Multi-Task Combinatorial Bandits for Budget Allocation

Lin Ge
linvge@amazon.com
Amazon
Seattle, WA, USA

Yang Xu
yxu63@ncsu.edu
North Carolina State University
Raleigh, NC, USA

Jianing Chu
jianichu@amazon.com
Amazon
Seattle, WA, USA

David Cramer
dxcramer@amazon.com
Amazon
Seattle, WA, USA

Fuhong Li
fuhonli@amazon.com
Amazon
Seattle, WA, USA

Kelly Paulson
paulsonk@amazon.com
Amazon
Seattle, WA, USA

Rui Song
ruisong@amazon.com
Amazon
Seattle, WA, USA

Abstract

Today's top advertisers typically manage hundreds of campaigns simultaneously and consistently launch new ones throughout the year. A crucial challenge for marketing managers is determining the optimal allocation of limited budgets across various ad lines in each campaign to maximize cumulative returns, especially given the huge uncertainty in return outcomes. In this paper, we propose to formulate budget allocation as a multi-task combinatorial bandit problem and introduce a novel online budget allocation system. The proposed system: i) integrates a Bayesian hierarchical model to intelligently utilize the metadata of campaigns and ad lines and budget size, ensuring efficient information sharing; ii) provides the flexibility to incorporate diverse modeling techniques such as Linear Regression, Gaussian Processes, and Neural Networks, catering to diverse environmental complexities; and iii) employs the Thompson sampling technique to strike a balance between exploration and exploitation. Through extensive empirical evaluations with both synthetic data and real-world data from Amazon campaigns, our system demonstrates robustness and adaptability, effectively maximizing the overall cumulative returns. A Python implementation of the proposed procedure is available at https://anonymous.4open.science/r/MCMAB.

CCS Concepts

• **Computing methodologies** → **Sequential decision making**;
• **Information systems** → **Computational advertising**.

Keywords

Online Advertising, Budget Allocation, Combinatorial Bandits, Meta Bandits

ACM Reference Format:
Lin Ge, Yang Xu, Jianing Chu, David Cramer, Fuhong Li, Kelly Paulson, and Rui Song. 2025. Multi-Task Combinatorial Bandits for Budget Allocation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3690624.3709434

KDD '25, Toronto, ON, Canada.

ACM ISBN 979-8-4007-1245-6/25/08
https://doi.org/10.1145/3690624.3709434

1 Introduction

Budget allocation has been given wide attention in various fields across advertising market [7, 18, 33], clinical trial [9, 23], transportation [22, 28], and more. As a motivating example, advertisers and agencies that use Demand Side Platforms (DSP), like Amazon DSP, routinely manage hundreds of simultaneous campaigns. Each campaign encompasses various ad lines targeting specific audiences with diverse delivery settings. Daily, marketing managers allocate budgets across ad lines for each campaign within a daily budget, aiming to boost traffic to retail websites or maximize product sales. A key obstacle marketing managers face is the lack of understanding of the relationship between ads spending and performance outcomes, which, once obtained, reduces the task to an optimization problem solvable by various methods [10, 15].

Today's ADSP implements automated online performance optimizations that respond to signals on each campaign's own spend and conversion data. Such an approach often encounters two significant challenges: **First**, advertisers frequently launch campaigns sequentially or run multiple campaigns simultaneously. Without a design in the ADSP that effectively coordinates learning from past and concurrent campaigns, the learning process is inefficient. Ad lines typically begin with an equal budget distribution at the start of a campaign. Although automated tools can make real-time budget adjustments during campaigns, determining the optimal budget allocation mix can take up to three weeks. This delay would result in a considerable number of ineffective ad deliveries, a general issue particularly severe for short-lived campaigns and large advertisers or agencies participating in numerous campaigns each

year. **Second**, measurement uncertainty exists, primarily due to the lack of real-life counterfactual observations, necessitating the use of estimations. This challenge is exacerbated by the dynamic nature of advertising and business environments, where factors like seasonality, competitive bidding, privacy regulations, DSP functionality, and fluctuating customer behaviors make it even more complex to accurately estimate these values. Relying solely on historical data can lead to suboptimal solutions in sequential decision-making scenarios. Thus, the following question is addressed in this paper: ***How can we wisely utilize i) learnings from past campaigns and ii) insights from other ongoing concurrent campaigns to accelerate the learning process for optimal budget allocation?***

To address the inherent uncertainty in digital advertising, Bandits algorithms are known for effectively balancing exploration and exploitation and have recently been applied to budget allocation, formalizing it as a combinatorial multi-armed bandit (CMAB) problem [19, 20, 37]. However, none of them investigated how to share information across multiple ad lines and campaigns. Figure 1 depicts the average clicks received for different ad lines from various campaigns, with varying levels of budget assigned, underscoring the significant influence of ad/campaign characteristics on budget-performance relationships. Additionally, according to Figure 1, it is important to note that the available features do not fully determine the expected performance of an ad line with an assigned specific budget. In other words, even when conditioned on informative features, the expected performance for a given budget level still exhibits a degree of variability that cannot be captured by features (referred to as *inter-arm heterogeneity*). To strategically boost information sharing across tasks, we model budget allocation in ADSP as a multi-task CMAB problem where each ad campaign represents a single CMAB task, and introduce a feature-based Bandit algorithm augmented by a Bayesian hierarchical model.

Contributions. Our main contributions are multi-fold.

First, our study is the first, to our knowledge, to investigate budget allocation for online advertising from the perspective of platforms and agencies managing multiple campaigns, with a meta objective of optimizing performance over the campaign distribution. This contrasts with existing studies that primarily focus on optimizing a single campaign from an advertiser's viewpoint [12, 19, 20].

Second, to capture the budget-performance relationship, we propose a general Bayesian hierarchical model. Driven by similar motivations, Han and Arndt [12] introduced a contextual bandit-based system using augmented data generated from a global model to share information across ad lines. However, not accounting for the uncertainty of the fitted global model, its effectiveness relies heavily on the global model's performance and can lead to suboptimal decisions, especially in data-limited scenarios. In contrast, through the construction of the Bayesian hierarchical model, which incorporates an arm-specific random effect to capture the information not being explained by the feature information, we effectively tackle the inter-arm heterogeneity observed in Figure 1.

Third, we develop algorithms to support both parametric and non-parametric modeling. While none of the aforementioned work addresses the ubiquitous inter-arm heterogeneity, recent advancements in meta bandits [29, 30] have begun to address it under simplified settings, relying heavily on parametric linear assumptions (detailed in Section 2). Our approach distinguishes itself by allowing for non-parametric modeling, thereby relaxing these conventional assumptions that frequently fail in practice, as shown in Figure 1. We demonstrate the applicability of our method to more complex nonlinear budget-performance relationships by leveraging Gaussian Process and Neural Network in prior specification, as two instances.

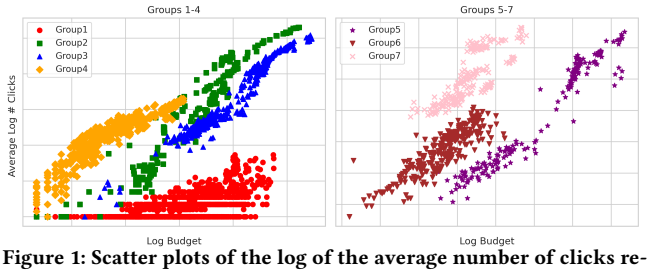

Figure 1: Scatter plots of the log of the average number of clicks received and the log of the budget allocated. Observations are classified based on factors such as the advertiser's industry, channel, supply source, and target audience.

Finally, we implemented our proposed framework in an offline study using real campaign data from ADSP and deployed it for an online experiment. The results consistently indicate that our framework achieves faster convergence and lower cumulative regret (i.e., higher cumulative reward), thereby leading to a better budget allocation strategy in the long term. This is also supported by extensive simulations validating its enhanced performance.

2 Related Work

Budget allocation has been widely discussed in the past decades. Different from traditional recommendation systems that employ bandits for personalized suggestions, the budget allocation problem poses unique challenges due to two main reasons: (1) budget is naturally continuous, which results in a continuous action space in the bandits problem and is thus more tricky to address; and (2) the resource constraint needs to be satisfied. We summarize several lines of work closely related to our proposed approach below.

Budget Allocation without Exploration. The first strand of literature formalizes budget allocation as a pure optimization problem [4, 10, 34, 36]. In the context of online advertising, particularly real-time bidding, DSPs continuously seek the optimal attribution strategy to maximize expected total rewards. Traditional methods explicitly define impression value, budget consumption, and resource constraints within an optimization problem, obtaining a solution through Lagrangian methods such as Karlsson [15]. To improve estimation efficiency, Geng et al. [10] introduced an "active-set" optimization algorithm that jointly optimizes bid prices and budgets, avoiding direct estimation of complex input functions from data. Further details can be found in the survey by Ou et al. [21]. However, as noted in the previous section, this approach lacks the flexibility to explore new resource allocation strategies, potentially resulting in suboptimal solutions.

Bandits with Resource Constraints. To handle the trade-off between exploration and exploitation, recent literature focuses on constrained optimization in bandits. Depending on how to set budget constraints, existing work can be roughly divided into two categories:

The first type of approach formalize the problem as *bandits with knapsack* [1–3, 25]. Instead of setting separate budget limits at each round, these approaches assume a d-dimensional consumption vector aggregated over time. The algorithm stops when any consumption factor reaches its budget limit. Although closely related to our general topic, this approach investigates cumulative constraints that ensure cumulative budget consumption remains within limits over time, differing from our focus on step-wise constraints requiring adherence to budget limits at each decision point.

Our work has more in common with the second type of work, which involves splitting the budget within each round under the framework of CMAB [11, 19, 20, 31, 32, 37]. Here, budgets are discretized into finite proportions, aligning with the nature of combinatorial bandits in slate recommendation. Notably, Zuo and Joe-Wong [37] leveraged the CMAB framework by defining a super arm as an (ad line, budget) tuple for action assignment, and naturally extended this idea to continuous budget allocation scenarios with additional Lipschitz continuity assumptions. In the work by Xu et al. [32], the authors studied resource allocation problem with concave objective and fairness constraints. Nuara et al. [19, 20] proposed a joint bid/budget optimization algorithm based on Bayesian bandits update with Gaussian process. In the work by Gupta et al. [11], the authors proposed a correlated combinatorial bandit framework to capture the structural correlations between reward functions.

