AdaO2B: Adaptive Online to Batch Conversion for Out-of-Distribution Generalization

Xiao Zhang,[1] **Sunhao Dai,**[1] **Jun Xu,**[1,*] **Yong Liu,**[1] **Zhenhua Dong**[2]

[1] Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[2] Huawei Noah's Ark Lab, Shenzhen, China
{zhangx89,sunhaodai,junxu,liuyonggsai}@ruc.edu.cn
dongzhenhua@huawei.com

Abstract

Online to batch conversion involves constructing a new batch learner by utilizing a series of models generated by an existing online learning algorithm, for achieving generalization guarantees under i.i.d assumption. However, when applied to real-world streaming applications such as streaming recommender systems, the data stream may be sampled from time-varying distributions instead of persistently being i.i.d. This poses a challenge in terms of out-of-distribution (OOD) generalization. Existing approaches employ fixed conversion mechanisms that are unable to adapt to novel testing distributions, hindering the testing accuracy of the batch learner. To address these issues, we propose AdaO2B, an adaptive online to batch conversion approach under the bandit setting. AdaO2B is designed to be aware of the distribution shifts in the testing data and achieves OOD generalization guarantees. Specifically, AdaO2B can dynamically combine the sequence of models learned by a contextual bandit algorithm and determine appropriate combination weights using a context-aware weighting function. This innovative approach allows for the conversion of a sequence of models into a batch learner that facilitates OOD generalization. Theoretical analysis provides justification for why and how the learned adaptive batch learner can achieve OOD generalization error guarantees. Experimental results have demonstrated that AdaO2B significantly outperforms state-of-the-art baselines on both synthetic and real-world recommendation datasets.

Introduction

Online learning aims at conducting sequential decision-making by capturing the dynamic nature of data stream, which generates an updated model at each round and uses it for the next decision-making round (Cesa-Bianchi and Lugosi 2006; Shalev-Shwartz 2011; Zhang and Liao 2019). To achieve regret guarantees of the decision process, online learning algorithms need to update the model incrementally upon receiving the new instances. Due to its high computational cost and decision instability, updating the model fully online is rather unrealistic for many real-world applications (*e.g.*, streaming recommender systems). An effective approach is *online to batch conversion* (Littlestone 1989; Cesa-Bianchi, Conconi, and Gentile 2004), which constructs a batch learner based on the model sequence generated by an existing online learning algorithm. The *batch learner* is fixed during the testing process, which aims to benefit from the sequence of existing models for good generalization abilities.

Classic online to batch (O2B) conversion approaches typically assume that the instances in data streaming are independently and identically distributed (i.i.d.) according to a fixed but unknown distribution. Under the i.i.d. assumption, O2B conversion becomes a process of selecting a representative model or averaging multiple models (Dekel and Singer 2005; Dekel 2008; Cutkosky 2019). But in real-world applications, distribution shifts between training and testing data are ubiquitous, posing new challenges of achieving out-of-distribution (OOD) generalization guarantees through O2B conversions. For example, in streaming recommender systems, users often change their preference dynamically (Hamidzadeh and Moradi 2021; Zhang et al. 2021a; Dai et al. 2023), *e.g.*, a user may prefer different categories of videos due to changes in weather or mood, and the features of a video may change on different timestamp.

Existing O2B conversion technologies are not suitable for the OOD scenarios due to their fixed conversion mechanisms, which can not adjust strategies of combining or selecting models for adaptation to novel (or similar) testing distributions. Figure 1 shows an empirical study on the real-world video recommendation data, where the user preferences in testing data may differ due to the different data-collection time that could incur OOD problem in the testing phase. We can observe that, compared with the fully online learning algorithm, the recommendation performances of classic O2B conversions significantly decreased during the testing phase. The results clearly indicate that the out-of-distribution data could be damaging for O2B conversion in terms of hurting the testing accuracy of a batch learner.

Analysis shows that, in online learning, the received instances in the data stream may be sampled from multiple distributions that violate the i.i.d. assumption. Thus, the models generated by an online learning algorithm have already been trained on data from these distributions, raising the possibility

*Corresponding author: Jun Xu (junxu@ruc.edu.cn).
Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education.

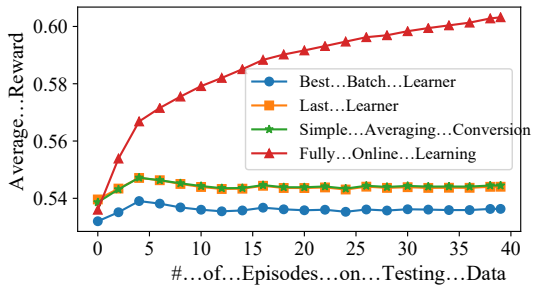


Figure 1: Generalization performances (average rewards on testing data) of classic online to batch conversion technologies and online learning algorithm on `KuaiRec` data, where "Best Batch Learner" selects the model that achieves the highest cumulative reward among all episodes for batch testing, "Simple Averaging Conversion" uses the average over the model sequence for testing (Cesa-Bianchi, Conconi, and Gentile 2004), "Last Learner" chooses the model obtained in the last episode for testing (Shalev-Shwartz 2007), "Fully Online Learning" keeps updating the learner on the testing data in an online manner, and the online learning backbone is sequential batch UCB (Han et al. 2020).

of constructing a batch learner for OOD generalization. Motivated by the potential capacity of online learning models, in this paper, we propose an adaptive O2B conversion approach for OOD generalization under the bandit feedback setting of online learning. Specifically, we first demonstrate the OOD generalization error bounds of O2B conversion, theoretically establishing the relationship between the weighted regret and the OOD generalization error of a batch learner. Then, taking the theoretical results as guidance, we propose AdaO2B that can be aware of distribution shifts through an adaptive weighting network and adaptively combines the model sequence for decision-making. AdaO2B uses the observed rewards as the supervised signals and can be efficiently trained using different data selection approaches based on the bandit feedback. Moreover, our AdaO2B is model-agnostic and can be applied to any contextual bandit algorithm.

We summarize the major contributions: (1) Rigorous theoretical analysis that establishes the relationship between the regret in online learning and the OOD generalization error of a batch learner in O2B conversion; (2) A model-agnostic O2B conversion framework called AdaO2B for achieving OOD generalization guarantees under bandit feedback setting; (3) Comprehensive empirical studies showed the effectiveness of AdaO2B in terms of improving the OOD generalization ability of different bandit algorithms and its superiority over state-of-the-art O2B conversion baselines.

