$

where a_k^t is the resource displayed in rank k and K is the total number of rankings.

In each trial t the algorithm needs to solve the problem of choosing which resources to present to the user. For each ranking position k, the algorithm must choose a resource a_k^t with a high utility for a user. More precisely the resource a_k^t chosen by the algorithm at rank k must maximize the binary utility reward f_k^t, which is 1 if the user selects the resource and 0 otherwise. The observed utility reward f_k^t for each rank k is a function Ψ of multiple objectives rewards, $\mathbf{r}_{a_k^t}^t = (r_{1,a_k^t}^t, \ldots, r_{d,a_k^t}^t)$ (i.e., a linear combination of the multiple-objective rewards):

$$\Psi : \mathbf{r}_{a_k^t}^t \mapsto f_k^t \tag{2}$$

When the algorithm presents a ranked list to the user, for each resource a_k^t that the user selects (and presumably found useful), the algorithm observes multiple rewards for the different objectives. These rewards $\mathbf{r}_{a_k^t}^t$ are observed by the algorithm, but ultimately a reward $r_{i,a_k^t}^t$ for objective i can be treated as a function Θ of a context feature vector $\mathbf{x}_{i,a_k^t}^t$, which combines context features of the user u_t and context features of the resource a_k^t for the objective i:

$$\Theta : \mathbf{x}_{i,a_k^t}^t \mapsto r_{i,a_k^t}^t \tag{3}$$

In the MOCR-B problem setting the utility function Ψ and the objective reward mapping function Θ are unknown apriori. The key to solving the MOCR-B problem is to learn Ψ and Θ while offering a minimal number of sub-optimal resources to users.

III. MOR-LinUCB Algorithm

A. Overview

This section describes the MOR-LinUCB algorithm we designed to solve the MOCR-B problem. The MOR-LinUCB algorithm handles ranking by having a Multi-Armed Bandit (MAB) instance for each of the different ranking positions $k \in \{1, ..., K\}$, where K is the total number of rankings. The MAB instance in each rank k takes in a user u_t and a set of resources $\mathcal{A}_t$, and uses these to construct a context vector $\mathbf{x}^t_{i,a^t_k}$ (which is user and resource $a^t_k \in \mathcal{A}_t$ features) for each objective $i \in \{1, \ldots, d\}$.

Each MAB instance in rank k aims to choose a resource a^t_k to maximize the utility reward f^t_k which is a function of multiple objective rewards $\mathbf{r}^t_{a^t_k} = (r^t_{1,a^t_k}, \ldots, r^t_{d,a^t_k})$ (see Equation 2). The MOR-LinUCB algorithm parameterizes the utility reward f^t_k as a *weighted linear combination* of the multiple objective rewards $\mathbf{r}^t_{a^t_k}$. These weights are represented by the vector $\boldsymbol{\beta}^k_{u_t}$ which is specific to a user u_t in rank k. Furthermore, the MOR-LinUCB algorithm parameterizes an objective reward r^t_{i,a^t_k} as a *weighted linear combination* of the context vector $\mathbf{x}^t_{i,a^t_k}$. These weights are represented by the vector $\boldsymbol{\theta}_{i,a^t_k}$ which is specific for each arm a^t_k in objective i .The pseudo-code for MOR-LinUCB is given in Algorithm 1. We now elaborate each component of the MOR-LinUCB algorithm that was just described.

B. Ranking

Formally, MOR-LinUCB runs a MAB instance, denoted as MAB_k, for *each rank* $k \in \{1, \ldots, K\}$, where K is the total number of ranking positions. Let Π be the set of these MAB instances. Formally, $\Pi = \{\text{MAB}_k | 1 \leq k \leq K\}$. When ranking resources for user u_t, $\text{MAB}_k \in \Pi$ selects the resource to be shown at rank k.

C. Utility Function As A Linear Combination Of Multiple Rewards

In a given trial t, the algorithm takes in a user u_t and a set of resources $\mathcal{A}_t$, which provides a context vector $\mathbf{x}^t_{i,a^t_k}$ that captures user features and resource features for each objective i and a resource $a^t_k \in \mathcal{A}_t$. Each MAB_k instance selects a resource a^t_k which maximizes the value of the utility reward f^t_k which is a function of observed multiple-objective rewards and it is parameterized by an unknown user-specific coefficient vector $\boldsymbol{\beta}^{k*}_{u_t}$. We modeled the expected utility f^t_k of an arm a in the k^{th} rank as a linear function of the rewards $\mathbf{r}^t_{a^t_k} = (r^t_{1,a^t_k}, \ldots, r^t_{d,a^t_k})$ in the multiple objective rewards with the coefficients $\boldsymbol{\beta}^{k*}_{u_t}$. Namely for all t,

$$\mathbb{E}[f^t_k | \mathbf{r}^t_{a^t_k}] = (\mathbf{r}^t_{a^t_k})^\top \boldsymbol{\beta}^{k*}_{u_t} \tag{4}$$