However, both categories of approaches outlined above either fail to utilize contextual information [3, 25, 31, 37] or rely on restrictive modeling assumptions in rewards and resource consumptions [1, 2, 32], limiting their applicability to more general applications. Recently, Han and Gabor [13] introduced a two-level system with a global model that uses a random machine learning regression model to capture the relationship between contextual features and the reward and then generates pseudo-observations to assist in updating local linear Thompson Sampling (TS) agents for final budget optimization. However, its performance would highly rely on the accuracy of the fitted global model, which is unrealistic given that no perfect prediction is guaranteed in the real world. In contrast, we use a single TS agent with a Bayesian hierarchical model (3) as the backbone to share information, accounting for the inherent variability in the expected reward around g (i.e., inter-arm heterogeneity) that is not captured by features, thereby enhancing the robustness of the learning process.

Multi-Task Bandits and Meta Bandits. To handle the budget allocation across multiple campaigns, it is important to conduct information sharing via a multi-task framework such as meta bandits [6, 17, 29, 30]. Notably, Kveton et al. [17] proposed a meta Thompson Sampling algorithm to address the update of multiple sequential bandit instances, assuming the reward vectors of each task are sampled from some prior distribution without using feature information. To enhance learning efficiency further, Wan et al. [29] proposed a multi-task bandits (MTB) algorithm by incorporating the feature information via a Bayesian hierarchical model. However, typical MTB and meta bandits are restricted to parametric models in reward modeling procedure (such as linear mixed effects model). Motivated by the complex environment in budget allocation problems, we extend this framework to constrained CMAB settings and incorporate state-of-the-art nonparametric models, which is nontrivial and requires further derivations.

3 Preliminaries

3.1 Budget Allocation

Consider a platform hosting a collection of M advertising campaigns running either concurrently or sequentially. In each campaign $m \in [M] = \{1, 2, \ldots, M\}$, there are K_m ad lines designated for daily budget allocation, and the campaign duration is denoted as T_m. For a given campaign m, we assume a total daily budget of B_m. At each time round t, the budget assigned to each ad line $k \in [K_m] = \{1, 2, \ldots, K_m\}$ is denoted as $a_{m,k,t}$ and a constraint exists such that the sum of the assigned budgets across all ad lines in campaign m satisfies $\sum_{k=1}^{K_m} a_{m,k,t} \leq B_m$. After assigning budget $a_{m,k,t}$ for each ad line k in campaign m, we consequently observe a random reward $R_{m,k,t}(a_{m,k,t})$. Our goal is to maximize the cumulative reward function across all ad lines, all campaigns and all rounds:

$$\begin{aligned} &\underset{a_{m,k,t}}{\text{maximize}} \sum_{m=1}^{M} \sum_{t=1}^{T_m} \sum_{k=1}^{K_m} \mathbb{E}\{R_{m,k,t}(a_{m,k,t})\}, \\ &\text{subject to} \sum_{k=1}^{K_m} a_{m,k,t} \leq B_m \quad \forall m, t. \end{aligned} \tag{1}$$

3.2 Combinatorial Multi-Armed Bandits

To connect the optimization problem described above with the framework of CMAB, we begin by introducing the fundamental concept of CMAB. A typical CMAB problem consists of K arms associated with a set of random variables $R_{k,t}$. The random variable $R_{k,t}$ indicates the random reward of arm $k \in [K] = \{1, 2, \ldots, K\}$ at time round t. The set of all possible subsets of arms is a power set $\mathcal{S} = 2^{[K]}$. We refer to every set of arms $S \in \mathcal{S}$ as a super arm and every arm in S as a base arm. At each time round, one super arm $S \in \mathcal{S}$ is played and the rewards of all base arms in this super arm are observed.

We consider each campaign as a single-task CMAB problem where each ad line within the campaign corresponds to a base arm. The first notable challenge is that, unlike the conventional CMAB problem where the decision revolves around whether to play the base arm or not, in our scenario, we must also determine the budget allocation for each chosen base arm. As the budget amount is continuous, we confront an infinite number of potential base arms. The second challenge arises from the presence of daily budget constraints. This implies that, at each round, only a subset of super arms is viable for play, restricted by the limitations imposed by the available daily budget. Even upon overcoming these challenges, conventional CMAB are designed to accommodate only a single campaign. However, advertisers often initiate new campaigns or run multiple campaigns concurrently. To mitigate the cold start issue associated with new campaigns and enhance data utilization, the proposed CMAB must facilitate information sharing across different campaigns.

4 Methodology

4.1 Problem Formulation

Without loss of generality, we define the continuous action space as $\mathcal{A} = [0, 1]$, where $a_{m,k,t} \in \mathcal{A}$ represents the proportion of the

total budget allocated to ad line k in campaign m at time round t. The corresponding budget can be expressed as $B_m \cdot a_{m,k,t}$. We further discretize the action space by partitioning the continuous budget into different proportions: $\mathcal{A}_d = \{0, \frac{1}{N}, \frac{2}{N}, \cdots, \frac{N-1}{N}, 1\}$, with N denoting a user-specified integer constant. The rationale behind the discretization is twofold. First, in practical scenarios, campaign budgets are commonly assigned in rounded percentages, such as 10% or 25%, rather than extremely precise amounts. Second, discretization eliminates the need for smoothness assumptions typically required for continuous budget optimization. Accordingly, we consider maintaining a set of base arms in campaign m as $\{(k, a) \mid k \in [K_m], a \in \mathcal{A}_d\}$. Each base arm contains metadata $x_{m,k}$,which comprises specific information on campaign and ad line configurations. At each time round t, we can only play one arm (k, a) for each ad line k in campaign m. This implies the allocation of a budget proportion $a_{m,k,t}$ to ad line k in campaign m. Let $\theta_{m,k,a} \equiv \mathbb{E}\{R_{m,k}(a)\}$. We can rewrite our goal as:

$$\begin{aligned} &\underset{a_{m,k,t}}{\text{maximize}} \sum_{m=1}^{M} \sum_{t=1}^{T_m} \sum_{k=1}^{K_m} \theta_{m,k,a_{m,k,t}}, \\ &\text{subject to} \sum_{k=1}^{K_m} a_{m,k,t} \le 1, \\ &\qquad \sum_{n=0}^{N} I\left(a_{m,k,t} = \frac{n}{N}\right) = 1, \quad \forall m, t. \end{aligned} \tag{2}$$

Thus, let $\boldsymbol{a}_{m,t} = (a_{m,1,t}, \cdots, a_{m,K_m,t})$, the full allocation space for each campaign m at each decision point is $\mathcal{S}_m = \{\boldsymbol{a}_{m,\cdot} \mid a_{m,k,\cdot} \in \mathcal{A}_d, \sum_{k=1}^{K_m} a_{m,k,\cdot} \le 1, \sum_{n=0}^{N} I\left(a_{m,k,\cdot} = \frac{n}{N}\right) = 1, \forall k\}$.

4.2 Multi-task Bayesian Hierarchical CMAB Framework

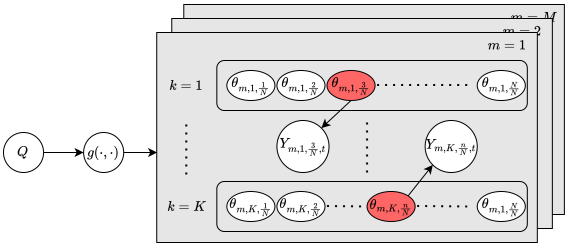

Figure 2: Graphical representation of multi-task Bayesian hierarchical CMAB framework. Red nodes are the selected base arm.

To tackle the optimization problem (2), a key step is to learn the reward distribution of $\theta_{m,k,a}$, which we propose to accomplish using the following Bayesian hierarchical model:

$$\begin{aligned} &\text{(Prior)} && \text{Prior information Q related to } g, \\ &\text{(Generalization)} && \theta_{m,k,a} \mid \mathbf{x}_{m,k}, a = g(\mathbf{x}_{m,k}, a) + \delta_{m,k,a}, \\ &&& \forall m \in [M], k \in [K_m], a \in \mathcal{A}_d, \\ &\text{(Observation)} && Y_{m,k,a_{m,k,t},t} = \theta_{m,k,a_{m,k,t}} + \epsilon_{m,k,t}, \\ &\text{(Reward)} && R_{m,t} = \sum_{k \in [K_m]} Y_{m,k,a_{m,k,t},t}, \end{aligned} \tag{3}$$

where $\epsilon_{m,k,t} \sim N(0, \sigma_\epsilon^2)$ is the random noise for some known σ_ϵ, $Y_{m,k,a,t}$ is the observed reward for ad line k in campaign m with an allocated budget of $B_m \cdot a$, and $R_{m,t}$ is the observed total reward aggregated across all ad lines within campaign m. The prior layer, Q, contains the prior belief of g, which can be derived from logged data or from domain experts, while the last two equations of (3) build up the classical assumption used in stochastic combinatorial bandits. See Figure 2 for a graphical illustration.

The essence of (3) lies in the two-way decomposition of $\theta_{m,k,a}$, which splits $\theta_{m,k,a}$ into i) $g(\mathbf{x}_{m,k}, a)$, an arbitrary model capturing the average impact of available features $\mathbf{x}_{m,k}$ and action a on the reward, and ii) $\delta_{m,k,a}$, a random effect that accounts for the inter-arm heterogeneity conditioned on $\mathbf{x}_{m,k}$ and a. Let $\boldsymbol{\delta}_m = \left[\delta_{m,1,0}, \cdots, \delta_{m,K_m,\frac{N}{N}}\right]$, we assume that $\boldsymbol{\delta}_m \sim N(\mathbf{0}, \Sigma)$ for some known covariance matrix Σ (e.g., $\sigma_m^2 \mathbf{I}$). Motivated by the observed fact that almost none of the off-the-shelf machine learning models can guarantee perfect estimation of g without error, no matter how informative $x_{m,k}$ is, $\delta_{m,k,a}$ is added for accounting for the additional randomness of $\theta_{m,k,a}$ that can not be explained by $x_{m,k}$. Intuitively, as such, we utilize i) the shared information across campaigns and ad lines via g and ii) the observations $Y_{m,k,a_{m,k,t},t}$ from all correlated base arms, to infer $\theta_{m,k,a}$.

For g, either a parametric or non-parametric model can be used. In this work, we consider three working models as examples: i) Linear regression (LR), assuming $g(\mathbf{x}_{m,k}, a) = \phi(\mathbf{x}_{m,k}, a)^T \boldsymbol{\gamma}$, where $\phi(\mathbf{x}_{m,k}, a)$ is a certain transformation of features and actions, and $\boldsymbol{\gamma}$ is a vector of parameters with a prior $Q := \boldsymbol{\gamma} \sim N(\boldsymbol{\mu}_{\boldsymbol{\gamma}}, \Sigma_{\boldsymbol{\gamma}})$. ii) Neural network (NN) regression, considering $g(\mathbf{x}_{m,k}, a)$ as a fully connected NN of depth $L \ge 2$, the collection of parameters of which, $\boldsymbol{\gamma}$, has a prior $Q := \boldsymbol{\gamma} \sim N(\boldsymbol{\mu}_{\boldsymbol{\gamma}}, \Sigma_{\boldsymbol{\gamma}})$ [14]. iii) Gaussian process (GP) regression, assuming that $g(\mathbf{x}_{m,k}, a)$ follows a Gaussian process prior Q, such that $g \sim \text{GP}(\mu_{\boldsymbol{\gamma}}, \mathcal{K}_{\boldsymbol{\gamma}})$.

