Functional Multi-armed Bandit and the Best Function Identification Problems

Yuriy Dorn
AI Center & IAI MSU
Lomonosov Moscow State University
Moscow, Russia
dornyv@my.msu.ru

Aleksandr Katrutsa
AI Center
Lomonosov Moscow State University
Moscow, Russia
amkatrutsa@gmail.com

Ilgam Latypov
AI Center & IAI MSU
Lomonosov Moscow State University
Moscow, Russia
i.latypov@iai.msu.ru

Anastasia Soboleva
AI Center & IAI MSU
Lomonosov Moscow State University
Moscow, Russia
a.soboleva@iai.msu.ru

ABSTRACT

We consider the model selection problem, where we have a set of candidate parametric functions and need to identify the function with the smallest minimum and corresponding minimizer. This problem arises in the competitive training of neural networks, where a set of candidates is given, and the limited computational budget prevents the use of a brute-force search. To address this problem, we propose generalizations of the classical multi-armed bandit (MAB) and best arm identification (BAI) setups, since using classical MAB and BAI setups leads to infeasible computational costs. We refer to the proposed setups as the *functional* multi-armed bandit problem (FMAB) and the best *function* identification (BFI) problems, respectively. For these problems, we establish lower regret bounds for different classes of candidate functions. To solve FMAB and BFI problems, we propose a novel reduction scheme to construct the F-LCB algorithm, which is a UCB-type algorithm based on basic algorithms for nonlinear optimization with known convergence rates. The F-LCB algorithm combines the arm selection step and the update of the current optimum approximation. We provide regret upper bounds for F-LCB based on the known convergence rates of the underlying base algorithms. The regret upper bounds match with the derived lower bounds up to the logarithmic factor. Numerical experiments confirm that the proposed approach correctly identifies the optimal function and provides the minimizer for it in both smooth and non-smooth convex cases. Similarly, F-LCB converges faster than SuccessiveHalving and Hyperband algorithms for the model selection problem, where the candidate functions are neural networks and only a stochastic gradient estimate is available.

KEYWORDS

Multi-armed bandits; functional multi-armed bandits; best function identification

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/SFMN9947

ACM Reference Format:
Yuriy Dorn, Aleksandr Katrutsa, Ilgam Latypov, and Anastasia Soboleva. 2026. Functional Multi-armed Bandit and the Best Function Identification Problems. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/SFMN9947

1 INTRODUCTION

The stochastic MAB problem could be defined as follows: an agent chooses arm A_t at each time step $t = 1, \ldots, T$ from the given set of arms $S = \{a_1, \ldots, a_k\}$ and observes loss $l_t(A_t)$. The agent can observe losses only for the chosen action at each step. This is referred to as *bandit feedback*. For each arm a, the loss distribution $\mathcal{D}_a$ with expectation $\mathbb{E}_{x\sim\mathcal{D}_a}[x] = \mu(a)$ is fixed but unknown to the agent. At each round t, loss $l_t(A_t)$ is sampled from distribution $\mathcal{D}_{A_t}$ independently after arm A_t is chosen. The agent's goal is to construct a learning algorithm that minimizes expected regret

$$\mathbb{E}[R(T)] = \sum_{t=1}^{T} \left[\mu(A_t) - \mu^*\right], \tag{1}$$

where $\mu^* = \arg\min_{a\in S} \mu(a)$. Surprisingly, there seem to be no works that properly generalize the multi-armed bandit setup to functions, where one models an unknown function as an arm rather than a random variable. This setup is appropriate for black-box optimization with multiple objectives involved. For example, when developing an AI service, the appropriate model architecture and hyperparameter settings must be selected and optimized. The main challenge is that the optimal architecture or hyperparameter setting is unknown in advance. Thus, as in the MAB problem, one must explore different models and setups. However, exploration comes with costs that are negligible for small models but can be astronomically high for large-scale models like modern LLMs. This challenge motivates modifications to the standard MAB setup, presented in Section 2. We propose the functional multi-armed bandit and best function identification problems, where each arm corresponds to an unknown function and is equipped with a black-box oracle.

The main contributions of our study are the following:

- We propose the Functional Multi-armed Bandit (FMAB) and Best Function Identification (BFI) setups, which are appropriate for the model selection problem.

- We establish regret lower bounds for the introduced setups and provide particular forms for the different classes of candidate functions.
- We develop the novel F-LCB algorithm for FMAB and BFI problems and prove the regret upper bounds for general FMAB and deterministic BFI setups.
- F-LCB outperforms the SuccessiveHalving and Hyperband baselines in competitive neural networks training.

Related works. The multi-armed bandit (MAB) problem has a rich history dating back to the seminal works of Thompson [39] and Robbins [30]. Over decades, an enormous body of literature has accumulated, with various aspects of the problem covered in several foundational textbooks [5, 8, 23, 35]. While there have been numerous attempts to generalize the MAB framework—particularly for hyperparameter optimization and complex decision spaces—most existing models primarily modify the feedback structure while maintaining the core correspondence between arms and random variables. For example, $\mathcal{X}$-bandits [6, 7] generalize the MAB problem on an arbitrary measurable space of arms. In functional bandits [40], the agent plays arm i, the random variable X_i is sampled, say x_i^t, and the value $f(x_i^t)$ is observed. Similarly, contextual bandits [9, 34, 44] and Lipschitz bandits [1, 20] assume that arms represent unknown distributions or functions of observable contexts.

In the domain of AutoML, several frameworks have gained prominence. Bayesian Optimization (BO) [33, 36] models the objective as a single black-box function, typically using Gaussian Processes. However, BO relies on shared structure across the parameter space and notoriously struggles in high-dimensional settings. In contrast, our Functional MAB (FMAB) framework assumes that arms represent separate functions, potentially from high-dimensional spaces, without requiring a shared global surrogate. While hybrid methods like BOHB [12] combine BO with Successive Halving, they inherit these limitations when the underlying function landscapes are unrelated. Another influential approach is ASHA [25], an asynchronous variant of Hyperband that utilizes relative rankings for early stopping. While computationally efficient, ASHA operates as a heuristic ranking mechanism and does not exploit the specific mathematical properties of the convergence trajectories.

A distinct direction is represented by Population-Based Training (PBT) [16] and its derivatives like PB2 [29]. PBT evolves a population of models by directly manipulating their parameters through explore/exploit operations and weight transfers (warm-starting). Our FMAB framework differs fundamentally as it allocates computation across independent optimization tasks without relying on weight sharing. Crucially, BO and PBT treat model selection as a purely black-box problem, ignoring the convergence guarantees of the underlying optimization routines, while our approach assumes that each arm corresponds to an optimization task with known convergence properties. This enables us to derive formal regret guarantees and identification bounds that are typically unavailable for purely heuristic AutoML methods. To the best of our knowledge, this is the first tractable and theoretically grounded setup that generalizes the concept of arms to functions by leveraging the internal dynamics of the optimization process.

2 PROBLEM STATEMENT

This section presents the functional modifications to the MAB problem and to the best-arm identification problems, respectively. We denote these modifications as FMAB and BFI, formally introduce them, derive the lower bounds for the deterministic setting, and discuss an application that best fits the FMAB problem statement.

2.1 Functional multi-armed bandit problem (FMAB)

Given convex objective functions $f_1 : \mathbb{R}^{n_1} \to \mathbb{R}, \dots, f_K : \mathbb{R}^{n_K} \to \mathbb{R}$ and convex decision sets $\mathcal{X}_1, \dots, \mathcal{X}_K$ at each $t \in [0, T]$ round, the agent chooses index $i_t \in \{1, \dots, K\}$ and the decision vector $x^{t,i_t} \in \mathcal{X}_{i_t} \subseteq \mathbb{R}^{n_{i_t}}$; the agent receives oracle feedback $O_{i_t}(x^{t,i_t})$, e.g., gradient $f'_{i_t}(x^{t,i_t})$ in the case of the first-order oracle (more details about oracle concept is presented in Section A in Appendix of [11]). The regret is defined as:

$$R_O(T) = \sum_{t=1}^{T} \left[f_{i_t}(x^{t,i_t}) - f^* \right], \tag{2}$$

where $f^* = \min\limits_{1 \le i \le K} \min\limits_{x \in \mathcal{X}_i} f_i(x)$. The agent aims to minimize regret R_O through specific rules for selection index i_t and decision vector x^{t,i_t}.

