Rewards and errors in multi-arm bandits for interactive education

Anonymous Author 1
Unknown Institution 1

Anonymous Author 2
Unknown Institution 2

Anonymous Author 3
Unknown Institution 3

1 Introduction

We consider sequential, interactive systems for educational platforms aiming at optimizing an objective function with initially unknown parameters. Specifically, we want to estimate the outcome of different learning options, while *at the same time* provide an educational benefit for users.

This setting fits the bandit framework, where the algorithm has access to a set of *arms* (options) with unknown expected *rewards*. The most common objective in bandits is to minimize the regret, i.e., the difference between the highest reward and the reward of the arms pulled by the algorithm. Since the arms are unknown in advance, this requires balancing *exploration* of the arms and *exploitation* of the mean estimates. An alternative setting is *active exploration* [Antos et al., 2010, Carpentier et al., 2011], where the algorithm's performance is only evaluated upon the termination in terms of accuracy in estimating the value of all arms.

The *two objectives* have been studied separately. However, consider the increasingly-prevalent situation where users participate in research studies (e.g., for education) that are designed to collect reliable data for an accurate assessment of the options (arms). In such situations, the users themselves rarely care about the underlying research questions, but may reasonably wish to gain their own benefit, such as students seeking to learn new material. In order to serve them and gather generalizable knowledge at the same time, we formalize this situation as a *multi-objective* bandit problem, where a designer seeks to trade off cumulative regret minimization (providing good direct reward for participants), with informing scientific knowledge about the strengths and limitations of the various conditions (active exploration to estimate all arm means). This trade-off is especially needed in education when running online experiments, where poor experience makes users leave the platform permanently. While in general, it may not be possible to be simultaneously optimal for both active exploration and reward maximization [Bubeck et al., 2009], we study trade-offs between these two objectives, as Liu et al. [2014] did in the context of education.

2 Learning problem

Objective function. We consider K arms with distributions $\{\nu_i\}_{i=1}^K$ characterized by mean μ_i and variance σ_i^2. Given a sequence of n arms $\mathcal{I}_n = (I_1, I_2, .., I_n)$, where $I_t \in [K]$ is the arm pulled at time t, average reward and estimation errors are defined as

$$\rho(\mathcal{I}_n) = \mathbb{E}\left[\frac{1}{n}\sum_{t=1}^{n} X_{I_t,T_{I_t,t}}\right] = \frac{1}{n}\sum_{i=1}^{K} T_{i,n}\mu_i, \qquad (1)$$

$$\varepsilon(\mathcal{I}_n) = \frac{n}{K}\sum_{i=1}^{K} \mathbb{E}\left[\left(\widehat{\mu}_{i,n} - \mu_i\right)^2\right] = \frac{n}{K}\sum_{i=1}^{K} \frac{\sigma_i^2}{T_{i,n}}, \qquad (2)$$

where $T_{i,t} = \sum_{s=1}^{t-1} \mathbb{I}\{I_s = i\}$ is the number of times arm i is selected up to step $t-1$ and $\widehat{\mu}_{i,n}$ is the empirical average of the $T_{i,n}$ samples. Notice that we *normalize* ε by multiplying the estimation errors by n to make their magnitude (as a function of n) comparable to $\rho(\mathcal{I}_n)$.

We define a trade-off objective function obtained by a continuous relaxation of the two functions above as:

$$f_w(\boldsymbol{\lambda}; \{\nu_i\}_i) = w\sum_{i=1}^{K} \lambda_i\mu_i - \frac{(1-w)}{K}\sum_{i=1}^{K} \frac{\sigma_i^2}{\lambda_i}, \qquad (3)$$