Related Work

Online model averaging/selection is an important topic in online learning, which aims at adjusting the model class for prediction in an online manner (Zhang, Liao, and Liao 2019; Zhang and Liao 2020). Online model averaging has been studied in online kernel learning, which typically reduces the model averaging problem to a prediction problem with expert advice (Jin, Hoi, and Yang 2010; Orabona, Fornoni, and Caputo 2010). Zhang *et al.* (Zhang and Liao 2018) presented an efficient online model selection approach using incremental sketched kernel alignment, which formulates an unbiased selection criterion of kernel models. Model selection under bandit settings has received increased attention over the past several years, which can adapt the reward model class of optimal policy (Foster, Krishnamurthy, and Luo 2019; Ghosh, Sankararaman, and Kannan 2021). Since existing online model averaging/selection approaches finally obtain fixed strategies for weighting or selecting the learners, they are not suitable for adapting the OOD testing data.

Transfer learning has received considerable attention in machine learning literature, which aims to improve the generalization performance on target domain by transferring the knowledge contained in previous related source domains (Pan and Yang 2009; Weiss, Khoshgoftaar, and Wang 2016; Zhuang et al. 2020). There are several types of technologies that are commonly used in transfer learning, including weighted ERM (Huang et al. 2006; Reddi, Poczos, and Smola 2015), feature mapping (Ando, Zhang, and Bartlett 2005), regularization (Duan et al. 2009; Kuzborskij and Orabona 2017). More recently, a series of works lies in the idea of transferring the model trained on streaming data for adapting the target domain (Zhao, Cai, and Zhou 2020; Tao and Lu 2020). Unlike the O2B conversion, these works need to specify the data source or the distribution assumption and re-train the model on data from the target source.

Problem Formulation

Let $[m] = \{1, 2, \ldots, m\}$, $\mathcal{S} \subseteq \mathbb{R}^d$ be the context space whose dimension is d. Since bandit feedback is ubiquitous in real-world applications, in this paper we focus on the batched version of contextual bandit (BCB) setting, which is an extension of the classic contextual bandit setting (Perchet et al. 2015; Han et al. 2020; Esfandiari et al. 2021). In BCB setting, the sequential decision-making process is partitioned into N *episodes*, where each episode includes B decision-making steps (B is also called the *batch size*). Specifically, at step b in the n-th episode, a bandit algorithm receives a *candidate context set* $\mathcal{S}_{n,b} \subseteq \mathcal{S}$, where $\mathcal{S}_{n,b}$ contains M contexts $\mathcal{S}_{n,b} = \{\boldsymbol{s}_I\}_{I\in[M]}$ and each context corresponds to a candidate action. Then, the bandit algorithm chooses a context $\boldsymbol{s}_{I_{n,b}} \in \mathcal{S}_{n,b}$ following the *policy* (*i.e.*,, decision model) $f_n : \mathcal{S}_{n,b} \to [M]$ parameterized by $\boldsymbol{\theta}_n$, where $I_{n,b} \in [M]$ can be seen as the index of the *executed action* at step b in the n-th episode. After choosing the context, algorithm will observe a *reward* $R_{n,b}$. Note that, during the decision process in each episode, the bandit policy is fixed. That is, the policy f_n is only updated at the end of the n-th episode based on a *data buffer* $\mathcal{D}_n = \{(\mathcal{S}_{n,b}, I_{n,b}, R_{n,b})\}_{b\in[B]}$.

Figure 2 illustrates the online to batch (O2B) conversion problem for out-of-distribution (OOD) generalization in the above BCB setting. In the *online learning phase*, the policy is incrementally updated on data steam generated from multiple distributions. O2B conversion aims to formulate a batch learner based on the collected data for OOD testing in the

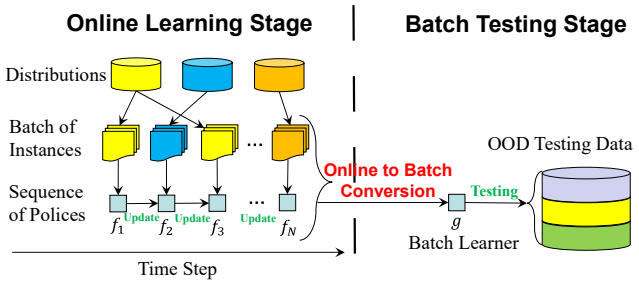


Figure 2: Online to batch conversion in BCB setting on out-of-distribution testing data.

batch testing phase, where the testing distributions may be similar but different. Next, we introduce the formal definition of O2B conversion for OOD generalization in BCB setting.

Definition 1 (O2B Conversion for OOD Generalization)**.** *Consider the BCB setting that includes N episodes. Let $\mathcal{F}_N = \{f_1, f_2, \ldots, f_N\}$ be a sequence of policies (i.e, decision models) generated by performing a bandit algorithm $\mathcal{A}$, and $\mathcal{D} = \{\mathcal{D}_n\}_{n\in[N]}$ be the sequence of data buffers that store the interaction history. Assume that the context-reward pairs are generated according to a distribution $\mathbb{P}$ (may be a mixture distributions[1]). O2B conversion aims to find a O2B function f_{o2b}: $f_{\text{o2b}} : \mathcal{F}_N \times \mathcal{D} \to g \in \mathcal{G}$, where g is a* ***batch learner*** *in a policy space $\mathcal{G}$ which has generalization guarantees on a testing distribution $\mathbb{Q}$ that may be different from the distribution $\mathbb{P}$.*

From Definition 1, we can observe that the key ingredients of O2B conversion for OOD generalization contain: the distribution assumptions of rewards as well as $\mathbb{P}$ and $\mathbb{Q}$, the policy space $\mathcal{G}$ (*i.e.,*, class of the batch learner g), the evaluation metric of OOD generalization.

OOD Generalization Analysis

In this section, we first specify the key ingredients in Definition 1 and then carry out an OOD generalization justification on why and how to conduct O2B conversion in environments with distribution shifts.

We specify the key ingredients in Definition 1 as follows.

Reward r. Following the setups in linear contextual bandit literature (Dimakopoulou et al. 2019; Yang et al. 2021; Li et al. 2010), for any context $\boldsymbol{s}_i \in \mathcal{S} \subseteq \mathbb{R}^d$, we assume that the expectation of the observed reward R_i is determined by unknown *true reward parameters* $\boldsymbol{\theta}^* \in \mathbb{R}^d$: $\mathbb{E}[R_i \mid \boldsymbol{s}_i] = \langle \boldsymbol{\theta}^*, \boldsymbol{s}_i \rangle$. The linear reward will be extended to the convex function case in Corollary 1.

Distributions $\mathbb{P}$ and $\mathbb{Q}$. Given the linear reward assumption, the distribution shift of the context-reward pairs can be described as the shift of context distribution. Thus, we define that $\mathbb{P}$ and $\mathbb{Q}$ are the distributions on the context space $\mathcal{S}$. Besides, $\mathbb{P}$ and $\mathbb{Q}$ are unknown to the algorithm and may be both are mixtures of multiple distributions defined in Corollary 1.

[1]In Corollary 1, we will give the formulation of the mixture distribution representing the sampling process according to multiple distributions.

