Constraint-Aware RoBoT: Feasible-Precision@K Guarantees for Training-Free NAS Under Deployment Budgets

Liz Lemma Future Detective

January 20, 2026

Abstract

Training-free neural architecture search (NAS) ranks architectures using zero-cost proxies, but modern deployment requires satisfying strict latency/memory/energy budgets; unconstrained proxy rankings often concentrate mass on infeasible models. Inspired by RoBoT (ICLR 2024), we propose a constraint-aware framework that makes constraints first-class in both the metric robustification and the exploitation stage. We define Feasible-Precision@K, the overlap between the proxy top-K and true top-K architectures restricted to the feasible set, and show it is the right quantity to quantify the estimation gap in deployment-aware NAS. Our main theoretical contribution is a constrained exploitation theorem bounding the expected true feasible rank of the returned architecture as a function of Feasible-Precision@K, together with a matching minimax lower bound (up to constants) for any method restricted to evaluating within proxy top-K feasible candidates. We further handle noisy feasibility via constraint predictors, proving graceful degradation in terms of false-feasible and false-infeasible rates. Finally, we instantiate a constraint-aware RoBoT algorithm that uses Bayesian optimization to learn a robust linear proxy ensemble while optimizing Feasible-Precision@K, then allocates the remaining evaluation budget to greedy exploitation among the proxy's top feasible architectures. Empirical results on hardware-aware NAS benchmarks (measured latency tables and on-device constraints) demonstrate improved feasible-model quality and better wall-clock efficiency versus unconstrained training-free and hybrid NAS baselines.

Table of Contents

2. 2. Related Work: training-free proxies; constraint-aware NAS/BO; RoBoT and partial monitoring; hardware-aware benchmarks/latency prediction.

3. 3. Problem Setup and Metrics: formal definition of feasible set, constrained rankings, Feasible-Precision@K; discussion of what can be observed and what is queried.

4. 4. Constrained Exploitation Theory: derive expected feasible-rank bounds for greedy top-K evaluation; provide matching lower bound; interpretability and regime analysis (small/large K, sparse feasible sets).

5. 5. Noisy Constraints: prediction model, false-feasible/false-infeasible rates; effective K and effective precision; refined bounds and budget-wastage analysis.

6. 6. Constraint-Aware RoBoT Algorithm: BO over proxy-combiner weights with feasibility filtering; exploitation policy; practical considerations (normalization, constraint predictors, tie-breaking, batch queries).

7. 7. Experiments (would strengthen contribution): hardware-aware NAS-Bench variants; on-device latency budgets; ablations on (i) constrained vs unconstrained Precision@K, (ii) constraint-noise, (iii) choice of proxy ensemble and acquisition; comparisons to constraint-aware BO baselines.

8. 8. Discussion and Extensions: multi-constraint Pareto front variants; distribution-free certificates as future work; implicit search spaces and approximate maximizers; limitations.

9. 9. Conclusion: summary of guarantees and deployment relevance.

1 Related Work

Training-free proxy metrics and zero-cost predictors. A substantial body of work seeks to avoid expensive objective queries by scoring candidate architectures using *training-free* or *zero-cost* signals computed from random initialization, a few forward/backward passes, or simple structural statistics. Representative examples include gradient- and sensitivity-based criteria (e.g., SynFlow and related path-norm surrogates), Jacobian/NTK-style conditioning measures, Fisher-information or diagonal approximations thereof, and structure-induced measures of feature diversity or activation pattern separation (e.g., NASWOT-type scores); surveys and comparative studies appear in, e.g., **??**. These proxies are typically evaluated by correlation between the proxy ranking and the fully trained objective ranking, often via Spearman or Kendall statistics, or by reporting the best objective found when selecting the proxy top-K architectures under an *unconstrained* candidate set. From our perspective, two common limitations follow. First, a global rank correlation over the full set is neither necessary nor sufficient for *top-set retrieval* under a small evaluation budget, since only the upper tail of the distribution matters. Second, when deployment feasibility constrains the admissible set, proxy alignment should be assessed *after* restricting to the feasible subset; a proxy that is well correlated globally may misorder the feasible frontier. These considerations motivate the use of top-K retrieval-style metrics restricted to feasibility, rather than a single global correlation coefficient, when the downstream decision is constrained selection.

Constrained and multi-objective neural architecture search. Constraint-awareness has long been a central requirement in practical NAS, particularly for latency, memory, and energy budgets. Earlier approaches handle constraints by incorporating a penalty or Lagrangian term into the objective, by casting the problem as multi-objective optimization, or by using constrained Bayesian optimization (BO) with separate models for the objective and constraints **??**. In the NAS literature, hardware-aware methods such as MnasNet, ProxylessNAS, and FBNet incorporate latency or FLOPs into the search objective or impose hard constraints during sampling **???**. In one-shot and weight-sharing regimes, constraint handling is often implemented by restricting the sampled subnets, using resource-aware regularizers, or learning distributions over architectures conditioned on budgets **?**. These methods, however, generally presuppose either (i) access to relatively accurate constraint measurements for many candidates, or (ii) the ability to amortize evaluation costs through shared weights. Our setting differs in that we explicitly target *query efficiency* in terms of expensive objective evaluations, and we separate cheap metric computation from expensive objective access. Moreover, rather than optimizing a scalarized objective, we emphasize the induced *constrained ranking* within the feasible set, since the end goal is to

output a single feasible architecture with high true objective.