where $w \in [0,1]$ and $\boldsymbol{\lambda} \in \mathcal{D}_K$ belongs to the simplex to define an allocation of arms. Notice that for $w = 1$ we recover the reward maximization problem, while for $w = 0$ the problem reduces to minimizing the average estimation error. The function f_w can be also obtained as a Lagrangian relaxation of a constrained optimization problem where we intend, e.g., to maximize the reward subject to a desired level of estimation accuracy. We define the optimal allocation and its performance as $\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}\in\mathcal{D}_K} f_w(\boldsymbol{\lambda}; \{\nu_i\}_i)$ and $f^* = f_w(\boldsymbol{\lambda}^*; \{\nu_i\}_i)$ respectively. Intuitively, $\boldsymbol{\lambda}^*$ favors arms with large means and large variance since allocating a large portion of the resources to them contribute to minimizing f_w by increasing the reward ρ and reducing the error ε. Finally, we define the performance measure of a learning algorithm. Let $\widetilde{\boldsymbol{\lambda}}_n$ be the empirical frequency of pulls ($\widetilde{\lambda}_{i,n} = T_{i,n}/n$), we define the regret to the optimal allocation as $R_n(\widetilde{\boldsymbol{\lambda}}_n) = f^* - f_w(\widetilde{\boldsymbol{\lambda}}_n; \{\nu_i\}_i)$.

Input: forcing parameter η, weight w
for $t = 1, \ldots, n$ **do**
  if $\min T_{i,t} < \eta\sqrt{t}$ **then**
    Select arm $I_t = \arg\min T_{i,t}$ (**forcing**)
  else
    Compute optimal estimated allocation
$$\widehat{\boldsymbol{\lambda}}_t = \arg\max_{\boldsymbol{\lambda}\in\mathcal{D}_K} f_w(\boldsymbol{\lambda}; \{\widehat{\nu}_{i,t}\}_i)$$
    Select arm (**tracking**)
$$I_t = \arg\max_{i=1,\ldots,K} \widehat{\lambda}_{i,t} - \widetilde{\lambda}_{i,t}$$
  end if
  Pull arm I_t, observe $X_{I_t,t}$, update $\widehat{\nu}_{I_t}$.
end for

Figure 1: The FORCING algorithm.

Learning algorithm. We introduce the FORCING algorithm (Fig. 1), inspired by the GAFS-MAX algorithm [Antos et al., 2010]. At each step t, the algorithm first checks the number of pulls of each arm and selects any arm with less than $\eta\sqrt{t}$ samples. If all arms have been sufficiently pulled, the allocation $\widehat{\boldsymbol{\lambda}}_t$ is computed using the empirical estimates of the arms' means and variances $\widehat{\mu}_{i,t} = \frac{1}{T_{i,t}} \sum_{s=1}^{T_{i,t}} X_{i,s}$ and $\widehat{\sigma}^2_{i,n} = \frac{1}{T_{i,t}-1} \sum_{s=1}^{T_{i,t}} \left(X_{i,s} - \widehat{\mu}_{i,t}\right)^2$. Once the allocation $\widehat{\boldsymbol{\lambda}}_t$ is computed, an arm is selected. FORCING explicitly tracks the allocation $\widehat{\boldsymbol{\lambda}}_n$ by selecting the arm I_t that is under-pulled the most so far. This tracking step allows us to force $\widetilde{\boldsymbol{\lambda}}_n$ to stay close to $\widehat{\boldsymbol{\lambda}}_n$ (and its performance) at each step. The parameter η defines the amount of exploration forced by the algorithm. A large η forces all arms to be pulled many times. While this guarantees accurate estimates $\widehat{\mu}_{i,t}$ and $\widehat{\sigma}^2_{i,n}$ and an optimal estimated allocation $\widehat{\boldsymbol{\lambda}}_t$ that rapidly converges to $\boldsymbol{\lambda}^*$, the algorithm would perform the tracking step very rarely and thus $\widetilde{\boldsymbol{\lambda}}_t$ would not track $\widehat{\boldsymbol{\lambda}}_t$ fast enough.