The interpretation of optimized functions as arms appears in application-related works, such as [45, 46]. The goal is to optimize multiple functions simultaneously with a limited compute budget. Model selection problem [24] is a particular case of this setting. The discussion of relations of the introduced functional version of MAB setup and the classic MAB is prtesented in Section 2.3.

2.2 Best function identification problem (BFI)

Given convex objective functions $f_1 : \mathbb{R}^{n_1} \to \mathbb{R}, \dots, f_K : \mathbb{R}^{n_K} \to \mathbb{R}$ and convex decision sets $\mathcal{X}_1, \dots, \mathcal{X}_K$ at each $t \in [0, T]$ round, the agent chooses index $i_t \in \{1, \dots, K\}$ and the decision vector $x^{t,i_t} \in \mathcal{X}_{i_t} \subseteq \mathbb{R}^{n_{i_t}}$; the agent observes the loss $f_{i_t}(x^{t,i_t})$. We assume that the agent has access to the oracles $O_i(x)$ for each objective function f_i and the oracle is the only source of information provided for each subproblem $\mathcal{P}_i$ defined by f_i and $\mathcal{X}_i$ (i.e., we use the black-box assumption). At the end of T rounds, the agent selects an arm, denoted by J_T, and aims to minimize the regret R_B defined as:

$$R_B(T) = \min_{x \in \mathcal{X}_{J_T}} f_{J_T}(x) - f^*, \tag{3}$$

where $f^* = \min\limits_{1 \le i \le K} \min\limits_{x \in \mathcal{X}_i} f_i(x)$. We call this problem to as best function identification problem (BFI), which is is an analog for the well-known best arm identification problem [2].

2.3 Relation between MAB and FMAB settings

Let us recap the stochastic MAB problem setup with time horizon T and K arms, each arm i is equipped with an unknown distribution $\mathcal{D}_i$ with an expected value $\mu_i > 0$. At each step t the agent chooses arm i_t and observes reward $\mu_{i_t} + \xi_t$ sampled from $\mathcal{D}_{i_t}$, where the noise ξ_t, is unbiased $\mathbb{E}[\xi_t] = 0$ by construction. The regret is defined by (1). Let us show how FMAB models this setup.

MAB as FMAB with zero-order oracle. Consider FMAB setting with $f_i(x) = -\mu_i$ equipped by the zero-order oracle $O_i(x_t)$ that

provides noised observations $O_i(x_t) = -\mu_i - \xi_t = f_i(x_t) - \xi_t$, sampled from $\mathcal{D}_i$ with "minus", with $\mathbb{E}[O_i(x_t)] = f_i(x_t)$. Then regret (2) is exactly the same as in (1). This direct reduction is formally correct. Let us further consider a reduction scheme that utilizes first-order oracles in FMAB.

MAB as FMAB with first-order oracle. Consider FMAB setting with $f_i(x) = \frac{1}{2}(x-\mu_i)^2 - \frac{\mu_i^2}{2}$. Here x estimates the expected value μ_i. Note that this setting has a few nice properties:

- Optimal objective values represent arms: $f_i^* = -\frac{\mu_i^2}{2}$,
- Objective value $f_i(x_t) = \frac{x_t^2}{2} - x_t\mu_i$ can be estimated via tractable $\hat{f}_i(x_t) = \frac{x_t^2}{2} - x_t(\mu_i + \xi_t)$, where $\mu_i + \xi_t$ is a sampled from reward distribution $\mathcal{D}_i$. Such estimation is unbiased $\mathbb{E}[\hat{f}_i(x)] = f_i(x)$.
- We can observe $O_i(x_t) = x_t - \mu_i - \xi_t$, which is unbiased gradient estimation $\mathbb{E}[O_i(x_t)] = x_t - \mu_i = f_i'(x)$.

In this approach, regret R_O (2) does not explicitly represent regret for MAB setup (1).

Reduction of FMAB to MAB. We have shown how MAB can be treated as particular cases of FMAB. However, the natural question of whether FMAB can handle cases which are intractable by MAB is still actual. By definition of FMAB, it could be represented as MAB with a continuous number of arms indexed by pair (i, x), where $i \in \{1, \dots, K\}$ and $x \in \mathbb{R}^{n_i}$ or approximated by MAB with an infinite number of arms via discretisation for the continuous space of x. Both cases require additional structural assumptions on arms to become tractable (see [43] as an example). If these assumptions are in place, then, just like in Bayesian optimization (BO) algorithms, sampled arms make decisions based on feedback. This is not avoidable in general, but in FMAB we assume that each arm i is equipped with a learning algorithm $\mathcal{A}_i$, that could optimize the corresponding function $f_i(x)$. These assumptions are natural for many applications (see the next subsection as an example) and are new and not considered in the general MAB setting. It allows, in general, to learn much faster compared to classical MAB or BO algorithms. So FMAB could be solved as MAB, but it is much better to solve it as FMAB to avoid sub-exponential costs.

2.4 Applications

Competitive neural network training. Modern neural networks are very costly to train [14]. Therefore, the standard trial-and-run approach is very inefficient. Within our framework, sequential training of all candidate models could be avoided and the most accurate model is identified automatically. Assume that there are k candidate models. Each model $i \in \{1, \dots, k\}$ is denoted by the number of parameters n_i, feasible decision set $\mathcal{X}_i \subseteq \mathbb{R}^{n_i}$ and domain-specific quality metric w.r.t. training cost $f_i : \mathcal{X}_i \to \mathbb{R}$. Then, regret R_O represents the sum of training costs for the optimal model and costs for experiments for other models.

Optimization method selection. Modern large-scale optimization tasks often present a dilemma: the most suitable algorithm for a given problem is unknown a priori. While theoretical convergence rates are well established, practical performance depends heavily on the structural properties of the data, such as sparsity, condition number, and feature space dimensionality. In this regime, researchers face trade-offs among model and solver classes. For instance, in high-dimensional settings, one must often choose between an ℓ_1-regularized model solved via FISTA [4] and a dense counterpart utilizing variance-reduction techniques such as SVRG [18]. The efficiency of such choices hinges on the underlying signal sparsity, which is typically not observable before training.

A prominent example of such complexity is found in optimal transport, where large-scale problems admit multiple formulations and a wide array of solvers [41]. The choice among primal-dual accelerated methods, Sinkhorn-based iterations, and stochastic approaches often depends on the desired precision and the strength of regularization.

Convex relaxations. Our framework is applicable if a complex original problem admits a convex relaxation. Such approximations are vital for a broad spectrum of tasks, including the weighted maxmin dispersion problem [15], optimal distributed control [13], mixed-integer programs [32], and polynomial optimization [19]. Since these convex relaxations are solved using iterative optimization methods, their performance is governed by analytical convergence bounds, such as $O(1/t^2)$ for accelerated gradient schemes if objective function is L-smooth. Our F-LCB algorithm leverages these properties to evaluate the quality of a relaxation "on the fly", enabling the early identification of the most promising problem formulation and the optimal allocation of computational resources.

2.5 Notation for function classes

We summarize here the main notation and function classes used throughout the paper. For each $i \in \{1, \dots, K\}$, let f_i be the objective function defined on a convex domain $\mathcal{X}_i \subset \mathbb{R}^{n_i}$ and optimized by a base algorithm $\mathcal{A}_i$. We denote $f^* = \min_{1 \le i \le K} \min_{x \in \mathcal{X}_i} f_i(x)$, and let i^* be the corresponding optimal index. The diameter of $\mathcal{X}_i$ is denoted by $R_i = \sup_{x,y \in \mathcal{X}_i} \|x - y\|_2$, and we also define the global diameter $R = \max_{1 \le i \le K} R_i$.