Using proxy rankings: beyond rank correlation. Several works interpret training-free proxies as a ranking mechanism and evaluate their quality via top-K hit rates or success probabilities, sometimes under additional heuristics (e.g., re-ranking, ensembling, or iterative refinement) **?**. Nevertheless, the dominant evaluation still reports aggregate correlation on a benchmark search space. In constrained settings, this can conflate two effects: (i) whether the proxy correlates with the objective, and (ii) whether the proxy selects candidates that are feasible in the first place. Our development can be viewed as making this separation explicit by measuring alignment *within* the feasible subset and treating the proxy as a retrieval rule operating on $\mathcal{F}$ (or its prediction). This viewpoint is aligned with classical information retrieval metrics and with best-arm identification under a restricted candidate pool, but we focus on constrained NAS where feasibility itself can be uncertain and must be verified upon query.

RoBoT, rank-based BO, and partial monitoring. RoBoT-style methods perform BO over a low-dimensional space of *proxy combiners* (e.g., weights w in a linear score of normalized metrics) and only observe the expensive objective on the single architecture selected by the current proxy; this induces a feedback structure that is weaker than bandit feedback on a fixed action set. The partial-monitoring formalism provides a natural language for such problems: the learner chooses an action (here, a proxy parameter) and receives limited feedback (here, the objective of the proxy-maximizer), with performance measured relative to the best action in hindsight **?**. Under global observability, one can obtain regret rates of order $\widetilde{O}(T^{2/3})$ for suitable algorithms (e.g., IDS/UCB variants) in a kernelized setting; RoBoT instantiates this idea for training-free NAS by learning proxy weights using few objective evaluations **?**. Our contribution is complementary: we incorporate feasibility filtering into both the definition of proxy performance and the selection rule, and we propagate the resulting top-K proxy quality (restricted to feasibility) into explicit bounds on the expected feasible rank of the final output.

Hardware-aware benchmarks and latency/constraint prediction. Empirical progress in NAS has been accelerated by tabular benchmarks such as NAS-Bench-101/201 and their variants, which provide precomputed accuracies for large numbers of architectures **??**. For deployment-oriented work, benchmarks and datasets increasingly include hardware measurements (e.g., latency on specific devices) or simulators thereof, and they expose the nontrivial gap between proxy constraints (FLOPs, parameter count) and real constraints (end-to-end latency, peak memory) **?**. Constraint prediction is

often implemented via lookup tables, additive layerwise models, or learned regressors (e.g., MLPs, gradient-boosted trees, or graph-based predictors over the architecture DAG) trained on a limited set of measurements **?**. Such predictors exhibit both false-feasible and false-infeasible errors due to measurement noise, operator fusion, memory effects, and device-specific kernels. This motivates a design in which predicted feasibility is used as a *filter* for candidate generation, but feasibility is ultimately verified upon expensive query. Our analysis formalizes this practice by parameterizing predictor quality through explicit misclassification rates and showing how these rates reduce the effective top-K budget and the attainable proxy precision within the true feasible set.

2 Problem Setup and Metrics

We study constrained architecture selection over a finite candidate set. Let $\mathcal{A}$ be a finite set of N architectures. Each $A \in \mathcal{A}$ is associated with an (unknown) scalar objective value $f(A) \in \mathbb{R}$ that we wish to maximize, and a (possibly unknown) constraint vector $c(A) \in \mathbb{R}^d$ that must lie below a given budget $b \in \mathbb{R}^d$ componentwise. The *true feasible set* is

$$\mathcal{F} \;:=\; \{A \in \mathcal{A} : c(A) \le b\}.$$

Our output must be feasible, i.e., we must return an architecture $\widehat{A} \in \mathcal{F}$.

Information access and query budget. We assume access to a collection of M *training-free metrics* $\{M_i\}_{i=1}^{M}$, where each $M_i(A)$ can be computed cheaply for any $A \in \mathcal{A}$ (e.g., from random initialization and a small number of forward/backward passes, or from graph/statistical properties of the architecture). In contrast, evaluating $f(A)$ requires an expensive query (e.g., training and validation), and we impose a strict budget of at most T objective queries. Depending on the setting, constraints may be (i) known tabularly for all architectures, in which case $\mathcal{F}$ is known at selection time, or (ii) measurable only upon query (e.g., device latency measurement), in which case feasibility must be verified for any queried candidate before it can be output. Our goal is query-efficient: using cheap metric computation and at most T expensive evaluations of f, we aim to output a feasible architecture with large objective value (equivalently, small rank among feasible candidates).

Metric normalization and proxy scores. Because different metrics may have incompatible scales, we work with normalized values $\widetilde{M}_i(A) \in [0,1]$ derived from $M_i(A)$ by a fixed normalization scheme (e.g., min–max normalization over the candidate pool, or rank-based normalization). For a weight

vector $w \in \mathbb{R}^M$, we define the induced proxy score

$$S_w(A) \;:=\; \sum_{i=1}^{M} w_i \, \widetilde{M}_i(A).$$

The family $\{S_w\}$ provides a low-dimensional control space (the weights) over which we may search, while the architecture space $\mathcal{A}$ itself can be extremely large. In the sequel, w is treated as a decision variable that determines how we combine training-free metrics into a single proxy ranking.

Constrained rankings. Since feasibility restricts the admissible set, we evaluate both the objective and the proxy *within* $\mathcal{F}$. For any scoring function $g : \mathcal{A} \to \mathbb{R}$, we define the feasible rank $R_g^{\mathcal{F}}(A)$ of an architecture $A \in \mathcal{F}$ as its position when sorting $\mathcal{F}$ in descending order of g (rank 1 is best). When ties occur, we assume an arbitrary but fixed tie-breaking rule (or equivalently, uniform tie-breaking; none of our definitions depend materially on the choice). We will be interested in $R_f^{\mathcal{F}}(A)$ (true objective rank among feasible architectures) and $R_{S_w}^{\mathcal{F}}(A)$ (proxy rank among feasible architectures). Our evaluation criterion is inherently *tail-focused*: because we can only afford a small number of objective evaluations, only the top portion of these rankings is operationally relevant.