Policy space $\mathcal{G}$. We assume that $\mathcal{G}$ is the set containing all possible linear combinations of policies in the sequence of policies $\mathcal{F}_N = \{f_1, f_2, \ldots, f_N\}$ generated by a bandit algorithm. Formally, the *adaptive batch learner* $g \in \mathcal{G}$ can be formulated as follows:

$$g(\boldsymbol{s}) = \sum_{n\in[N]} \beta_n \, f_n(\boldsymbol{s})/\beta_{1:N}, \quad \forall \boldsymbol{s} \in \mathcal{S}, \tag{1}$$

where $\{\beta_n\}_{n\in[N]}$ denotes the *combination weights* that are outputs of a context-aware *weighting function* $h : \mathcal{S} \to \mathbb{R}^N$ for adapting the distribution shifts between $\mathbb{P}$ and $\mathbb{Q}$, and $\beta_{1:N}$ denotes the sum of weights over the number of episodes N, *i.e.,*, $\beta_{1:N} := \sum_{n\in[N]} \beta_n$. More specifically, for the linear true reward, bandit policy f_n typically chooses action (*i.e.,*, the context) according to the *estimated reward* $r_n(\boldsymbol{s}) = \langle \boldsymbol{\theta}_n, \boldsymbol{s} \rangle$, where $\boldsymbol{\theta}_n$ denotes the *estimated reward parameters* in f_n from the n-the episode. Then, the adaptive batch learner g in (1) can be represented by the combination of the estimated reward parameters[2]

$$\boldsymbol{\theta}_{\text{ada}} = \sum_{n\in[N]} \beta_n \boldsymbol{\theta}_n / \beta_{1:N}. \tag{2}$$

For the sake of simplicity, we denote the adaptive batch learner g by $\boldsymbol{\theta}_{\text{ada}}$.

Evaluation metric of OOD generalization. Given an unknown distribution (may be a mixture distribution) of the contexts $\boldsymbol{s} \in \mathcal{S}$, define the *expected reward* of a policy with estimated reward parameters $\boldsymbol{\theta}$ w.r.t. $\mathbb{Q}$ to be $\text{ER}_{\mathbb{Q}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{s}\sim\mathbb{Q}}[\langle \boldsymbol{\theta}, \boldsymbol{s} \rangle]$. Then, we define the *generalization error* (also called *expected risk*) of $\boldsymbol{\theta}$ w.r.t. $\mathbb{Q}$ as follows:

$$\text{GE}_{\mathbb{Q}}(\boldsymbol{\theta}) = C_{\text{ER}} - \text{ER}_{\mathbb{Q}}(\boldsymbol{\theta}), \tag{3}$$

where C_{ER} denotes the upper bound of the absolute values of expected reward w.r.t. any distribution. Then, $\text{GE}_{\mathbb{Q}}(\boldsymbol{\theta}_{\text{ada}}$ can be used for evaluating the OOD generalization performance of the adaptive batch learner $\boldsymbol{\theta}_{\text{ada}}$ on testing distribution $\mathbb{Q}$.

Following the above setup of O2B conversion problem, we first demonstrate the OOD generalization error bound of the adaptive batch learner $\boldsymbol{\theta}_{\text{ada}}$ in (2).

Theorem 1 (OOD Generalization Error Bound of Adaptive Batch Learner)**.** *Consider the BCB setting with N episodes and the batch size B. Let $\boldsymbol{\theta}_n, n \in [N]$ be the reward parameters estimated by policy f_n, $\boldsymbol{\theta}^*$ be the true reward parameters, $\boldsymbol{s}_{I_{n,b}} \in \mathcal{S}_{n,b}$ be the context chosen by f_n at step b, and C_{ER} be the upper bound defined in* (3)*. Define the weighted regret as*

$$\text{WReg}(N, B) = \sum_{n\in[N], b\in[B]} \beta_n \left\langle \boldsymbol{\theta}^* - \boldsymbol{\theta}_n, \boldsymbol{s}_{I_{n,b}} \right\rangle, \tag{4}$$

and assume that the weighted regret is bounded by C_{WReg}, i.e., $\text{WReg}(N, B) \le C_{\text{WReg}}$*. Then,*

$$\begin{aligned} &\text{GE}_{\mathbb{Q}}(\boldsymbol{\theta}_{\text{ada}}) - \text{GE}_{\mathbb{P}}(\boldsymbol{\theta}^*) \\ &\le \mathbb{E}_{\mathbb{P}}\left[\frac{C_{\text{WReg}}}{B\beta_{1:N}}\right] + C_{\text{ER}}\sqrt{2\text{D}_{\text{JS}}(\mathbb{Q}\|\mathbb{P})}, \end{aligned} \tag{5}$$

[2]In the batch testing phase, since the batch learner is fixed, we omit the exploration term in the original policy (*e.g.*, the uniform distribution term in EXP3 policy).

where $\mathrm{D}_{\mathrm{JS}}(\mathbb{Q}\|\mathbb{P})$ *denotes the Jensen-Shannon (JS) divergence between the distributions* $\mathbb{Q}$ *and* $\mathbb{P}$.

Remark 1 (Tighten the Bound)**.** *On one hand, the above theorem tells us that the weighted regret upper bounds the OOD generalization error of the adaptive bath learner* $\boldsymbol{\theta}_{\mathrm{ada}}$. *That is, to achieve good OOD generalization performances, the batch learner's objective should be to minimize the weighted regret in* (4) *by adjusting the combination weights* $\{\beta_n\}_{n\in[N]}$. *On the other hand, a smaller JS divergence between the training distribution* $\mathbb{P}$ *and the testing distribution* $\mathbb{Q}$ *leads to a tighter bound in* (5).

Remark 2 (Simply to i.i.d.)**.** *In* (5)*, setting* $\beta_n = 1$ *for all* $n \in [N]$ *and* $\mathbb{Q} = \mathbb{P}$ *yields the i.i.d. risk bound of a simple averaging learner* $\boldsymbol{\theta}_{\mathrm{avg}} = \sum_{n\in[N]} \boldsymbol{\theta}_n/N$ *in bandit setting. Analogous to the case of full-information O2B conversion (Shalev-Shwartz 2007), the result in* (5) *becomes* $\mathrm{GE}_{\mathbb{P}}(\boldsymbol{\theta}_{\mathrm{avg}}) \leq \mathrm{GE}_{\mathbb{P}}(\boldsymbol{\theta}^*) + \mathbb{E}_{\mathbb{P}}\left[C_{\mathrm{Reg}}/T\right]$, *where* C_{Reg} *is the upper bound of worse-case regret and* $T = N \times B$.

As illustrated in Figure 2, the distributions of $\mathbb{P}$ and $\mathbb{Q}$ may be mixture distributions, yielding the following result.