Algorithm 1 MOR-LinUCB

Require: $\alpha, \gamma \in \mathbb{R}_+$ ▷ Exploration parameters
Initialize $\zeta = \{1, \ldots . d\}$ ▷ Set of objectives
Initialize $\Pi = \{\text{MAB}_1, \ldots, \text{MAB}_K\}$ ▷ LinUCB instances
for $t = 1 \ldots T$ **do**
  Observer user u_t
  for all $a \in \mathcal{A}_t$ **do**
    if a is new **then**
      Initialize $\mathbf{A}_{i,a} = \mathbf{I}_l, \forall i \in \zeta$
      Initialize $\mathbf{b}_{i,a} = \mathbf{0}_{l \times 1}, \forall i \in \zeta$
    end if
    Observe $\mathbf{x}^t_{i,a}, \forall i \in \zeta$ of user u_t and a
    Compute $\hat{\boldsymbol{\theta}}_{i,a} \leftarrow \mathbf{A}^{-1}_{i,a} \mathbf{b}_{i,a}, \forall i \in \zeta$
    Compute $\hat{r}^t_{i,a} \leftarrow (\mathbf{x}^t_{i,a})^\top \hat{\boldsymbol{\theta}}_{i,a} + \alpha \sqrt{\mathbf{x}^{t\top}_{i,a} \mathbf{A}^{-1}_{i,a} \mathbf{x}^t_{i,a}}, \forall i \in \zeta$
    $\hat{\mathbf{r}}^t_a \leftarrow (\hat{r}^t_{i,a}, \forall i \in \zeta)$
  end for
  Initialize $\mathcal{O}_t = \{\hat{\mathbf{r}}^t_a, \forall a \in \mathcal{A}_t\}$ ▷ All estimated rewards
  $\tilde{\mathcal{A}}_t \leftarrow \text{RANK}(\Pi, \mathcal{A}_t, \mathcal{O}_t, u_t)$
  Display $\tilde{\mathcal{A}}_t = \{a^t_1, \ldots, a^t_K\}$ to u_t
  Record clicks and feedback from u_t
  for $k = 1, \ldots, K$ **do** ▷ Update on feedback
    $\mathbf{r}^t_{a^t_k} \leftarrow \hat{\mathbf{r}}^t_{a^t_k}$
    if user clicked a^t_k **then**
      Observe $\mathbf{r}^t_{a^t_k} \leftarrow (r^t_{i,a^t_k}, \forall i \in \zeta)$ from u_t
      Update $\mathbf{A}_{i,a^t_k} \leftarrow \mathbf{A}_{i,a^t_k} + \mathbf{x}^t_{i,a^t_k} (\mathbf{x}^t_{i,a^t_k})^\top, \forall i \in \zeta$
      Update $\mathbf{b}_{i,a^t_k} \leftarrow \mathbf{b}_{i,a^t_k} + r^t_{i,a^t_k} \mathbf{x}^t_{i,a^t_k}, \forall i \in \zeta$
    end if
    Update MAB_k parameters
  end for
end for

Algorithm 2 Ranking The Resources

function $\text{RANK}(\Pi, \mathcal{A}_t, \mathcal{O}_t, u_t)$
  for $k = 1 \ldots K$ **do** ▷ K ranking positions
    for $a \in \mathcal{A}_t$ **do**
      if u_t is new **then**
        $\mathbf{A}^{\text{MAB}_k}_{u_t} \leftarrow \mathbf{I}_d$; $\mathbf{b}^{\text{MAB}_k}_{u_t} \leftarrow \mathbf{0}_{d \times 1}$
      end if
      $\hat{\boldsymbol{\beta}}^{\text{MAB}_k}_{u_t} = (\mathbf{A}^{\text{MAB}_k}_{u_t})^{-1} \mathbf{b}^{\text{MAB}_k}_{u_t}$
      $p_a \leftarrow \mathbf{r}^\top_a \hat{\boldsymbol{\beta}}^{\text{MAB}_k}_{u_t} + \gamma \sqrt{\mathbf{r}^\top_a (\mathbf{A}^{\text{MAB}_k}_{u_t})^{-1} \mathbf{r}_a}$
    end for
    $\hat{a}^t_k \leftarrow \text{argmax}_{a \in \mathcal{A}_t} p_a$
    if $\hat{a}^t_k \in \{a^t_1, \ldots, a^t_{k-1}\}$ **then** ▷ Avoid repeats
      $a^t_k \leftarrow$ arbitrary unselected resource $\in \mathcal{A}'_t$
    else $a^t_k \leftarrow \hat{a}^t_k$
    end if
  end for
  $\tilde{\mathcal{A}}_t \leftarrow \{a^t_k | 1 \leq k \leq K\}$
  return $\tilde{\mathcal{A}}_t$
end function
function $\text{UPDATE}(\text{MAB}_k, f^t_k, \mathbf{r}^t_{a^t_k}, u_t)$
  $\mathbf{A}^{\text{MAB}_k}_{u_t} \leftarrow \mathbf{A}^{\text{MAB}_k}_{u_t} + \mathbf{r}^t_{a^t_k} (\mathbf{r}^t_{a^t_k})^\top$
  $\mathbf{b}^{\text{MAB}_k}_{u_t} \leftarrow \mathbf{b}^{\text{MAB}_k}_{u_t} + f^t_k \mathbf{r}^t_{a^t_k}$
end function