Feasible-Precision@K. Fix a top-K parameter $K \le T$. Given a proxy score S_w, we define the *Feasible-Precision@K* of S_w with respect to f as

$$\rho_K^{\mathcal{F}}(S_w, f) \;:=\; \frac{1}{K} \left| \left\{ A \in \mathcal{F} : R_{S_w}^{\mathcal{F}}(A) \le K \;\wedge\; R_f^{\mathcal{F}}(A) \le K \right\} \right|.$$

Thus $\rho_K^{\mathcal{F}}(S_w, f) \in [0, 1]$ is the fraction of architectures that lie simultaneously in the proxy top-K feasible set and the true top-K feasible set. This is a constrained, top-set retrieval metric: it is insensitive to global correlation far from the optimum and directly quantifies how many "truly good" feasible architectures are exposed by querying within the proxy top-K feasible list. In particular, if we can afford to evaluate (up to) K candidates suggested by the proxy within $\mathcal{F}$, then $\rho_K^{\mathcal{F}}(S_w, f)$ controls the expected availability of truly top-ranked feasible candidates inside that queried pool. We further denote the best attainable value over the score family by

$$\rho_K^{\mathcal{F},*} \;:=\; \max_w \rho_K^{\mathcal{F}}(S_w, f),$$

which serves as a benchmark for learning a good combiner.

Unknown feasibility and predicted-feasible filtering. When $c(A)$ is not known a priori, we assume access to a cheap predictor $\widehat{c}(A)$ (e.g., a learned latency model), yielding a *predicted feasible set*

$$\widehat{\mathcal{F}} \;:=\; \{A \in \mathcal{A} : \widehat{c}(A) \le b\}.$$

We parameterize predictor quality by misclassification rates: a false-feasible bound $\mathbb{P}(A \in \widehat{\mathcal{F}} \mid A \notin \mathcal{F}) \leq \alpha$ and a false-infeasible bound $\mathbb{P}(A \notin \widehat{\mathcal{F}} \mid A \in \mathcal{F}) \leq \beta$. Operationally, we use $\widehat{\mathcal{F}}$ as a filter for candidate generation and ranking, but we only certify feasibility by verifying $c(A) \leq b$ upon querying (when possible). This separation is essential: false-feasible errors can waste objective queries on infeasible architectures, while false-infeasible errors can remove good feasible candidates from consideration; both effects will appear explicitly in our rank guarantees via an effective reduction of the usable top-K budget and attainable precision within $\mathcal{F}$.

Optimization goal. The learner adaptively selects up to T architectures for expensive evaluation of $f(A)$ (and, when needed, verification of $c(A)$). Among all queried and verified-feasible architectures, we output $\widehat{A}$ with the largest observed objective value. Our performance metric is the expected feasible rank $\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})]$ (equivalently, the expected objective value subject to feasibility), and the central role of $\rho_K^{\mathcal{F}}(S_w, f)$ is to connect cheap proxy computation to explicit, budget-aware guarantees on this final feasible rank.

3 Constrained Exploitation Theory

In this section we isolate the *exploitation* component: we fix a weight vector w (learned by any means), restrict attention to feasible architectures, and ask what can be guaranteed by the greedy policy that evaluates the proxy top-K and returns the best feasible objective observed. The key point is that, once w is fixed, the only quantity connecting the proxy ranking to the objective ranking in the relevant tail is $\rho_K^{\mathcal{F}}(S_w, f)$.

Proxy top-K feasible pool and intersection size. For a fixed w, define the proxy-induced queried pool within $\mathcal{F}$ by

$$\mathcal{T}_K(w) \;:=\; \{A \in \mathcal{F} : R_{S_w}^{\mathcal{F}}(A) \leq K\}.$$

If we evaluate f on all $A \in \mathcal{T}_K(w)$ and output $\widehat{A} = \arg\max_{A \in \mathcal{T}_K(w)} f(A)$, then our output rank is entirely determined by which elements of $\mathcal{T}_K(w)$ actually belong to the true objective top-K within $\mathcal{F}$. Let

$$\mathcal{I}_K(w) \;:=\; \{A \in \mathcal{F} : R_{S_w}^{\mathcal{F}}(A) \leq K \;\wedge\; R_f^{\mathcal{F}}(A) \leq K\}.$$

By definition, $|\mathcal{I}_K(w)| = \rho_K^{\mathcal{F}}(S_w, f) \cdot K$. Write $m := |\mathcal{I}_K(w)|$. Note that elements in $\mathcal{T}_K(w) \setminus \mathcal{I}_K(w)$ have feasible objective rank strictly larger than K and hence cannot beat any element of $\mathcal{I}_K(w)$ in terms of feasible rank.

Expected feasible-rank bound via order statistics. We now quantify the expected best feasible rank obtained by the above exploitation rule. The only stochasticity we require is a mild symmetry assumption: conditional on membership in $\mathcal{I}_K(w)$, the objective ranks $\{R_f^{\mathcal{F}}(A) : A \in \mathcal{I}_K(w)\}$ behave like an (unordered) uniform m-subset of $\{1, 2, \ldots, K\}$.[1] Under this assumption, the exploitation output rank is the minimum rank among those m intersected items:

$$R_f^{\mathcal{F}}(\widehat{A}) \;=\; \min_{A \in \mathcal{I}_K(w)} R_f^{\mathcal{F}}(A),$$

since evaluating all of $\mathcal{T}_K(w)$ guarantees we observe the best objective value among the intersected set (and nothing outside the intersection can improve upon it). A standard identity for the expected minimum of a uniform m-subset of $\{1, \ldots, K\}$ yields

$$\mathbb{E}\left[\min_{A \in \mathcal{I}_K(w)} R_f^{\mathcal{F}}(A)\right] \;=\; \frac{K+1}{m+1} \;=\; \frac{K+1}{\rho_K^{\mathcal{F}}(S_w, f)\, K + 1},$$

provided $\rho_K^{\mathcal{F}}(S_w, f) > 0$ (otherwise $m = 0$ and no nontrivial rank guarantee is possible under the stated information). This is the bound we will use throughout: it translates a retrieval-quality quantity ($\rho_K^{\mathcal{F}}$) into a final expected feasible-rank guarantee for the natural top-K exploitation strategy.

