Non-Stationary Dueling Bandits for Online Learning to Rank

Shiyin Lu[1], Yuan Miao[2], Ping Yang[2], Yao Hu[2], and Lijun Zhang(✉)[1]

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{lusy, zhanglj}@lamda.nju.edu.cn
[2] Alibaba Group, Hangzhou 311121, China
{miaoyuan.my, yangping.yangping, yaoohu}@alibaba-inc.com

Abstract. We study online learning to rank (OL2R), where a parameterized ranking model is optimized based on sequential feedback from users. A natural and popular approach for OL2R is to formulate it as a multi-armed dueling bandits problem, where each arm corresponds to a ranker, i.e., the ranking model with a specific parameter configuration. While the dueling bandits and its application to OL2R have been extensively studied in the literature, existing works focus on static environments where the preference order over rankers is assumed to be stationary. However, this assumption is often violated in real-world OL2R applications as user preference typically changes with time and so does the optimal ranker. To address this problem, we propose non-stationary dueling bandits where the preference order over rankers is modeled by a time-variant function. We develop an efficient and adaptive method for non-stationary dueling bandits with strong theoretical guarantees. The main idea of our method is to run multiple dueling bandits gradient descent (DBGD) algorithms with different step sizes in parallel and employ a meta algorithm to dynamically combine these DBGD algorithms according to their real-time performance. With straightforward extensions, our method can also apply to existing DBGD-type algorithms. Extensive experiments on public datasets demonstrate that our method can improve the ranking performance of DBGD-type algorithms in non-stationary environments with affordable time and space complexities.

Keywords: Online learning to rank · Dueling bandits · Non-Stationary environments.

1 Introduction

As a powerful ranking optimization paradigm, learning to rank has found applications in a variety of information retrieval scenarios such as web search, online advertising, and recommendation systems [12, 14]. In the classical offline learning to rank, a parameterized ranking model is first trained on collected queries and documents with relevance labels, and then deployed to respond to users' queries with predicted relevant documents. A drawback of offline learning to

rank is that the process of collecting training data with relevance labels is highly time-consuming and expensive in large-scale applications [6]. Furthermore, as the ranking model is fixed after being deployed, it cannot track the evolution of user needs [9].

To address these issues, recent advances in information retrieval have introduced online learning to rank (OL2R), where the ranking model is optimized based on its interactions with users on the fly [4]. Compared to its offline counterpart, OL2R has lighter computational overhead and higher updating frequency. At the heart of OL2R lies the trade-off between exploring new rankers and exploiting the seemingly optimal ranker. Thus, a natural and popular approach for OL2R is to formulate it as a dueling bandits problem [27, 26], where each ranker is viewed as an arm and the ranking model is optimized through sequential noisy comparisons between rankers. While the dueling bandits based methods have been widely studied for OL2R, they are limited in that the preference order over rankers is assumed to follow stationary probability distributions. However, in real-world scenarios, user preference typically changes with time, making the stationary assumption invalid.

To better cope with real-world ranking tasks, we investigate dueling bandits with non-stationary preference probability distributions for OL2R. Specifically, let $\mathbf{w}$ and $\mathbf{w}'$ be two points in the parameter space of the ranking model. We model the probability that users prefer the ranking results produced by a ranker with parameter $\mathbf{w}$ over those of a ranker with parameter $\mathbf{w}'$ by a composite function $f_t(\mathbf{w}, \mathbf{w}') = \sigma(v_t(\mathbf{w}) - v_t(\mathbf{w}'))$, where σ is a static link function, and v_t denotes the utility function in round t. Compared to the existing works on dueling bandits, the novelty of our model is that the utility function can change with time t, capturing the non-stationarity of user preference. Since v_t and $v_{t'}$ can be different for $t \neq t'$, the optimal parameter $\mathbf{w}_t^*$ that maximizes v_t and hence the optimal ranker can change with time, making the non-stationary dueling bandits much harder to deal with than its stationary counterpart.

Nevertheless, by drawing inspiration from recent progress in dynamic online optimization [28, 29], we develop an efficient and adaptive method for non-stationary dueling bandits. Our method follows the prediction with expert advice framework [1] and has a two layer hierarchical structure: multiple dueling bandits gradient descent (DBGD)[27] algorithms running parallel in the bottom and a meta algorithm aggregating the outputs of DBGDs in the top. Generally speaking, DBGDs aim at balancing the exploration-exploitation tradeoff, which also exists in the classical stationary dueling bandits, and the meta algorithm is responsible for tracking the change of utility functions, which is a new task arising only in our non-stationary setting. Under mild assumptions, we prove that our method guarantees no-regret learning, indicating that when the number of rounds goes infinity, the average performance of our method is the same as that of a clairvoyant who knows the optimal ranker in each round. Furthermore, we show that our method, while developed in the context of DBGD, can be also straightforwardly extended to existing variants of DBGD. Finally, we conduct

extensive experiments on public datasets to demonstrate the effectiveness and efficiency of our method for OL2R in non-stationary environments.

2 Related Work

In their seminal work [27], Yue and Joachims proposed a dueling bandits formulation for OL2R, where the ranking model is continuously updated based on noisy comparisons between different rankers. By using an interleaving method to infer user preference and estimate gradients, they developed an online algorithm termed as Dueling Bandits Gradient Descent (DBGD) and established a sub-linear regret bound for DBGD, indicating that the average performance of DBGD converges to that of the best offline ranker.

Due to its simplicity and effectiveness, DBGD has been a popular basis in OL2R and various extensions of DBGD have been proposed. Hoffman et al. first investigated the idea of reusing historical data to speed up learning [7]. They designed a method where the exploratory rankers are filtered according to historical comparisons. While this method enjoys faster learning, it suffers the bias issue of historical data and thus its ranking quality could degrade over time [17]. Wang et al. proposed to accelerate learning by reducing exploration from the whole parameter space to a null space of recently badly performing gradients and selectively constructing candidate rankers for comparisons [25]. Later, a variance reduction method which projects gradients into the document space while maintaining the unbiased property of gradients is developed [24]. It is showed that this method can be combined with existing DBGD-type algorithms for improving ranking performance [24].

Another way to speed up learning is to simultaneously evaluate multiple rankers in each round. Schuth et al. extended DBGD to the Multileave Gradient Descent (MGD) algorithm [20], which replaces the interleaving method in DBGD with multileaving methods [21] so as to compare multiple exploratory directions. In a subsequent work, Zhao and King proposed the Dual-Point Dueling Bandits Gradient Descent algorithm [30], which samples two exploratory vectors with opposite directions for variance reduction. Oosterhuis and Rijke focused on the trade-off between faster learning and higher ranking quality and developed a cascade method which initially optimizes a simple model and switches to a complex model when convergence of the simple model is detected [15]. They empirically showed that the cascade method can learn user preference faster without sacrificing the ranking quality.

A common feature of the above works is that they fall into the static dueling bandits framework. To the best of our knowledge, we are the first to study dueling bandits as well as its application to OL2R in non-stationary environments. Finally, departing from dueling bandits, there also exist other lines of research that study OL2R in other bandits frameworks such as cascading bandits [11], contextual bandits [10], and ranked bandits [22]. However, these works deal with each query independently and thus are not suitable for large-size OL2R applications.