For each convex function f_i, we assume that

$$\frac{\mu_i}{2}\|x-y\|_2^2 \le f_i(y) - f_i(x) - \langle f_i'(x), y-x\rangle \le \frac{L_i}{2}\|x-y\|_2^2 + M_i\|x-y\|_2,$$

where $\mu_i \ge 0$ is the strong convexity constant, $L_i \ge 0$ is the smoothness parameter, and $M_i \ge 0$ is the Lipschitz constant of f_i. Different combinations of (μ_i, L_i, M_i) correspond to different standard classes of convex optimization problems:

- if $M_i > 0$, $\mu_i = 0$, f_i is convex M_i-Lipschitz function
- if $L_i > 0$ and $\mu_i = 0$, f_i is L_i-smooth convex function
- If $\mu_i > 0$, $M_i > 0$, f_i is μ_i-strongly convex and M_i-Lipschitz
- If $\mu_i > 0$, $L_i > 0$, f_i is μ_i-strongly convex and L_i-smooth.

For convenience, we introduce the following global parameters:

$$M = \max_i M_i, \qquad L = \max_i L_i, \qquad \mu = \min_i \mu_i, \qquad \kappa = \max_i \frac{L_i}{\mu_i}.$$

3 LOWER BOUNDS FOR DETERMINISTIC SETUPS

We present minimax lower bounds for FMAB and BFI problems in deterministic settings. The idea is to reduce these problems to standard optimization tasks with known lower bounds [26, 27].

We use $\mathcal{F}$ to denote a class of optimization problems defined by a family of objective functions $\{f_i\}_{i=1}^K$, their corresponding domains $\{\mathcal{X}_i\}_{i=1}^K$, and oracle types. We assume that all problems belong to the same class $\mathcal{F}$, i.e. they share the same structural properties (e.g., convexity, smoothness, strong convexity) and oracle complexity.

Consider a family of optimization problems $\mathcal{P}_i$:

$$\min_{x\in\mathcal{X}_i} f_i(x), \qquad i = 1,\dots,K, \tag{4}$$

where all f_i belong to the same class $\mathcal{F}$ and have the oracles with the same complexity. For such problems, the classical minimax lower bound states that for any algorithm $\mathcal{A}$ and any $t \in \mathbb{N}$ there exists a problem instance such that

$$f(x_t) - \min_{x\in\mathcal{X}} f(x) \;\geq\; g(H,t), \tag{5}$$

where $g(H,t)$ is the complexity function depending on $H = (f, \mathcal{D})$.

For many standard classes, $g(H,t)$ factorizes as

$$g(H,t) \;=\; \phi(H)\, t^{-\alpha}, \tag{6}$$

allowing us to define the class-wide hardness function

$$\underline{g}(t) = \inf_H g(H,t), \tag{7}$$

typically of order $\Omega(t^{-\alpha})$.

Theorem 3.1 (Minimax lower bound for BFI). *For any BFI algorithm and any $T \in \mathbb{N}$, there exists a family $\{P_i\}_{i=1}^K \subset \mathcal{F}$ such that*

$$R_B(T) \;\geq\; \underline{g}^{-1}\left(\frac{T}{K}\right), \tag{8}$$

where $\underline{g}^{-1}$ is the functional inverse of $\underline{g}$.

Theorem 3.2 (Minimax lower bound for FMAB). *For any FMAB algorithm and any $T \in \mathbb{N}$, there exists a family $\{P_i\}_{i=1}^K \subset \mathcal{F}$ such that*

$$R_O(T) \;\geq \inf_{\{k_i\geq 0:\ \sum_{i=1}^K k_i = T\}} \sum_{i=1}^K G(k_i), \tag{9}$$

where

$$G(m) = \sum_{s=1}^{m} \underline{g}(s). \tag{10}$$

For homogeneous problems (all P_i have the same hardness $\underline{g}$), this yields the following orders:

- Convex M-Lipschitz: $\underline{g}(s) = \Omega(MR/\sqrt{s}) \;\Rightarrow\; R_O(T) = \Omega(MR\sqrt{T})$,
- L-smooth convex: $\underline{g}(s) = \Omega(LR^2/s^2) \;\Rightarrow\; R_O(T) = \Omega(LR^2)$,
- μ-strongly convex, M-Lipschitz: $\underline{g}(s) = \Omega(M^2/(\mu s)) \;\Rightarrow\; R_O(T) = \Omega\big(\frac{M^2}{\mu}\log T\big)$,
- μ-strongly convex, L-smooth: $\underline{g}(s) = \Omega(R^2 e^{-s/\sqrt{\kappa}}) \;\Rightarrow\; R_O(T) = \Omega(R^2)$.

These lower bounds match, up to logarithmic factors, the upper bounds we derive later for the proposed F-LCB algorithm.

Remark 1. *Unlike static "explore-first" schedules that are optimal only for indistinguishable worst-case scenarios,* F-LCB *adaptively focuses computational effort on promising arms. It reacts immediately when an arm's LCB becomes suboptimal, avoiding the resource waste inherent in static allocations.*

4 F-LCB ALGORITHM

This section presents the novel F-LCB algorithm to solve the stated FMAB (2) and BFI (3) problems and proof of the corresponding regret rates. However, for the reader's convenience, we first introduce the necessary notations important for further presentation.

Definition 4.1. An algorithm

$$x_{k+1} = \mathcal{A}\left(x_0, O(x_0), \dots, x_k, O(x_k)\right)$$

An algorithm is called a $g(k,\delta)$-bounded algorithm if, for any $k \in \mathbb{N}$ and $\delta > 0$ inequality

$$f(x_k) - f(x^*) \leq g(k,\delta)$$

holds with a probability of at least $1-\delta$. If there exists a function $g(k)$ such that $f(x_k) - f(x^*) \leq g(k)$, we say that the algorithm $\mathcal{A}$ is $g(k)$-bounded.

Function $g(k,\delta)$ (or $g(k)$ in the deterministic case) represents the convergence rate for algorithm $\mathcal{A}$. The notation $g(k)$ is more convenient for deterministic algorithms with exact oracles, while $g(k,\delta)$ is more appropriate for stochastic methods or methods utilizing inexact oracles. Now, we are ready to present our F-LCB algorithm for both FMAB and BFI problems, taking $g(k,\delta)$- or $g(k)$-bounded algorithms as the main ingredient; see Algorithm 1.

Algorithm 1 F-LCB algorithm

Require: number of functions K, $g_i(k,\delta)$-bounded optimization method $\mathcal{A}_i$ for $i = 1,\dots,K$, period T, initial estimates $x_0^{\mathcal{P}_1},\dots,x_0^{\mathcal{P}_K}$, parameter δ ($\delta = 0$ for deterministic setup).