Interpretation and regimes. The expression $\frac{K+1}{\rho K+1}$ has two immediate regimes (writing $\rho = \rho_K^{\mathcal{F}}(S_w, f)$ for brevity). First, when $\rho K \gg 1$ (meaning the proxy top-K contains many truly top-K feasible architectures), we obtain

$$\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})] \;\approx\; \frac{1}{\rho},$$

so the expected feasible rank is controlled primarily by the inverse precision and is essentially independent of K. Second, when $\rho K \ll 1$, the guarantee deteriorates to $\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})] \approx K + 1$, reflecting that the proxy top-K likely contains no truly top-K feasible item, hence querying K candidates is not sufficient to reliably access the very best feasible architectures. In particular, increasing K can help only insofar as it increases ρK, i.e., the *absolute* number of true top-K feasible elements exposed by the proxy list.

The bound is also sensitive to the feasible set size only through the meaning of K: we tacitly require $K \le |\mathcal{F}|$ (otherwise one replaces K by $K' := \min\{K, |\mathcal{F}|\}$ in the definitions). Notably, the feasible fraction $|\mathcal{F}|/|\mathcal{A}|$ does not enter: exploitation operates entirely within the feasible ranking. The operational difficulty in sparse-feasibility settings is therefore not the

[1] Equivalently, the set of true ranks appearing among the intersected items is uniformly distributed over all $\binom{K}{m}$ possibilities. This is the natural "no additional structure" assumption: the proxy identifies m genuinely top-K feasible items, but does not privilege which of the K ranks they occupy.

rank analysis itself, but rather whether the learner can *construct* $\mathcal{T}_K(w)$ without spending objective budget on infeasible candidates; we address this explicitly in the noisy-feasibility setting.

A matching minimax lower bound. We next argue that the dependence on ρ in the above guarantee cannot be substantially improved (up to constants) if an algorithm is constrained to query only within the proxy top-K feasible set. Consider the class of instances in which $\mathcal{T}_K(w)$ is fixed and only the placement of the $m = \rho K$ "good" (true top-K) feasible items inside $\mathcal{T}_K(w)$ is unknown. An adversary may choose the objective values so that exactly m of the K candidates belong to the true top-K set, and these m positions are drawn uniformly at random. Under this construction, any adaptive querying strategy that evaluates at most K items in $\mathcal{T}_K(w)$ is effectively attempting to find the minimum of an unknown uniform m-subset of $\{1, \ldots, K\}$; by Yao's principle, even randomized algorithms cannot achieve expected rank better than

$$\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})] \;\geq\; \Omega\left(\frac{K}{\rho K + 1}\right),$$

which matches the $\Theta\left(\frac{K+1}{\rho K+1}\right)$ behavior above up to constant factors. Consequently, without additional structure beyond knowledge (or learnability) of $\rho_K^{\mathcal{F}}(S_w, f)$, no method restricted to the proxy top-K feasible pool can guarantee expected feasible rank $o(1/\rho)$ when ρK is moderately large. This justifies our focus on (i) designing proxies that maximize feasible precision at the relevant K, and (ii) learning w so as to approach the best attainable $\rho_K^{\mathcal{F},*}$.

A self-contained statement. Fix a weight vector w and an integer K. Let $\mathcal{T}_K(w) = \{A \in \mathcal{F} : R_{S_w}^{\mathcal{F}}(A) \leq K\}$ be the proxy top-K feasible pool, and let $\mathcal{I}_K(w) = \{A \in \mathcal{F} : R_{S_w}^{\mathcal{F}}(A) \leq K \wedge R_f^{\mathcal{F}}(A) \leq K\}$ be the intersection with the objective top-K feasible set. We denote $m := |\mathcal{I}_K(w)| = \rho_K^{\mathcal{F}}(S_w, f)\, K$. The exploitation policy under consideration evaluates f on all architectures in $\mathcal{T}_K(w)$ (or on as many as the remaining budget allows) and outputs the feasible queried architecture with maximal f.

To avoid pathologies due to ties, we assume that $R_f^{\mathcal{F}}$ and $R_{S_w}^{\mathcal{F}}$ are defined using independent random tie-breaking whenever equal scores occur; the resulting ranks are then well-defined random variables on $\{1, \ldots, |\mathcal{F}|\}$.

Why the intersection controls the output rank. By definition, every $A \in \mathcal{T}_K(w) \setminus \mathcal{I}_K(w)$ satisfies $R_f^{\mathcal{F}}(A) > K$, i.e., it lies strictly below the objective top-K feasible set. Consequently, even if such an architecture has the best objective value among the non-intersecting elements, it cannot have

feasible rank better than any member of $\mathcal{I}_K(w)$. Thus, on the event $m \geq 1$, the feasible rank of the exploitation output is exactly

$$R_f^{\mathcal{F}}(\widehat{A}) \;=\; \min_{A \in \mathcal{I}_K(w)} R_f^{\mathcal{F}}(A).$$

When $m = 0$ there is, under our information model, no nontrivial guarantee: the proxy top-K feasible list contains no truly top-K feasible architecture, and the output rank may be arbitrarily large.