4.3 Learning Strategy

4.3.1 Posterior Distributions. To sequentially update the parameter estimation in an online setting, a key step is to derive the posterior distribution of parameters and functions in the hierarchical model (3). Define $\boldsymbol{\theta}_m = \{\theta_{m,k,a}\}_{1 \le k \le K_m, 1 \le a \le N}$ as the reward expectation vector for campaign m.We propose a two-step iteration procedure via Thompson sampling to update the posterior of each $\boldsymbol{\theta}_m$ given the constantly updating historical data $\mathcal{H}$. Since $\mathbb{P}(\boldsymbol{\theta}_m \mid \mathcal{H}) \propto \mathbb{P}(\boldsymbol{\theta}_m \mid \mathcal{H}, g)\mathbb{P}(g \mid \mathcal{H})$, we split the posterior derivation into two steps: 1) $\mathbb{P}(g \mid \mathcal{H})$, and 2) $\mathbb{P}(\boldsymbol{\theta}_m \mid \mathcal{H}, g)$.

In Step 1), the posterior distribution of $g \mid \mathcal{H}$ shares a general structure under LR, GP and NN. Let $\boldsymbol{\Psi}_{1:H}$ be a matrix comprising features of the K selected arms (one for each ad line) collected from round 1 to round H. Similarly, $\boldsymbol{Y}_{1:H} = (\boldsymbol{Y}_1^T, \cdots, \boldsymbol{Y}_H^T)^T$ denotes the observed rewards of all base arms offered up to round H. The posterior mean and variance of $g \mid \mathcal{H}$ are given below:

$$\begin{aligned} &\mathbb{E}(g(\cdot) \mid \mathcal{H}) = \mu(\cdot) + \mathcal{K}(\cdot, \boldsymbol{\Psi}_{1:H})(\Phi + \sigma_\epsilon^2 I)^{-1}(\boldsymbol{Y}_{1:H} - \mu(\boldsymbol{\Psi}_{1:H})) \\ &\text{Cov}(g(\cdot) \mid \mathcal{H}) = \mathcal{K}(\cdot, \cdot) - \mathcal{K}(\cdot, \boldsymbol{\Psi}_{1:H})(\Phi + \sigma_\epsilon^2 I)^{-1}\mathcal{K}(\boldsymbol{\Psi}_{1:H}, \cdot). \end{aligned}$$

Here, $\Phi = \mathcal{K}(\boldsymbol{\Psi}_{1:H}, \boldsymbol{\Psi}_{1:H}) + \Sigma_{1:H}$. Notably, the first term $\mathcal{K}(\boldsymbol{\Psi}_{1:H}, \boldsymbol{\Psi}_{1:H})$ represents the variance induced in the prior distribution, and the second term $\Sigma_{1:H}$ denotes the variance induced by the random effect. See detailed definitions and derivations in Appendix A.

In LR, $\mu(x) = x^T\boldsymbol{\gamma}$ takes a specific linear form, and $\mathcal{K}(x, x') = x^T\Sigma_{\boldsymbol{\gamma}}x'$. In GP, $\mu(x)$ as the prior mean can adopt any function form of x, and $\mathcal{K}(x, x')$ is a general kernel function. It can be a linear kernel $\mathcal{K}(x, x') = \langle x, x'\rangle$, an RBF kernel $\mathcal{K}(x, x') = \exp\{-|x - x'|^2/(2\tilde{l}^2)\}$ with $\tilde{l}$ as a hyperparameter representing the standard deviation, or other kernel functions. In NN, $\mu(x)$ represents a fully-connected neural network, and $\mathcal{K}(x, x')$ represents the neural tangent kernel.

In Step 2), given g, we can derive the posterior of $\boldsymbol{\theta}_m \mid \mathcal{H}, g$ under the standard Bayesian estimation process with Gaussian white noise [26]. Let $C_{m,k,n}$ be the number of observations in $\mathcal{H}$ that correspond to base arm (m, k, n), and $\boldsymbol{Z}_{1:H}$ be an indicator matrix where the (i, l)-th entry of $\boldsymbol{Z}_{1:H}$ denotes whether the lth observation belongs to the ith (m, k, a) tuple. The posterior mean and variance of $\boldsymbol{\theta}_m \mid \mathcal{H}, g$ are given below.

$$\mathbb{E}(\boldsymbol{\theta}_m \mid \mathcal{H}, g) = \mathrm{Cov}(\boldsymbol{\theta}_m \mid \mathcal{H}, g)\left[\Sigma^{-1}g(\boldsymbol{\Psi}_m) + \sigma_\epsilon^{-2}\boldsymbol{Z}_{1:H,m}\boldsymbol{Y}_{1:H}\right],$$
$$\mathrm{Cov}(\boldsymbol{\theta}_m \mid \mathcal{H}, g) = \left(\Sigma^{-1} + \sigma_\epsilon^{-2}diag(C_{m,1,1}, \cdots, C_{m,K,N})\right)^{-1}.$$

4.3.2 TS and Optimization. Based on the derived posterior distributions in the previous subsection, we propose a four-step Thompson sampling method to sample estimated rewards $\tilde{\boldsymbol{\theta}}_m$ for each campaign m: i) update $\mathbb{P}(g \mid \mathcal{H})$, ii) sample $\tilde{g}$ from $\mathbb{P}(g \mid \mathcal{H})$, iii) update $\mathbb{P}(\boldsymbol{\theta}_m \mid \tilde{g}, \mathcal{H})$, and iv) sample $\tilde{\boldsymbol{\theta}}_m$ from $\mathbb{P}(\boldsymbol{\theta}_m \mid \tilde{g}, \mathcal{H})$.

For instance, consider linear regression $g(x) = x^T\gamma$ as the working model. We start by updating the posterior distribution of γ using observations from all campaigns and ad lines according to the proposed Bayesian hierarchical model. We then sample a $\tilde{\gamma}$, which is used to update the prior of $\theta_{m,k,a}$ to $\mathcal{N}(x^T\tilde{\gamma}, \sigma_m^2)$. Next, we re-derive the posterior distribution of $\theta_{m,k,a}$ by incorporating this new prior along with the historical data specific to action (m, k, a), and sample $\tilde{\boldsymbol{\theta}}_m$ accordingly. By doing so, we are able to integrate all collected data and thereby establishing a comprehensive, feature-based informative prior for each $\theta_{m,k,a}$, which guides the subsequent learning of $\theta_{m,k,a}$ through its continuous interaction with the environment, enhancing the overall efficiency of the learning process. We refer to this as a feature-guided (FG) approach.

With the sampled $\tilde{\boldsymbol{\theta}}_m$, the next phase is to select the best super arm that solves (2). Since each campaign has independent budget limit B_m, we can optimize each campaign $m \in [M]$ separately. At each time round t, we need to solve $\underset{\boldsymbol{a}_{m,t} \in \mathcal{S}_m}{\mathrm{argmax}} \sum_{k\in[K_m]} \tilde{\theta}_{m,k,a_{m,k,t}}$, which can be regarded as a Multiple-Choice Knapsack Problem (MCKP) [16]. Specifically, MCKP is a generalization of the ordinary knapsack problem, where the set of items are originally partitioned into K groups. Instead of making binary choices regarding each item, MCKP only allows (at most) one item in the same group to be chosen. Similar to the ordinary knapsack problem, one can utilize dynamic programming to find the optimal solution. We summarize the entire learning strategy in Algorithm 1.

4.3.3 Offline-training-online-deployment. In practice, updating g at every decision time is unnecessary and adopting an offline-training-online-deployment paradigm can be more efficient. Specifically, we would update the posterior of g at particular time points using all accumulated information. Then, a function instance $\tilde{g}$ is sampled and utilized as the prior for subsequent learning of $\boldsymbol{\theta}$ until g is retrained. The retraining frequency can be tailored based on available computational resources or operational requirements. For example, given that marketing managers often allocate budgets across campaigns at the start of each day and collect observations at the end of each day, we will sample $\tilde{g}$ just once daily in practice. Furthermore, to achieve efficient optimization, we used a dynamic programming solution with a time complexity of $O(N^2K)$. Empirically, the daily computational cost per campaign of the offline-training-online-deployment variant is less than 1.1s even when $MKN = 200,000$. For more details on such a computationally efficient variant, refer to Appendix B.

Algorithm 1: Multi-Task Combinatorial Bandits for Budget Allocation (MCMAB)

Input : Specification of g and the corresponding prior; known parameters (i.e., σ_ϵ, Σ) of the hierarchical model

Set $\mathcal{H} = \{\}$

for *every decision point j* **do**

- Retrieve the campaign index m;
- Update the posterior for g as $\mathbb{P}(g|\mathcal{H})$, according to the hierarchical model (3);
- Sample a $\tilde{g} \sim \mathbb{P}(g|\mathcal{H})$;
- Given $\tilde{g}$, update the posterior for $\boldsymbol{\theta}_m$ as $\mathbb{P}(\boldsymbol{\theta}_m|\tilde{g}, \mathcal{H})$;
- Sample an utility vector $\tilde{\boldsymbol{\theta}}_m \sim \mathbb{P}(\boldsymbol{\theta}_m|\tilde{g}, \mathcal{H})$;
- Take action $\boldsymbol{a}_{m,t} = argmax_{\boldsymbol{a}_{m,\cdot} \in \mathcal{S}_m} \sum_{k\in[K_m]} \tilde{\theta}_{m,k,a_{m,k,\cdot}}$;
- Receive reward $R_{m,t}$;
- Update the dataset as $\mathcal{H} \leftarrow \mathcal{H} \cup \{(m, \boldsymbol{a}_{m,t}, R_{m,t})\}$

end

5 Offline Experiments

We assess the performance of MCMAB on both synthetically generated problems and real-world situations simulated using Amazon's campaign data. Without loss of generality, we assume $T_m = T$ and $K_m = K$ are the same across campaigns. Bayes regret [17] is used to evaluate the performance of different approaches:

$$BR(T, M, K) = \mathbb{E}\sum_{m=1}^{M}\sum_{t=1}^{T_m}\sum_{k=1}^{K_m}\max_{\boldsymbol{a}_{m,t}\in\mathcal{S}_m}\theta_{m,k,a} - \theta_{m,k,a_{m,k,t}},$$

where the expectation is taken over the distribution of task instances $\{(\mathbf{x}_{m,k}, \boldsymbol{\theta}_{m,k})\}$ and randomness caused by algorithms.