1: Run $\mathcal{A}_i$ for each function i ($i = 1,\dots,K$) to compute $x_1^{\mathcal{P}_i} = \mathcal{A}_i(x_0^{\mathcal{P}_i}, O_{\mathcal{P}_i}(x_0^{\mathcal{P}_i}))$.
2: For each function i ($i = 1,\dots,K$) set $k_i = 1$ and initialize $LCB_i(k_i,\delta) = f_i(x_1^{\mathcal{P}_i}) - g_i(k_i,\delta)$.
3: **for** $t = 1,\dots,T$ **do**
4: Choose function $i_t = \operatorname{argmin}_{1\leq i\leq K} LCB_i(k_i,\delta)$.
5: Compute
$$x_{k_{i_t}+1}^{\mathcal{P}_{i_t}} = \mathcal{A}_{i_t}(x_0^{\mathcal{P}_{i_t}}, O_{\mathcal{P}_{i_t}}(x_0^{\mathcal{P}_{i_t}}),\dots,x_{k_{i_t}}^{\mathcal{P}_{i_t}}, O_{\mathcal{P}_{i_t}}(x_{k_{i_t}}^{\mathcal{P}_{i_t}})).$$
6: Update LCB index of the played function and preserve others:
$$LCB_{i_t}(k_i+1,\delta) = \begin{cases} LCB_i(k_i,\delta), & i \neq i_t, \\ f_{i_t}(x_{k_{i_t}+1}^{\mathcal{P}_{i_t}}) - g_{i_t}(k_{i_t}+1,\delta), & i = i_t. \end{cases}$$
7: **if** $g_{i_t}(k_{i_t+1},\delta) < \frac{\varepsilon}{2}$ **then**
8: return f_{i_t}
9: **end if**
10: Increase iteration counter for the played arm: $k_{i_t} := k_{i_t} + 1$.
11: **end for**

The main idea of Algorithm 1 is to treat base optimization algorithm's convergence rate as confidence intervals to construct the lower confidence bound on the objective value of the chosen arm. So, the overall scheme is as follows: each optimization problem $\mathcal{P}_i$ defined by $(f_i, \mathcal{X}_i)$ equipped with $g_i(k,\delta)$-bounded algorithm $\mathcal{A}_i$, suitable for $\mathcal{P}_i$ problem class. Then, at each time step t, our algorithm chooses the i_t-th arm. Therefore, we run an iteration of $\mathcal{A}_{i_t}$

based on the current optimistic estimation $LCB_i(x_t^{\mathcal{P}_i})$ of the corresponding objectives' optimal values $f_{i_t}^*$.

Remark 2. *If $f_i(x)$, $1 \le i \le K$, is accessed by an inexact oracle O_i, the oracle should be sampled multiple times at each point. The corresponding algorithm $\mathcal{A}_i$ usually controls the number of samples.*

Remark 3. *In* F-LCB*, each iteration simultaneously reduces uncertainty and expected regret. Thus, the LCB index acts as a unified criterion that is both regret-optimal and information-optimal (up to logarithmic factors), allowing the BFI stopping criterion to integrate naturally without distinct exploration phases.*

This is a direct application of ideas introduced in the seminal paper [3] if one uses convergence rates for optimization algorithms instead of concentration rates of statistical estimators. This approach was proposed in [10] for MAB with heavy tails. Note that one could use different base optimization algorithms $\mathcal{A}_i$ for different i. Next, we present the regret rates for the F-LCB algorithm in Table 1, which are proved formally in Sections 4.1.

Table 1: Summary of regrets R_O and R_B rates for BFI and FMAB problems in the deterministic setup. The proofs for these rates are follows from Theorem 4.3. Here, we assume that functions f_i belong to the same class and base optimizers $\mathcal{A}_i$ are the same and equal to those reported in column 2. PGD denotes Projected Gradient Descent, and AGD denotes Accelerated Gradient Descent, and $\kappa = \frac{L}{\mu}$.

Function	Base optimizer	$g(k)$	# iter for $R_B \le \varepsilon$	$R_O(T)$
Convex M-Lipschitz	PGD	$\frac{RM}{\sqrt{k}}$	$\sum_{i=1}^{K}\left\lceil \frac{M_i^2R_i^2}{\max(f_i^*-f^*-\frac{\varepsilon}{2},\frac{\varepsilon}{2})^2}\right\rceil$	$O\left(\sqrt{T\cdot\sum_{i=1}^K M_i^2R_i^2}\right)$
Convex L-smooth	AGD	$\frac{LR^2}{k^2}$	$\sum_{i=1}^{K}\left\lceil \sqrt{\frac{L_iR_i^2}{\max(f_i^*-f^*-\frac{\varepsilon}{2},\frac{\varepsilon}{2})}}\right\rceil$	$O\left(\sum_{i=1}^K L_iR_i^2\right)$
μ-strongly convex M-Lipschitz	PGD	$\frac{M^2}{\mu k}$	$\sum_{i=1}^{K}\left\lceil \frac{M_i^2}{\mu_i\max(f_i^*-f^*-\frac{\varepsilon}{2},\frac{\varepsilon}{2})}\right\rceil$	$O\left(\left(\sum_{i=1}^K \frac{M_i^2}{\mu_i}\right)\log T\right)$
μ-strongly convex L-smooth	AGD	$R^2\exp\{-\frac{k}{\sqrt{\kappa}}\}$	$\sum_{i=1}^{K}\left\lceil \sqrt{\kappa}\log\left(\frac{R_i^2}{\max(f_i^*-f^*-\frac{\varepsilon}{2},\frac{\varepsilon}{2})}\right)\right\rceil$	$O\left(\sum_{i=1}^K \frac{R_i^2}{\exp\left\{\frac{1}{\sqrt{\kappa_i}}\right\}-1}\right)$

4.1 Determenistic case

This section presents the regret bounds for FMAB and BFI problems in terms of the convergence rates $g_i(k)$ for the base optimizers $\mathcal{A}_i$ equipped with the deterministic oracles. After that, the substitution of the particular forms of $g_i(k) = g(k)$ corresponding to the base optimizer (AGD or PGD) leads to the regret bounds from Table 1.

FMAB. This paragraph focuses on the deterministic FMAB problem, which aims to minimize the regret R_O (2). Lemma 4.2 shows how to bound R_O in general for $g_i(k)$-bounded base optimizers. Theorem 4.3 specializes this result to the case where all functions $g_i(k)$ decrease at the same polynomial rate, i.e., scale with a common convergence **r**ate r.

Lemma 4.2. *Assume $\mathcal{A}_i$ $(i = 1,\dots,K)$ be $g_i(k)$-bounded base algorithms. Then for Algorithm 1 for all $\tau \in \overline{1,T}$ holds:*

$$R_O(\tau) \le \sum_{t=1}^{\tau} g_{i_t}(k_{i_t,t}) = \sum_{i=1}^{K}\sum_{k=1}^{k_{i,\tau}} g_i(k), \tag{11}$$

where $k_{i,t}$ is a number of calls for the i-th function by time t.

Proof. Let i_t be the arm selected at time t. Then, its LCB value is the smallest one among the arms. That is, for all j:

$$LCB_{i_t}(k_{i_t,t}) = f_{i_t}(x^{i_t,k_{i_t,t}}) - g_{i_t}(k_{i_t,t}) \le$$
$$\le f_j(x^{j,k_{j,t}}) - g_j(k_{j,t}) \le f_j^*.$$

In particular, this holds w.r.t. the best arm, yielding an estimate of the per-step regret:

$$f_{i_t}(x^{i_t,k_{i_t,t}}) - g_{i_t}(k_{i_t,t}) \le f^* \Rightarrow f_{i_t}(x^{i_t,k_{i_t,t}}) - f^* \overset{(\star)}{\le} g_{i_t}(k_{i_t,t}). \tag{12}$$

Summing up inequality $(\star)$ for $t = 1,\dots,\tau$ we get regret rate:

$$\sum_{t=1}^{\tau} f_{i_t}(x^{i_t,k_{i_t}}) - f^* \overset{(1)}{\le} \sum_{t=1}^{\tau} g_{i_t}(k_{i_t}) \overset{(2)}{=} \sum_{i=1}^{K}\sum_{t=1}^{k_{i,\tau}} g_i(t). \tag{13}$$

Equality (2) is obtained by grouping terms over arms. □

Theorem 4.3. *Let $r > 0$. Assume $\mathcal{A}_i$ $(i = 1,\dots,K)$ are $g_i(k)$-bounded base algorithms with $g_i(k) = \frac{\beta_i}{k^r}$. Then for Algorithm 1, for all $\tau \in \overline{1,T}$ the following regret bounds hold:*

- *if $r \in (0,1)$: $R_O(\tau) \le O\left(\left(\sum_{i=1}^K \beta_i^{\frac{1}{r}}\right)^r \tau^{1-r}\right)$;*
- *if $r = 1$: $R_O(\tau) \le O\left(\sum_{i=1}^K \beta_i \log\tau\right)$;*
- *if $r > 1$: $R_O(\tau) \le O\left(\sum_{i=1}^K \beta_i\right)$.*

Proof idea. The bounds follow from summing the convergence rates $\sum \beta_i k_i^{-r}$ and applying Hölder's inequality subject to the budget constraint $\sum k_i \le \tau$ for each regime of r.