Order-statistic calculation under a symmetry assumption. We impose the following symmetry ("no privileged ranks") condition: conditional on $\mathcal{I}_K(w)$ having size m, the multiset of objective ranks $\{R_f^{\mathcal{F}}(A) : A \in \mathcal{I}_K(w)\}$ is distributed as an unordered uniformly random m-subset of $\{1, 2, \ldots, K\}$. Equivalently, every subset of $\{1, \ldots, K\}$ of cardinality m is equally likely to be realized as the set of ranks occupied by the intersected architectures.

Under this assumption, letting $Z := \min_{A \in \mathcal{I}_K(w)} R_f^{\mathcal{F}}(A)$, we have for each $r \in \{0, 1, \ldots, K-1\}$,

$$\mathbb{P}(Z > r) = \frac{\binom{K-r}{m}}{\binom{K}{m}},$$

since the event $\{Z > r\}$ means all m ranks fall in $\{r+1, \ldots, K\}$, which has size $K - r$. Therefore

$$\mathbb{E}[Z] = \sum_{r=0}^{K-1} \mathbb{P}(Z > r) = \frac{1}{\binom{K}{m}} \sum_{r=0}^{K-1} \binom{K-r}{m}.$$

Using the hockey-stick identity $\sum_{u=m}^{K} \binom{u}{m} = \binom{K+1}{m+1}$ (substitute $u = K - r$), we obtain

$$\mathbb{E}[Z] = \frac{\binom{K+1}{m+1}}{\binom{K}{m}} = \frac{K+1}{m+1} = \frac{K+1}{\rho_K^{\mathcal{F}}(S_w, f)\, K + 1}.$$

This gives the exact exploitation bound claimed in Theorem 1 under the stated symmetry model.

Interpretation: dependence on K, ρ, and feasible-set sparsity. The expression $\frac{K+1}{\rho K+1}$ exhibits two informative regimes. If $\rho K \gg 1$ then $\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})] \approx 1/\rho$, so the expected feasible rank is governed mainly by the inverse feasible precision and is weakly sensitive to further increases in K. If instead $\rho K \ll 1$, then $\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})] \approx K + 1$, reflecting that the proxy top-K feasible pool is unlikely to contain any truly top-K feasible architecture. Importantly, the feasible fraction $|\mathcal{F}|/|\mathcal{A}|$ does not appear explicitly: once we can restrict attention to $\mathcal{F}$, exploitation is a ranking problem internal to $\mathcal{F}$. The practical difficulty in sparse-feasibility regimes lies in *constructing* $\mathcal{T}_K(w)$ without spending objective budget on infeasible architectures, which motivates the next section.

A matching lower bound under proxy-top-K restrictions. We now formalize tightness in the standard minimax sense. Consider any algorithm (possibly randomized and adaptive) that is restricted to issuing objective queries only within the fixed set $\mathcal{T}_K(w)$ and makes at most K such queries. We define a hard instance distribution as follows: an adversary selects exactly $m = \rho K$ architectures in $\mathcal{T}_K(w)$ to be the objective top-K feasible items (equivalently, to have feasible ranks in $\{1, \dots, K\}$) and places these m "good" items uniformly at random among the K positions of the proxy list; objective values are then assigned so that their induced feasible ranks realize this placement. By Yao's minimax principle, it suffices to lower bound the expected performance of any deterministic strategy under this random instance distribution.

Under this construction, before querying, the algorithm has no information about which proxy positions contain good items; adaptivity cannot create information that is not revealed by queries. The best feasible rank achievable after up to K queries is therefore lower bounded (up to constant factors) by the expected minimum rank among the m good items, which is $\mathbb{E}[Z] = (K+1)/(m+1) = \Theta\Big(\frac{K}{\rho K+1}\Big)$. Hence there exists an instance for which

$$\mathbb{E}[R_f^{\mathcal{F}}(\widehat{A})] \;\geq\; \Omega\left(\frac{K}{\rho K+1}\right),$$

matching the upper bound's dependence on ρ and K up to constants. In particular, without additional structural assumptions beyond the single alignment quantity $\rho_K^{\mathcal{F}}(S_w, f)$, no method confined to $\mathcal{T}_K(w)$ can guarantee expected feasible rank $o(1/\rho)$ in the regime $\rho K \to \infty$.

Noisy feasibility model. We now treat the case in which feasibility is not known a priori, but instead is inferred from a cheap predictor $\widehat{c} : \mathcal{A} \to \mathbb{R}^d$ (e.g., a regressor trained on measured latencies) that induces a predicted-feasible set

$$\widehat{\mathcal{F}} \;:=\; \{A \in \mathcal{A} : \widehat{c}(A) \leq b\}.$$

We assume that true feasibility can be *verified* when an architecture is actually evaluated (e.g., we can measure latency/memory during the expensive evaluation), so that we may discard infeasible queried items and still guarantee feasibility of the final output. The statistical role of $\widehat{c}$ is only to reduce wasted expensive queries. Accordingly, we quantify its errors by two one-sided misclassification bounds:

$$\mathbb{P}\Big(A \in \widehat{\mathcal{F}} \mid A \notin \mathcal{F}\Big) \leq \alpha, \qquad \mathbb{P}\Big(A \notin \widehat{\mathcal{F}} \mid A \in \mathcal{F}\Big) \leq \beta,$$

interpreted as a false-feasible rate α and a false-infeasible rate β. These bounds may be understood either in a distributional sense over a random draw of A from the candidate pool (or from the proxy-ranked list), or as

empirical bounds after calibration on a held-out measurement set; for our purpose, they serve as parameters controlling how far $\widehat{\mathcal{F}}$ deviates from $\mathcal{F}$.