5.1 Synthetic Experiments

Settings. Assume there are $M = 50$ different campaigns, each with $K = 5$ ad lines. For each ad line, we consider discretizing the action space into $N = 50$ discrete levels. Suppose that the budget for each campaign $B_m \sim \mathrm{Uniform}(20, 30)$. $\mathbf{x}_{m,k}$ is a $(1 + d_m + d_k)$-dimensional vector consisting of B_m, d_m campaign-specific features sampled from $\mathcal{N}(\mathbf{0}_{d_m}, \boldsymbol{I}_{d_m})$, sharing across ad lines within the same campaign, and d_k ad-line-specific features sampled from $\mathcal{N}(\mathbf{0}_{d_k}, \boldsymbol{I}_{d_k})$. Evaluating the proposed algorithm with LR, we consider a linear environment setting with $d_m = d_k = 3$. Specifically, we set $\phi(\mathbf{x}_{m,k}, a) = (1, B_m * a, \mathbf{x}_{m,k}^{-B_m})^T$, where $\mathbf{x}_{m,k}^{-B_m}$ denote the

$\mathbf{x}_{m,k}$ with the B_m removed, and set $g(\mathbf{x}_{m,k}, a) = \phi(\mathbf{x}_{m,k}, a)^T \boldsymbol{\gamma}$ with $\boldsymbol{\gamma} \sim \mathcal{N}(0, \boldsymbol{I})$. For algorithms with NN and GP as the working model, we consider a nonlinear environment setting with $\phi(\mathbf{x}_{m,k}, a) = (B_m * a, \mathbf{x}_{m,k}^{-B_m})^T$ and $g(\mathbf{x}_{m,k}, a) = log(|\phi^2(\mathbf{x}_{m,k}, a)^T \boldsymbol{\gamma}|)$ with $\boldsymbol{\gamma} \sim \mathcal{N}(0, \boldsymbol{I})$. We set $d_m = d_k = 1$ for GP and $(d_m = 1, d_k = 2)$ for NN. Following the model (3), we further set $\Sigma = \sigma_m^2 \boldsymbol{I}$ and $\sigma_\epsilon = 1$ for all settings. We vary the value of σ_m for each setting, a higher value of which implies a larger component of the sub-arms' reward not being explained by the observed feature information. Clipping techniques are employed to ensure that the observed outcomes are non-negative.

To evaluate the performance of MCMAB in both use cases of leveraging learnings from past campaigns and utilizing information from concurrent campaigns, we consider both the concurrent and sequential settings. Specifically, all 50 campaigns run simultaneously in the concurrent setting; under the sequential setting, campaigns only start when the previous campaign ends.

Baselines. For each setting, we consider feature agnostic (*FA-ind*) and feature determined (*FD*) approaches as baselines, where *FA-ind* learns the reward distribution of each base arm (k, a) in each campaign independently, *FD* assumes the same model assumption as the proposed *MCMAB* approach but with $\sigma_m = 0$. Additionally, the computationally efficient variants for both *FD* and *MCMAB* are reported, denoted as *FD-approx* and *MCMAB-approx*. Furthermore, we compare the proposed method to the information-sharing framework proposed in Han and Arndt [12], referred to as *Han2021*. For the experiment of LR approaches, we use linear regression as the global model for *Han2021*, fit the local model for each ad line as a Bayesian linear model, and estimate its posterior based on an augmented dataset consisting of 30 predicted rewards generated by the fitted global model and the ad line's observed history. For the experiments of GP and NN approaches, we use random forest [5] as the global model. For experiments of LR approaches, we use the environment model as the prior for *MCMAB* approaches and derive the setting of prior distributions for *FA-ind*, *FD*, and the local model of *Han2021* by MCMC for a fair comparison. For experiments of GP and NN approaches, uninformative priors are used for all algorithms. We also present the performance of the oracle-TS as our skyline. See Table 1 for a summary.

Models	Utilize $\boldsymbol{x}$	Heterogeneity	Linear Assumption
MCMAB (ours)	✓	✓	×
FD	✓	×	×
FA-ind	×	✓	×
Han2021	✓	×	✓

Table 1: MCMAB and baseline approaches.

Results. Figures 3 and 4 depict the results of experiments conducted in both concurrent and sequential settings, respectively, with aggregation over 100 random seeds. Our observations can be summarized as follows. First, the *MCMAB* method outperforms all baselines, achieving an average regret closest to that of the oracle-TS. Second, under the concurrent setting, both *FA-ind* and *MCMAB* exhibit sublinear regret in T, but *FA-ind* exhibits a significantly slower learning rate, resulting in substantially higher average regret compared to *MCMAB*. This trend continues in the sequential setting, where *FA-ind*'s lack of information sharing across campaigns and ad lines leads to notably higher regret than *MCMAB*, further emphasizing the benefits of information sharing. Third, *FD* underperforms *MCMAB* in all settings, especially when the metadata is less useful (i.e., when σ_m is large). This emphasizes the need to use the feature information wisely. Assuming that feature information alone determines the reward distributions would induce further bias, resulting in higher regret. Fourth, while *Han2021* demonstrates similar limitations to *FD* in the evaluation of LR-based approaches, it exhibits even more pronounced inferiority in settings with nonlinear environments. This underperformance is not just from neglecting the uncertainty of the global model and inter-arm heterogeneity, but also crucially due to its linear model assumptions being ill-suited to the complex, nonlinear relationships between $\theta_{m,k,a}$ and the corresponding budget a.

Lastly, the computationally efficient variants generally match the performance of their respective standard counterparts, except for the *FD* approach with NN as the working model. With NN, *FD-approx* yields significantly higher regret than *FD*. This increased regret stems not just from the inefficient use of information due to delayed updating but also from the inadequacies in batch updating of the NN-based *FD* approach that we adopted [35], as discussed in Dai et al. [8]. Further investigation of integrating our hierarchical model with the batch Neural TS [8] is worthwhile but outside the scope of this study.

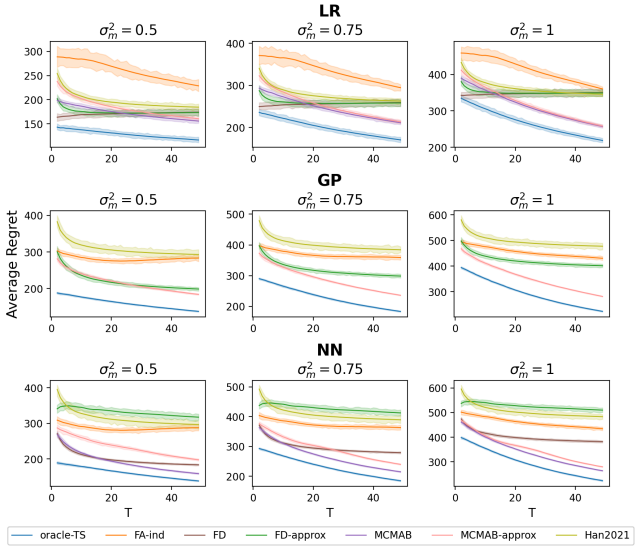

Figure 3: Simulation results for the concurrent setting. Shaded areas indicate the 95% confidence interval (CI).

Additional Results. Results under other settings of $(M, K, N, d, \sigma_\epsilon)$ are reported in Appendix C. Overall, *MCMAB* consistently yields lower regrets. With an increase in M/K, the availability of more metadata aids *MCMAB* in learning the task distribution more effectively, thus rapidly approaching oracle-TS performance. However, as $d_m/d_k/\sigma_\epsilon/N$ increase, the learning challenge intensifies for all algorithms, yet *MCMAB* maintains superior performance.

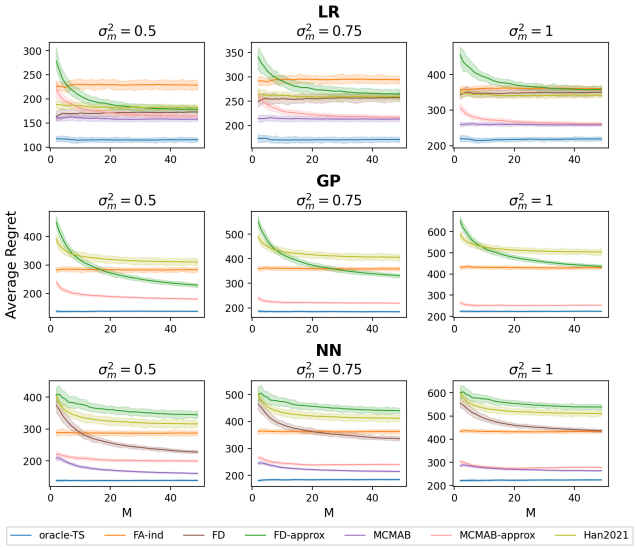

Figure 4: Simulation results for the sequential setting. Shaded areas indicate the 95% CI.

The trend involving N may initially appear counterintuitive, as a finer discretization of the action space, leading to more precise recommendations, is typically expected to improve performance. However, as Figure 5 illustrates, the reality is more complex due to the uncertainty in reward distributions for base arms. While a coarse discretization (small N) falls short of achieving the true optimal, a finer division (large N) significantly escalates the complexity of the learning process, ultimately hindering *MCMAB* performance. Finding N values between 30 and 50 yields relatively optimal performance without significant difference, we set $N = 50$ for the subsequent offline study of Amazon's campaign data.

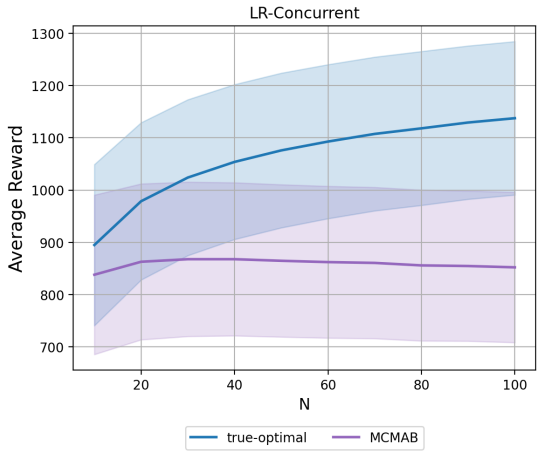

Figure 5: The average reward at $T = 50$ corresponds to various N specifications in the concurrent linear environment setting. Shaded regions represent the 95% CI.

5.2 Offline Evaluation on Amazon Campaigns

To evaluate the effectiveness of the proposed approach and compare different working models in real-world settings before online deployment, we use Amazon's campaign data from Q1 2023.