BFI. This paragraph focuses on the deterministic BFI problem, which aims to minimize the regret R_B (3). Theorem 4.4 shows how many steps are required for Algorithm 1 to get R_B smaller than ε from the selected $g_i(k)$-bounded base optimizers.

Theorem 4.4. *Consider a deterministic BFI problem. We denote by $f^* = \min_{1\le i\le k} f_i^*$. To achieve regret $R_B(T) = \min\limits_{x\in\mathcal{D}_{J_T}} f_{J_T}(x) - f^* \le \varepsilon$, Algorithm 1 requires at most*

$$T = 1 + \sum_{i=1}^{k} g_i^{-1}\left(\max\left[f_i^* - f^* - \frac{\varepsilon}{2}, \frac{\varepsilon}{2}\right]\right) \tag{14}$$

iterations, where $g_i^{-1}(\varepsilon) \triangleq \min\{\tau \mid f_i(x_t) - f_i^* \le \varepsilon,\ \forall t \ge \tau\}$.

Proof. Assume without loss of generality that $f^* = f_1^*$. Let $k_{i,t}$ denote the total number of times f_i is updated in Algorithm 1 *(line 6)* by iteration t. Then, since in every iteration of Algorithm 1 only one function is updated, the number of iterations T can be computed as $T = \sum_{i=1}^K k_{i,T}$. Note that from the definition 4.1 it follows that for all $k_{1,t} \ge 1$, we have $f_1(x^{1,t}) - g_1(k_{1,t}) \le f_1^* = f^*$. At step $T - 1 = \sum_{i=1}^K k_{i,T-1}$, there exists a subproblem $\mathcal{P}_j$ such that the corresponding function f_j was updated at least $k_j^* = g_j^{-1}(\max(f_j^* - f^* - \frac{\varepsilon}{2}, \frac{\varepsilon}{2}))$ times, i.e. $k_{j,T-1} \ge k_j^*$, since, otherwise, $\sum_{i=1}^k k_{i,T-1} < T - 1$. Note that if $k_{j,t} \ge k_j^*$, then

$$g_j(k_{j,t}) \le g_j(k_j^*), \tag{15}$$

which means the larger the number of iterations, the smaller the gap between f_j and f_j^*.

Let us show that if Algorithm 1 selects function f_j *(line 6)* after making exactly $k_{j,t} = k_j^*$ corresponding updates, then it reaches the stopping criterion *(lines 7-9)* and returns f_j such that $f_j^* - f^* \le \varepsilon$. According to *lines 4* in Algorithm 1 the following inequality holds:

$$f_j(x^{j,t}) - g_j(k_{j,t}) \le f_1(x^{1,t}) - g_1(k_{1,t}) \le f_1^* = f^*. \quad (16)$$

If $f_j^* - f^* > \varepsilon$, then $\max(f_j^* - f^* - \frac{\varepsilon}{2}, \frac{\varepsilon}{2}) = f_j^* - f^* - \frac{\varepsilon}{2}$ and $g(k_j^*) = f_j^* - f^* - \frac{\varepsilon}{2}$. Taking into account the inequality (15), we have the following inequalities:

$$\begin{aligned} & f_j(x^{j,t}) - g_j(k_{j,t}) \ge f_j(x^{j,t}) - g_j(k_j^*) = \\ & = f_j(x^{j,t}) - (f_j^* - f_1^* - \frac{\varepsilon}{2}) = f_1^* + (f_j(x^{j,t}) - f_j^*) + \frac{\varepsilon}{2} \ge \\ & \ge f_1^* + \frac{\varepsilon}{2}. \end{aligned} \quad (17)$$

Thus, we have a contradiction with inequality (16), and conclude that $f_j^* - f^* \le \varepsilon$. Therefore, $\max(f_j^* - f^* - \frac{\varepsilon}{2}, \frac{\varepsilon}{2}) = \frac{\varepsilon}{2}$, $k_j^* = g_j^{-1}(\frac{\varepsilon}{2})$ and the stopping criterion (*lines 7-9*) holds. So, we demonstrate that $k_{i,t} \le k_i^*, i = 1, \dots, K$. Thus, if the stopping criterion was not reached before iteration $T-1$, then Algorithm 1 made exactly k_i^* updates of function f_i for $i = 1, \dots, K$ by the iteration $T-1$. After that, in the iteration T, Algorithm 1 selects and returns f_j such that $k_{j,T-1} = k_j^*$, and the stopping criterion holds. □

Corollary 4.5 (BFI polynomial). *If for each base algorithm $g_i(k) = \frac{\beta_i}{k^r}$ with $r > 0$, $\beta_i > 0$. Then $g_i^{-1}(\varepsilon) = \min\left\{\tau \mid \frac{\beta_i}{\tau^r} \le \varepsilon\right\} = \left\lceil \left(\frac{\beta_i}{\varepsilon}\right)^{1/r} \right\rceil$, and to achieve $R_B(T) \le \varepsilon$ Algorithm 1 requires at most T iterations, where $T = 1 + \sum_{i=1}^{k} \left\lceil \left(\frac{\beta_i}{\max\left\{f_i^* - f^* - \frac{\varepsilon}{2}, \frac{\varepsilon}{2}\right\}} \right)^{1/r} \right\rceil$.*

4.2 Stochastic case

This section presents bounds for regrets R_O and R_B, if $g_i(k,\delta)$-bounded base optimizers use inexact oracles. After that, we consider particular classes of functions f_i, select the corresponding $g(k,\delta)$-bounded base optimizers, and derive the final regret bounds.

FMAB. To prove the bound for R_O regret, we define a *clean event* as follows: $\mathcal{E}_{\text{clean}} \triangleq \bigcap_{i\in[K], t\in[T]} \left\{f_i(x_{i,t}) - f_i^* \le g(k_{i,t},\delta)\right\}$

The probability of this event can be bounded from below. Indeed, if the concentration inequality $\mathbb{P}\left[f_i(x_{i,t}) - f_i^* \ge g(k_{i,t},\delta)\right] \le \delta$ holds, then applying the union bound, we get: $\mathbb{P}[\mathcal{E}_{\text{clean}}] \ge 1 - TK\delta$.

Conjecture 4.6. *Functions f_i are bounded above, i.e.* $\max_{1\le i\le K} \max_{x_i\in D_i} f_i(x_i) \le A$.

With Assumption 4.6, we get the following regret bound:

$$\mathbb{E}[R_O(T)] \le \mathbb{E}[R_O(T) \mid \mathcal{E}_{\text{clean}}] + KT^2 A\delta. \quad (18)$$

Lemma 4.7. *Assume $\mathcal{A}_i$ be $g_i(k,\delta)$-bounded algorithms for the corresponding problem $\min_{x\in\mathcal{D}_i} f_i(x)$ for each $1 \le i \le K$. Then for Algorithm 1 the following inequality holds:*

$$\mathbb{E}\left[R_O(T)\right] \le \mathbb{E}\left[\sum_{i=1}^{K}\sum_{t=1}^{k_{i,T}} g_i(t,\delta) \middle| \mathcal{E}_{clean}\right] + \delta KT^2 \cdot A, \quad (19)$$

where $A = \max_{1\le i\le K} \max_{x_i\in D_i} f_i(x_i)$.

Proof idea. Expected regret (18) under a complement to a clean event is lower than $\delta KT^2 A$. In the conditions of a clean event, for all realizations of the optimization process, the regret is bounded as in Theorem 4.2. It only remains to add them up according to (11). Here, expectation is over realization in a clean event. □

(The complete proof is presented in Appendix B.1 in [11].