Predicted-feasible exploitation and effective top-K. Fix w and K as before, but replace the proxy top-K feasible pool by the proxy top-K *predicted*-feasible pool

$$\widehat{\mathcal{T}}_K(w) \;:=\; \left\{A \in \widehat{\mathcal{F}} :\; R^{\widehat{\mathcal{F}}}_{S_w}(A) \le K\right\}.$$

The exploitation policy queries f on architectures in $\widehat{\mathcal{T}}_K(w)$ (or a prefix thereof) and, upon observing the true constraint vector (or at least a feasibility flag), retains only those queries that satisfy $c(A) \le b$, outputting the queried feasible item with maximal f.

Because $\widehat{\mathcal{T}}_K(w)$ may contain infeasible elements, the *effective* number of feasible evaluations can be smaller than K. A convenient conservative summary is

$$K_{\text{eff}} \;:=\; \lfloor (1-\alpha)K \rfloor,$$

which reflects the worst-case expectation that an α fraction of predicted-feasible selections are in fact infeasible. More precisely, if we let W be the number of infeasible architectures among the K queried members of $\widehat{\mathcal{T}}_K(w)$, then $\mathbb{E}[W] \le \alpha K$ by the false-feasible bound, and hence $\mathbb{E}[K-W] \ge (1-\alpha)K$. In the sequel we reason as if at least K_{eff} feasible evaluations are available, noting that this is a lower bound in expectation (and can be upgraded to a high-probability statement under additional concentration assumptions on the predictor errors along the proxy list).

Effective precision under false-infeasible errors. Even when we succeed in evaluating K_{eff} feasible candidates, we may have excluded some truly good feasible architectures from consideration because they were incorrectly filtered out of $\widehat{\mathcal{F}}$. This is precisely the impact of β. To capture it at the same level of granularity as our earlier analysis, we define an *effective* feasible precision

$$\rho_{\text{eff}} \;:=\; (1-\beta)\,\rho^{\mathcal{F}}_{K_{\text{eff}}}(S_w, f),$$

which may be read as follows: among the true feasible proxy top-K_{eff} list (the list we *would* have formed if $\mathcal{F}$ were known), only a $(1-\beta)$ fraction of its members are expected to survive the predicted-feasible filter. In particular, if β is small (conservative predictor), the proxy–objective alignment measured by $\rho^{\mathcal{F}}_{K_{\text{eff}}}$ is largely preserved; if β is large, good architectures are systematically excluded and no proxy-based method that relies on $\widehat{\mathcal{F}}$ can avoid this loss without incurring more infeasible trials.

Rank bound with verification (graceful degradation). We couple the predicted-feasible exploitation procedure with an idealized exploitation procedure restricted to $\mathcal{F}$ in two steps. First, we account for false-feasible inclusions: the K predicted-feasible queries contain at most αK infeasible elements in expectation, so at least K_{eff} of them are (in expectation) feasible and hence comparable to queries drawn from a proxy-ranked list within $\mathcal{F}$. Second, we account for false-infeasible exclusions: relative to the proxy ranking within $\mathcal{F}$, a fraction at most β of feasible architectures are removed by the filter, which reduces the expected size of the intersection between the queried set and the true objective top-K_{eff} feasible set by a factor $(1-\beta)$. Under the same symmetry assumption used previously for order statistics on the intersection ranks, the preceding reductions allow us to invoke the exploitation calculation with (K, m) replaced by $\big(K_{\text{eff}},\, \rho_{\text{eff}} K_{\text{eff}}\big)$, yielding

$$\mathbb{E}\Big[R_f^{\mathcal{F}}(\widehat{A})\Big] \;\le\; \frac{K_{\text{eff}}+1}{\rho_{\text{eff}} K_{\text{eff}}+1}.$$

Two limiting cases are worth recording. If $\alpha = 0$ and $\beta = 0$ we recover the noiseless-feasibility expression with $K_{\text{eff}} = K$ and $\rho_{\text{eff}} = \rho_K^{\mathcal{F}}(S_w, f)$. If $\alpha > 0$ but $\beta = 0$, the only effect is budget wastage: we effectively spend αK expensive evaluations on infeasible designs, but the proxy alignment among the remaining feasible trials is unchanged.

Budget wastage and practical tuning. The quantity αK is an upper bound on the *expected* number of wasted expensive objective evaluations during exploitation due to infeasibility. This bound clarifies the operational tradeoff in choosing the predictor threshold (or safety margin) used to form $\widehat{\mathcal{F}}$: tightening the filter decreases α (fewer infeasible queries) but typically increases β (more feasible candidates, including potentially high-f ones, are discarded). In settings with scarce objective budget T, a conservative choice that controls α is often preferable because it preserves effective sample size; in settings where feasibility violations are cheap to detect early (e.g., latency can be measured without full training), one can afford a larger α to reduce β and increase ρ_{eff}. These considerations will be reflected explicitly in the algorithmic design via feasibility filtering, early rejection, and the allocation between exploration and exploitation.

4 Constraint-Aware RoBoT (C-RoBoT): BO over proxy combiners with feasibility filtering

We now instantiate the preceding rank-based guarantees in a concrete procedure that allocates the expensive-query budget across (i) learning a proxy combiner and (ii) exploiting the resulting proxy ranking, while enforcing feasibility by construction.