D. Objective Reward As A Linear Function of Context

We refer to the reward r^t_{i,a^t_k} for each objective i, as the objective reward. The objective reward r^t_{i,a^t_k} for resource a^t_k is

a function of the context vector $\mathbf{x}^t_{i,a^t_k}$ and it is parameterized by an unknown coefficient vector $\boldsymbol{\theta}^*_{i,a^t_k}$ that is specific for each resource and objective pair. For each resource a^t_k, we modeled the expected reward r^t_{i,a^t_k} in each objective i as a linear function of the observed context vector $\mathbf{x}^t_{i,a^t_k}$ with the coefficients $\boldsymbol{\theta}^*_{i,a^t_k}$. Namely for all t,

$$\mathbb{E}[r^t_{i,a}|\mathbf{x}^t_{i,a}] = (\mathbf{x}^t_{i,a})^\top \boldsymbol{\theta}^*_{i,a}. \quad (5)$$

As rewards for each objective can be different in various contexts, we modeled the expected reward $r^t_{i,a}$ in each objective separately for each arm. Note that $r^t_{i,a}$ is the output of function Θ given in Equation 3.

E. Exploration And Exploitation With UCB

The estimated utility reward f^t_k is computed using unknown coefficient vectors $\boldsymbol{\beta}^{k*}_{u_t}$. The estimated objective reward r^t_{i,a^t_k} is computed using unknown coefficient vectors $\boldsymbol{\theta}^*_{i,a^t_k}$. Therefore, we need to trade off exploration and exploitation to select the best arm to display in each rank and learn the unknown coefficients to minimize the regret over the T trials. MOR-LinUCB tradeoff exploration and exploitation by computing the upper confidence bound (UCB) [3] of the expected utility value and the multiple objective rewards. Once each MAB_k instance has selected the resource a^t_k for rank k, the ranked list $\tilde{\mathcal{A}}_t$ is displayed to the user.

F. Updating Estimated Coefficient Vector

On each trial t, a user u_t considers the ranked list $\tilde{\mathcal{A}}_t$, generated as explained above, in order and selects relevant resources and reveals multiple objective rewards for selected resources. The algorithm uses the observed utility function value (selected or not) and the observed rewards in multiple objectives to update its coefficient vectors as follows. Let a^{t*}_k be a selected resource in trial t at rank k and $\mathbf{r}^t_{a^{t*}_k}$ be the multiple objective rewards for a^{t*}_k. Based on $\mathbf{r}^t_{a^{t*}_k}$ MOR-LinUCB updates the estimated $\boldsymbol{\theta}^*_{i,a^{t*}_k}$ as shown in Algorithm 1 (lines 21 - 26) to minimize the objective reward estimation error in each objective. Additionally, based on the observed utility function value, MOR-LinUCB updates the estimated $\boldsymbol{\beta}^{k*}_{u_t}$ in each MAB_k and learns the user utility function over the multiple objectives for each rank simultaneously. If a^{t*}_k was selected for the MAB_k instance and clicked by u_t the utility reward for the MAB_k instance, denoted as f^t_k, is 1 else it is 0. As we have used the LinUCB Disjoint algorithm [3] in each MAB_k instance, the respective coefficient vector will be updated as shown in Algorithm 2 (lines 19-22).

Using this model, MOR-LinUCB learns to estimate rewards based on the observed context in multiple objectives and learns the user's utility over the multi-objective space for ranking. In the evaluation section, we empirically show that MOR-LinUCB achieves competitive regret over the T trials compared to other algorithms that do not consider context information or multiple objectives.

G. Computing The UCB

In the MOR-LinUCB algorithm since the coefficients $\boldsymbol{\theta}^*_{i,a}$ are unknown and estimated over the T trials, there is an uncertainty in the output value which is the objective reward $r^t_{i,a}$. Thus, we compute the upper confidence bound (UCB) of the objective reward $r^t_{i,a}$ and use the UCB as the estimated objective reward. In each rank k based on the UCB of the multiple objective rewards in each resource and the estimated $\boldsymbol{\beta}^{k*}_{u_t}$ coefficients, the MOR-LinUCB algorithm gets a weighted UCB for the utility reward f^t_k for all the resources. MOR-LinUCB chooses to display the resource with the highest UCB for the utility reward f^t_k in each rank.