Corollary 1 (OOD Generalization Error for Mixture Distributions)**.** *Assume that the conditions in Theorem 1 hold, and* $\mathbb{P}, \mathbb{Q}$ *are two mixture distributions with distribution densities* $p(\boldsymbol{s}) = \sum_{i\in[W]} \pi_i \cdot p_i(\boldsymbol{s})$ *and* $q(\boldsymbol{s}) = \sum_{i\in[W]} \tau_i \cdot q_i(\boldsymbol{s})$, *respectively, where* $\{p_i\}_{i\in[W]}$ *and* $\{q_i\}_{i\in[W]}$ *are densities of Gaussian distributions in* $\mathbb{P}$ *and* $\mathbb{Q}$, $\boldsymbol{\pi} = \{\pi_i\}_{i\in[W]}$ *and* $\boldsymbol{\tau} = \{\tau_i\}_{i\in[W]}$ *denotes multi-dimensional Bernoulli distributions. Then,*

$$\mathrm{GE}_{\mathbb{Q}}(\boldsymbol{\theta}_{\mathrm{ada}}) - \mathrm{GE}_{\mathbb{P}}(\boldsymbol{\theta}^*) \leq \mathbb{E}_{\mathbb{P}}\left[\frac{C_{\mathrm{WReg}}}{B\beta_{1:N}}\right] + C_{\mathrm{ER}}\sqrt{2\log K + K\sum_{i\in[W]} \pi_i\tau_i \cdot \mathrm{D}_{\mathrm{BKL}}(p_i\|q_i)}, \tag{6}$$

where $\mathrm{D}_{\mathrm{BKL}}(p_i\|q_i) := \frac{1}{2}\left[\mathrm{D}_{\mathrm{KL}}(p_i\|q_i) + \mathrm{D}_{\mathrm{KL}}(q_i\|p_i)\right]$ *denotes the bidirectional Kullback-Leibler (KL) divergence between* p_i *and* q_i *(Liang et al. 2021), and* $\mathrm{D}_{\mathrm{KL}}(p\|q$ *denotes the KL divergence between the distributions* p *and* q.

Remark 3 (Weighted Bidirectional KL Divergence)**.** *The weight term* $\pi_i\tau_i$ *can be seen as the probability of drawing distributions* p_i *and* q_i *for online learning and OOD testing, respectively. Then, the weighted bidirectional KL divergence in* (6) *indicates that, if distributions in* $\mathbb{P}$ *and* $\mathbb{Q}$ *with similar probability of occurrence have similar distribution densities, tighter OOD generalization error bound can be achieved.*

We further give a more general version of Theorem 1, where the expectation of true rewards is convex.

Corollary 2 (OOD Generalization Error for Convex Rewards)**.** *The adaptive batch learner* $\boldsymbol{\theta}_{\mathrm{ada}}$ *enjoys the same upper bound of OOD generalization error as in* (5).

The above results provide theoretical guidance to implement the adaptive O2B learner, and we will derive guiding principles as well as the algorithm in the next section.

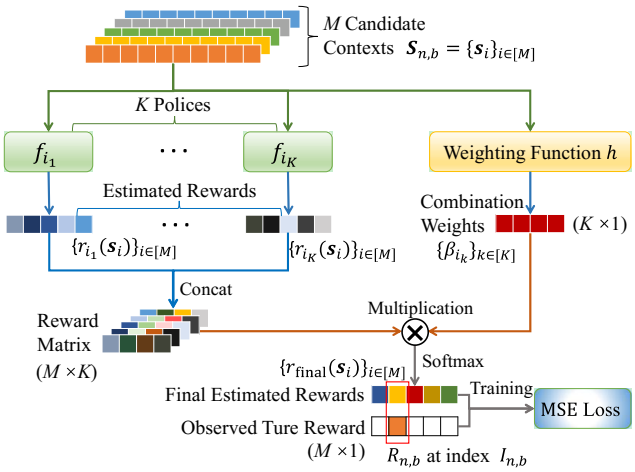


Figure 3: The training process of the proposed AdaO2B based on the data collected at step b in the n-th episodes, where $\{i_k\}_{k\in[K]}$ denotes the index set $\mathcal{K}$.

AdaO2B: The Proposed Algorithm

From Remark 1&3, we can derive the following guiding principles for designing the adaptive O2B algorithm: (a) The data for training the adaptive batch learner should be as close as possible to the testing distribution; (b) The combination weights $\beta_n, n \in [N]$ should be computed based on the received candidate contexts at each step, which can be aware of the changes in test distribution; (c) The objective of training the weighting function should contain the component of minimizing the weighted regret in (4) or its surrogate.

Next, we provide a practical implementation of the adaptive batch learner $\boldsymbol{\theta}_{\mathrm{ada}}$ in (2) named AdaO2B.

Data Selection

We use subsets of the whole sequences of policies and data buffers in Definition 1 for O2B conversion. Specifically, to reduce the difference between the training distribution of O2B conversion and the testing distribution, we propose the following three approaches for maintaining the data buffers (denoted by $\overline{\mathcal{D}}_{\mathcal{K}} := \{\mathcal{D}_n\}_{n\in\mathcal{K}} \subseteq \mathcal{D}$) and the candidate policy set (denoted by $\mathcal{F}_{\mathcal{K}} := \{f_n\}_{n\in\mathcal{K}} \subseteq \mathcal{F}_N$), where $\mathcal{K} \subseteq [N]$ denotes the index set of the selected data buffers which is also the index set of the policies trained on these selected buffers, and $K := |\mathcal{K}|$ denotes the cardinality of $\mathcal{K}$ (*i.e.,*, the cardinality of $\overline{\mathcal{D}}_{\mathcal{K}}$ and $\mathcal{F}_{\mathcal{K}}$):

1) Sliding window approach. Motivated by the fact that streaming data from neighboring periods typically has similar distributions in the streaming applications (Bendada, Salha, and Bontempelli 2020; Zhang et al. 2021b), we use the data buffers and policies from the most recent K episodes for training the adaptive O2B learner, *i.e.,*, $\mathcal{K} = \{N - K + 1, N-K+2, \ldots, N\}$. More specifically, the training process is based on $\overline{\mathcal{D}}_{\mathcal{K}} = \{\mathcal{D}_n\}_{n=N-K+1}^{N}$ that are the data buffers in the recent K episodes, and the candidate policy set $\mathcal{F}_{\mathcal{K}} = \{f_n\}_{n=N-K+1}^{N}$.

2) Reservoir sampling approach. The index set $\mathcal{K}$ can be

sampled uniformly from $[N]$. Without knowing the number of episodes N, we can use a reservoir sampling approach (Knuth 1998) to determining which data buffers and the corresponding candidate policies are stored for O2B conversion, such that every data buffer in $\overline{\mathcal{D}}_{\mathcal{K}}$ (as well as every policy in $\mathcal{F}_{\mathcal{K}}$) has the same probability of being selected.