Dataset. The dataset is sourced from an advertiser that promotes Amazon products and services, which is the largest advertiser and the primary focus of our forthcoming online experiment. To mitigate the influence of extreme values and outliers, we filtered out data points with a cost exceeding 1000 USD. Additionally, we focused on ad lines sourced from the top two suppliers, APS and 3PX, and limited our analysis to six primary channels: Video Mobile App Out-Stream, Video Mobile App In-Stream, Web Billboard, Mobile App, Video Web, and Web. The refined dataset encompasses a total of 39,212 individual observations, which are spread across 1278 ad lines from nine campaigns. By employing the random forest algorithm, we found that the combination of the supply source, channel, and the logarithm of budget cost can effectively account for 94.7% of the observed variation in the logarithm of clicks obtained (i.e., $log(\theta_{m,k,a})$). This suggests that supply source, channel, and budget cost are crucial factors that significantly impact clicks. However, there are still variations that remain unexplained to us.

Design. To simulate real-world scenarios, we first determined g, σ_m, and σ_ϵ. The selection of g involved comparing linear regression, random forest, and CatBoost [24] via cross-validation, where CatBoost demonstrated the best performance with the lowest mean squared error (MSE). Subsequently, let $\Sigma = \sigma_m^2 \boldsymbol{I}$, σ_m and σ_ϵ were determined to be 0.35 and 0.40, respectively, using maximum likelihood estimation based on the fitted CatBoost model and under the assumption that both the random effects and noise are normally distributed. It is important to note that in our application of CatBoost, we do not enforce any parametric modeling assumptions regarding the relationship between the features and the rewards.

To mimic the concurrent scenario, where multiple campaigns runs simultaneously, we randomly construct 50 distinct campaigns ($M = 50$). Each campaign comprises 5 randomly selected ad lines ($K = 5$). Assuming each campaign has a daily budget limit of 300 USD ($B_m = 300$), we allocate the budget across the 5 ad lines within each campaign separately on each day t. The stochastic observations for the total number of clicks are generated using the base model (3) with $g, \sigma_m, \sigma_\epsilon$ as parameters. Similarly, to mimic the sequential scenario, where campaigns comes in sequence, each campaign is randomly constructed with 5 ad lines ($K = 5$) and lasts 50 days ($T = 50$) with a daily budget of 300 USD ($B_m = 300$).

Baselines. Our studies compare ten approaches, including the proposed MCMAB algorithm with LR, NN, and GP as working models to quantify the relationship between $log(\theta_{m,k,a})$ and features, alongside their *FD* counterparts. LR-based approaches use feature information $\boldsymbol{x}_{m,k}$, encompassing channels and supply source of the corresponding ad line, and the budget limit of the corresponding campaign. They assume that $g(\boldsymbol{x}_{m,k}, a) = \phi(\boldsymbol{x}_{m,k}, a)^T \boldsymbol{\gamma}$, with ϕ being a deterministic function that transforms the tuple information $(\boldsymbol{x}_{m,k}, a)$ to further include interaction terms between channels, supply sources, and the logarithm of budget shares. GP-based approaches assume $g(\boldsymbol{x}_{m,k}, a)$ follows a Gaussian Process, while NN-based approaches employ a fully connected 3-layer neural network g with a width of 30 for the concurrent setting and a width

of 26 for the sequential setting. Additionally, we explored *Hibou*, the current method used in the ADSP system, which posits linear relationships between ad-line performance and assigned budgets, allocating budgets solely on estimated gradients without further exploration or information sharing. The study also includes an evaluation of *Han2021_RF* and *Han2021_LR*, the contextual bandits approaches from Han and Arndt [12], utilizing Random Forest and Linear Regression as the global model, respectively. The local model for each ad line is fitted as a Bayesian linear model using an augmented dataset consisting of 30 predicted returns generated by the fitted global model and the ad line's observed history. Lastly, we considered *FA-ind*, a baseline approach that independently learns the distribution of each $\theta_{m,k,a}$.

To ensure a fair comparison, we applied uninformative priors for all methods. Specifically, for *FD-LR* and *MCMAB-LR*, we used $\boldsymbol{\gamma} \sim \mathcal{N}(0, 20\boldsymbol{I})$; for *FD-GP* and *MCMAB-GP*, we used zero-mean priors with RBF kernels; for *FD-NN* and *MCMAB-NN*, we initialized the networks with all weights sampled from normal distributions with zero mean Zhang et al. [35]; for *Hibou*, we started with an even allocation; and for *Han2021_LR* and *Han2021_RF*, we used $\mathcal{N}(0, 20\boldsymbol{I})$ as the prior for local model parameters[12].

Results. Figure 6 depicts the average reward (i.e., the average number of clicks) received after implementing the budget allocation strategies suggested by each approach. Overall, feature-guided approaches (*MCMAB-LR*, *MCMAB-GP*, and *MCMAB-NN*) demonstrated greater average reward, outperforming other methods. Compared to *Hibou*, *MCMAB* showed an 18% increase in the average number of clicks obtained at the conclusion of the experiment in the concurrent setting, and a 16% increase in the sequential setting.

Failing to utilize any feature information, *FA-ind* struggles with the curse of dimensionality and limited interaction opportunities, resulting in a significantly slower learning process with the lowest average reward, for both settings. Under the concurrent setting, *Hibou*, which uses only the budget information and learns the reward distribution for each ad line independently, continues to show a lower average reward than approaches that utilize additional ad line metadata information. On the other hand, feature-determined approaches (*FD-LR*, *FD-GP*) and *Han2021* initially outperform feature-guided approaches (*MCMAB-LR*, *MCMAB-GP*) but ultimately sustain lower average reward due to their restricted model assumptions. It should be noted that because the current network structure is naively specified without carefully fine-tuning its width and depth, *FD-NN* and *MCMAB-NN* perform worse than other feature-determined approaches, indicating that the current network structure fails to capture the relationship's complexity well. We could expect that the NN-based approach will perform better with further fine-tuning.

Under the sequential setting, *Hibou* demonstrates superior performance during the initial stages. This is because when campaigns are introduced sequentially, the metadata information is limited initially, impeding reasonable estimations for other feature-based approaches. As the system accumulates data from an increasingly diverse range of campaigns, the average reward for approaches that utilize metadata for information sharing shows a marked increase. In contrast, *Hibou*'s average reward converges to be constant, reflecting its inability to leverage learnings from past campaigns. Similar to what we observed in the concurrent setting, FD approaches and *Han2021* yield a lower average reward compared to *MCMAB*. This underperformance is primarily attributed to their failure to adequately address the heterogeneity among base-arms.

Finally, *MCMAB-LR* performs better than *MCMAB-NN* and *MCMAB-GP* under the concurrent setting. This is presumably due to the linear properties of the dataset we used, which reduce the efficacy of the more complex GP and NN models, potentially leading to overfitting. In contrast, *MCMAB-GP* performs better than *MCMAB-LR* and *MCMAB-NN* under the sequential setting, with *MCMAB-LR* gradually approaching the performance level of *MCMAB-GP* as more campaigns are completed. This is mainly due to the initial scarcity of metadata, which hinders *MCMAB-LR*'s ability to establish a reliable linear model, whereas the Gaussian process, by using its kernel function, can effectively focus on more relevant features and ignore features containing less information. It is worth noting that fine-tuning kernels and hyperparameters can improve the performance of GP-based approaches, while the performance of NN-based approaches can be enhanced by further adjusting the neural network structure and learning rate.

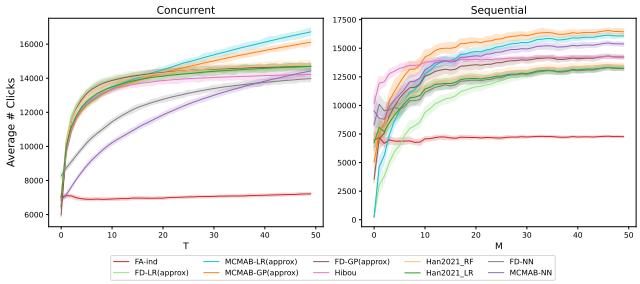

Figure 6: Simulation results on Amazon's campaign data, averaged over 100 random seeds. Shaded areas represent the 95% CI.

6 Online Experiments

To further assess the proposed methods, we initiated an online experiment by allocating 3% of the budget from an active campaign on Amazon DSP. This campaign targeted 15 distinct audience groups, each comprising 22 ad lines. For the experiment, we constructed 30 campaigns, each consisting of a single audience group with 10 ad lines, summing up to 300 ad lines in total. The 3% of the budget was isolated by carving out 3% of Amazon demand by geo from the main campaign and directing them exclusively to the experiment.

Within the 3% of geos in the experiment, we implemented an A/B test, dividing the budget equally between two groups: one subjected to our newly developed MCMAB-based budget allocation strategy and the other continuing with the traditional Hibou budget optimization as a control group. Based on the previous offline evaluation, we selected **LR** as the working model, incorporating features such as the suppliers of the ad lines and the channels used. An informative prior for the MCMAB-LR was constructed using data collected from similar campaigns run by the same advertiser in 2023. The experiment demonstrated promising potential, achieving a notable 12.7% reduction in cost-per-click using the proposed method compared to the standard practice after three weeks.

7 Conclusion

Motivated by the potential of leveraging information sharing among campaigns for enhancing the accuracy of return predictions and hence optimizing budget allocation strategies, we introduce a sophisticated multi-task combinatorial bandit framework building upon Bayesian hierarchical models. This innovative approach has demonstrated considerable promise in numerical studies, effectively maximizing cumulative returns through utilization of the metadata of campaigns and ad lines, as well as the budget size. Specifically, an offline study conducted on Amazon's campaign data reveals an average improvement of 18% in total clicks obtained under the concurrent setting compared to the currently deployed Hibou system and an average improvement of 16% under the sequential setting. Additionally, an online experiment shows a notable reduction of 12.7% in cost-per-click.

As a future direction, recognizing the impact of dynamic contextual factors such as the week of the month, day of the week, and holidays on advertising return variability, we are considering an extension to include contextual bandits. This development aims to incorporate these temporal factors, thereby improving the model's ability to take the seasonality of advertising returns into account. Furthermore, while we assume independence between different campaign activities, internal competition is ubiquitous due to limited ad resources[27]. Optimizing each campaign individually may result in a local optimum, with the possibility of excessive demand exceeding ad line supply. To achieve a global optimum, it is worthwhile to consider internal competition across campaigns.

References

[1] Shipra Agrawal and Nikhil Devanur. 2016. Linear contextual bandits with knapsacks. *Advances in Neural Information Processing Systems* 29 (2016).

[2] Shipra Agrawal and Nikhil R Devanur. 2014. Bandits with concave rewards and convex knapsacks. In *Proceedings of the fifteenth ACM conference on Economics and computation*. 989–1006.

[3] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. 2018. Bandits with knapsacks. *Journal of the ACM (JACM)* 65, 3 (2018), 1–55.

[4] Santiago R Balseiro and Yonatan Gur. 2019. Learning in repeated auctions with budgets: Regret minimization and equilibrium. *Management Science* 65, 9 (2019), 3952–3968.