Suppose $\hat{\boldsymbol{\beta}}^k_{u_t}$ is the estimated coefficient vector in a given trial t for user u_t at rank k and f^t_k is the utility reward. Walsh et al. [11] have shown that if Ridge Regression is used to estimate $\hat{\boldsymbol{\beta}}^k_{u_t}$ the uncertainty of the utility reward f^t_k can be bounded as,

$$\left|(\mathbf{r}^t_a)^\top \hat{\boldsymbol{\beta}}^k_{u_t} - \mathbb{E}[f^t_k|\mathbf{r}^t_a]\right| \leq \gamma\sqrt{(\mathbf{r}^t_a)^\top (\mathbf{D}^{k\top}_{u_t}\mathbf{D}^k_{u_t} + \mathbf{I}_d)^{-1}\mathbf{r}^t_a} \quad (6)$$

where $\mathbf{D}_{u_t} \in \mathbb{R}^{m\times d}$ is the design matrix, $\mathbf{I}_d$ is the $d \times d$ identity matrix (d is the number of objectives and m is the number of training instances for user u_t), and γ is a constant. We use the one-sided confidence interval given above to trade off exploration and exploitation in MOR-LinUCB as shown in Algorithm 2 (line 8). As a result, we estimate the upper confidence bound of the utility reward f^t_k in trial t as,

$$(\mathbf{r}^t_a)^\top \hat{\boldsymbol{\beta}}_{u_t} + \gamma\sqrt{(\mathbf{r}^t_a)^\top (\mathbf{D}^\top_{u_t}\mathbf{D}_{u_t} + \mathbf{I}_d)^{-1}\mathbf{r}^t_a} \quad (7)$$

IV. Evaluation

We benchmark our algorithm with a modified PUCB1 algorithm [12] and a modified LinUCB algorithm suitable for ranking. PUCB1, as mentioned in the Related Work, is a version of a UCB algorithm that considers Pareto-optimality for resource selection. The evaluation measures we used are as follows [4], [13].

1) *Click Through Rate (CTR)*: The ratio of the average number of clicks an algorithm received and the maximum number of clicks the algorithm can receive.
2) *Mean Reciprocal Rank (MRR)*: When T is the total number of trials and k_t is the rank of the clicked resource in trial t, then $MRR = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{k_t}$
3) *Precision at rank k (P@k)*: Represents the accuracy of the model in retrieving the correct document at the k^{th} rank. $\text{P@}k = \frac{\text{\# of relevant items @}k}{\text{\# of recommended items @k}}$

A. Evaluation On A Synthetic Dataset

We simulated a synthetic dataset where we randomly sampled the optimal objective reward coefficient vectors $\boldsymbol{\theta}^*_{i,a}$ and user utility coefficient vectors $\boldsymbol{\beta}^*_u$ from a uniform grid. We then evaluated how each algorithm learned the predefined optimal coefficient vectors in the MOCR-B setting. To be precise, the simulation process was as follows.

1) Generating The Synthetic Dataset: Suppose $\mathcal{A}_t$ denotes the set of simulated resources and l is the dimension of the context feature vector. To obtain the context modeling coefficient vectors $\boldsymbol{\theta}^*_{i,a}$, we drew unit vectors uniformly from $\mathbb{R}^l$ for each objective i in each arm $a \in \mathcal{A}_t$. Suppose $\mathcal{U}$ denotes the set of simulated users and d is the number of objectives. To obtain the users utility modeling coefficient vectors $\boldsymbol{\beta}^*_u$, we drew unit vectors uniformly from $\mathbb{R}^d$ for all $u \in \mathcal{U}$. In trial t, we sampled the contexts $\mathbf{x}^t_{i,a} \in \mathbb{R}^l$ from a uniform distribution and a user u was picked randomly from $\mathcal{U}$. When an algorithm returned a ranked list $\tilde{\mathcal{A}}_t$ of size K in trial t, the simulator calculates the *actual* objective reward $r^t_{i,a}$ for each objective i in all $a \in \tilde{\mathcal{A}}_t$ as $r^t_{i,a} = \boldsymbol{\theta}^{*\top}_{i,a}\mathbf{x}^t_{i,a} + \upsilon_t$, where υ_t is a random noise. Suppose $\mathbf{r}^t_a = (r^t_{i,a} | 1 \leq i \leq d)$ contains the actual rewards for all objectives for an arm a. Then in the simulation, a resource in $\tilde{\mathcal{A}}_t$ is considered to be "clicked" if $(\mathbf{r}^t_a)^\top \boldsymbol{\beta}^*_u \geq 0.5$. If at least one resource in the ranked list $\tilde{\mathcal{A}}_t$ has a payoff greater than 0.5, we consider it as a positive reward for the displayed ranked list. The simulation parameters were set as $T = 1000, K = 10, |\mathcal{A}_t| \in \{10, \ldots, 30\}, i \in \{2, \ldots, 10\}$, $|\mathcal{U}| = 10$ and $l \in \{2, \ldots, 10\}$.