3) Data-dependent approach. Since the decrease or low growth of the rewards received in two episodes indicates severe distribution shifts, we can maintain the candidate policies with average rewards of low growth rates and the corresponding data buffers. Specifically, we present a data-dependent approach for selecting the elements in $\overline{\mathcal{D}}_{\mathcal{K}}$ and $\mathcal{F}_{\mathcal{K}}$: the index set $\mathcal{K}$ is constructed by selecting the indices with the bottom-K values of $(\overline{R}_{n+1} - \overline{R}_n)/\overline{R}_n, \forall n \in [N]$, where $\overline{R}_n$ denotes the average reward in the n-th episode.

Model Formulation and Model Training

Formally, by only retaining the exploitation term in the policy for batch testing, the k-th policy in $\mathcal{F}_{\mathcal{K}} = \{f_n\}_{n\in\mathcal{K}}$ can be written as $f_{i_k} = \arg\max_{j\in[M]} \langle \boldsymbol{\theta}_{i_k}, \boldsymbol{s}_j \rangle,\ \boldsymbol{s}_j \in \mathcal{S}_{i_k,b}$, which makes decision according to the estimated reward using the estimated reward parameters $\boldsymbol{\theta}_{i_k}$ given the contexts. Then, given the index set $\mathcal{K} = \{i_k\}_{k\in[K]}$, the adaptive batch learner in (2) can be expressed as $\boldsymbol{\theta}_{\mathrm{ada}} = \sum_{k\in[K]} \beta_{i_k}\boldsymbol{\theta}_{i_k}/\beta_{\mathcal{K}},\ i_k \in \mathcal{K}$, where $\beta_{\mathcal{K}} = \sum_{k\in[K]} \beta_{i_k}$. The combination weights $\{\beta_{i_k}\}_{k\in[K]}$ are obtained using the context-aware weighting function $h : \mathcal{S} \to \mathbb{R}^K$. We implement the weighting function h using one MLP (Multi-Layer Perceptron), denoted by MLP_h. As shown in Figure 3, at each step, the weighting function h (*i.e.,*, MLP_h) takes the candidate contexts as input, and obtains a K-dimensional vector $\{\beta_{i_k}\}_{k\in[K]}$.

To incorporate the weighted regret in (4) into the training objective, we transform the weighted regret into a loss function. Given the index set $\mathcal{K}$, for the candidate context set $\mathcal{S}_{i_k,b}$ collected at step b in the i_k-th episodes, the weighted regret truncated with K becomes $\sum_{k\in[K]} \beta_{i_k} \langle \boldsymbol{\theta}^* - \boldsymbol{\theta}_{i_k}, \boldsymbol{s}_j \rangle,\ \boldsymbol{s}_j \in \mathcal{S}_{i_k,b}$, yielding that

$$\left\langle \sum_{k\in[K]} \frac{\beta_{i_k}}{\beta_{\mathcal{K}}} \boldsymbol{\theta}^*, \boldsymbol{s}_j \right\rangle - \langle \boldsymbol{\theta}_{\mathrm{ada}}, \boldsymbol{s}_j \rangle, \quad \boldsymbol{s}_j \in \mathcal{S}_{i_k,b}, \quad (7)$$

where the first term can be seen as an estimate of the true reward, and the second term is the *final estimated reward* estimated by $\boldsymbol{\theta}_{\mathrm{ada}}$: $r_{\mathrm{final}}(\boldsymbol{s}) = \langle \boldsymbol{\theta}_{\mathrm{ada}}, \boldsymbol{s} \rangle$. Then, as a surrogate objective of minimizing (7), we perform the training process of $\boldsymbol{\theta}_{\mathrm{ada}}$ by minimizing the difference between the observed true reward $R_{n,b}$ in data buffers and the final estimated reward. More specifically, in the adaptive batch learner $\boldsymbol{\theta}_{\mathrm{ada}}$, model parameters that need to be trained include the parameters in the weighting function MLP_h. All these trainable parameters in MLP_h are denoted as $\boldsymbol{\Theta}_h$, and trained based on $\overline{\mathcal{D}}_{\mathcal{K}}$. Then, the task of training the weighting function h amounts to minimizing the following mean squared error (MSE) loss:

$$\mathcal{L}_{\boldsymbol{\Theta}_h} = \frac{1}{\left|\overline{\mathcal{D}}_{\mathcal{K}}\right|} \sum_{(I_{n,b},R_{n,b})\in\overline{\mathcal{D}}_{\mathcal{K}}} \left[R_{n,b} - r_{\mathrm{final}}(\boldsymbol{s}_{I_{n,b}})\right]^2 + \lambda\|\boldsymbol{\Theta}_h\|_2^2, \quad (8)$$

Algorithm 1: AdaO2B

INPUT: Batch size B, number of episodes N, bandit algorithm $\mathcal{A}$, number of candidate policies K
OUTPUT: Adaptive batch learner $\boldsymbol{\theta}_{\mathrm{ada}}$

Initialize policy $\boldsymbol{\theta}_1$ using the uniform distribution
Initialize parameter set of policies $\mathcal{F}_{\boldsymbol{\theta}} \leftarrow \emptyset$
// Online Learning Phase
for $n = 1$ **to** N **do**
  for $b = 1$ **to** B **do**
    Receive candidate context set $\mathcal{S}_{n,b}$ and choose the context $s_{I_{n,b}} \in \mathcal{S}_{n,b}$ following policy $\boldsymbol{\theta}_n$
    Observe the reward $R_{n,b}$
  end for
  Store interactions into a data buffer $\mathcal{D}_n \leftarrow \{(\mathcal{S}_{n,b}, I_{n,b}, R_{n,b})\}_{b\in[B]}$
  Store the policy $\mathcal{F}_{\boldsymbol{\theta}} \leftarrow \mathcal{F}_{\boldsymbol{\theta}} \cup \{\boldsymbol{\theta}_n\}$ if $n \in \mathcal{K}$
  Update the policy $\boldsymbol{\theta}_n$ as $\boldsymbol{\theta}_{n+1} \leftarrow \Delta(\boldsymbol{\theta}_n)$ on $\mathcal{D}_n$ using bandit algorithm $\mathcal{A}$
end for
// Online to Batch Conversion Phase
Collect K data buffers $\overline{\mathcal{D}}_{\mathcal{K}} \leftarrow \{\mathcal{D}_n\}_{n\in\mathcal{K}}$
Compute estimated rewards $r_n(\boldsymbol{s})$ using policies in $\mathcal{F}_{\boldsymbol{\theta}}$ for all $n \in \mathcal{K}$ and $\boldsymbol{s} \in \cup_{n\in\mathcal{K},b\in[B]}\{\mathcal{S}_{n,b}\}$
Compute the combination weights $\{\beta_n\}_{n\in\mathcal{K}}$ using MLP_h for each candidate context set in $\overline{\mathcal{D}}_{\mathcal{K}}$
Compute final estimated rewards $\{r_{\mathrm{final}}(\boldsymbol{s}_i)\}_{i\in[M]}$ for each candidate context set in $\overline{\mathcal{D}}_{\mathcal{K}}$
Optimizing the loss in (8) on $\overline{\mathcal{D}}_{\mathcal{K}}$ using Adam and output the model parameters $\boldsymbol{\Theta}_h$ of MLP_h
return $\mathcal{F}_{\boldsymbol{\theta}} = \{\boldsymbol{\theta}_n\}_{n\in\mathcal{K}}, \boldsymbol{\Theta}_h$

where $\|\boldsymbol{\Theta}_h\|_2^2$ is a regularizer for avoiding over-fitting, and $\lambda \geq 0$ is the regularization parameter. Besides, as shown in Figure 3, each final estimated reward can be efficiently computed through the multiplication of the reward matrix that concatenates the estimated rewards, and the vector of combination weights. Then, the obtained final estimated rewards are normalized through a Sofmax function. Finally, Adam (Kingma and Ba 2014) is used to conduct the optimization.