[5] Gérard Biau and Erwan Scornet. 2016. A random forest guided tour. *Test* 25 (2016), 197–227.

[6] Craig Boutilier, Chih-Wei Hsu, Branislav Kveton, Martin Mladenov, Csaba Szepesvari, and Manzil Zaheer. 2020. Differentiable meta-learning of bandit policies. *Advances in Neural Information Processing Systems* 33 (2020), 2122–2134.

[7] Chun-Hung Chiu, Tsan-Ming Choi, Xin Dai, Bin Shen, and Jin-Hui Zheng. 2018. Optimal advertising budget allocation in luxury fashion markets with social influences: A mean-variance analysis. *Production and Operations Management* 27, 8 (2018), 1611–1629.

[8] Zhongxiang Dai, Yao Shu, Bryan Kian Hsiang Low, and Patrick Jaillet. 2022. Sample-then-optimize batch neural thompson sampling. *Advances in Neural Information Processing Systems* 35 (2022), 23331–23344.

[9] Michael Fairley, Lauren E Cipriano, and Jeremy D Goldhaber-Fiebert. 2020. Optimal allocation of research funds under a budget constraint. *Medical Decision Making* 40, 6 (2020), 797–814.

[10] Tong Geng, Fangzhou Sun, Di Wu, Wei Zhou, Harikesh Nair, and Zhangang Lin. 2021. Automated bidding and budget optimization for performance advertising campaigns. *Available at SSRN 3913039* (2021).

[11] Samarth Gupta, Jinhang Zuo, Carlee Joe-Wong, Gauri Joshi, and Osman Yağan. 2022. Correlated combinatorial bandits for online resource allocation. In *Proceedings of the Twenty-Third International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*. 91–100.

[12] Benjamin Han and Carl Arndt. 2021. Budget Allocation as a Multi-Agent System of Contextual & Continuous Bandits. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2937–2945.

[13] Benjamin Han and Jared Gabor. 2020. Contextual Bandits for Advertising Budget Allocation. *Proceedings of the ADKDD* 17 (2020).

[14] Arthur Jacot, Franck Gabriel, and Clément Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems* 31 (2018).

[15] Niklas Karlsson. 2020. Feedback control in programmatic advertising: The frontier of optimization in real-time bidding. *IEEE Control Systems Magazine* 40, 5 (2020), 40–77.

[16] Hans Kellerer, Ulrich Pferschy, David Pisinger, Hans Kellerer, Ulrich Pferschy, and David Pisinger. 2004. The multiple-choice knapsack problem. *Knapsack Problems* (2004), 317–347.

[17] Branislav Kveton, Mikhail Konobeev, Manzil Zaheer, Chih-wei Hsu, Martin Mladenov, Craig Boutilier, and Csaba Szepesvari. 2021. Meta-thompson sampling. In *International Conference on Machine Learning*. PMLR, 5884–5893.

[18] Yossi Luzon, Rotem Pinchover, and Eugene Khmelnitsky. 2022. Dynamic budget allocation for social media advertising campaigns: optimization and learning. *European Journal of Operational Research* 299, 1 (2022), 223–234.

[19] Alessandro Nuara, Francesco Trovo, Nicola Gatti, and Marcello Restelli. 2018. A combinatorial-bandit algorithm for the online joint bid/budget optimization of pay-per-click advertising campaigns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[20] Alessandro Nuara, Francesco Trovò, Nicola Gatti, and Marcello Restelli. 2022. Online joint bid/daily budget optimization of internet advertising campaigns. *Artificial Intelligence* 305 (2022), 103663.

[21] Weitong Ou, Bo Chen, Xinyi Dai, Weinan Zhang, Weiwen Liu, Ruiming Tang, and Yong Yu. 2023. A Survey on Bid Optimization in Real-Time Bidding Display Advertising. *ACM Transactions on Knowledge Discovery from Data* 18, 3 (2023), 1–31.

[22] Okan Örsan Özener and Özlem Ergun. 2008. Allocating costs in a collaborative transportation procurement network. *Transportation Science* 42, 2 (2008), 146–165.

[23] Martin Posch and Peter Bauer. 2013. Adaptive budgets in clinical trials. *Statistics in Biopharmaceutical Research* 5, 4 (2013), 282–292.

[24] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems* 31 (2018).

[25] Karthik Abinav Sankararaman and Aleksandrs Slivkins. 2018. Combinatorial semi-bandits with knapsacks. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1760–1770.

[26] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. 2018. A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of mathematical psychology* 85 (2018), 1–16.

[27] Guangyuan Shen, Shenjie Sun, Dehong Gao, Shaolei Li, Libin Yang, Yongping Shi, and Wei Ning. 2023. Cross-channel Budget Coordination for Online Advertising System. *arXiv preprint arXiv:2305.06883* (2023).

[28] Yue Shi, Yisha Xiang, Hui Xiao, and Liudong Xing. 2021. Joint optimization of budget allocation and maintenance planning of multi-facility transportation infrastructure systems. *European Journal of Operational Research* 288, 2 (2021), 382–393.

[29] Runzhe Wan, Lin Ge, and Rui Song. 2021. Metadata-based multi-task bandits with bayesian hierarchical models. *Advances in Neural Information Processing Systems* 34 (2021), 29655–29668.

[30] Runzhe Wan, Lin Ge, and Rui Song. 2023. Towards scalable and robust structured bandits: A meta-learning framework. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1144–1173.

[31] Huanle Xu, Yang Liu, Wing Cheong Lau, Jun Guo, and Alex Liu. 2019. Efficient online resource allocation in heterogeneous clusters with machine variability. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 478–486.

[32] Huanle Xu, Yang Liu, Wing Cheong Lau, and Rui Li. 2020. Combinatorial Multi-Armed Bandits with Concave Rewards and Fairness Constraints.. In *IJCAI*. 2554–2560.

[33] Yanwu Yang, Daniel Zeng, Yinghui Yang, and Jie Zhang. 2015. Optimal budget allocation across search advertising markets. *informs Journal on Computing* 27, 2 (2015), 285–300.

[34] Weinan Zhang, Ying Zhang, Bin Gao, Yong Yu, Xiaojie Yuan, and Tie-Yan Liu. 2012. Joint optimization of bid and budget allocation in sponsored search. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1177–1185.

[35] Weitong Zhang, Dongruo Zhou, Lihong Li, and Quanquan Gu. 2020. Neural thompson sampling. *arXiv preprint arXiv:2010.00827* (2020).

[36] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized cost per click in taobao display advertising. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2191–2200.

[37] Jinhang Zuo and Carlee Joe-Wong. 2021. Combinatorial multi-armed bandits for resource allocation. In *2021 55th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 1–4.

A Posterior Updating

Recall that $\mathbf{x}_{m,k}$ denotes all features related to the adline k in campaign m. For simplicity of notations, we assume $K \equiv K_m$ for all $m \in [M]$. We further define $\mathbf{x}_m = (\mathbf{x}_{m,1}, \cdots, \mathbf{x}_{m,K})$ as all features involved in campaign m, and $\boldsymbol{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_M)$ as all features involved in all campaigns. Likewise, we denote $\boldsymbol{\theta}_{m,k} = (\theta_{m,k,1}, \cdots, \theta_{m,k,N})^T$, $\boldsymbol{\theta}_m = (\boldsymbol{\theta}_{m,1}^T, \cdots, \boldsymbol{\theta}_{m,K}^T)^T$, and $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^T, \cdots, \boldsymbol{\theta}_M^T)^T$.

Suppose at decision point t, the $m(t)$th campaign was involved in the posterior updating process. Define $a_{m,k,t}$ as the action selected for the kth adline at round t, and $\boldsymbol{Y}_t$ is a $K \times 1$ vector containing the observed reward for the K selected base arms. Let $\boldsymbol{\psi}_t = \{\phi(\mathbf{x}_{m(t),k,t}, a_{m(t),k,t})\}_{1 \le k \le K}$ be a $d \times K$-dimensional matrix containing features of all base arms offered at round t after a general transformation $\phi(\mathbf{x}, a)$. Subsequently, $\boldsymbol{\Psi}_{1:H} = (\boldsymbol{\psi}_1, \cdots, \boldsymbol{\psi}_H)$ is a $d \times KH$ matrix including all transformed features from round 1 to round H. Likewise, $\boldsymbol{Y}_{1:H} = (\boldsymbol{Y}_1^T, \cdots, \boldsymbol{Y}_H^T)^T$ includes observed rewards of all base arms offered till round H.

Then, we define an $MKN \times KH$-dimensional indicator matrix $\boldsymbol{Z}_{1:H}$, such that the (i,l)-th entry of $\boldsymbol{Z}_{1:H}$ denotes whether the lth observation belongs to the ith (m,k,a) tuple. More specifically, $\{\boldsymbol{Z}_{1:H}\}_{i,l} = \mathbb{I}\{\tilde{g}(l) = i\}$, where the function $\tilde{g}(l)$ maps the l-th observation to its corresponding base arm index. For each observation l, let $m(l)$, $k(l)$, and $a(l)$ represent the indices of the campaign, ad line, and budget level, respectively. The mapping function is defined as $\tilde{g}(l) = (m(l)-1) \times KN + (k(l)-1) \times N + a(l)$.

Based on the hierarchical model in Equation (3), the core of our methodology is to learn the reward distribution $P(\theta_{m,k,a} \mid x)$, relying on which to make optimal decisions. To learn the $P(\theta_{m,k,a} \mid x)$, we make a model assumption such that $\theta_{m,k,a} \sim N(g(x_{m,k}, a), \sigma_m^2)$. Without knowledge of the true generation model g, any parameterized linear or non-parameterized model, such as GP or NN, can be used as the working model for g. Motivated by the observed fact that almost none of the off-the-shelf machine learning models can guarantee perfect estimation of g without error, no matter how informative $x_{m,k}$ is, $\delta_{m,k,a}$ is added for allowing the additional randomness of $\theta_{m,k,a}$ that can not be explained by $x_{m,k}$. As outlined in the main paper, the posterior updating process follows the bayesian updating rule

$$\mathbb{P}(\boldsymbol{\theta} \mid \mathcal{H}) \propto \mathbb{P}(\boldsymbol{\theta} \mid \mathcal{H}, g)\mathbb{P}(g \mid \mathcal{H}),$$

which naturally split the posterior derivation into two steps: 1) $\mathbb{P}(g \mid \mathcal{H})$, and 2) $\mathbb{P}(\boldsymbol{\theta} \mid \mathcal{H}, g)$. In the subsequent sections, we will first derive the posterior distribution of g given $\mathcal{H}$ as a function form, and then derive the posterior distribution of $\boldsymbol{\theta}_m$ given $\mathcal{H}$ and g. Since both LR and NN can be regarded as a special case of multivariate normal distribution given some transformed contextual information vector x, we will focus on the posterior derivation under Gaussian Process, and the expressions for LR and NN can be similarly derived with slight modifications, which are provided at the end of each subsection below.