Theorem 4.8. *Let $r > 0$. Assume $\mathcal{A}_i$ $(i = 1, \dots, K)$ are $g_i(k,\delta)$-bounded base algorithms with $g_i(k,\delta) = \frac{\beta_i}{k^r} c(\delta)$. set $\delta = \frac{1}{KT^2A}$. Then for Algorithm 1, for all $\tau \in \overline{1,T}$ the following regret bounds hold:*

- *if $r \in (0,1)$: $R_O(\tau) \le O\left(\left(\sum_{i=1}^K \beta_i^{\frac{1}{r}}\right)^r \tau^{1-r} c(\frac{1}{KT^2A})\right)$;*
- *if $r = 1$: $R_O(\tau) \le O\left(\sum_{i=1}^K \beta_i \log(\tau) c(\frac{1}{KT^2A})\right)$;*
- *if $r > 1$: $R_O(\tau) \le O\left((\sum_{i=1}^K \beta_i) c(\frac{1}{KT^2A})\right)$.*

The multiplier $c(\delta)$ typically grows at most polylogarithmically with respect to $\frac{1}{\delta}$, i.e. $c(\delta) = \text{polylog}\left(\frac{1}{\delta}\right)$, as is standard in high-probability convergence bounds for stochastic first-order methods (see, e.g., [22, 31]). Consequently, δ can be chosen to decay polynomially with T to ensure that the additive term $\delta KT^2 A$ remains negligible compared to the main regret term, while only affecting $c(\delta)$ by a logarithmic factor.

Further, we consider algorithms from [22, 31] with known convergence rates $g(t,\delta)$. These algorithms are developed for unconstrained stochastic optimization problems that typically appear in applications [42]. Logarithmic factors in the complexity bounds are omitted for clarity. Convergence guarantees of considered algorithms rely on the following assumptions.

Conjecture 4.9. *The algorithm $\mathcal{A}_i$ has access to the unbiased stochastic first-order oracle, returning $G_i(x,\xi)$. There exists a set $\mathcal{X}_i \in \mathbb{R}^d$ and values $\sigma \ge 0, \alpha \in (1,2]$ such that for all $x \in \mathcal{X}_i$:* $\mathbb{E}_\xi\left[\|G_i(x,\xi) - \nabla f_i(x)\|^\alpha\right] \le \sigma^\alpha$.

Conjecture 4.10. *For any x we have:* $\mathbb{E}\exp\{\|G_i(x,\xi) - \nabla f_i(x)\|^2/\sigma_i^2\} \le 1$.

The resulting R_O regret bounds for the stochastic setup are summarized in Table 2. The proofs for these bounds are based on Theorem 4.8, available $g(k,\delta)$ in base optimizers.

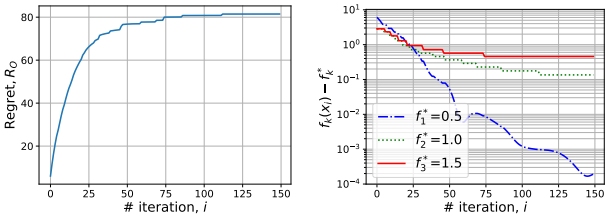
Remark 4. *In the stochastic assumptions, $\alpha \in (1,2]$ characterizes the noise distribution. For the considered base algorithms, the convergence rate parameter r is an explicit function of α: for* clipped-SSTM *(convex problems) $r = 1 - \frac{1}{\alpha}$, and for* R-clipped-SSTM *(strongly convex problems) $r = 2\left(1 - \frac{1}{\alpha}\right)$.*

5 NUMERICAL EXPERIMENTS

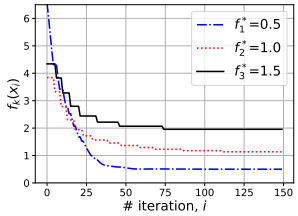
To illustrate the performance of the proposed approach, we consider synthetic test cases for convex smooth and nonsmooth functions. The case of smooth convex functions with inexact first-order oracles is also included in our experimental evaluation. Finally, we consider the CIFAR100 image classification task and use the F-LCB algorithm to automatically identify the best neural network from the given candidate set. We share the source code in the repository at https://github.com/IAIOnline/FMAB to reproduce the presented results.

Table 2: Regret bounds for FMAB problem in the stochastic case. All algorithms require Assumption 4.6. SSTM algorithms [31] require Assumption 4.9 ($\alpha \in (1, 2]$). AGD [22] requires Assumption 4.9 ($\alpha = 2$) and Assumption 4.10.

Function	Base optimizer	$R_O(T)$
Convex L-smooth	clipped-SSTM	$O\left(\max\left[KLR^2, \alpha\sigma RK^{1-\frac{1}{\alpha}}T^{\frac{1}{\alpha}}\log(AKT)\right]\right)$
μ-strongly convex, L-smooth	R-clipped-SSTM	$O\left(\max\left[K\sqrt{\frac{L}{\mu}}, \frac{\sigma^2}{\mu}K^{2\frac{\alpha-1}{\alpha}}T^{\frac{2}{\alpha}-1}\log(AKT)\right]\right)$
μ-strongly convex, M-Lipschitz	AGD	$O\left(\sqrt{KT}\sigma R\log(AKT)\right)$



(a) Cumulative regret **(b) Convergence rate**



(c) Function values

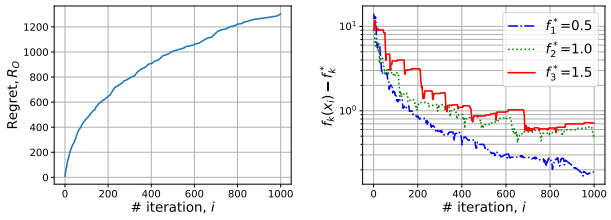
Figure 1: Dependence of cumulative regret (upper left), convergence rate (upper right), and function values (bottom) on iterations of F-LCB algorithms for FMAB setup with smooth convex functions (20). Regret stops increasing after 75 iterations, and F-LCB minimizes only f_1 with $f_1^* < f_2^*$ and $f_1^* < f_3^*$.

5.1 FMAB: smooth convex functions

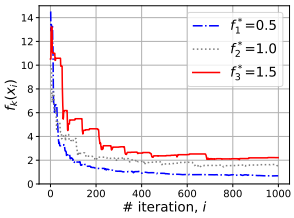
We consider the following set of smooth convex functions:

$$f_i(\mathbf{x}) = \sqrt{1 + (\mathbf{x} - \mathbf{x}_i^*)^\top \Sigma_i (\mathbf{x} - \mathbf{x}_i^*)} + c_i, \tag{20}$$

where $\Sigma_i = \mathrm{diag}(\sigma_1, \ldots, \sigma_d)$ is a diagonal matrix, where $\sigma_i > 0$. We set $\sigma_1 = 1$ and $\sigma_i = \exp(-5\xi)$, where $\xi \sim U[0, 1]$ for $i = 2, \ldots, d$. Here, functions are not strongly convex but have a Lipschits gradient with $L_i = \max_{i=1,\ldots,d} \sigma_i$. We use accelerated gradient descent [27, 37] as a base optimizer in this setup. According to [38], the function g has the following form for this base optimizer: $g_i(t) = \frac{2L_i\|\mathbf{x}^{0,i} - \mathbf{x}_\star\|^2}{t^2+5t+6}$ and hense the regret is bounded by constant $O(KLR^2)$. We generate $K = 3$ instances of functions with $d = 20$ and run the algorithm for $T = 200$ steps. Figure 1 shows that F-LCB algorithm automatically selects the function with the smallest optimal value. After some iterations, it minimizes only this function, while the target variables for other functions are not updated. The stepwise decreasing of f_3 in Figure 1c illustrates such behavior. Thus, F-LCB identifies the smooth convex function f_1 among other similar functions $\{f_2, f_3\}$ such that $f_1^* < f_2^*$ and $f_1^* < f_3^*$.