3 Problem Setup

We study non-stationary dueling bandits for online learning to rank, which proceeds in a sequence of rounds. Let $\mathcal{W} \subseteq \mathbb{R}^d$ be the parameter space of a ranking model and T be the number of rounds. Following previous work [16, 25, 24], we refer to the ranking model with a specific parameter configuration as a ranker. In each round $t \in [T] = \{1, \ldots, T\}$, firstly a learner chooses two rankers with parameters $\mathbf{w}_t \in \mathcal{W}$ and $\mathbf{w}'_t \in \mathcal{W}$, respectively. Then, the ranking lists produced by the rankers are merged by an interleaving method [19, 8]. The merged list is displayed to a user and a noisy preference order over the rankers is inferred from the user's click feedback. Specifically, the ranker whose ranking list receives more clicks is preferred. Finally, the learner updates the parameter of the ranking model based on the inferred preference order.

We denote by $\mathbf{w} \succ \mathbf{w}'$ the event that users prefer the ranking list produced by the ranker $\mathbf{w}$ than that of the ranker $\mathbf{w}'$. While the existing works only consider the setting where the probability of this event is fixed, we allow the probability to change with time so as to capture the non-stationary nature of user preference. Specifically, in round t, the probability of the event $\mathbf{w} \succ \mathbf{w}'$ is defined as

$$\Pr(\mathbf{w} \succ \mathbf{w}'|t) = f_t(\mathbf{w}, \mathbf{w}') = \sigma(v_t(\mathbf{w}) - v_t(\mathbf{w}')) \tag{1}$$

where σ is a static link function, and v_t denotes the utility function in round t. Following previous work [27, 24], we make some standard assumptions as follows:

- The parameter space of the ranking model $\mathcal{W}$ is bounded

$$\max_{\mathbf{w} \in \mathcal{W}} \|\mathbf{w}\|_2 \leq R. \tag{2}$$

- The link function σ is rotation-symmetric

$$\sigma(x) = 1 - \sigma(-x). \tag{3}$$

- The link function σ is monotonically increasing and satisfies

$$\sigma(-\infty) = 0, \quad \sigma(0) = 1/2, \quad \sigma(\infty) = 1.$$

- The link function σ is L_σ-Lipschitz, and all utility functions $v_t, t \in [T]$ are L_v-Lipschitz. Furthermore, the link function σ is also second order L_2-Lipschitz.[3]

Denoting $L = L_\sigma L_v$, the above assumptions directly imply the functions $f_t, t \in [T]$ are L-Lipschitz in both arguments.

Let $\mathbf{w}_t^* = \arg\max_{\mathbf{w} \in \mathcal{W}} v_t(\mathbf{w})$ denote the optimal ranker achieving the maximum utility in round t. We adopt dynamic regret as performance metric, defined as

$$\mathrm{DR}(T) = \sum_{t=1}^{T} \big(f_t(\mathbf{w}_t^*, \mathbf{w}_t) + f_t(\mathbf{w}_t^*, \mathbf{w}'_t) - 2 f_t(\mathbf{w}_t^*, \mathbf{w}_t^*) \big).$$

Our goal is to design an online learning method for minimizing the above dynamic regret.

[3] In OL2R, a widely used link function is the sigmoid function $\sigma(x) = 1/\big(1+\exp(-x)\big)$, which satisfies all of our assumptions.

4 Method

In this section, we first review the dueling bandits gradient descent (DBGD) algorithm and derive its dynamic regret bound, then present our method as well as its theoretical guarantee, and finally discuss the extensions of our method to existing DBGD-type algorithms.

4.1 Dueling Bandits Gradient Descent

As outlined in Algorithm 1, DBGD has two hyperparameters δ and γ, corresponding to the step sizes of exploration and exploitation, respectively. In each round t, DBGD first draws a vector $\mathbf{u}_t$ uniformly at random from the unit sphere $\mathbb{S} \triangleq \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\|_2 = 1\}$ as an exploratory direction. Then, a candidate ranker is created with parameter

$$\mathbf{w}'_t = \Pi_{\mathcal{W}}[\mathbf{w}_t + \delta \mathbf{u}_t] \tag{4}$$

where $\mathbf{w}_t$ is the current parameter of the ranking model and $\Pi_{\mathcal{W}}[\cdot]$ denotes the operation of projecting a point to the parameter space $\mathcal{W}$. Next, the two rankers $\mathbf{w}_t$ and $\mathbf{w}'_t$ are compared by the probabilistic interleaving method [8], which can merge the ranking lists produced by the two rankers and infer a preference order over the two rankers from user clicks on the merged ranking list. Finally, based on the preference order, DBGD updates the parameter of the ranking model for the next round. Specifically, if $\mathbf{w}'_t$ wins, which reveals that the exploratory direction leads to better ranking performance, then the parameter of the ranking model moves along the exploratory direction with step size γ:

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}[\mathbf{w}_t + \gamma \mathbf{u}_t].$$

Otherwise, the ranking model remains unchanged.

We rigorously analyze the learning properties of DBGD and derive a sublinear dynamic regret bound as follows.

Theorem 1. *Let C_T be the path length of the optimal rankers over T rounds, defined as*

$$C_T = \sum_{t=2}^{T} \|\mathbf{w}_t^* - \mathbf{w}_{t-1}^*\|_2. \tag{5}$$

By setting $\delta = \sqrt{\frac{2\lambda d}{(11+2\lambda)L\sqrt{T}}}$ and $\gamma = \sqrt{\frac{5R^2+2RC_T}{T}}$, the dynamic regret of DBGD satisfies

$$\mathbb{E}[\mathrm{DR}(T)] \leq \sqrt{2(11+2\lambda)\lambda d L}\left(1+\sqrt{5R^2+2RC_T}\right)T^{\frac{3}{4}}.$$

Algorithm 1 DBGD

Require: step sizes of exploration δ and exploitation γ
1: Initialize a ranker $\mathbf{w}_1 \in \mathcal{W}$ arbitrarily
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Draw a vector $\mathbf{u}_t$ uniformly at random from $\mathbb{S}$
4: Create an exploratory ranker $\mathbf{w}'_t = \Pi_{\mathcal{W}}[\mathbf{w}_t + \delta \mathbf{u}_t]$
5: Compare $\mathbf{w}_t$ and $\mathbf{w}'_t$ by probabilistic interleaving
6: **if** $\mathbf{w}'_t \succ \mathbf{w}_t$ **then**
7: Set $\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}[\mathbf{w}_t + \gamma \mathbf{u}_t]$
8: **else**
9: Set $\mathbf{w}_{t+1} = \mathbf{w}_t$
10: **end if**
11: **end for**

4.2 DBGD Meets Meta Learning

While DBGD can achieve a sub-linear dynamic regret bound for $C_T = o(\sqrt{T})$, it requires the value of the path-length C_T for tuning the step size γ, which is clearly impossible in practice since C_T depends on the unknown optimal rankers $\mathbf{w}_1^*, \ldots, \mathbf{w}_T^*$. To address this issue, we employ the meta learning technique to automatically tune the step size γ, which has exhibited successes in online convex optimization [3, 28, 29]. The basic idea is to run multiple DBGDs in parallel, each of which is configured with a different step size γ and admits the sub-linear dynamic regret bound for a class of path length. We develop our method in the prediction with expert advice framework, where each DBGD is viewed as an expert and the outputs of DBGDs are combined by an expert-tracking algorithm.