We summarize the above steps in Algorithm 1, called AdaO2B. To facilitate the understanding of the whole process, we involve the online learning phase in AdaO2B. We can specify the data selection process in AdaO2B using three approaches, denoted by AdaO2B-S (sliding window approach), AdaO2B-R (reservoir sampling approach), and AdaO2B-D (data-dependent approach), respectively.

Experiments

We conducted experiments to test the performance of AdaO2B on synthetic data and real-world data.

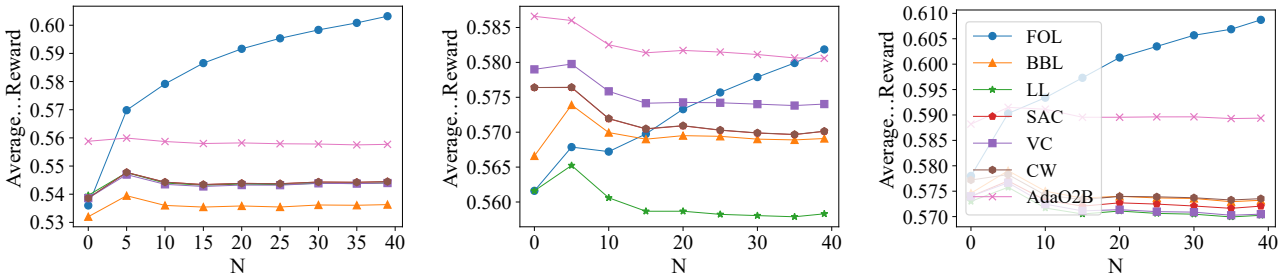
Experimental Settings

Baselines AdaO2B was compared with several classic online to batch conversion algorithms as well as their variants:

Best Batch Learner (BBL) chooses the model with the highest cumulative reward (Dekel and Singer 2005); **Last**

Model Type	Dataset	Synthetic			KuaiRec		
	Algorithm	SBUCB	EXP3-B	BLTS-B	SBUCB	EXP3-B	BLTS-B
Oracle	FOL	0.7644	0.6665	0.7651	0.6032	0.5819	0.6063
Baselines	BBL	0.7354	0.6388	0.7453	0.5363	0.5691	0.5633
	LL	0.7388	0.6441	0.7494	0.5440	0.5583	0.5642
	SAC-S	0.7356	0.6404	0.7397	0.5445	0.5604	0.5661
	SAC-R	0.6929	0.6216	0.7019	0.5417	0.5691	0.5699
	SAC-D	0.7070	0.6230	0.7130	0.5426	0.5701	0.5647
	VC-S	0.7365	0.6396	0.7402	0.5441	0.5595	0.5659
	VC-R	0.6969	0.6086	0.7040	0.5457	0.5723	0.5667
	VC-D	0.7091	0.6194	0.7165	0.5421	0.5712	0.5655
	CW-S	0.7332	0.6418	0.7404	0.5445	0.5603	0.5658
	CW-R	0.7004	0.6228	0.7054	0.5424	0.5688	0.5675
	CW-D	0.7098	0.6252	0.7166	0.5427	0.5701	0.5647
Ours	AdaO2B-S	**0.7848*** (+6.23%)	0.6971* (+8.23%)	**0.7811*** (+4.23%)	0.5556* (+1.81%)	**0.5861*** (+2.41%)	0.5806* (+1.88%)
	AdaO2B-R	0.7692* (+4.11%)	0.7154* (+11.07%)	0.7672* (+2.38%)	**0.5563*** (+1.94%)	**0.5861*** (+ 2.41%)	**0.5832*** (+2.33%)
	AdaO2B-D	0.7777* (+5.27%)	**0.7380*** (+14.58%)	0.7783* (+3.86%)	0.5532* (+1.37%)	0.5833* (+ 1.92%)	0.5796* (+1.7%)

Table 1: Comparisons of average reward (w.r.t. all episodes) for AdaO2B and baselines on synthetic and real-world KuaiRec dataset. Bold values are the best of the proposed algorithm, while the best values in baselines are underlined. '$*$': improvements over baselines are statistical significant (t-test, p-value< 0.05) compared to the best baseline.



(a) Bandit backbone: SBUCB (b) Bandit backbone: EXP3-B (c) Bandit backbone: BLTS-B

Figure 4: Average rewards of baselines and the proposed AdaO2B equipped with the best data selection approach on testing data of KuaiRec dataset, where N denotes the number of episodes in the batch testing phase.

Learner (LL) uses the model from the last episode (Shalev-Shwartz 2007); **Simple Averaging Conversion (SAC)** averages models over episodes (Cesa-Bianchi, Conconi, and Gentile 2004; Shalev-Shwartz 2011); **Voting Conversion (VC)** uses majority voting among candidate policies (Freund and Schapire 1999; Dekel and Singer 2005); **Constant Weight (CW)** averages models using constant weights based on normalized average rewards. These algorithms use different data selection appoaches: Sliding Window (S), Reservoir Sampling (R), and Data-dependent (D). For example, SAC with Sliding Window is "SAC-S".

AdaO2B and these baselines are model-agnostic and can be applied to various bandit algorithms: **Sequential Batch UCB (SBUCB)** (Han et al. 2020) is a batched extension of LinUCB (Li et al. 2010); **Batched EXP3 (EXP3-B)** is a batched version of the adversarial bandit EXP3 (Bistritz et al. 2019); **BLTS-B** uses the Thompson sampling for selecting parameters of estimated rewards (Dimakopoulou et al. 2019).

To compare the performance between O2B conversion and fully online learning, we introduce the following version as oracle for each bandit backbone: **Fully Online Learning (FOL)** keeps updating the bandit policy on the testing data in an online manner.

Evaluation For both synthetic and real-world datasets, we split them into two subsets for the online learning phase (as well as the O2B conversion phase) and the batch testing phase, respectively, denoted by **OL-Data** and **BT-Data**. Following the standard practice in (Zhang et al. 2021b, 2024), we use the average reward to evaluate the accuracy of algorithms as $\frac{1}{nB}\sum_{k=1}^{n}\sum_{b=1}^{B} R_{k,b}$ for the first n episodes, where $R_{k,b}$ is the observed reward at step b in the k-th episode.

