Optimizing Freshness of Machine Learning Model: A Timing (Multi-Armed) Bandit Approach

Qiulin Lin, Liyuan Wang, Junyan Su, and Minghua Chen

***Abstract*—Maintaining the freshness of machine learning (ML) models, i.e., the time elapsed since their last update, is crucial for AI systems to deliver optimal performance. This is particularly relevant for resource-intensive models like Large Language Models, where frequent retraining or fine-tuning incurs significant costs. We address the challenge of jointly optimizing model freshness and update expense by carefully timing and tailoring ML model updates. We formulate this puzzle as a novel timing (multi-armed) bandit problem, where each "arm" represents a specific update timing with associated update costs and unknown, stochastic, model staleness cost. We develop an efficient online algorithm for the problem with provable sublinear regret. Our approach enables the dynamic allocation of resources towards retraining or fine-tuning, striking a balance between maintaining ML model freshness and minimizing update expenses.**

I. Introduction

Machine learning models are powering critical applications such as recommendation engines, autonomous driving, and IoT network operations. These models are trained on data sets and used for making predictions and decisions and improving system performance. Over time, their effectiveness can diminish due to concept drift or changes in data distributions. To preserve their performance, models need regular updates, which can be expensive. For example, updating large-scale models demands considerable computational resources like high-performance GPUs and extensive energy. Additionally, acquiring and preparing high-quality training data is labor-intensive and costly. Therefore, it is essential to balance maintaining model performance with the costs of these updates.

A viable approach to this challenge is to strategically determine update timings. Previous research has focused on either purely offline analysis [1] or heavily on historical data [2] for predicting future conditions, which may not always be applicable in practice. Additionally, the highly uncertain and dynamic nature of model staleness adds complexity to system design and analysis. It necessitates the optimization of update timings in an online manner, without prior knowledge of the uncertain model staleness.

In this paper, we demonstrate our online approach to determining the learning model update timings and balancing the model staleness costs and update costs. We make the following *contributions*. By exploring the offline optimal structure of the problem, we show that the online problem is essentially learning the optimal inter-update time. We frame the problem as a novel variant of the multi-armed bandit (MAB) problem, where each arm represents an inter-update time with associated update cost and accumulated staleness cost. We propose an online strategy that achieves an improved sublinear regret compared with directly applying standard approaches for MAB. We conduct simulations based on real-world traces to evaluate the empirical performance. Please refer to [3] for detailed results.

Q. Lin, L. Wang, J. Su and M. Chen are with Department of Data Science, City University of Hong Kong, Hong Kong. E-mail: {qiulin.lin@, liyuawang9-c@my., junyan.su@my., minghua.chen@}cityu.edu.hk.

II. Model and Problem Formulation

We consider a slotted time horizon with T slots. The learning model is implemented at the beginning of the first slot. We denote the age of a learning model as a_t at slot t to represent the number of elapsed slots since the previous update, starting from one. That is,

$$a_t = \begin{cases} 1, & \text{if } t = 1 \text{ or the model is updated at } t; \\ a_{t-1} + 1, & \text{otherwise.} \end{cases} \tag{1}$$

We assume that there is a minimum update requirement that the model can not stay un-updated for more than K slots, i.e., $a_t \leq K, \forall t$. We consider the model staleness cost as the operation error of applying the learning model, e.g., false recognition in image classification. The operation error can be influenced by various factors such as training data quality, model bias, and dynamic inference requests, leading to uncertain model staleness cost. We address this by considering random staleness cost. We assume that the model staleness costs at different slots are independent and their distributions only dependent on the model's age.[1] We denote the random model staleness cost at slot t as $g_t(a_t)$ and its expectation as $\tilde{g}(a_t)$. We assume that $g_t(a_t)$ is upper bounded by a constant ($\triangleq M$) and $\tilde{g}(a_t)$ is non-decreasing in a_t.

We consider the model update cost as a function of model age a, denoted as $C(a)$, which is non-decreasing in a and is upper bounded by a constant ($\triangleq C$). We note that we consider a model update at the beginning of a slot. If we update at slot t, a model with age a_{t-1} is updated, which incurs cost $C(a_{t-1})$.

We consider the following problem formulation.

$$\min \quad \sum_{t=1}^{T} \left(\tilde{g}(a_t) + C(a_{t-1}) \cdot x_t\right) \tag{2}$$

$$\text{s.t.} \quad a_t = \begin{cases} 1, & t = 1 \text{ or } x_t = 1, \\ a_{t-1} + 1, & t > 1 \text{ and } x_t = 0, \end{cases} \tag{3}$$

$$\text{var.} \quad x_t \in \{0, 1\}, a_t \leq K, \forall t \in [T]; x_1 = 0. \tag{4}$$

We use $x_t \in \{0, 1\}$ to indicate the update decisions at slot t. Our goal is to minimize the expected model staleness cost and

[1] We use real-world traces in our simulations to evaluate the performance of our algorithm without this assumption.

update cost by optimizing update decisions. Our problem is closely related to the single-source time-varying cost of AoI study in [4]; please refer to [3] for a detailed discussion.

III. Our Approach and Results

Our first finding is regarding the offline optimal solution for minimizing the asymptotic average of the total cost in (2).

Theorem 1. *The optimal solution is to update the learning model periodically with the optimal period k^*:*

$$k^* = \arg\min_{k\in[K]} \frac{\sum_{j=1}^{k} \tilde{g}(j) + C(k)}{k}. \tag{5}$$

The optimal average cost is given by $[\sum_{j=1}^{k^*} \tilde{g}(j)+C(k^*)]/k^*$.