We now describe our method in detail, which is termed as DBGD Meets Meta Learning (DM^{2}L) and consists of a meta algorithm and an expert algorithm.

Meta Algorithm As outlined in Algorithm 2, at the beginning of the meta algorithm, we invoke the expert algorithm with different step size γ. According to our theoretical analysis, we maintain

$$N = \left\lceil \log_2 \sqrt{1 + 4T/5} \right\rceil + 1 \tag{6}$$

experts and the step size γ of the i-th expert is configured as

$$\gamma_i = 2^{i-1} R \sqrt{5/T}, \quad i = 1, \ldots, N. \tag{7}$$

Each expert $i \in [N]$ is associated with a time-variant weight π_t^i, which is dynamically adjusted according to the real time performance of expert i. For deriving a tighter dynamic regret bound, we take a nonuniform initialization of weights:

$$\pi_1^i = \frac{N+1}{i(i+1)N}, \quad i = 1, \ldots, N. \tag{8}$$

In each round t, we first receive a ranker $\mathbf{w}_t^i$ from each expert $i \in [N]$ and aggregate these rankers according to the weights of experts $\pi_t^i, i \in [N]$ as

$$\mathbf{w}_t = \sum_{i=1}^{N} \pi_t^i \mathbf{w}_t^i.$$

Then, we sample a vector $\mathbf{u}_t$ from the unit sphere $\mathbb{S}$ uniformly at random and compare $\mathbf{w}_t$ with $\mathbf{w}_t' = \Pi_{\mathcal{W}}[\mathbf{w}_t + \delta \mathbf{u}_t]$ by invoking the probabilistic interleaving method, which returns a noisy preference order $\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}$. Next, we update the weight of each expert using an exponential scheme

$$\pi_{t+1}^i = \frac{\pi_t^i \exp(-\alpha \ell_t(\mathbf{w}_t^i))}{\sum_{j=1}^{N} \pi_t^j \exp(-\alpha \ell_t(\mathbf{w}_t^j))}, \quad i = 1, \ldots, N \tag{9}$$

where $\ell_t(\mathbf{w})$ is a surrogate loss function, defined as

$$\ell_t(\mathbf{w}) = -\frac{d}{\delta} \langle \mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}} \mathbf{u}_t, \mathbf{w} - \mathbf{w}_t \rangle$$

which approximately evaluates the real-time performance of the experts. Finally, both the preference order $\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}$ and the exploratory direction $\mathbf{u}_t$ are sent to each expert so that they can update their own rankers accordingly.

Expert Algorithm As summarized in Algorithm 3, the expert algorithm is a variant of DBGD. In each round t, each expert $i \in [N]$ first sends its current ranker $\mathbf{w}_t^i$ to the meta algorithm. Then, each expert receives the same preference order $\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}$ and exploratory direction $\mathbf{u}_t$ from the meta algorithm. Finally, each expert updates its own ranker as

$$\mathbf{w}_{t+1}^i = \Pi_{\mathcal{W}}[\mathbf{w}_t^i + \gamma_i \mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}} \mathbf{u}_t], \quad i = 1, \ldots, N. \tag{10}$$

Different from DBGD, we here take the same updating direction $\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}} \mathbf{u}_t$ for all experts so that only two rankers $\mathbf{w}_t, \mathbf{w}_t'$ need to be compared in each round. While the updating direction is no longer opposite to the gradient of the smoothed function $\nabla h_t(\mathbf{w}_t^i)$, it is the inverse of the gradient of the surrogate loss function $\nabla \ell_t(\mathbf{w}_t^i)$. Thus, the updating rule of each expert can still be viewed as gradient descent and the dynamic regret of each expert can be analyzed following the proof of Theorem 1.

We present the theoretical guarantee of our method DM^{2}L in the following theorem. Compared to DBGD, the main advantage of DM^{2}L is that it can achieve the sub-linear dynamic regret bound without prior knowledge of the path length C_T and thus can adapt to unknown non-stationarity of environments.

Theorem 2. *By setting* $\delta = \sqrt{\frac{3\lambda d}{(11+2\lambda)L\sqrt{T}}}$ *and* $\alpha = 4/\sqrt{T}$ *and using the configurations in (6) and (7), DM^{2}L achieves the following dynamic regret bound*

$$\mathbb{E}[\mathrm{DR}(T)] \leq \sqrt{3(11+2\lambda)\lambda d L} \left(1 + \sqrt{5R^2 + 2RC_T}\right) T^{\frac{3}{4}} + \lambda(1 + \ln(N+1))\sqrt{T}.$$

Algorithm 2 DM^{2}L: Meta Algorithm

Require: number of experts N, step size of exploration δ, step sizes of exploitation $\gamma_1, \ldots, \gamma_N$, learning rate α

1: Invoke Algorithm 3 with γ_i for each expert $i \in [N]$
2: Initialize the weights of experts $\pi_1^i, i \in [N]$ by (8)
3: **for** $t = 1, 2, \ldots, T$ **do**
4: Receive ranker $\mathbf{w}_t^i$ from each expert $i \in [N]$
5: Aggregate the rankers as $\mathbf{w}_t = \sum_{i=1}^N \pi_t^i \mathbf{w}_t^i$
6: Draw a vector $\mathbf{u}_t$ uniformly at random from $\mathbb{S}$
7: Create an exploratory ranker $\mathbf{w}_t' = \Pi_{\mathcal{W}}[\mathbf{w}_t + \delta \mathbf{u}_t]$
8: Compare $\mathbf{w}_t$ and $\mathbf{w}_t'$ by probabilistic interleaving
9: Update the weight of each expert $\pi_t^i, i \in [N]$ by (9)
10: Send $\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}$ and $\mathbf{u}_t$ to each expert $i \in [N]$
11: **end for**

Algorithm 3 DM^{2}L: Expert Algorithm

Require: step size of exploitation γ_i

1: Initialize a ranker $\mathbf{w}_1^i \in \mathcal{W}$ arbitrarily
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Send ranker $\mathbf{w}_t^i$ to Algorithm 2
4: Receive $\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}$ and $\mathbf{u}_t$ from Algorithm 2
5: Update ranker $\mathbf{w}_{t+1}^i = \Pi_{\mathcal{W}}[\mathbf{w}_t^i + \gamma_i \mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}} \mathbf{u}_t]$
6: **end for**

Remark 1. The above theorem implies that the gap between the average performance of our method and that of the optimal rankers can be bounded by $O\big(T^{-1/4}\sqrt{1 + C_T}\big)$, which converges to zero for $C_T = o(\sqrt{T})$ when T goes to infinity.