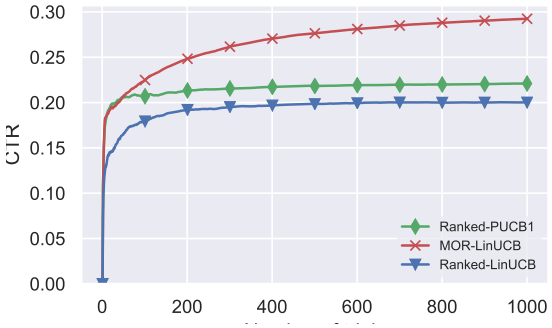

Fig. 2: CTR for the ranked lists over 1000 trials.

2) Click Through Rate Analysis: Figure 2 shows the CTR of the ranked lists over the T trials for MOR-LinUCB and other benchmark algorithms in the synthetic dataset. A higher CTR for MOR-LinUCB confirms it ranked a larger number of relevant resources for users compared to other algorithms. MOR-LinUCB differs from Ranked-PUCB1 in two ways; (1) MOR-LinUCB models context information over the multiple objectives while Ranked-PUCB1 is context-free, and (2) MOR-LinUCB learns user preferences over the multi-objective space to rank the items while Ranked-PUCB1 considers arms in the Pareto-front to be equally likely. As a result MOR-LinUCB is more robust in learning the optimal ranked list in different context for different users. Furthermore, the CTRs for Ranked-LinUCB, which only models the context information for a single objective, is considerably lower than MOR-LinUCB. Therefore, we argue that modeling context information across multiple objectives is important for a context-aware ranking.

As a first attempt to solve the MOCR-B problem we modeled the rewards to be linear functions of contextual information and learned the coefficient vectors separately, instead of conjunctively. It may be argued that if rewards are modeled as non-linear functions of the contextual information and user preferences, and the coefficient vectors are learned conjunctively then non-convex points in the multi-objective optimization may get captured better. We leave this enhancement to MOR-LinUCB as future work.

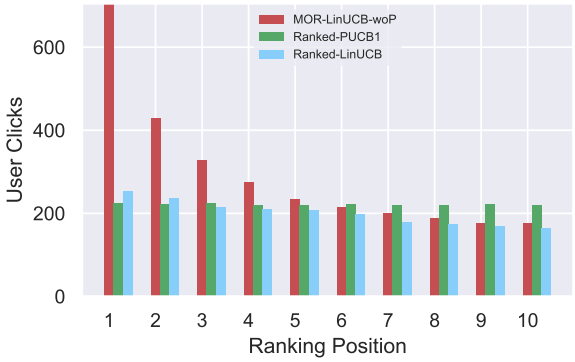

Fig. 3: User clicks for each rank $k \in \{1, \ldots, 10\}$.

3) MRR And P@1 Analysis: To get an intuition about the click distribution across the ranks in each algorithm we plot the number of user clicks in each rank over the T trials in Figure 3. As can be seen in Figure 3, the algorithm with the highest number of clicks in rank 1 is MOR-LinUCB. Also the number of clicks for each subsequent rank decreases at a higher rate for MOR-LinUCB compared to Ranked-PUCB1 and Ranked-LinUCB algorithms. These results suggest that the accuracy of the MOR-LinUCB model in retrieving the correct (clicked) document at the first position is higher compared to other benchmark algorithms. To further validate this argument we present the MRR and P@1 values for the MOR-LinUCB algorithm in Table I. The MRR value 0.59 for MOR-LinUCB algorithm suggests that on average users find relevant resources ranked in the 1^{st} or 2^{nd} positions. On average the relative improvement of the MRR values in MOR-LinUCB compared to Ranked-PUCB1 and Ranked-LinUCB is 0.92 and 1.19 respectively. Similarly, as can be seen in Table I, MOR-LinUCB has a higher P@1 value compared to other algorithms. These results further strengthen our argument that MOR-LinUCB is more effective in context-aware multi-objective ranking.

Method	**CTR**	**MRR**	**P@1**	**P@2**	**P@3**	**P@4**	**P@5**
MOR-LinUCB	0.26	0.79	0.71	0.56	0.48	0.42	0.39
Ranked-PUCB1	0.22*	0.41*	0.22*	0.22*	0.22*	0.22*	0.22*
Ranked-LinUCB	0.19*	0.36*	0.25*	0.23*	0.22*	0.22*	0.21*

TABLE I: CTR, MRR and P@1 values for the synthetic dataset. The $*$ indicates statistically significant differences between the MOR-LinUCB and other algorithms.