Under the online setting, we do not have access to the distribution or the expected value of $g_t(a_t)$. Instead, we need to learn them while making decisions with the observed samples.

Our second finding is to link our problem to the multi-armed bandit paradigm with a special feedback structure. We interpret the inter-update time $k \in [K]$ as an arm k with a normalized mean cost between $[0, 1]$,

$$\mu_k \triangleq \frac{\sum_{j=1}^{k} \tilde{g}(j) + C(k)}{k} \frac{1}{C+M}. \tag{6}$$

Our goal is to achieve a low regret compared with always pulling the optimal arm k^*. We call the problem *timing* multi-armed bandit to emphasize the decisions on the update timing and its specific feedback structures. That is, we determine an inter-update time k after the last update, i.e., pull arm k. In the following k slots, we receive feedback on the staleness cost of a learning model with age from one to k (instead of arm k only), and it incurs a cost of $(C + M) \cdot k \cdot \mu_k$. We make the next arm-pulling decision afterward. We show that the expected regret of an algorithm can be evaluated as follows, where $n_k(T)$ is the expected number of selecting arm k of the algorithm.

$$\tilde{R}(T) \triangleq (C + M) \cdot \left[\sum_{k=1}^{K} k \cdot \mu_k \cdot n_k(T) - T \cdot \mu_{k^*}\right]. \tag{7}$$

Our third finding is regarding a lower bound on directly applying the standard multi-armed bandit approach to our timing bandit problem. Specifically, it treats each arm k independently with mean cost μ_k. Meanwhile, each pulling of arm k will occupy k slots, suffer k times of its cost and receive one feedback of its cost. This makes it harder than the standard MAB where each pulling takes one slot and only suffers one unit of cost to get one feedback. We show that if we directly apply the standard algorithms (more generally, all online algorithms that ignore the additional structures of the problems) will suffer a regret of at least $\Omega(K\sqrt{T}/\log K)$ in the worst case, which is larger than the lower bound $\Omega(\sqrt{KT})$ in the standard MAB problem.

Our final finding is an algorithm that achieves an improved $\tilde{O}(\sqrt{KT})$-regret (ignoring the logarithmic factor in K and T). Our algorithm adapts the arm elimination approach for standard multi-armed bandit problem with the feedback structure of our problem. We maintain a candidate set $\mathcal{A}$ of arms. After making the last update, we check the empirical average cost of the arms in $\mathcal{A}$ and eliminate those that cost significantly larger than the minimum one we observed. We pull the arm with the *maximum index* in $\mathcal{A}$. According to the feedback structure we identify in our second finding, such a design allows us to receive staleness cost feedback of all model ages within the maximum index and update the cost estimations of all arms in $\mathcal{A}$. It efficiently utilizes the feedback structure in that we pull one arm but could update the estimations for all arms in $\mathcal{A}$. In comparison, the standard arm elimination solution requires pulling each arm in $\mathcal{A}$ once to update their estimations. We summarize our result on the regret analysis of the algorithm.

Theorem 2. *The expected regret of our variant of arm elimination algorithm is upper bounded by $\tilde{O}(\sqrt{KT})$.*

IV. Simulation

We consider a recommendation system for recommending business entities to customers. We use a convolutional neural network to build and update the recommendation system using the Yelp dataset[2]. At each update, we will incorporate all the data observed since the previous update to retrain the recommendation system. At each slot, we randomly pick a group of customers from the dataset and apply the current recommendation system to recommend suitable business entities to them. We define the model degradation cost $g(a_t)$ to be the portion of unsuccessful recommendation. We consider that the update cost $C(a) = s \cdot a + b$, where s and b are parameters. We compare our proposed algorithm with two existing baselines, 1) CARA proposed in [2] that updates the model whenever the cumulative degradation cost exceeds a pre-defined threshold, and 2) CSA proposed in [1] that updates the model whenever the improvement in learning performance over the update cost is larger than a pre-defined threshold. We show the performance under varying lengths of the horizon and the maximum update interval in Fig. 1. It shows that our proposed algorithm consistently outperforms alternatives, achieving a performance improvement of at least 18.4%.

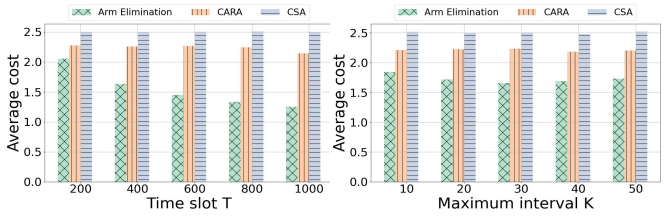

Fig. 1. Performance comparison under varying T and K.

References

[1] I. Žliobaitė, M. Budka, and F. Stahl, "Towards cost-sensitive adaptation: When is it worth updating your predictive model?" *Neurocomputing*, vol. 150, pp. 240–249, 2015.

[2] A. Mahadevan and M. Mathioudakis, "Cost-aware retraining for machine learning," *Knowledge-Based Systems*, vol. 293, p. 111610, 2024.

[3] Q. Lin, L. Wang, J. Su, and M. Chen, "Optimizing freshness of machine learning model: A timing (multi-armed) bandit approach," *Technical Report*, 2025, https://lin-qiulin.github.io/TB.TR.pdf.

[4] V. Tripathi and E. Modiano, "An online learning approach to optimizing time-varying costs of aoi," in *Proceedings of ACM Mobihoc '21*, 2021, p. 241–250.

[2]Yelp open dataset," https://www.yelp.com/dataset, Yelp Inc., 2024.