3 Evaluation on educational data

Treefrog Treasure is an educational math game in which players navigate through a world of number lines. The players must find and jump through the appropriate fraction on each number line. To analyze the effectiveness of our algorithm when parameters are drawn from a real-world setting, we use data from an experiment in *Treefrog Treasure* to estimate the means and variances of a 64-arm experiment. Each arm corresponds to a different experimental condition: after a tutorial, 34,197 players each received a pair of number lines with different properties, followed by the same (randomized) distribution of number lines thereafter. We measured how many number lines students solved conditioned on the type of this initial pair; the hope is to learn which type of number line encourages player persistence on a wide variety of number lines afterwards. There were a total of $K = 64$ conditions, formed from choosing between 2 representations of the target fraction, 2 representations of the label fractions on the lines themselves, adding or withholding tick marks at regular intervals on the number line, adding or removing hinting animations if the problem was answered incorrectly, and 1-4 different rates of backoff hints that would progressively offer more and more detailed hints as the player made mistakes. The details of both the experiments and the experimental conditions are taken from Liu et al. [2014], though we emphasize that *we measure a different outcome*, i.e., the player persistence as opposed to the chance of a correct answer.

	$\frac{\rho(\boldsymbol{\lambda})}{\mu_{\max}}$	$\sqrt{\frac{\varepsilon(\boldsymbol{\lambda})}{\sigma^2_{\max}}}$	R_n	*RankErr*
UCB	**0.930**	6.220	0.070	5.424
GAFS	0.919	**5.891**	0.027	**5.151**
FORCE	0.922	5.912	**0.005**	5.291
Unif	0.916	5.880	0.055	-
$\boldsymbol{\lambda}^*$	0.922	5.904	-	-

Figure 2: Results on the educational data.

We run FORCING, UCB [Auer et al., 2002], GAFS-MAX over $n = 25,000$. Both FORCING and GAFS-MAX use $\eta = 1$ and w is set to 0.95 to give priority to the player's experience and entertainment. We study the performance in terms of average reward $\rho(\boldsymbol{\lambda})$, estimation error $\varepsilon(\boldsymbol{\lambda})$, regret R_n, and *RankErr* that measures how well arms are ranked on the basis of their mean.[1] Small values of *RankErr* mean that arms are estimated well enough to correctly rank them and can allow the experiment designer to later reliably remove the worst performing arms. The results are reported in Fig. 2. UCB achieves the highest reward but performs very poorly in estimating the arms' mean and in ranking them. GAFS-MAX does not collect much reward but is very accurate in estimating the means. On the other hand, FORCING balances the two objectives and it achieves the smallest regret, thus preserving a very good estimation accuracy without compromising too much the average reward. Here, effectively balancing between the two objectives allows us to rank the different game settings in the right order while providing players with a good experience. Had we used UCB, the outcome for players would have been better, but the designers would have less insight into how the different ways of providing number lines affect player behavior for when they need to design the next game (high *RankErr*). Alternatively, using GAFS-MAX would give the designer excellent insight into how different number lines affect players; however, if some conditions are too difficult, we might have caused many players to quit. FORCING provides a useful feedback to the designer without compromising the players' experience (the *RankErr* is close to GAFS-MAX but the reward is higher).

[1] Let π^* be the true ranking and $\widehat{\pi}$ the estimated ranking, then $RankErr = 1/K \sum_{i=1}^{K} |\pi^*(i) - \widehat{\pi}(i)|$.

References

András Antos, Varun Grover, and Csaba Szepesvári. Active learning in heteroscedastic noise. *Theoretical Computer Science*, 411:2712–2728, June 2010.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory (ALT'09)*, 2009.

Alexandra Carpentier, Alessandro Lazaric, Mohammad Ghavamzadeh, Rémi Munos, and Peter Auer. Upper-confidence-bound algorithms for active learning in multi-armed bandits. In *Proceedings of the 22nd International Conference on Algorithmic Learning Theory (ALT'11)*, pages 189–203, 2011.

Yun-En Liu, Travis Mandel, Emma Brunskill, and Zoran Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In *Proceedings of the 7th International Conference on Educational Data Mining (EDM)*, 2014.