4.3 Extensions to DBGD-type Algorithms

While our meta learning method is developed in the context of DBGD, it be also straightforwardly extended to existing DBGD-type algorithms such as MGD [20] and NSGD-DSP [25, 24]. Note that the existing DBGD-type algorithms only differ in the exploratory direction and the updating direction. Thus, we can replace Steps 6-8 at Algorithm 2 with the corresponding exploration pseudocodes of the DBGD-type algorithm and set $\mathbf{u}_t$ used in Steps 9-10 at Algorithm 2 as the updating direction in the DBGD-type algorithm, while keeping Algorithm 3 and the other steps of Algorithm 2 unchanged. We termed the algorithms obtained by applying our meta learning method to MGD and NSGD-DSP as M^{3}L (MGD Meets Meta Learning) and NM^{2}L (NSGD-DSP Meets Meta Learning), respectively.

5 Analysis

In this section, we provide the proofs of Theorem 1 and Theorem 2.

5.1 Proof of Theorem 1

Let $\mathbb{I}_{\{\cdot\}}$ denote the indicator function. We can rewrite the updating rule of DBGD as

$$\mathbf{w}_{t+1} = \Pi_{\mathcal{W}}[\mathbf{w}_t + \gamma \mathbb{I}_{\{\mathbf{w}'_t \succ \mathbf{w}_t\}} \mathbf{u}_t].$$

It is well known that the above updating rule can be viewed as performing gradient descent over a smoothed function [27, 24]

$$h_t(\mathbf{w}) = \mathbb{E}_{\mathbf{u} \in \mathbb{B}}[f_t(\mathbf{w}_t, \Pi_{\mathcal{W}}[\mathbf{w} + \delta \mathbf{u}])]$$

thus accounting for the name of DBGD, where $\mathbb{B} \subseteq \mathbb{R}^d$ is the unit ball. Formally, we have the following lemma.

Lemma 1. *Let $\mathcal{F}_t$ be the learning history up to round t, i.e., the sigma-field generated by random variables $\{\mathbf{w}_s, \mathbf{w}'_s, \mathbb{I}_{\{\mathbf{w}_s \succ \mathbf{w}'_s\}}\}_{s=1,\ldots,t}$. For all $t = 1, \ldots, T$, we have*

$$\mathbb{E}[\mathbb{I}_{\{\mathbf{w}'_t \succ \mathbf{w}_t\}} \mathbf{u}_t | \mathcal{F}_{t-1}] = -\frac{\delta}{d} \nabla h_t(\mathbf{w}_t).$$

Proof. By Lemma 2 in [27], we have

$$\nabla h_t(\mathbf{w}_t) = \frac{d}{\delta} \mathbb{E}_{\mathbf{u} \in \mathbb{S}}[f_t(\mathbf{w}_t, \Pi_{\mathcal{W}}[\mathbf{w}_t + \delta \mathbf{u}]) \mathbf{u} | \mathcal{F}_{t-1}].$$

On the other hand, by (1), (3), and (4), we get

$$\begin{aligned}
\mathbb{E}[\mathbb{I}_{\{\mathbf{w}'_t \succ \mathbf{w}_t\}} \mathbf{u}_t | \mathcal{F}_{t-1}] &= \mathbb{E}_{\mathbf{u}_t \in \mathbb{S}}[f_t(\mathbf{w}'_t, \mathbf{w}_t) \mathbf{u}_t | \mathcal{F}_{t-1}] \\
&= \mathbb{E}_{\mathbf{u}_t \in \mathbb{S}}[\mathbf{u}_t | \mathcal{F}_{t-1}] - \mathbb{E}_{\mathbf{u}_t \in \mathbb{S}}[f_t(\mathbf{w}_t, \mathbf{w}'_t) \mathbf{u}_t | \mathcal{F}_{t-1}] \\
&= -\mathbb{E}_{\mathbf{u}_t \in \mathbb{S}}[f_t(\mathbf{w}_t, \Pi_{\mathcal{W}}[\mathbf{w}_t + \delta \mathbf{u}_t]) \mathbf{u}_t | \mathcal{F}_{t-1}].
\end{aligned}$$

Combing the above two equalities, we finish the proof.

To proceed, we show that the smoothed function $h_t(\mathbf{w})$ enjoys nice properties as follows.

Lemma 2. *For $\delta \in \left(0, \frac{L_\sigma}{L_v L_2}\right)$ and $t = 1, \ldots, T$,*

- *the smoothed function $h_t(\mathbf{w})$ is close to $f_t(\mathbf{w}_t, \mathbf{w})$:*

$$|h_t(\mathbf{w}) - f_t(\mathbf{w}_t, \mathbf{w})| \leq \delta L, \ \forall \mathbf{w} \in \mathcal{W}$$

- *the smoothed function $h_t(\mathbf{w})$ is almost convex:*

$$h_t(\mathbf{w}_t) - h_t(\mathbf{w}^*_t) \leq \lambda \langle \nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}^*_t \rangle + (3 + \lambda) \delta L$$

where $\lambda = \frac{L_\sigma}{L_\sigma - \delta L_v L_2}$.

Proof. The first inequality follows from the Lipschitz continuity of f_t, and the second inequality is analogous to Theorem 1 in [27], the proof of which is thus omitted here.

Equipped with the above two lemmas, we are now ready to prove Theorem 1. We first bound the dynamic regret as

$$\begin{aligned}
\mathrm{DR}(T) &= \sum_{t=1}^{T} \big(f_t(\mathbf{w}_t^*, \mathbf{w}_t) + f_t(\mathbf{w}_t^*, \mathbf{w}_t') - 2f_t(\mathbf{w}_t^*, \mathbf{w}_t^*)\big) \\
&\leq 2\sum_{t=1}^{T} \big(f_t(\mathbf{w}_t^*, \mathbf{w}_t) - f_t(\mathbf{w}_t^*, \mathbf{w}_t^*)\big) + \delta LT \\
&= 2\sum_{t=1}^{T} \big(f_t(\mathbf{w}_t, \mathbf{w}_t) - f_t(\mathbf{w}_t, \mathbf{w}_t^*)\big) + \delta LT \qquad (11) \\
&\leq 2\sum_{t=1}^{T} \big(h_t(\mathbf{w}_t) - h_t(\mathbf{w}_t^*)\big) + 5\delta LT \\
&\leq 2\lambda \sum_{t=1}^{T} \langle \nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^* \rangle + (11 + 2\lambda)\delta LT
\end{aligned}$$

where the first inequality holds since f_t is L-Lipschitz in both arguments, the second equality is due to $\sigma(0) = 1/2$, and the last two inequalities follow from Lemma 2.