(a) Cumulative regret **(b) Convergence rate**



(c) Function values

Figure 2: Dependence of cumulative regret (upper left), convergence rate (upper right), and functions values (bottom) on the number of iterations of F-LCB for FMAB setup with nonsmooth convex functions (21). The function f_1 with the smallest minimal value is identified, and the smallest minimum $f_1^* = 0.5$ is achieved. Spikes in plots (b) and (c) indicate the switching between the minimization of $\{f_1, f_2, f_3\}$.

5.2 FMAB: nonsmooth convex functions

To illustrate the performance of our algorithm in the nonsmooth convex setup, we consider piece-wise linear functions with feasible sets $D_i = [-4, 4]^d$ and $d = 20$:

$$f_i(\mathbf{x}) = \max_{k=1,\ldots,p} (\mathbf{a}_{ki}^\top \mathbf{x} + b_k^i) + c_i. \tag{21}$$

We consider $K = 3$ functions and run the algorithm for $T = 1000$ steps. We use $p = \{5, 10, 12\}$ linear functions for given minimal values $\{0.5, 1, 1.5\}$ respectively. We use the Subgradient Method with Triple Averaging [28] as a base optimizer for such functions. For this base optimizer, we have $g_i(t) = \frac{M_i R_i}{\sqrt{t}}$. Hence, cumulative regret is bounded by $O\left(RM\sqrt{KT}\right)$. The resulting cumulative regret and function values are presented in Figure 2. The convergence of F-LCB demonstrates that the minimization process for the target objective function f_1 leads to faster convergence to the minimum.

5.3 FMAB: smooth convex functions with inexact oracle

To emulate the inexact oracle in the smooth convex setup, we consider functions (20) but add noise to the gradients. The gradient estimate is computed as $G_i(\mathbf{x}, \xi) = \nabla f_i(\mathbf{x}) + \frac{\sigma_i}{\sqrt{d}}\xi$, where $\xi \sim \mathcal{N}(0, I)$. We use stochastic accelerated gradient descent as a base optimizer and parameters from proposition 4.5 in [22] that give $\mathbb{E}[f(\overline{\mathbf{x}}_t)] - f^* \leq O(1)\left(2\gamma R + \frac{4\sqrt{2}(M^2+\sigma^2)}{3\gamma}\right)\frac{1}{\sqrt{t}}$. We consider $K = 3$ functions, $T = 1500$ steps, dimension $d = 20$ and $\sigma = 2$. Figure 3 shows that although gradient is inexact, our algorithm finds the best

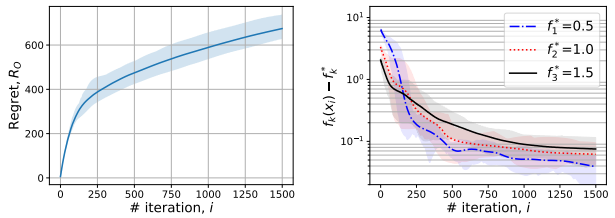
Table 3: Summary of selected neural networks for image clsssification task used to evaluate the F-LCB. The models are ranked according to their Top-1 accuracy on the validation set.

Model	top-1 acc, %	# params, $\cdot 10^6$
mobilenetv2_x1_4	75.98	4.50
mobilenetv2_x1_0	74.20	2.35
shufflenetv2_x1_5	73.91	2.58
mobilenetv2_x0_75	73.61	1.48
resnet56	72.63	0.86
shufflenetv2_x1_0	72.39	1.36
resnet44	71.63	0.67
mobilenetv2_x0_5	70.88	0.82
resnet32	70.16	0.47
resnet20	68.83	0.28

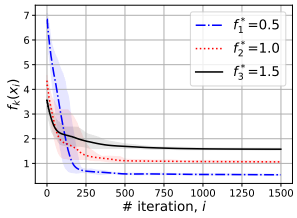
Table 4: Mean rank of the best model selected by each algorithm. The closer the rank to one, the better the algorithm works. F-LCB algorithm demonstrates strong performance for tested budgets. The smallest mean model ranks are bold. SH denotes the SuccessiveHalving algorithm.

Budget, T	Hyperband	SH	F-LCB
50	4.6 ± 2.7	2.5 ± 0.9	$\mathbf{2.2 \pm 2.7}$
100	3.4 ± 3.2	2.9 ± 0.5	$\mathbf{1.1 \pm 0.3}$
200	3.8 ± 2.9	1.1 ± 0.3	$\mathbf{1.1 \pm 0.3}$
350	1.7 ± 0.9	1.0 ± 0.0	$\mathbf{1.0 \pm 0.0}$
500	2.7 ± 2.2	1.0 ± 0.0	$\mathbf{1.0 \pm 0.0}$

function. In contract to the deterministic setup, the convergence curves shown in Figure 3b are less distinguished. However, Figure 3c shows that F-LCB pays more attention to the minimization of f_1 rather than f_2 or f_3.



(a) Cumulative regret **(b) Convergence rate**



(c) Function values

Figure 3: Dependence of cumulative regret (upper left), convergence rate (upper right), and functions values (bottom) on iterations of the F-LCB algorithm in the FMAB setup for smooth convex functions with inexact oracles. The best function is found automatically even if the initial guess is poor.

5.4 BFI: neural network selection

We compare the performance of our algorithm with SuccessiveHalving [17] (denoted as SH) and Hyperband [24] methods. We evaluate how well these algorithms can distinguish between neural network architectures with similar performance. We consider models for the image classification task and train them in the CIFAR-100 dataset [21] on the single GPU P100. The selected models have fewer than 5M parameters and are represented as arms in the BFI problem. Table 3 provides a summary of the selected models.

LCB estimation. In this setup, the pull of the i-th arm is 40 updates of the i-th model parameters. After that, the validation loss and accuracy are computed to update the corresponding LCBs. Since training neural networks is a non-convex problem, convergence guarantees do not directly apply. Therefore, to define the function g, we use a heuristic approach inspired by the stochastic optimization theory [22]. In particular, we define $g(t) = \frac{2 \cdot f_i(x^{1,i})}{\sqrt{t}}$, where the nominator estimates the maximum function deviation during the training process, and $x^{1,i}$ is obtained after the first 40 updates. Since the training process could lead to a large variance in validation losses, we compute the LCBs based on the best validation loss for each model computed before the current step. (More details about the experimental setup and used hyperparameters are presented in Appendix C of [11]).

Results. The experimental comparison of the F-LCB algorithm is presented in Table 4, where the mean and standard deviation of the selected model rank are reported. Each algorithm run is repeated 10 times. These ranks demonstrate that F-LCB can identify the best model using a smaller training budget. The selected models provide the smallest validation loss for each budget and algorithm. In contrast to the competitors, F-LCB does not discard models permanently. Since Hyperband runs SH multiple times and splits the budget between them, it shows poor performance. The parameters for Hyperband are chosen so that the total training steps are approximately equal to the given budget.

6 CONCLUSION AND LIMITATIONS

This work investigates strategies for the functional multi-armed bandit problem (FMAB) and for best function identification (BFI). We propose a UCB-type algorithm that uses basic optimizers with known large deviation bounds to construct LCB estimates. It establishes regret rate guarantees for FMAB and BFI problems that match corresponding lower bounds up to a logarithmic factor. Extensive experimental evaluation demonstrates the efficiency of the proposed F-LCB algorithm. First, our approach identifies the best functions among smooth and non-smooth functions with base optimizers equipped with deterministic oracles. Second, we show that F-LCB can process the base optimizer with an inexact oracle for smooth objective functions. Finally, we compare F-LCB with SuccessiveHalving and Hyperband to identify the best neural network for the given task. Our method outperforms competitors if a small computing budget is available. In settings with moderate or large budgets, F-LCB performs similarly to competitors and identifies the best model for the task.

Limitations. Our approach requires the function $g(k, \delta)$ for the base optimizer, since $g(k, \delta)$ is directly used for LCB computation. Therefore, one must know the objective's functional class and convergence rates for the base algorithm in advance. One also needs to observe objective values or their approximations. This requirement is satisfied for most classical ML models and corresponding training algorithms. However, it often fails for NN-based models since there are no convergence rates (in terms of objective value) for algorithms used for NN training aligned with definition 4.1.