Action space: weight vectors and induced candidates. We treat the combiner weights $w \in \mathbb{R}^M$ as the *continuous* decision variables and view each w as inducing a *single* architecture choice

$$A(w) \in \arg\max_{A \in \mathcal{G}} S_w(A),$$

where $\mathcal{G}$ denotes the candidate pool after feasibility filtering. In the tabular-known-feasibility setting we set $\mathcal{G} = \mathcal{F}$, whereas in the predictor setting we set $\mathcal{G} = \widehat{\mathcal{F}}$ and subsequently verify true feasibility upon query. In either case, the expensive oracle is accessed only through values of the form $f(A(w))$, which converts the discrete constrained search over $\mathcal{A}$ into a continuous black-box optimization problem over w with structured feedback. To avoid degeneracies, we restrict w to a compact set (e.g., $\|w\|_1 \leq 1$ or $w \in \Delta^{M-1}$), which also makes the BO surrogate well-posed.

Two-phase allocation: exploration then exploitation. C-RoBoT follows an explore–exploit template. In the *exploration* phase, we run BO over w and use each selected w_t to propose $A_t = A(w_t)$, which we then evaluate with the expensive oracle (and, when necessary, measure $c(A_t)$ to verify feasibility). We maintain a queried set $\mathcal{Q}$ so that if multiple w_t yield the same maximizer A_t, we do not re-spend budget; instead we record the duplicate proposal as additional information about the proxy landscape. We denote by T_0 the number of *distinct* objective evaluations consumed during exploration. In the *exploitation* phase, we fix the best weight vector $w^\star$ found by exploration (according to observed $f(A(w))$ among feasible evaluations) and spend the remaining budget to evaluate a proxy-ranked list under $S_{w^\star}$ within the feasible filter, finally returning the best feasible item seen.

Exploration details: BO on induced objective values. Concretely, we model the mapping $w \mapsto f(A(w))$ with a GP surrogate and choose w_t via an acquisition rule such as UCB or information-directed sampling. Although $A(w)$ is piecewise-constant in w (since it is defined by an arg max over a finite set), the surrogate remains effective in practice because the acquisition process only needs to identify regions of w that induce high-f candidates, and the induced discontinuities are aligned with proxy score ties. When objective observations are noisy (e.g., due to stochastic training), we may either treat the noise as homoscedastic in the GP likelihood or, if the budget permits, repeat evaluations of a small set of promising candidates and average their f values; in all cases the final selection is based on verified-feasible queried points.

Exploitation policy: top-K within the feasible filter. Given $w^\star$, we form the proxy ranking induced by $S_{w^\star}$ restricted to the feasibility filter $\mathcal{G}$. Let $A^{(1)}, A^{(2)}, \ldots$ be the sorted list in decreasing proxy score (breaking ties

deterministically; see below). We then query f on the first L distinct elements of this list not already in $\mathcal{Q}$, where typically $L = \min\{K, T - T_0\}$, though one may use all remaining budget when $T - T_0 > K$. If feasibility is only predicted, each queried item is immediately checked against the true constraint budget, and infeasible items are discarded (but still count against the budget). The final output is the feasible queried architecture with maximum observed objective value. This exploitation step matches the setting of our rank analysis, with the predictor-induced degradation summarized by (α, β) through $(K_{\text{eff}}, \rho_{\text{eff}})$.

Normalization and metric preprocessing. Since the proxy family is linear in normalized metrics, careful normalization is operationally important. We compute $\widetilde{M}_i(A) \in [0, 1]$ using either min–max scaling over the candidate pool or robust quantile clipping (e.g., mapping the 5th and 95th percentiles to 0 and 1) to prevent a single heavy-tailed metric from dominating the linear score. If a metric is missing for some architectures, we either impute it with a conservative value (e.g., the median) or restrict the candidate pool to architectures with complete metric vectors; the former preserves coverage while the latter simplifies reproducibility.

Feasibility predictors and safety margins. When $\widehat{c}$ is used, we recommend explicitly incorporating a safety margin to control α: instead of $\widehat{c}(A) \le b$ one may impose $\widehat{c}(A) \le b - \gamma$ componentwise, with $\gamma \ge 0$ tuned to trade off false-feasible and false-infeasible errors. The bounds above then apply with (α, β) corresponding to the calibrated predictor under the chosen margin. In latency-constrained settings, we may also implement *early rejection*: measure latency first, and only if the candidate passes, proceed with the expensive training/validation required to obtain $f(A)$, thereby reducing the realized cost of infeasible trials.

Tie-breaking, duplicates, and stable rankings. Because $A(w)$ is defined by an $\arg\max$, ties in S_w can create instability. We enforce a stable tie-breaking rule, e.g., lexicographic order on architecture identifiers or preference for lower predicted constraint usage, so that repeated runs with the same w yield identical selections. During exploitation, we similarly ensure that the top-K list is a list of *distinct* architectures; if duplicates arise from numerical ties, we skip to the next item. These conventions are immaterial to the theory but essential for a well-defined implementation.

Batch queries and parallelism. When the evaluation environment supports batch size $B > 1$, we extend both phases by selecting a batch of weight vectors (e.g., via batched UCB) and collecting the corresponding architectures $\{A(w_{t,1}), \ldots, A(w_{t,B})\}$, deduplicating them before dispatch. In

exploitation, we simply evaluate the next B items in the proxy-ranked list. This parallel variant preserves the same accounting of distinct objective evaluations and differs only in wall-clock time, not in budget usage.

5 Experiments

We evaluate C-RoBoT in settings where (i) feasibility is known from a tabular benchmark, (ii) feasibility is only partially observed through a learned predictor with controlled error, and (iii) the relevant constraint is measured on-device (latency) and is therefore both costly and noisy. Across all experiments, we report the best *verified-feasible* architecture found within the objective query budget T, and we additionally report rank-based diagnostics that directly instantiate our theory, including empirical estimates of $\rho_K^{\mathcal{F}}(S_w, f)$ and the realized feasible rank $R_f^{\mathcal{F}}(\widehat{A})$ when the benchmark provides ground-truth rankings.