B. Evaluation On A Real-World Dataset

To evaluate MOR-LinUCB on a real-world dataset, we used the TripAdvisor [14]–[16] data. The dataset has information about user profiles, hotels profiles, hotel reviews, date, location, number of readers for reviews, number of helpful votes for reviews and overall rating for a hotel. Additionally it has the explicit ratings for 8 different objectives. They are value aspect rating, rooms aspect rating, location aspect rating, cleanliness aspect rating, check in/front desk aspect rating, service aspect rating, sleep quality aspect rating and business service aspect rating. Aspect ratings ranges from 0 to 5 stars. This dataset consists of 878,561 reviews from 4,333 hotels crawled from TripAdvisor. Once the invalid entries, users with less than 10 reviews and hotels with just 1 review were removed the dataset contained 777 users and 1875 hotels.

The online ranking task in this dataset is to generate a ranked list of hotels for different users in different contexts considering their preferences for different aspects in hotels. We used MOR-LinUCB to present a ranked list of *potential* hotels to the users. The size of the list was 10. User feedback for the MOR-LinUCB output was captured over 5000 trials. If the ranked list contained a hotel rated by the user, it was considered as a user click.

1) Feature Extraction: As we mentioned before, we incorporate the context features, the users features and the hotel features to make accurate prediction. To specify which hotel is chosen by a certain user, we also need to consider the hotelID and userID as features. However, the vocabulary size of the categorical values can be very large (e.g., there are 1875 hotels and 777 uses in our dataset), which will lead to extremely heavier computational costs.

To address this issue, inspired by the success of embedding methods in different research areas [8]–[10], [17], [18], we also employ such method to encode these categorical features. Comparing with the one-hot encoding [19], the embedding method mainly has two advantages. First, the embedding method effectively reduces the input dimension and thus it is more computationally efficient to our algorithm. Moreover, it has been shown that the categorical values with similar semantic meaning are usually embedded to the close locations [19]. Thus, the embedding method helps find and share similar patterns among different hotels as well as users. Notice that location also plays an important role in our recommendation system [20], we also embed locationID into shorter representations.

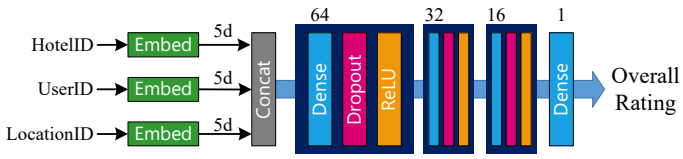

Fig. 4: Network structure for embedding task, where the output dimension of dense layers are indicated over themselves respectively.

In this study, a supervised learning framework is utilized to obtain the embedding vectors. We use userID, hotelID and locationID as inputs to predict the overall rating of each transaction. More specifically, as depicted in Figure 4, we embed userID, hotelID and locationID to $\mathbb{R}^5$ respectively. The embedding vector is randomly initialized but will be refined during the training phase. Once we get their embedding vectors, we simply concatenate them as the input of a multi-layer perceptron to perdict the overall rating. During the experiments, we adopt Adam [21] optimizer to train the parameters. The learning rate of Adam is $5e^{-4}$ and the batch size during training is 32. Our model is implemented with PyTorch 1.0 on the server with one Titan V.

In order to run the MOR-LinUCB algorithm we concatenated the learned embedding vectors with additional context features. For example, when modeling the location aspect rating for a given user and hotel, we concatenated the users embedding vector, hotels embedding vector, location embedding vector, users context vector and the hotels context vector. The dimension of the final context feature vector was $\mathbb{R}^{22}$.

2) Experimental Results For CTR: Figure 5 shows the CTR for MOR-LinUCB, Ranked-LinUCB, and Ranked-PUCB1 for the Tripadvisor dataset. MOR-LinUCB obtains a higher CTR for the ranked lists it displays to the user compared to the other two approaches. The results imply that users find suitable hotels more often in the ranked list generated by MOR-LinUCB compared to other algorithms. The relatively high

Fig. 5: CTR for the ranked lists presented to TripAdvisor users

CTR for MOR-LinUCB compared to Ranked-LinUCB shown in Figure 5, provides evidence that there is value addition in optimizing the ranking for multiple objectives in dynamic environments instead of optimizing for a single objective. The improved results for MOR-LinUCB can be attributed to the diverse needs of users when choosing hotels to stay in different contexts. Some users would always choose hotels that are in a good location but may be have a lower star-rate. Other users may prefer hotels with a higher star-rate but not necessarily in a good location. Since MOR-LinUCB can capture these diverse preferences over multiple objectives in various context, in contrast to a single objective ranking approach, we see the increased CTR.

Fig. 6: User-clicks for each rank in the TripAdvisor dataset.

3) Experimental Results for MRR and P@1: Next, we evaluate the quality of ranking in each of the algorithms. Figure 6 plots the number of clicks in the top 10 ranks after 5000 trials. MOR-LinUCB has significantly more clicks in rank 1 compared to other ranks. On the contrary, the clicks are distributed more uniformly across the ranks in Ranked-PUCB1 and Ranked-LinUCB algorithms. These dissimilarities suggest that by modeling context information in multiple objectives and considering user preferences over the multiple objectives MOR-LinUCB has captured more information about the users' search intends in TripAdvisor compared to Ranked-PUCB1 and Ranked-LinUCB. We present the MRR and P@1 values based on the top 2 ranks in Table II to validate our argument further.