Define $\hat{\mathbf{w}}_{t+1} = \mathbf{w}_t + \gamma \mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}} \mathbf{u}_t$ and $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\mathcal{F}_{t-1}]$. By Lemma 1, we have

$$\nabla h_t(\mathbf{w}_t) = -\frac{d}{\delta}\mathbb{E}_t[\mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}\mathbf{u}_t] = -\frac{d}{\delta\gamma}\mathbb{E}_t[\hat{\mathbf{w}}_{t+1} - \mathbf{w}_t].$$

It follows that

$$\begin{aligned}
\langle \nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^* \rangle &= \frac{d}{\delta\gamma}\mathbb{E}_t[\langle \mathbf{w}_t - \hat{\mathbf{w}}_{t+1}, \mathbf{w}_t - \mathbf{w}_t^* \rangle] \\
&= \frac{d}{2\delta\gamma}\mathbb{E}_t\left[\|\mathbf{w}_t - \mathbf{w}_t^*\|_2^2 - \|\hat{\mathbf{w}}_{t+1} - \mathbf{w}_t^*\|_2^2 + \|\mathbf{w}_t - \hat{\mathbf{w}}_{t+1}\|_2^2\right] \\
&\leq \frac{d}{2\delta\gamma}\mathbb{E}\left[\|\mathbf{w}_t - \mathbf{w}_t^*\|_2^2 - \|\mathbf{w}_{t+1} - \mathbf{w}_t^*\|_2^2 + \gamma^2\right] \\
&= \frac{d}{2\delta\gamma}\mathbb{E}\left[\|\mathbf{w}_t\|_2^2 - \|\mathbf{w}_{t+1}\|_2^2 + 2\langle \mathbf{w}_{t+1} - \mathbf{w}_t, \mathbf{w}_t^* \rangle + \gamma^2\right]
\end{aligned}$$

where the inequality holds since $\mathbf{w}_{t+1} = \mathit{\Pi}_{\mathcal{W}}[\hat{\mathbf{w}}_{t+1}]$ and $\mathbf{u}_t \in \mathbb{S}$. Summing the above inequality over $t = 1, \ldots, T$ and taking expectations in both sides, we get

$$
\begin{aligned}
&\mathbb{E}\big[\textstyle\sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^*\rangle\big] \\
&\quad\leq \frac{d}{\delta}\mathbb{E}\left[\frac{\|\mathbf{w}_1\|_2^2}{2\gamma} + \frac{1}{\gamma}\sum_{t=1}^{T}\langle \mathbf{w}_{t+1} - \mathbf{w}_t, \mathbf{w}_t^*\rangle + \frac{\gamma T}{2}\right] \\
&\quad\leq \frac{d}{\delta}\mathbb{E}\left[\frac{R^2}{2\gamma} + \sum_{t=2}^{T}\frac{\langle \mathbf{w}_{t-1}^* - \mathbf{w}_t^*, \mathbf{w}_t\rangle}{\gamma} + \frac{\langle \mathbf{w}_{T+1}, \mathbf{w}_T^*\rangle - \langle \mathbf{w}_1, \mathbf{w}_1^*\rangle}{\gamma} + \frac{\gamma T}{2}\right] \\
&\quad\leq \frac{5dR^2 + 2dRC_T}{2\delta\gamma} + \frac{d\gamma T}{2\delta}
\end{aligned}
$$

where the last two inequalities are due to (2) and (5). Substituting the above inequality into (11) completes the proof.

5.2 Proof of Theorem 2

Following (11), we have

$$
\mathrm{DR}(T) \leq 2\lambda\sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^*\rangle + (11 + 2\lambda)\delta LT.
$$

Fix an expert i. We divide $\sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^*\rangle$ into

$$
\sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^i\rangle + \sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t^i - \mathbf{w}_t^*\rangle.
$$

Following the proof of Theorem 1 and utilizing the updating rule in (10), we can bound the second term as

$$
\mathbb{E}\left[\sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t^i - \mathbf{w}_t^*\rangle\right] \leq \frac{5dR^2 + 2dRC_T}{2\delta\gamma_i} + \frac{d\gamma_i T}{2\delta}.
$$

It remains to bound the first term. Let $\mathbf{z}_t \triangleq [z_t^1, \ldots, z_t^N]$ be a vector with $z_t^j = \ell_t(\mathbf{w}_t^j), j \in [N]$. Then, the weight vectors $\boldsymbol{\pi}_t \triangleq [\pi_t^1, \ldots, \pi_t^N], t \in [T]$ are identical to the outputs of the Hedge algorithm [1] applied to a prediction with expert advice problem with loss vectors $\mathbf{z}_t, t \in [T]$. Thus, we can apply the theoretical guarantee of Hedge [1] and get

$$
\mathbb{E}\left[\sum_{t=1}^{T}\langle\boldsymbol{\pi_t}, \mathbf{z}_t\rangle\right] - \mathbb{E}\left[\sum_{t=1}^{T} z_t^i\right] \leq \frac{2\ln(N+1)}{\alpha} + \frac{\alpha T}{8}.
$$

On the other hand, by Lemma 1, we have

$$
\mathbb{E}\left[\sum_{t=1}^{T}\langle\nabla h_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}_t^i\rangle\right] = \mathbb{E}\left[\sum_{t=1}^{T} -\ell_t(\mathbf{w}_t^i)\right] = -\mathbb{E}\left[\sum_{t=1}^{T} z_t^i\right]
$$

and

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle \boldsymbol{\pi_t}, \mathbf{z}_t\rangle\right] = \mathbb{E}\left[\sum_{t=1}^{T}\sum_{j=1}^{N}\pi_t^j \ell_t(\mathbf{w}_t^j)\right]$$
$$= \mathbb{E}\left[\sum_{t=1}^{T}\sum_{j=1}^{N} -\pi_t^j \frac{d}{\delta}\langle \mathbb{I}_{\{\mathbf{w}_t' \succ \mathbf{w}_t\}}\mathbf{u}_t, \mathbf{w}_t^j - \mathbf{w}_t\rangle\right] = 0$$

where the last equality is due to $\mathbf{w}_t = \sum_{i=j}^{N}\pi_t^j \mathbf{w}_t^j$.

Putting everything together, we obtain for any expert i,

$$\mathbb{E}[\mathrm{DR}(T)] \leq 2\lambda\left(\frac{5dR^2 + 2dRC_T}{2\delta\gamma_i} + \frac{d\gamma_i T}{2\delta}\right) + (11+2\lambda)\delta LT + \frac{4\lambda\ln(N+1)}{\alpha} + \frac{\alpha\lambda T}{4}.$$

Define $\gamma^* = \sqrt{\frac{5R^2+2RC_T}{T}}$. Since the assumption in (2) implies $0 \leq C_T \leq 2RT$, by (7) there must exist an expert i whose step size γ_i satisfies $\gamma_i \leq \gamma^* \leq 2\gamma_i$. Thus, we have

$$\mathbb{E}[\mathrm{DR}(T)] \leq 2\lambda\left(\frac{5dR^2 + 2dRC_T}{\delta\gamma^*} + \frac{d\gamma^* T}{2\delta}\right) + (11+2\lambda)\delta LT + \frac{4\lambda\ln(N+1)}{\alpha} + \frac{\alpha\lambda T}{4}.$$

Substituting $\gamma^* = \sqrt{\frac{5R^2+2RC_T}{T}}$, $\delta = \sqrt{\frac{3\lambda d}{(11+2\lambda)L\sqrt{T}}}$, and $\alpha = 4/\sqrt{T}$ into the above inequality finishes the proof.