A.1 Posterior Distribution of g Given $\mathcal{H}$

In the Gaussian process, the Bayesian hierarchical model can be rewritten as

$$\begin{aligned} &\text{(Prior)} && g \sim GP(\mu, \mathcal{K}), \\ &\text{(Observation)} && \boldsymbol{Y}_{1:H} = g(\boldsymbol{\Psi}_{1:H}) + \boldsymbol{\delta}_{1:H} + \boldsymbol{\epsilon}_{1:H}, \end{aligned} \tag{4}$$

where we define $\boldsymbol{\delta} = (\boldsymbol{\delta}_1^T, \ldots, \boldsymbol{\delta}_M^T)^T$ as the collection of the random effect vectors for all campaigns, $\boldsymbol{\delta}_{1:H} = \boldsymbol{Z}_{1:H}^T \boldsymbol{\delta} \sim \mathcal{N}(0, \Sigma_{1:H})$, and $\boldsymbol{\epsilon}_{1:H} \sim \mathcal{N}(0, \sigma^2 I_H)$. Then we have $\Sigma_{1:H} = \boldsymbol{Z}_{1:H}^T diag(\Sigma, \cdots, \Sigma)\boldsymbol{Z}_{1:H}$. Define $\boldsymbol{g}_{1:H} = g(\boldsymbol{\Psi}_{1:H})$ as the realization of Gaussian process g under historical data $\boldsymbol{\Psi}_{1:H}$, and $\mathcal{K}_{1:H} = \mathcal{K}(\boldsymbol{\Psi}_{1:H}, \boldsymbol{\Psi}_{1:H})$ as the covariance matrix for $\boldsymbol{g}_{1:H}$. Following Bayesian rule, the posterior of $g \mid \boldsymbol{Y}_{1:H}$ is:

$$p(\boldsymbol{g}_{1:H} \mid \boldsymbol{Y}_{1:H}) \propto p(\boldsymbol{Y}_{1:H} \mid \boldsymbol{\Psi}_{1:H}, \boldsymbol{g}_{1:H}) p(\boldsymbol{g}_{1:H} \mid \boldsymbol{\Psi}_{1:H}) \propto \exp\left(-\frac{1}{2}(\boldsymbol{Y}_{1:H} - \boldsymbol{g}_{1:H})^T (\sigma^2 I + \Sigma_{1:H})^{-1} (\boldsymbol{Y}_{1:H} - \boldsymbol{g}_{1:H}) - \frac{1}{2}\boldsymbol{g}_{1:H}^T \mathcal{K}_{1:H}^{-1} \boldsymbol{g}_{1:H}\right) \sim \mathcal{N}(\tilde{\mu}, \tilde{\Sigma}) \tag{5}$$

where $\tilde{\Sigma} = \left\{\mathcal{K}_{1:H}^{-1} + (\sigma^2 I + \Sigma_{1:H})^{-1}\right\}^{-1}$, and $\tilde{\mu} = (\sigma^2 I + \Sigma_{1:H})^{-1}\tilde{\Sigma}\boldsymbol{Y}_{1:H} = \left\{(\sigma^2 I + \Sigma_{1:H}) + \mathcal{K}_{1:H}\right\}^{-1} \mathcal{K}_{1:H} \boldsymbol{Y}_{1:H}$.

When a new sample x^* comes in, the posterior distribution of $g^* \mid x^*, \mathcal{H}$ is given by

$$p(g^* \mid x^*, \mathcal{H}) = \int p(g^*, \boldsymbol{g}_{1:H} \mid x^*, \mathcal{H}) d\boldsymbol{g}_{1:H} = \int p(g^* \mid \boldsymbol{g}_{1:H}, x^*, \mathcal{H}) p(\boldsymbol{g}_{1:H} \mid x^*, \mathcal{H}) d\boldsymbol{g}_{1:H}. \tag{6}$$

Since $p(g^* \mid \boldsymbol{g}_{1:H}) \sim \mathcal{N}\big(\mathcal{K}(x^*, \boldsymbol{\Psi}_{1:H})\mathcal{K}_{1:H}\boldsymbol{g}_{1:H}, \mathcal{K}(x^*, x^*) - k(x^*, \boldsymbol{\Psi}_{1:H})\mathcal{K}_{1:H}\mathcal{K}(\boldsymbol{\Psi}_{1:H}, x^*)\big)$, we can combine it with (5) and (6) to obtain the posterior mean and variance for $p(g^* \mid x^*, \mathcal{H})$. After some tedious but conceptually straightforward manipulations [26], the posterior mean and variance of $g \mid \mathcal{H}$ can be derived as

$$\begin{aligned} \mu_{\text{post}}(x) &= \mu(x) + \mathcal{K}(x, \boldsymbol{\Psi}_{1:H})[\mathcal{K}(\boldsymbol{\Psi}_{1:H}, \boldsymbol{\Psi}_{1:H}) + \sigma^2 I + \Sigma_{1:H}]^{-1}(\boldsymbol{Y}_{1:H} - \mu(\boldsymbol{\Psi}_{1:H})) \\ \mathcal{K}_{\text{post}}(x, x') &= \mathcal{K}(x, x') - \mathcal{K}(x, \boldsymbol{\Psi}_{1:H})[\mathcal{K}(\boldsymbol{\Psi}_{1:H}, \boldsymbol{\Psi}_{1:H}) + \sigma^2 I + \Sigma_{1:H}]^{-1}\mathcal{K}(\boldsymbol{\Psi}_{1:H}, x') \end{aligned} \tag{7}$$

with $\Sigma_{1:H} = \boldsymbol{Z}_{1:H}^T diag(\Sigma, \cdots, \Sigma)\boldsymbol{Z}_{1:H}$.

In LR and NN, since the prior was imposed on a finite-dimensional parameter $\boldsymbol{\gamma}$, one can similarly derive the posterior of $\boldsymbol{\gamma} \mid \mathcal{H}$ as below:

$$\begin{aligned} \mathbb{E}(\boldsymbol{\gamma} \mid \mathcal{H}) &= \text{Cov}(\boldsymbol{\gamma} \mid \mathcal{H})\{\boldsymbol{\Psi}_{1:H}(\Sigma_{1:H} + \sigma^2 \boldsymbol{I})^{-1}\boldsymbol{Y}_{1:H} + \Sigma_{\boldsymbol{\gamma}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\gamma}}\} \\ \text{Cov}(\boldsymbol{\gamma} \mid \mathcal{H}) &= \left(\boldsymbol{\Psi}_{1:H}(\Sigma_{1:H} + \sigma^2 \boldsymbol{I})^{-1}\boldsymbol{\Psi}_{1:H}^T + \Sigma_{\boldsymbol{\gamma}}^{-1}\right)^{-1} = \Sigma_{\boldsymbol{\gamma}} - \Sigma_{\boldsymbol{\gamma}}\boldsymbol{\Psi}_{1:H}[\boldsymbol{\Psi}_{1:H}'\Sigma_{\boldsymbol{\gamma}}\boldsymbol{\Psi}_{1:H} + \sigma^2 I + \Sigma_{1:H}]^{-1}\boldsymbol{\Psi}_{1:H}'\Sigma_{\boldsymbol{\gamma}}, \end{aligned}$$

where the second equation in $\text{Cov}(\boldsymbol{\gamma} \mid \mathcal{H})$ is a direct application of the Woodbury matrix identity.

In both LR and NN, $g(x)$ shares a similar linear structure defined by $g(x) = \phi(x)^T\boldsymbol{\gamma}$, where $\phi(x)$ denoting different feature information representations. Specifically, for LR, $\phi(x)$ can be as simple as x; for NN, $\boldsymbol{\gamma} = \{\text{vec}(W_1), \cdots, \text{vec}(W_L)\}$ represents the collection of parameters of the neural network, L is the number of layers, and g is a fully connected neural network of depth $L \geq 2$. $\phi(x)$ is the gradient of the neural network. As such, the posterior mean and variance of $g(x) \mid \mathcal{H}$ can be obtained by directly multiplying $\phi(x)$ on the posterior of $\boldsymbol{\gamma} \mid \mathcal{H}$:

$$\begin{aligned}\mathbb{E}(g(x) \mid \mathcal{H}) &= \phi(x)'\boldsymbol{\gamma} + \phi(x)'\Sigma_{\boldsymbol{\gamma}}\boldsymbol{\Psi}_{1:H}[\boldsymbol{\Psi}'_{1:H}\Sigma_{\boldsymbol{\gamma}}\boldsymbol{\Psi}_{1:H} + \sigma^2 I + \Sigma_{1:H}]^{-1}(Y_{1:H} - \boldsymbol{\Psi}'_{1:H}\boldsymbol{\gamma}) \\ \text{Cov}(g(x) \mid \mathcal{H}) &= \phi(x)'\Sigma_{\boldsymbol{\gamma}}\phi(x) - \phi(x)'\Sigma_{\boldsymbol{\gamma}}\boldsymbol{\Psi}_{1:H}[\boldsymbol{\Psi}'_{1:H}\Sigma_{\boldsymbol{\gamma}}\boldsymbol{\Psi}_{1:H} + \sigma^2 I + \Sigma_{1:H}]^{-1}\boldsymbol{\Psi}'_{1:H}\Sigma_{\boldsymbol{\gamma}}\phi(x),\end{aligned} \tag{8}$$

which exactly aligns with the general posterior formulation we derived for GP in Equation (7).