Method	CTR	MRR	P@1	P@2
MOR-LinUCB	0.033	0.073	0.0282	0.019
Ranked-PUCB1	0.016	0.016	0.0052	0.0054
Ranked-LinUCB	0.024	0.042	0.0144	0.0152

TABLE II: CTR, MRR and P@1 values for the TripAdvisor dataset.

As shown in Table II there is a significant difference in the P@1 values for MOR-LinUCB in comparison to other benchmark algorithms. Likewise, the MRR value for MOR-LinUCB is significantly higher than that of Ranked-PUCB1 and Ranked-LinUCB. The results confirm that MOR-LinUCB accurately retrieves the relevant hotels in the top ranks compared to other approaches. MRR values confirm that when using MOR-LinUCB to rank resources in TripAdvisor, users find the relevant resources ranked in the top ranking positions on average, whereas when using Ranked-PUCB1 and Ranked-LinUCB users find the relevant resources across the ranked list on average.

We show the benefit of MOR-LinUCB empirically. A theoretical regret analysis of the MOCR-B problem and MOR-LinUCB will be a valuable addition in future. However, it was not the focus of our paper and remains as future work.

V. Related Work

In this section, we relate our work to similar bandit approaches that have focused on context-awareness, multiple objectives, and ranking. Existing bandit approaches such as LinUCB [3], [22], [23] primarily focus on context-awareness but not ranking or multiple objectives. PUCB1 [24], and Pareto Thompson sampling [25] algorithms focus on multi-objective optimization. Auer et al. [26] presented two algorithms to return all Pareto optimal points in a stochastic bandit feedback setting based on an elimination algorithm and UCBs. Turgay et al. have studies the problem of the presense of multiple objectives in a MAB setting with similarity information [27]. Above methods identify a Pareto-front, but in many practical problems, we need to order the Pareto-front based on a user/domain specific preference. MOR-LinUCB learns the preferences over the multiple reward objectives over time and ranks the Pareto-front.

Ranked Bandits [28] is a bandit approach for ranking. The key characteristic of ranked bandits algorithm is that each position in the list is an independent bandit problem, which is solved by some bandit algorithm. As a result, ranked bandits generate a diversified ranked list that maximizes click-through rates when learning to rank web resources. Using the MOR-LinUCB algorithm, we show how the ranked bandits algorithm can be used to determine the users utility over multiple objectives to populate a ranked list.

Several studies have looked into the fusion of contextual bandits, multi-objective multi-arm bandits and ranked bandits. Tekin et al. [29] defined a multi-objective contextual bandit problem for two objectives in which they have explicitly defined one objective as the dominant, and the other as the non-dominant. The MOC-MAB algorithm presented in their work maximizes the long-term reward of the non-dominant objective conditioned on the fact that it maximizes the long-term reward of the dominant objective. In contrast, our work does not assume that there is a dominant objective. Instead, we learn the user's utility over the multi-objective space. Otunba et al. [30] proposed an offline multi-objective pairwise ranking model that provides item recommendation and audience retrieval simultaneously using a scalarized approach for multi-objective optimization. When using an offline scalarized approach, the challenge lies in determining the appropriate scalarization function to use and its parameterization. Other studies [12], [24] have shown that Pareto-optimality, is more efficient than scalarization functions in finding the Pareto-front, but they require a large number of evaluations to have a reliable estimation of the objective values if the Pareto-optimal set is large. We tackle this problem by learning the unknown utility function per user across multiple reward objectives. We use context information and users feedback and learn the utility function. An existing work [31] that studies a problem similar to the MOCR-B problem uses a utility based view for multi-objective optimization in a bandit setting. However, the algorithms Utility-MAP UCB and Interactive Thompson Sampling [31] presented in their work does not rank resource

and there is an expensive pairwise reward vector comparison to learn user utility. MOR-LinUCB does not require pairwise comparisons as the users' preferences are learned independently based on context vectors. In summary, the value added by our work is the robust fusion of context-awareness, multi-objective modeling and ranking techniques in a bandit setting.

VI. Conclusion

There are important real-world online settings which require algorithms to provide users with ranked lists of relevant web resources based on rewards in multiple objectives. To address this online ranking problem, we defined the Multi-Objective Contextual Ranked Bandit (MOCR-B) problem and introduced one solution, the MOR-LinUCB algorithm. MOR-LinUCB manages the exploration-exploitation tradeoff in suggesting ranked lists and using user responses to discover: (1) models of how contextual information about users and resources maps to multiple reward objectives; and (2) models of how users utility of resources depends on multiple objective rewards. In both synthetic data and a real-world data set from a hotel recommender platform, we show that MOR-LinUCB leads to higher click-through rates and precision values than algorithms that do not consider context information or multiple reward objectives. We further illustrate how neural network embeddings can be used to encode the high-dimensional and unstructured contextual data into an uniformed representation for sequential decision making. Our work provides insight into how a wide range of future work can integrate ranking, sequential-decision making, contextual information, and multiple objective rewards for the online ranking problem.