ACKNOWLEDGMENTS

This work was supported by the The Ministry of Economic Development of the Russian Federation in accordance with the subsidy agreement (agreement identifier 000000C313925P4H0002; grant No 139-15-2025-012).

REFERENCES

[1] Rajeev Agrawal. 1995. The continuum-armed bandit problem. *SIAM journal on control and optimization* 33, 6 (1995), 1926–1951.
[2] Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. 2010. Best Arm Identification in Multi-Armed Bandits. In *Proceedings of the 23rd Conference on Learning Theory (COLT)*. JMLR, Haifa, Israel, 41–53.
[3] Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3, Nov (2002), 397–422.
[4] Amir Beck and Marc Teboulle. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences* 2, 1 (2009), 183–202.
[5] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning* 5, 1 (2012), 1–122.
[6] Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. 2011. X-Armed Bandits. *Journal of Machine Learning Research* 12, 5 (2011), 1655–1695.
[7] Sébastien Bubeck, Gilles Stoltz, Csaba Szepesvári, and Rémi Munos. 2008. Online optimization in X-armed bandits. *Advances in Neural Information Processing Systems* 21 (2008), 201–208.
[8] Nicolo Cesa-Bianchi and Gábor Lugosi. 2006. *Prediction, Learning, and Games*. Cambridge University Press, Cambridge, UK.
[9] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. 2011. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, Fort Lauderdale, FL, USA, 208–214.
[10] Yuriy Dorn, Aleksandr Katrutsa, Ilgam Latypov, and Andrey Pudovikov. 2024. Fast UCB-type algorithms for stochastic bandits with heavy and super heavy symmetric noise. *arXiv preprint arXiv:2402.07062* (2024). arXiv:2402.07062 [cs.LG]
[11] Yuriy Dorn, Aleksandr Katrutsa, Ilgam Latypov, and Anastasiia Soboleva. 2025. Functional multi-armed bandit and the best function identification problems. *arXiv preprint arXiv:2503.00509* (2025).
[12] Stefan Falkner, Aaron Klein, and Frank Hutter. 2018. BOHB: Robust and efficient hyperparameter optimization at scale. In *International conference on machine learning*. PMLR, 1437–1446.
[13] Ghazal Fazelnia, Ramtin Madani, Abdulrahman Kalbat, and Javad Lavaei. 2016. Convex relaxation for optimal distributed control problems. *IEEE Trans. Automat. Control* 62, 1 (2016), 206–221.
[14] Julia Gusak, Daria Cherniuk, Alena Shilova, Alexandr Katrutsa, Daniel Bershatsky, Xunyi Zhao, Lionel Eyraud-Dubois, Oleh Shliazhko, Denis Dimitrov, Ivan V Oseledets, et al. 2022. Survey on Efficient Training of Large Neural Networks.. In *IJCAI*. 5494–5501.
[15] Sheena Haines, Jason Loeppky, Paul Tseng, and Xianfu Wang. 2013. Convex relaxations of the weighted maxmin dispersion problem. *SIAM Journal on Optimization* 23, 4 (2013), 2264–2294.
[16] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. 2017. Population based training of neural networks. *arXiv preprint arXiv:1711.09846* (2017).
[17] Kevin Jamieson and Ameet Talwalkar. 2016. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*. PMLR, 240–248.
[18] Rie Johnson and Tong Zhang. 2013. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems* 26 (2013).
[19] André A Keller. 2014. Convex Relaxation Methods for Nonconvex Polynomial Optimization Problems. In *Mathematics and Computers in Sciences and Industry, Proc. Intern. Conf., Varna, Bulgaria*. 36–45.
[20] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. 2008. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*. 681–690.
[21] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images*. Technical Report. University of Toronto. https://www.cs.toronto.edu/~kriz/cifar.html
[22] Guanghui Lan. 2020. *First-order and stochastic optimization methods for machine learning*. Vol. 1. Springer.
[23] Tor Lattimore and Csaba Szepesvári. 2020. *Bandit algorithms*. Cambridge University Press.
[24] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research* 18, 185 (2018), 1–52.
[25] Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. 2020. A system for massively parallel hyperparameter tuning. *Proceedings of machine learning and systems* 2 (2020), 230–246.
[26] Arkadi Nemirovsky and David Yudin. 1983. *Problem complexity and method efficiency in optimization*. Wiley-Interscience.
[27] Y Nesterov. 1983. A Method for Solving a Convex Programming Problem with Convergence Rate $O(1/K^2)$. In *Soviet Mathematics. Doklady*, Vol. 27. 367–372.
[28] Yu Nesterov and Vladimir Shikhman. 2015. Quasi-monotone subgradient methods for nonsmooth convex minimization. *Journal of Optimization Theory and Applications* 165, 3 (2015), 917–940.
[29] Jack Parker-Holder, Vu Nguyen, and Stephen J Roberts. 2020. Provably efficient online hyperparameter optimization with population-based bandits. *Advances in neural information processing systems* 33 (2020), 17200–17211.
[30] Herbert Robbins. 1952. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.* (1952).
[31] Abdurakhmon Sadiev, Marina Danilova, Eduard Gorbunov, Samuel Horváth, Gauthier Gidel, Pavel Dvurechensky, Alexander Gasnikov, and Peter Richtárik. 2023. High-probability bounds for stochastic optimization and variational inequalities: the case of unbounded variance. In *International Conference on Machine Learning*. PMLR, Honolulu, Hawaii, USA, 29563–29648.
[32] Anureet Saxena, Pierre Bonami, and Jon Lee. 2010. Convex relaxations of non-convex mixed integer quadratically constrained programs: extended formulations. *Mathematical programming* 124, 1 (2010), 383–411.
[33] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. 2015. Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE* 104, 1 (2015), 148–175.
[34] Aleksandrs Slivkins. 2011. Contextual bandits with similarity information. In *Proceedings of the 24th annual Conference On Learning Theory*. JMLR Workshop and Conference Proceedings, 679–702.
[35] Aleksandrs Slivkins et al. 2019. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning* 12, 1-2 (2019), 1–286.
[36] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems* 25 (2012).
[37] Weijie Su, Stephen Boyd, and Emmanuel J Candes. 2016. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research* 17, 153 (2016), 1–43.
[38] Adrien B Taylor, Julien M Hendrickx, and François Glineur. 2017. Exact worst-case performance of first-order methods for composite convex optimization. *SIAM Journal on Optimization* 27, 3 (2017), 1283–1313.
[39] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3-4 (1933), 285–294.
[40] Long Tran-Thanh and Jia Yuan Yu. 2014. Functional bandits. *arXiv preprint arXiv:1405.2432* (2014).
[41] Nazarii Tupitsa, Pavel Dvurechensky, Darina Dvinskikh, and Alexander Gasnikov. 2022. Numerical methods for large-scale optimal transport. *arXiv preprint arXiv:2210.11368* (2022).
[42] Stein W Wallace and William T Ziemba. 2005. *Applications of stochastic programming*. SIAM.
[43] Yizao Wang, Jean-Yves Audibert, and Rémi Munos. 2008. Algorithms for infinitely many-armed bandits. *Advances in Neural Information Processing Systems* 21 (2008).
[44] Michael Woodroofe. 1979. A one-armed bandit problem with a concomitant variable. *J. Amer. Statist. Assoc.* 74, 368 (1979), 799–806.
[45] Wentao Wu and Chi Wang. 2024. Budget-aware Query Tuning: An AutoML Perspective. *ACM SIGMOD Record* 53, 3 (2024), 20–26.
[46] Xinyi Zhang, Zhuo Chang, Hong Wu, Yang Li, Jia Chen, Jian Tan, Feifei Li, and Bin Cui. 2023. A unified and efficient coordinating framework for autonomous dbms tuning. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–26.