6 Experiments

In this section, we present experimental results to demonstrate the effectiveness and efficiency of our meta learning method. We adopt DBGD and the following OL2R algorithms as baselines:

- MGD [20]: Multiple directions are explored in each round. A multileaving method is applied to simultaneously evaluate these directions and the parameter of the ranking model is updated towards the mean of the winning directions.
- PDGD [16]: Instead of estimating gradients from user clicks on interleaved lists, PDGD constructs gradients based on the inferred preferences between document pairs.
- NSGD-DSP [25, 24]: The exploratory directions are sampled from a null space of recently badly performing gradients rather than the whole unit sphere. The winning directions are first projected into a space spanned by the feature vectors of examined documents and then used to update the parameter of the ranking model.

Except for PDGD, all the baseline algorithms belong to the class of DBGD-type algorithms and our meta learning method can apply to them as discussed in the previous section. We illustrate the efficacy of our meta learning method by comparing the performance of each DBGD-type algorithms, with and without our meta learning method applied. The NSGD-DSP and PDGD algorithms are the two state-of-the-art OL2R algorithms and the latter does not fall into the DBGD framework. We thus also compare NSGD-DSP with our meta learning method applied against PDGD to show the superiority of our meta learning method for OL2R in non-stationary environments.

6.1 Setup

Datasets We perform experiments on five datasets, including NP2003 [13], MQ2007 [13], MQ2008 [13], MSLR [18], and the Yahoo! Learning to Rank Challenge dataset [2]. All the datasets are publicly available and have been widely used in the literature [24].

The NP2003 dataset is constructed based on the the name-page finding task at TREC 2003 [23]. It contains about 150 queries, each of which is associated with over 1000 documents that need to be ranked. Each query-document pair is represented by a feature vector with 64 dimensions and labelled with either 0 (not relevant) or 1 (relevant).

The MQ2007 and MQ2008 datasets originate from the Million Query track of TREC in 2007 and 2008, consisting of about 1700 and 800 queries, respectively. Both MQ2007 and MQ2008 represent query-document pairs by 46-dimensional feature vectors and the relevance of each query-document pair is categorized into three grades: 0 (not relevant), 1 (relatively relevant), and 2 (very relevant).

The MSLR dataset is collected from the Web search logs of Microsoft Bing. The number of queries in MSLR is exactly 10000 and the number of associated documents per query is about 125. Each query-document pair is encoded into a 136-dimensional vector, the components of which represent commonly-used ranking features including TF-IDF, PageRank, etc. The relevance label provided by MSLR is more fine-grained and ranges from 0 (not relevant) to 4 (very relevant).

The last dataset used in our experiments is the Yahoo! Learning to Rank Challenge dataset, which is collected from the Web logs of the Yahoo! search engine. The Yahoo! dataset contains about 36000 queries and 883000 documents. Each query-document pairs has 700 ranking features and is also labelled with five-level relevance assessment ranging from 0 (not relevant) to 4 (very relevant).

Experimental Framework We construct an online and non-stationary experimental framework to evaluate the performance of the examined OL2R algorithms. The framework proceeds in a sequence of rounds $t = 1, \ldots, T$. In each round t, firstly a query is uniformly sampled from the dataset, which, together with the documents that need to be ranked, is passed to an OL2R algorithm. Then, the algorithm outputs a ranked list of the documents, which is displayed to a user. Finally, the user makes click feedback on the displayed documents, which is revealed to the algorithm for its inner updating.

Table 1: Click probability $p_c(r)$ and stop probability $p_s(r)$ of each type of user

User	$p_c(0)$	$p_c(1)$	$p_c(2)$	$p_c(3)$	$p_c(4)$	$p_s(0)$	$p_s(1)$	$p_s(2)$	$p_s(3)$	$p_s(4)$
Perfect	0.0	0.2	0.4	0.8	1.0	0.0	0.0	0.0	0.0	0.0
Navigational	0.05	0.3	0.5	0.7	0.95	0.2	0.3	0.5	0.7	0.9
Informational	0.4	0.6	0.7	0.8	0.9	0.1	0.2	0.3	0.4	0.5

Following previous work [16, 25, 24], we use the cascade click model [5] to simulate user's interactions with the displayed documents. This model assumes that user scans through the list of documents from top to bottom. For each document, user clicks on it with probability $p_c(r)$, where r denotes the relevance label of the document. To simulate non-stationary environments, we randomly swap the relevance labels of the documents. Specifically, in each round t, let $y_{t,i}, i = 1, 2, \ldots$ denote the relevance labels of the documents that need to be ranked. With probability 0.3, we modify each relevance label from $y_{t,i}$ to $y'_{t,i}$ as

$$y'_{t,i} = \begin{cases} 2, & y_{t,i} = 1 \\ 1, & y_{t,i} = 2 \\ 4, & y_{t,i} = 3 \\ 3, & y_{t,i} = 4 \end{cases}$$

for all datasets except NP2003 and $y'_{t,i} = 1 - y_{t,i}$ for the NP2003 dataset. In this way, the optimal ranker varies with time and the experimental environments are hence non-stationary. After clicking and visiting a document, user can either scan through the remaining documents or stop, and the latter occurs with probability $p_s(r)$. By configuring the cascade click model with different $p_c(r)$ and $p_s(r)$, we can simulate different types of user. As outlined in Table 1, we consider three types of user including: perfect user who clicks on all relevant documents and never stops until the document list ends, navigational user who often stops when a relevant document is visited, and informational user who sometimes clicks on irrelevant documents.

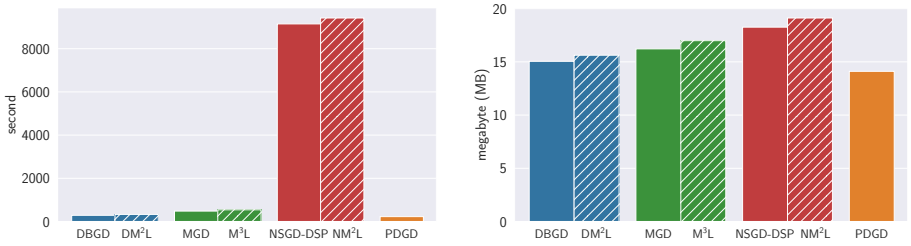
Evaluation Metric Following previous work[16, 25, 24], we use the Normalized Discounted Cumulative Gain (NDCG) as the ranking metric. For each OL2R algorithm, we evaluate its overall ranking performance during the T rounds by the cumulative NDCG@10:

$$\text{cumulative NDCG@10} = \sum_{t=1}^{T} 0.995^{t-1} \text{NDCG@10}(t)$$

where NDCG@10(t) corresponds to the algorithm's ranking performance in round t and 0.995 is a discount factor. This factor is commonly used in OL2R literature [25] for better modelling user experience as user may abandon the OL2R

Table 2: Cumulative NDCG@10 of each OL2R algorithm on the five datasets with the three user types. The standard deviations are shown in the small font within parentheses. The relative improvement of each DBGD-type algorithm after applying our meta learning method is denoted by percentages. The best performance on each dataset with each user type is indicated in bold.