Implementation details We trained AdaO2B based on the last 10 (*i.e.,*, $K = 10$) data buffers and history policies. We

tuned the hyper-parameters as follows: the learning rate was tuned within the range of $\{1e{-}2, 1e{-}3, 1e{-}4, 1e{-}5\}$, the weight decay was tuned among $\{1e{-}3, 1e{-}4, 1e{-}5, 1e{-}6\}$, and the batch size was tuned in $\{256, 512, 1024, 2048\}$.

Experiments on Synthetic Data

We first conducted experiments on synthetic dataset which simulates the OOD scenario in online recommendation.

Data To simulate the OOD scenario, we generated synthetic data with $N = 40$ episodes, a batch size of $B = 5,000$, $M = 10$ candidates, and a context dimension of $d = 10$. For OL-Data, candidate context set $\mathcal{S}_{n,b} \subseteq \mathbb{R}^d$ were drawn from a Gaussian distribution $\mathcal{N}(\mu_{\mathrm{s}}\mathbf{1}_d, \sigma_{\mathrm{s}}^2\boldsymbol{I}_d)$, with means μ_{s} ranging from 1 to -2.6 in the first 20 episodes and a standard deviation of $\sigma_{\mathrm{s}} = 0.05$. To simulate the mixture distribution, we set the mean of the first candidate context to 1.2 in the last 20 episodes. For BT-Data, the mean of the first candidate context was set to 1.4 to simulate distribution shifts in testing data. The observed reward was modeled as a sigmoid function $\mathrm{sigmoid}(\langle \boldsymbol{w}_{\mathrm{r}}, \boldsymbol{s}_{n,b}\rangle)$, with coefficients$\boldsymbol{w}_{\mathrm{r}} \in \mathbb{R}^d$ sampled from $\mathcal{N}(0.1, 0.01^2)$.

Results & discussions Table 1 reports the average reward w.r.t. all episodes for the proposed AdaO2B and the baselines on the synthetic dataset. From the result, we can observe that AdaO2B significantly outperformed all the baselines with all three different bandit backbones in terms of the average rewards. Specifically, AdaO2B outperformed the best baseline (LL) by 6.23% with SBUCB, 14.58% with EXP3-B, and 4.23% with BLTS-B. These results verified the effectiveness and model agnosticism of AdaO2B for capturing the distribution shifts and improving the performance of online to conversion in the out-of-distribution scenario. Furthermore, the proposed AdaO2B even outperformed FOL on synthetic data. The reason is that FOL used the exploration and exploitation trade-off strategy on testing data, where the exploration may hurt the accuracy of the bandit policy in a synthetic testing environment.

For the three data selection method used in AdaO2B, we can conclude that: (1) the data-dependent approach achieved higher average rewards than the reservoir sampling approach, indicating the advantage of the data-dependent approach for slight distribution shifts; (2) Sliding window approach typically outperformed other data selection approaches since the distribution of the recent data buffers was more similar to that of the testing data in this synthetic environment.

Experiments on Real-World Data

We also compared the proposed AdaO2B with the baselines on a real-world short video recommendation dataset.

Data We used the KuaiRec dataset[3], which provides a fully observed user-item interaction matrix from the popular video-sharing app Kuaishou. After filtering, the dataset includes 6,980 users, 973 videos, and 400,000 interactions. It also offers detailed side information for both users and videos, including daily item features that vary over time, aligning with our research goals. We processed categorical features as one-hot vectors and concatenated them with integer features, reducing the final feature dimension to 50 using principal component analysis (PCA) following the practices in (Zhang et al. 2021b, 2022; Yoshikawa and Imai 2018), *i.e.,*, $d = 50$. For the candidate set, we retained the original item for each interaction and then randomly sampled 99 extra items from the entire item set (except the original item).

Evaluation protocol In online recommendation, we cannot guarantee that the corresponding feedback of each recommended item to the user can be found in the log data. To overcome this issue and facilitate ground-truth evaluations, following (Lyu et al. 2022; Zhang et al. 2022), we created a simulated online environment to test all the algorithms. More specifically, we first trained a matrix factorization (MF) model (Koren, Bell, and Volinsky 2009) using both OL-Data and BT-Data. AUC of the trained MF model were both over 83%, which assures that the online environment can provide nearly realistic feedbacks of users. At each step, the online environment received a selected context (*i.e.,*, a recommended item) from the algorithm, and returned a user feedback (1 or 0) according to $\mathbb{I}(\hat{y} > \gamma)$, where $\mathbb{I}(\cdot)$ is an indicator function, $\hat{y}$ is the predicted score by the trained MF model and γ is a tuned threshold that the AUC can achieve the highest score.

Results & discussions The results of average rewards of the baselines and the proposed AdaO2B on the KuaiRec dataset are reported in Table 1. We can observe that, AdaO2B outperformed the best baseline (VC-R and SAC-R) by 1.94% with SBUCB, 2.41% with EXP3-B, and 2.33% with BLTS-B. Figure 4 shows the curves of average rewards on KuaiRec dataset, where AdaO2B achieved the highest average reward for all bandit backbones approximating the performances of the oracle FOL. The results verified the effectiveness of the model-agnostic AdaO2B framework in improving existing bandit models for real-world online to batch conversion problem.

Note that FOL with EXP3-B has lower average rewards than FOL with other bandit backbones. This phenomenon is due to the randomized exploration term used in the EXP3-B policy that is more suitable for online adversarial environments. Besides, AdaO2B equipped with the reservoir sampling approach had the best performance than other data selection approach, indicating that a simple reservoir sampling approach could capture the severe distribution drift in real-world applications.

Conclusion

This paper aims to address the out-of-distribution generalization problem in online to batch conversion. Specifically, we propose an adaptive online to batch conversion approach called AdaO2B. The proposed AdaO2B is aware of distribution shifts through adaptively combining the model sequence using a weighting network, takes rigorous theoretical analyses as guidance, and achieves OOD generalization guarantees under the bandit feedback setting. Experimental results demonstrated the effectiveness of AdaO2B in out-of-distribution scenarios.

[3]https://kuairec.com

Acknowledgments

This work was partially supported by the National Science and Technology Major Project (2022ZD0114802), the National Natural Science Foundation of China (No. 62376275, 62377044), Intelligent Social Governance Interdisciplinary Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. Supported by fund for building world-class universities (disciplines) of Renmin University of China. Supported by Public Computing Cloud, Renmin University of China. Supported by the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China (23XNKJ13).

References

Ando, R. K.; Zhang, T.; and Bartlett, P. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(11).

Bendada, W.; Salha, G.; and Bontempelli, T. 2020. Carousel personalization in music streaming apps with contextual bandits. In *Proceedings of the 14th ACM Conference on Recommender Systems*, 420–425.