A.2 Posterior Distribution of θ Given $\mathcal{H}$ and g

According to the observation layer in (3), the posterior distribution of $Y_{1:H} \mid \boldsymbol{\Psi}_{1:H}, \boldsymbol{\theta}$ is given by

$$Y_{1:H} \mid \boldsymbol{\Psi}_{1:H}, \boldsymbol{\theta} \sim \mathcal{N}(Z_{1:H}^T\boldsymbol{\theta}, \sigma^2\boldsymbol{I}_{KH}).$$

Then, the posterior distribution of $\boldsymbol{\theta}$ given $\mathcal{H}$ and $\boldsymbol{\gamma}$ is

$$\begin{aligned}&\mathbb{P}(\boldsymbol{\theta} \mid Y_{1:H}, g) \propto \mathbb{P}(Y_{1:H} \mid \boldsymbol{\theta})\mathbb{P}(\boldsymbol{\theta} \mid g) \\ &\propto exp\Big(-\frac{1}{2}\sigma^{-2}(Y_{1:H} - Z_{1:H}^T\boldsymbol{\theta})^T(Y_{1:H} - Z_{1:H}^T\boldsymbol{\theta})\Big)exp\Big(-\frac{1}{2}(\boldsymbol{\theta} - g(\boldsymbol{\Psi}))^T diag(\Sigma^{-1}, \cdots, \Sigma^{-1})(\boldsymbol{\theta} - g(\boldsymbol{\Psi}))\Big) \\ &\propto exp\Big(-\frac{1}{2}\boldsymbol{\theta}^T\underbrace{(\sigma^{-2}Z_{1:H}Z_{1:H}^T + diag(\Sigma^{-1}, \cdots, \Sigma^{-1}))}_{\Sigma_{**}^{-1}}\boldsymbol{\theta} + \boldsymbol{\theta}^T\{\sigma^{-2}Z_{1:H}Y_{1:H} + diag(\Sigma^{-1}, \cdots, \Sigma^{-1})g(\boldsymbol{\Psi})\}\Big) \\ &\sim \mathcal{N}(\underbrace{\Sigma_{**}\{\sigma^{-2}Z_{1:H}Y_{1:H} + diag(\Sigma^{-1}, \cdots, \Sigma^{-1})g(\boldsymbol{\Psi})\}}_{\boldsymbol{\mu}_{**}}, \Sigma_{**}).\end{aligned}$$

Let $C_{m,k,n}$ be the number of observations in $\mathcal{H}$ that correspond to base arm (m, k, n). We have that

$$\sigma^{-2}Z_{1:H}Z_{1:H}^T + diag(\Sigma^{-1}, \cdots, \Sigma^{-1}) = \sigma^{-2}diag(C_{1,1,1}, \cdots, C_{M,K,N}) + diag(\Sigma^{-1}, \cdots, \Sigma^{-1}).$$

Therefore, for each campaign $m \in [M]$, the posterior of $\boldsymbol{\theta}_m$ follows a normal distribution with mean and variance as:

$$\mathbb{E}(\boldsymbol{\theta}_m \mid \mathcal{H}, g) = Cov(\boldsymbol{\theta}_m \mid \mathcal{H}, g)\left[\Sigma^{-1}g(\boldsymbol{\Psi}_m) + \sigma^{-2}Z_{1:H,m}Y_{1:H}\right]; \;\; Cov(\boldsymbol{\theta}_m \mid \mathcal{H}, g) = \left(\Sigma^{-1} + \sigma^{-2}diag(C_{m,1,1}, \cdots, C_{m,K,N})\right)^{-1},$$

where $Z_{1:H,m}$ is a subset of $Z_{1:H}$ that contains all rows corresponding to ad lines within campaign m. Regardless of the working model from which g is sampled, the posterior formula for $\boldsymbol{\theta}_m \mid \mathcal{H}, g$ remains identical for LR, GP, and NN.

B Computational Efficient Variants

Algorithm 2: Computational Efficient MCMAB-Concurrent

Input : Specification of g and the corresponding prior; known parameters (i.e., σ_ϵ, Σ) of the hierarchical model; Set $\mathcal{H} = \{\}$
for *every decision point t* **do**
  Update the posterior for g as $\mathbb{P}(g|\mathcal{H})$, according to the hierarchical model (3);
  Sample a $\tilde{g} \sim \mathbb{P}(g|\mathcal{H})$;
  for *every campaign m* **do**
    Given $\tilde{g}$, update the posterior for $\boldsymbol{\theta}_m$ as $\mathbb{P}(\boldsymbol{\theta}_m|\tilde{g}, \mathcal{H})$;
    Sample an utility vector $\tilde{\boldsymbol{\theta}}_m \sim \mathbb{P}(\boldsymbol{\theta}_m|\tilde{g}, \mathcal{H})$;
    Take action $\boldsymbol{a}_{m,t} = argmax_{\boldsymbol{a}_{m,\cdot} \in \mathcal{S}_m} \sum_{k \in [K_m]} \tilde{\theta}_{m,k,a_{m,k,\cdot}}$;
    Receive reward $R_{m,t}$;
    Update the dataset as $\mathcal{H} \leftarrow \mathcal{H} \cup \{(m, \boldsymbol{a}_{m,t}, R_{m,t})\}$
  end
end

In practice, for efficient deployment, it is not necessary to update g at every decision time. Especially when observations are received in batches and decisions are made in batch mode, it is more appropriate to follow an offline training-online deployment framework. Specifically, at certain time points, we would update the posterior of the function g based on all the information observed so far. Then, a function instance of g is sampled and used as the prior for subsequent learning of $\boldsymbol{\theta}$ until the next time when g is retrained and sampled. For example, Algorithm 2 outlines a computationally efficient variant of MCMAB tailored for concurrent scenarios. Here, g is trained once daily before delivering recommendations. The efficient variant for sequential settings is designed similarly, wherein g is retrained only at the commencement of each new campaign.

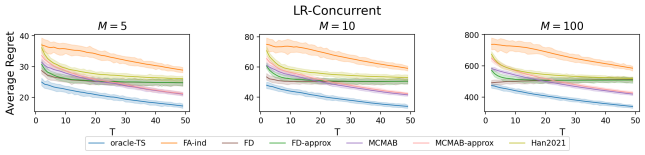

Figure 7: Simulation results for the concurrent linear environment setting with different M specifications. Shaded areas indicate the 95% confidence interval.

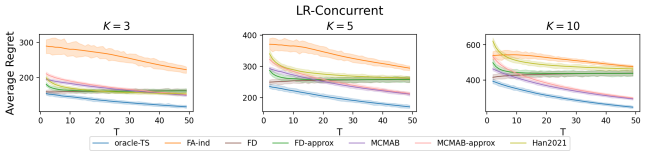

Figure 8: Simulation results for the concurrent linear environment setting with different K specifications. Shaded areas indicate the 95% confidence interval.

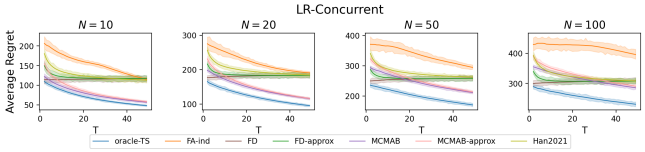

Figure 9: Simulation results for the concurrent linear environment setting with different N specifications. Shaded areas indicate the 95% confidence interval.

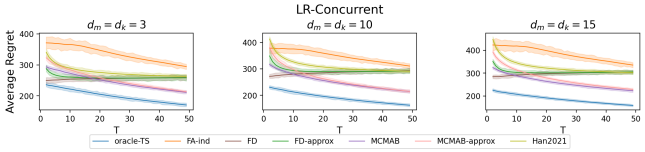

Figure 10: Simulation results for the concurrent linear environment setting with different d specifications. Shaded areas indicate the 95% confidence interval.

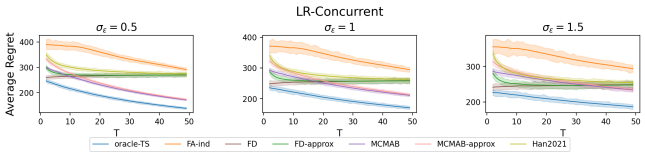

Figure 11: Simulation results for the concurrent linear environment setting with different σ_ϵ specifications. Shaded areas indicate the 95% confidence interval.

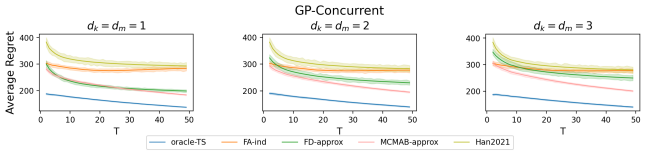

Figure 12: Simulation results for the concurrent nonlinear environment setting with different d specifications, evaluating GP-based algorithms. Uninformative priors with zero mean and an RBF kernel are used. Shaded areas indicate the 95% confidence interval.

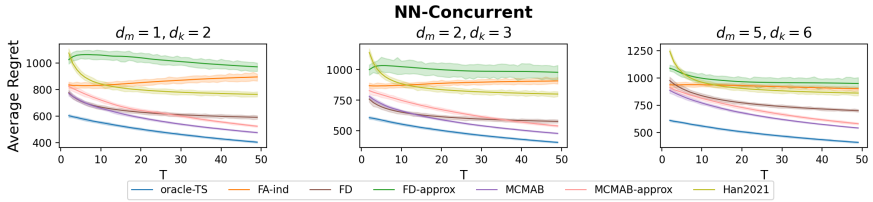

Figure 13: Simulation results for the concurrent nonlinear environment setting with different d specifications, evaluating NN-based algorithms. We set a regularization parameter of 1, and a learning rate of.01 for all NN-based approaches.$(L = 2, m = 12)$ for NN-based algorithms in the setting with $(d_m = 1, d_k = 2)$, $(L = 3, m = 14)$ for NN-based algorithms in the setting with $(d_m = 2, d_k = 3)$, and $(L = 3, m = 20)$ for NN-based algorithms in the setting with $(d_m = 5, d_k = 6)$. Shaded areas indicate the 95% confidence interval.

C Additional Experiment Results

In this section, we present additional results for the MCMAB's performance under various settings of $(M, K, N, d, \sigma_\epsilon)$. Specifically, Figures 7-11 examine MCMAB's behavior in a concurrent linear environment, using LR as the working model. To minimize computational costs, we set a baseline hyper-parameter configuration as $\mathbf{\Sigma} = \sigma_m^2 \boldsymbol{I} = .75^2 \boldsymbol{I}$, $M = 50$, $K = 5$, $N = 50$, $\sigma_\epsilon = 1$, and $d_m = d_k = 3$. All findings are averaged over 100 random seeds. Overall, MCMAB consistently outperforms other baseline methods, achieving lower regrets that gradually approach the performance of the oracle-TS. Our results can be summarized as follows: **i)** With an increase in M (Figure 7), MCMAB effectively learns the task distribution, benefiting from the availability of more diverse metadata, especially in the initial stages. **ii)** As K increases (Figure 8), MCMAB effectively learns the task distribution more quickly, despite the expected rise in optimization and learning complexity. This is attributed to the greater availability of data for environmental estimation and the robust performance of the efficient dynamic programming algorithm. **iii)** With a higher N (Figure 9), although the learning challenge intensifies for all algorithms, MCMAB maintains lower regret levels. **iv)** An increase in d_m/d_k (Figure 10) adds complexity to the learning process. Nevertheless, MCMAB begins to show superior performance, even with a limited time horizon of $T = 20$. **v)** As σ_ϵ rises (Figure 11), the learning task complexity increases, yet MCMAB still exhibits a clear advantage. Additionally, we explore the performance of MCMAB with GP and NN as working models in the concurrent nonlinear environments with different specifications of d_m/d_k in Figure 12 and Figure 13, respectively. Similar to what we observed in Figure 10, as d_m/d_k increases, the learning process gets more complicated, but MCMAB still demonstrates better performance than other baseline algorithms.