User	Algorithm	NP2003	MQ2007	MQ2008	MSLR	Yahoo!
Per.	DBGD	100.3 (2.8)	59.2 (4.3)	75.2 (5.2)	46.3 (3.6)	105.3 (3.2)
	DM^{2}L	118.1 (2.5)	63.3 (5.4)	80.3 (4.7)	50.3 (3.9)	111.1 (3.8)
		+17.7%	+6.9%	+6.8%	+8.6%	+5.5%
	MGD	107.7 (3.6)	57.7 (3.8)	73.7 (4.9)	47.0 (3.4)	103.8 (3.8)
	M^{3}L	127.8 (4.5)	64.9 (3.9)	81.1 (5.1)	52.0 (3.2)	113.4 (2.6)
		+18.7%	+12.5%	+10.0%	+10.6%	+9.2%
	NSGD-DSP	126.1 (3.0)	61.3 (3.1)	78.9 (4.2)	49.9 (2.5)	112.9 (2.8)
	NM^{2}L	**139.5 (2.5)**	**69.1 (3.4)**	**87.1 (5.3)**	**55.2 (3.1)**	**126.9 (2.1)**
		+10.6%	+12.7%	+10.4%	+10.6%	+12.4%
	PDGD	131.8 (3.9)	64.9 (2.8)	82.5 (5.5)	52.8 (4.7)	120.3 (3.0)
Nav.	DBGD	94.1 (7.4)	55.7 (5.2)	72.4 (5.6)	45.1 (3.7)	103.1 (4.8)
	DM^{2}L	98.2 (6.1)	61.3 (6.1)	77.3 (4.4)	48.7 (3.6)	108.5 (5.3)
		+4.4%	+10.1%	+6.8%	+8.0%	+5.2%
	MGD	102.3 (5.3)	56.5 (4.3)	73.6 (4.9)	46.7 (3.3)	103.3 (4.6)
	M^{3}L	116.9 (6.3)	62.5 (3.9)	80.6 (4.2)	50.3 (5.2)	109.8 (3.4)
		+14.3%	+10.6%	+9.5%	+7.7%	+6.3%
	NSGD-DSP	119.9 (4.9)	61.6 (2.8)	77.2 (5.0)	48.6 (2.9)	112.0 (4.0)
	NM^{2}L	**134.4 (5.6)**	**68.9 (3.1)**	**85.0 (4.5)**	**53.9 (3.4)**	**123.1 (3.7)**
		+12.1%	+11.9%	+10.1%	+10.9%	+9.9%
	PDGD	112.6 (5.0)	59.7 (3.6)	80.1 (4.1)	47.8 (5.1)	113.6 (5.7)
Inf.	DBGD	79.5 (12.4)	53.8 (5.1)	69.4 (6.2)	43.9 (3.8)	100.6 (4.7)
	DM^{2}L	86.6 (14.2)	57.9 (4.8)	73.7 (4.9)	46.7 (5.3)	105.8 (5.1)
		+8.9%	+7.6%	+6.2%	+6.4%	+5.2%
	MGD	95.3 (10.4)	52.8 (4.7)	70.6 (5.4)	44.6 (5.1)	101.7 (5.3)
	M^{3}L	105.3 (11.6)	58.6 (5.3)	76.4 (6.3)	47.3 (4.7)	107.7 (4.6)
		+10.5%	+11.0%	+8.2%	+6.1%	+5.9%
	NSGD-DSP	113.5 (7.1)	60.7 (4.0)	75.5 (5.0)	48.3 (3.4)	110.9 (4.9)
	NM^{2}L	**122.0 (7.9)**	**65.4 (3.2)**	**82.1 (4.1)**	**52.1 (3.8)**	**120.4 (5.2)**
		+7.5%	+7.7%	+8.7%	+7.9%	+8.6%
	PDGD	100.1 (9.3)	58.9 (4.4)	74.6 (5.4)	46.2 (6.0)	108.4 (6.3)



(a) time consumption (b) memory usage

Fig. 1: Total time consumption and memory usage of each OL2R algorithm over the five datasets and three user types.

algorithm after observing bad ranking results. Since $0.995^{1000} \leq 1\%$, we set the number of online rounds as $T = 1000$.

Another popular evaluation metric for OL2R algorithms is the offline NDCG, which is obtained by applying OL2R algorithms iterated after T rounds to held-out testing datasets. While in stationary environments studied by previous work [16, 25, 24], the offline NDCG can measure the final convergence of OL2R algorithms, it is meaningless for non-stationary environments investigated in this work, as the optimal ranker varies with time and there does not exist a global optimal ranker to which OL2R algorithms should converge.

6.2 Results and Analysis

We configure the hyperparameters of the baseline algorithms according to the suggestions in their original papers. The step size of exploration in our meta learning method is set as $\delta = 1$ in order to be consistent with the baseline algorithms and the other hyperparameters of our meta learning method are configured according to Theorem 2. We run each OL2R algorithm ten times and report the average cumulative NDCG over the ten runs in Table 2. Recall that the DM^{2}L, M^{3}L, and NM^{2}L algorithms are yielded by applying our meta learning method to DBGD, MGD, and NSGD-DSP, respectively. As can be seen from Table 2, our meta learning method consistently improves the cumulative NDCG of each DBGD-type algorithm. This ie expected as the existing DBGD-type algorithms are designed for stationary environments with the aim of learning the static optimal ranker, and our meta learning method enables them to track the time-varying optimal rankers in non-stationary environments, as shown by Theorem 2. While MGD is a refined variant of DBGD, its cumulative NDCG is smaller than that of DBGD in several cases, implying that in non-stationary environments only simultaneously exploring multiple directions does not necessarily boost the performance. This is distinct from the empirical observations in stationary environments [25, 24], where MGD dominates DBGD. By contrast, as the state-of-the-art DBGD-type algorithm, NSGD-DSP always outperforms MGD and DBGD, suggesting that the null space exploration and the document

space projection techniques, although originally designed for stationary environments [25, 24], can also lead to performance improvement in non-stationary environments.

Further comparing NSGD-DSP with PDGD, it can be seen that on all the five datasets, PDGD behaves better than NSGD-DSP for perfect user, while inferior to NSGD-DSP for informational user. Thus, we can conclude that these two state-of-the-art OL2R algorithms are generally not comparable and each has its own advantage in non-stationary environments, depending on the user type. Nevertheless, by equipping NSGD-DSP with our meta learning method, the NM^{2}L algorithm achieves the largest cumulative NDCG on all datasets for all user types, demonstrating the superiority of NM^{2}L and the value of our our meta learning method for non-stationary OL2R applications. We also observe that the standard deviation of the cumulative NDCG of NM^{2}L is comparable to that of NSGD-DSP, which implies that our meta learning method does not hinder the effect of NSGD-DSP in variance reduction.