Bistritz, I.; Zhou, Z.; Chen, X.; Bambos, N.; and Blanchet, J. 2019. Online EXP3 learning in adversarial bandits with delayed feedback. In *Advances in Neural Information Processing Systems 32*, 11349–11358.

Cesa-Bianchi, N.; Conconi, A.; and Gentile, C. 2004. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9): 2050–2057.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge University Press.

Cutkosky, A. 2019. Anytime online-to-batch, optimism and acceleration. In *Proceedings of the 36th International Conference on Machine Learning*, 1446–1454.

Dai, S.; Zhou, Y.; Xu, J.; and Wen, J.-R. 2023. Dually Enhanced Delayed Feedback Modeling for Streaming Conversion Rate Prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 390–399.

Dekel, O. 2008. From online to batch learning with cutoff-averaging. In *Advances in Neural Information Processing Systems 21*.

Dekel, O.; and Singer, Y. 2005. Data-driven online to batch conversions. *Advances in Neural Information Processing Systems 18*.

Dimakopoulou, M.; Zhou, Z.; Athey, S.; and Imbens, G. 2019. Balanced linear contextual bandits. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 3445–3453.

Duan, L.; Tsang, I. W.; Xu, D.; and Chua, T.-S. 2009. Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 289–296.

Esfandiari, H.; Karbasi, A.; Mehrabian, A.; and Mirrokni, V. S. 2021. Regret bounds for batched bandits. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 7340–7348.

Foster, D. J.; Krishnamurthy, A.; and Luo, H. 2019. Model selection for contextual bandits. *Advances in Neural Information Processing Systems 32*.

Freund, Y.; and Schapire, R. E. 1999. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3): 277–296.

Ghosh, A.; Sankararaman, A.; and Kannan, R. 2021. Problem-complexity adaptive model selection for stochastic linear bandits. In *International Conference on Artificial Intelligence and Statistics*, 1396–1404.

Hamidzadeh, J.; and Moradi, M. 2021. Online recommender system considering changes in user's preference. *Journal of AI and Data Mining*, 9(2): 203–212.

Han, Y.; Zhou, Z.; Zhou, Z.; Blanchet, J. H.; Glynn, P. W.; and Ye, Y. 2020. Sequential batch learning in finite-action linear contextual bandits. *CoRR*, abs/2004.06321.

Huang, J.; Gretton, A.; Borgwardt, K.; Schölkopf, B.; and Smola, A. 2006. Correcting sample selection bias by unlabeled data. *Advances in Neural Information Processing Systems 19*.

Jin, R.; Hoi, S. C.; and Yang, T. 2010. Online multiple kernel learning: Algorithms and mistake bounds. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory*, 390–404.

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

Knuth, D. E. 1998. *The Art of Computer Programming, Vol 2, Seminumerical Algorithms*. Addison-Wesley (2nd edition).

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

Kuzborskij, I.; and Orabona, F. 2017. Fast rates by transferring from auxiliary hypotheses. *Machine Learning*, 106(2): 171–195.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, 661–670.

Liang, X.; Wu, L.; Li, J.; Wang, Y.; Meng, Q.; Qin, T.; Chen, W.; Zhang, M.; and Liu, T.-Y. 2021. R-Drop: Regularized dropout for neural networks. In *Advances in Neural Information Processing Systems 34*, 10890–10905.

Littlestone, N. 1989. From on-line to batch learning. In *Proceedings of the 2nd Annual Workshop on Computational Learning Theory*, 269–284.

Lyu, Y.; Dai, S.; Wu, P.; Dai, Q.; Deng, Y.; Hu, W.; Dong, Z.; Xu, J.; Zhu, S.; and Zhou, X.-H. 2022. A Semi-Synthetic Dataset Generation Framework for Causal Inference in Recommender Systems. *arXiv preprint arXiv:2202.11351*.

Orabona, F.; Fornoni, M.; and Caputo, N., Barbara Cesa-Bianchi. 2010. OM-2: An online multi-class multi-kernel learning algorithm. In *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 43–50.

Pan, S. J.; and Yang, Q. 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10): 1345–1359.

Perchet, V.; Rigollet, P.; Chassang, S.; and Snowberg, E. 2015. Batched bandit problems. In *Proceedings of the 28th Conference on Learning Theory*, 1456.

Reddi, S.; Poczos, B.; and Smola, A. 2015. Doubly robust covariate shift correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Shalev-Shwartz, S. 2007. *Online learning: Theory, algorithms, and applications*. Ph.D. thesis, The Hebrew University of Jerusalem.

Shalev-Shwartz, S. 2011. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2): 107–194.

Tao, Y.; and Lu, S. 2020. From online to non-i.i.d. batch learning. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 328–337.

Weiss, K.; Khoshgoftaar, T. M.; and Wang, D. 2016. A survey of transfer learning. *Journal of Big data*, 3(1): 1–40.

Yang, J.; Hu, W.; Lee, J. D.; and Du, S. S. 2021. Impact of representation learning in linear bandits. In *Proceedings of the 9th International Conference on Learning Representations*.

Yoshikawa, Y.; and Imai, Y. 2018. A nonparametric delayed feedback model for conversion rate prediction. *arXiv preprint arXiv:1802.00255*.

Zhang, C.; Chen, S.; Zhang, X.; Dai, S.; Yu, W.; and Xu, J. 2024. Reinforcing Long-Term Performance in Recommender Systems with User-Oriented Exploration Policy. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1850–1860.

Zhang, S.; Liu, H.; He, J.; Han, S.; and Du, X. 2021a. Deep sequential model for anchor recommendation on live streaming platforms. *Big Data Mining and Analytics*, 4(3): 173–182.

Zhang, X.; Dai, S.; Xu, J.; Dong, Z.; Dai, Q.; and Wen, J.-R. 2022. Counteracting User Attention Bias in Music Streaming Recommendation via Reward Modification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2504–2514.

Zhang, X.; Jia, H.; Su, H.; Wang, W.; Xu, J.; and Wen, J. 2021b. Counterfactual reward modification for streaming recommendation with delayed feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 41–50.

Zhang, X.; and Liao, S. 2018. Online kernel selection via incremental sketched kernel alignment. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 3118–3124.

Zhang, X.; and Liao, S. 2019. Incremental randomized sketching for online kernel learning. In *Proceedings of the 36th International Conference on Machine Learning*, 7394–7403.

Zhang, X.; and Liao, S. 2020. Hypothesis sketching for online kernel selection in continuous kernel space. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 2498–2504.

Zhang, X.; Liao, Y.; and Liao, S. 2019. A survey on online kernel selection for online kernel learning. *WIREs Data Mining and Knowledge Discovery*, 9(2): e1295.

Zhao, P.; Cai, L.-W.; and Zhou, Z.-H. 2020. Handling concept drift via model reuse. *Machine Learning*, 109(3): 533–568.

Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; and He, Q. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1): 43–76.