References

[1] B. Xiang, D. Jiang, J. Pei, X. Sun, E. Chen, and H. Li, "Context-aware ranking in web search," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp. 451–458.

[2] H. Zamani, M. Bendersky, X. Wang, and M. Zhang, "Situational context for ranking in personal search," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 1531–1540.

[3] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 661–670.

[4] T.-Y. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225–331, Mar. 2009. [Online]. Available: http://dx.doi.org/10.1561/1500000016

[5] K. M. Svore, M. N. Volkovs, and C. J. Burges, "Learning to rank with multiple objective functions," in *Proceedings of the 20th international conference on World wide web*. ACM, 2011, pp. 367–376.

[6] P. Nguyen, J. Dines, and J. Krasnodebski, "A multi-objective learning to re-rank approach to optimize online marketplaces for multiple stakeholders," *arXiv preprint arXiv:1708.00651*, 2017.

[7] H. Robbins, "Some aspects of the sequential design of experiments," in *Herbert Robbins Selected Papers*. Springer, 1985, pp. 169–177.

[8] L. A. Amorim, M. F. Freitas, A. Dantas, E. F. de Souza, C. G. Camilo-Junior, and W. S. Martins, "A new word embedding approach to evaluate potential fixes for automated program repair," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[9] S. Zheng, H. Bao, J. Xu, Y. Hao, Z. Qi, and H. Hao, "A bidirectional hierarchical skip-gram model for text topic embedding," in *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016, pp. 855–862.

[10] A. Albuquerque, T. Amador, R. Ferreira, A. Veloso, and N. Ziviani, "Learning to rank with deep autoencoder features," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[11] T. J. Walsh, I. Szita, C. Diuk, and M. L. Littman, "Exploring compact reinforcement-learning representations with linear regression," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2009, pp. 591–598.

[12] M. M. Drugan and A. Nowé, "Designing multi-objective multi-armed bandits algorithms: A study," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–8.

[13] N. Fuhr, "Some common mistakes in ir evaluation, and how they can be avoided," in *ACM SIGIR Forum*, vol. 51, no. 3. ACM, 2018, pp. 32–41.

[14] L. Liu, N. Mehandjiev, and D.-L. Xu, "Multi-criteria service recommendation based on user criteria preferences," in *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 2011, pp. 77–84.

[15] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis on review text data: a rating regression approach," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 783–792.

[16] ——, "Latent aspect rating analysis without aspect keyword supervision," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 618–626.

[17] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "Geoman: Multi-level attention networks for geo-sensory time series prediction," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 2018, pp. 3428–3434.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[19] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Advances in neural information processing systems*, 2016, pp. 1019–1027.

[20] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*. Springer, 2011, pp. 217–253.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] N. Wanigasekara, J. Schmalfuss, D. Carlson, and D. S. Rosenblum, "A bandit approach for intelligent iot service composition across heterogeneous smart spaces," in *Proceedings of the 6th International Conference on the Internet of Things*. ACM, 2016, pp. 121–129.

[23] N. Wanigasekara, "A semi lazy bandit approach for intelligent service discovery in iot applications," in *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2015, pp. 503–508.

[24] M. M. Drugan and A. Nowé, "Scalarization based pareto optimal set of arms identification algorithms," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 2690–2697.

[25] S. Yahyaa and B. Manderick, "Thompson sampling for multi-objective multi-armed bandits problem," in *Proceedings*. Presses universitaires de Louvain, 2015, p. 47.

[26] P. Auer, C.-K. Chiang, R. Ortner, and M. Drugan, "Pareto front identification from stochastic bandit feedback," in *Artificial Intelligence and Statistics*, 2016, pp. 939–947.

[27] E. Turğay, D. Öner, and C. Tekin, "Multi-objective contextual bandit problem with similarity information," *arXiv preprint arXiv:1803.04015*, 2018.

[28] F. Radlinski, R. Kleinberg, and T. Joachims, "Learning diverse rankings with multi-armed bandits," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 784–791.

[29] C. Tekin and E. Turgay, "Multi-objective contextual multi-armed bandit problem with a dominant objective," *arXiv preprint arXiv:1708.05655*, 2017.

[30] R. Otunba, R. A. Rufai, and J. Lin, "Mpr: Multi-objective pairwise ranking," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 170–178.

[31] D. M. Roijers, L. M. Zintgraf, and A. Nowé, "Interactive thompson sampling for multi-objective multi-armed bandits," in *International Conference on Algorithmic DecisionTheory*. Springer, 2017, pp. 18–34.