Finally, we investigate the time and space complexities of our meta learning method when applied to DBGD-type algorithms. Theoretically, according to the pseudocodes in Algorithm 1, 2, 3 and the discussion in the previous section, the additional time and space complexities introduced by our meta learning method are both $O(dN) = O(d \log T)$ per round, where d is the dimension of feature vector and T is the time horizon. Since the complexities of each DBGD-type algorithm are at least $O(d)$ per round and $\log T$ grows very slowly with T, we can conclude that the complexities of our meta learning method are affordable. Empirically, we compare the actual time consumption and memory usage on the five datasets of each DBGD-type algorithm, with and without our meta learning method applied. As can be seen from Figure 1, the time and memory consumed by our meta learning method are ignorable compared to those used by each DBGD-type algorithm, which validates the time and space efficiency of our meta learning method.

7 Conclusion and Future Work

We have formulated a new bandits model for OL2R, termed as non-stationary dueling bandits, where the preference order over rankers can change with time. For this bandits model, we developed a meta learning method, which dynamically aggregates multiple DBGD algorithms with different step sizes. Theoretical analysis showed that under mild assumptions, our meta learning method enjoys a sub-linear dynamic regret bound. We also discuss the extensions of our meta learning method to existing DBGD-type algorithms. Extensive experiments on public datasets demonstrate the effectiveness and efficiency of our meta learning method for OL2R in non-stationary environments.

Currently, we only focus on the dueling bandits approach and our meta learning method can only apply to DBGD-type algorithms. As shown in the experiments, the PDGD algorithm, which is also designed for stationary environments but does not fall into the DBGD framework, outperforms the state-of-the-art

DBGD-type algorithm in several cases. Thus, it would be interesting to investigate whether PDGD can be extended to non-stationary environments for boosting performance, which we leave as a future work.

Acknowledgements. This work was partially supported by NSFC (61976112) and JiangsuSF (BK20200064). We thank the anonymous reviewers for their constructive suggestions.

References

1. Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge university press (2006)
2. Chapelle, O., Chang, Y.: Yahoo! learning to rank challenge overview. In: Proceedings of the learning to rank challenge. pp. 1–24 (2011)
3. van Erven, T., Koolen, W.M.: Metagrad: Multiple learning rates in online learning. In: Advances in Neural Information Processing Systems 29. pp. 3666–3674 (2016)
4. Grotov, A., Rijke, M.: Online learning to rank for information retrieval. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1215–1218 (2016)
5. Guo, F., Liu, C., Wang, Y.M.: Efficient multiple-click models in web search. In: Proceedings of the 2nd Acm International Conference on Web Search and Data Mining. pp. 124–131 (2009)
6. Hofmann, K.: Fast and reliable online learning to rank for information retrieval. Ph.D. Dissertation (2013)
7. Hofmann, K., Schuth, A., Whiteson, S., Rijke, M.: Reusing historical interaction data for faster online learning to rank for ir. In: Proceedings of the 6th ACM International Conference on Web Search and Data Mining. pp. 183–192 (2013)
8. Hofmann, K., Whiteson, S., Rijke, M.: A probabilistic method for inferring preferences from clicks. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management. pp. 249–258 (2011)
9. Hofmann, K., Whiteson, S., Rijke, M.: Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. Information Retrieval **16**(1), 63–90 (2013)
10. Hofmann, K., Whiteson, S., de Rijke, M., et al.: Contextual bandits for information retrieval. In: Proceedings of NeurIPS 2011 Workshop on Bayesian Optimization, Experimental Design, and Bandits, Granada (2011)
11. Kveton, B., Szepesvari, C., Wen, Z., Ashkan, A.: Cascading bandits: Learning to rank in the cascade model. In: Proceedings of the 32nd International Conference on Machine Learning. pp. 767–776 (2015)
12. Liu, T.Y.: Learning to rank for information retrieval. Foundations and Trends in Information Retrieval **3**(3), 225–331 (2009)
13. Liu, T., Xu, J., Qin, T., Xiong, W., Li, H.: Letor: Benchmark dataset for research on learning to rank for information retrieval. In: Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval. pp. 137–145 (2007)
14. Long, B., Bian, J., Chapelle, O., Zhang, Y., Inagaki, Y., Chang, Y.: Active learning for ranking through expected loss optimization. IEEE Transactions on Knowledge and Data Engineering **27**(5), 1180–1191 (2014)

15. Oosterhuis, H., de Rijke, M.: Balancing speed and quality in online learning to rank for information retrieval. In: Proceedings of the 2017 ACM Conference on Information and Knowledge Management. pp. 277–286 (2017)
16. Oosterhuis, H., Rijke, M.: Differentiable unbiased online learning to rank. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 1293–1302 (2018)
17. Oosterhuis, H., Schuth, A., Rijke, M.: Probabilistic multileave gradient descent. In: Proceedings of the 38th European Conference on Information Retrieval. pp. 661–668 (2016)
18. Qin, T., Liu, T.Y.: Introducing letor 4.0 datasets. arXiv preprint arXiv:1306.2597 (2013)
19. Radlinski, F., Kurup, M., Joachims, T.: How does clickthrough data reflect retrieval quality? In: Proceedings of the 17th ACM Conference on Information and Knowledge Management. pp. 43–52 (2008)
20. Schuth, A., Oosterhuis, H., Whiteson, S., Rijke, M.: Multileave gradient descent for fast online learning to rank. In: Proceedings of the 9th ACM International Conference on Web Search and Data Mining. pp. 457–466 (2016)
21. Schuth, A., Sietsma, F., Whiteson, S., Lefortier, D., de Rijke, M.: Multileaved comparisons for fast online evaluation. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management. pp. 71–80 (2014)
22. Slivkins, A., Radlinski, F., Gollapudi, S.: Ranked bandits in metric spaces: learning diverse rankings over large document collections. Journal of Machine Learning Research **14**(Feb), 399–436 (2013)
23. Voorhees, E.M., Harman, D.K., et al.: TREC: Experiment and evaluation in information retrieval, vol. 63. MIT press Cambridge (2005)
24. Wang, H., Kim, S., McCord-Snook, E., Wu, Q., Wang, H.: Variance reduction in gradient exploration for online learning to rank. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 835–844 (2019)
25. Wang, H., Langley, R., Kim, S., McCord-Snook, E., Wang, H.: Efficient exploration of gradient space for online learning to rank. In: Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 145–154 (2018)
26. Yue, Y., Broder, J., Kleinberg, R., Joachims, T.: The k-armed dueling bandits problem. Journal of Computer and System Sciences **78**(5), 1538–1556 (2012)
27. Yue, Y., Joachims, T.: Interactively optimizing information retrieval systems as a dueling bandits problem. In: Proceedings of the 26th International Conference on Machine Learning. pp. 1201–1208 (2009)
28. Zhang, L., Lu, S., Zhou, Z.H.: Adaptive online learning in dynamic environments. In: Advances in Neural Information Processing Systems 31. pp. 1323–1333 (2018)
29. Zhao, P., Wang, G., Zhang, L., Zhou, Z.H.: Bandit convex optimization in non-stationary environments. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics. pp. 1508–1518 (2020)
30. Zhao, T., King, I.: Constructing reliable gradient exploration for online learning to rank. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. pp. 1643–1652 (2016)