Benchmarks and constraints. We consider hardware-aware NAS-Bench variants where each architecture is annotated with validation accuracy (our f) and one or more resource measurements (our c), e.g., latency on a specified device, FLOPs, parameter count, peak memory, or energy proxies. When true constraints are tabulated, $\mathcal{F}$ is known exactly and we can compute $\rho_K^{\mathcal{F}}$ without estimation error. When only a subset of architectures has measured constraints, we construct $\widehat{c}$ by regressing the constraint on architecture descriptors (or on training-free metrics), and we form $\widehat{\mathcal{F}} = \{A : \widehat{c}(A) \leq b\}$, verifying feasibility for every queried candidate by measuring $c(A)$ upon evaluation.

On-device latency budgets. To stress the setting motivating feasibility filtering, we include a deployment-style protocol in which the budget b is given as a hard latency limit (e.g., $b = 20\,\text{ms}$) on a target device. Each queried architecture undergoes compilation and timed inference to obtain an empirical latency estimate, and is then trained/evaluated to obtain $f(A)$ if (and only if) it passes the latency filter, implementing the early-rejection policy. We treat the resulting feasibility label as ground truth for output validity, and we model measurement stochasticity by repeating latency measurements a small number of times and using either the mean or a high quantile to define $c(A)$; this gives a controlled way to increase or decrease effective α by changing the decision rule.

Training-free metric suite and proxy families. We instantiate $\{M_i\}_{i=1}^{M}$ with a standard suite of training-free signals (e.g., synaptic saliency and Jacobian-based measures) computed at initialization or after a few mini-batches, together with resource proxies (e.g., predicted latency, FLOPs)

when appropriate. We compare (a) single-metric proxies (choosing one $\widetilde{M}_i$), (b) fixed uniform combinations ($w_i \equiv 1/M$), and (c) learned linear combinations as in C-RoBoT. In all cases we normalize metrics to $[0, 1]$ on the candidate pool used in that experiment to ensure commensurability, and we fix K to match the exploitation budget so that $\rho_K^{\mathcal{F}}$ is directly interpretable.

Baselines. We compare against constraint-aware Bayesian optimization baselines that explicitly model feasibility, including methods that maintain independent surrogates for f and each component of c and use a constrained acquisition rule (e.g., feasibility-weighted UCB/expected improvement). We also include (i) random search with feasibility filtering (uniform over $\mathcal{F}$ when known, otherwise over $\widehat{\mathcal{F}}$ with verification), and (ii) simple proxy top-K exploitation without exploration over w (choose w by a small grid or by random sampling, then exploit). When the benchmark supports it, we additionally report an oracle proxy upper bound obtained by selecting w to maximize the *true* $\rho_K^{\mathcal{F}}(S_w, f)$, which operationalizes $\rho_K^{\mathcal{F},*}$.

Evaluation protocol and reporting. Each method receives the same total expensive-query budget T, where an expensive query consists of obtaining $f(A)$ (and, when necessary, measuring $c(A)$). For methods that may propose duplicates, we account in terms of distinct objective evaluations. We repeat each run over multiple random seeds (affecting, e.g., BO initialization, GP hyperparameters, and any stochasticity in objective evaluation), and we report mean and standard error of the best feasible objective value. In tabular settings, we also compute $R_f^{\mathcal{F}}(\widehat{A})$ exactly; in non-tabular settings, we approximate ranks by comparing against a large held-out set of feasible architectures.

Ablation I: constrained vs. unconstrained Precision@K. To isolate the conceptual role of feasibility in alignment, we compare weight selection based on (i) $\rho_K^{\mathcal{F}}(S_w, f)$ (our constrained notion) versus (ii) the analogous unconstrained precision computed on all of $\mathcal{A}$, i.e., replacing $\mathcal{F}$ by $\mathcal{A}$. Empirically, when constraints are active (the feasible fraction is small), unconstrained precision is a poor predictor of exploitation success: it can assign high weight to metrics that surface high-f but systematically infeasible designs. In contrast, optimizing for constrained alignment yields materially higher feasible precision and correspondingly lower realized feasible ranks, consistent with Theorem 1.

Ablation II: constraint-noise and predictor errors. We study graceful degradation by artificially corrupting feasibility in two ways: (a) adding noise to measured constraints before filtering, and (b) training $\widehat{c}$ on a limited labeled subset and varying a safety margin γ to trace the (α, β) trade-off.

We report (i) the fraction of wasted queries due to infeasibility during exploitation, (ii) the achieved best feasible objective, and (iii) an estimate of the effective parameters K_{eff} and ρ_{eff} used in Theorem 3. The observed performance curves track the predicted monotone dependence on $(1-\alpha)$ and $(1-\beta)$: aggressive filtering reduces wasted queries but may exclude top feasible architectures, while conservative filtering admits more infeasible trials but recovers feasibility coverage.

Ablation III: proxy ensemble and acquisition rule. We compare UCB-style and information-directed acquisitions in the exploration phase, and we test the sensitivity to the hypothesis class for S_w (linear over normalized metrics versus small nonlinear alternatives, where permitted). We also quantify the value of exploration itself by comparing against pure exploitation under a fixed w chosen by random search. The principal empirical pattern is that BO over w improves the attained $\rho_K^{\mathcal{F}}$ as T_0 increases, and that the downstream best-feasible objective improves accordingly, matching the qualitative content of Theorem 4.

Summary of findings. Across hardware-aware tabular benchmarks and on-device latency experiments, C-RoBoT consistently improves best-found feasible objective under tight budgets, with the largest gains when feasibility is stringent and naive proxy ranking surfaces infeasible architectures. Moreover, the measured $\rho_K^{\mathcal{F}}(S_{w^\star}, f)$ serves as an informative diagnostic: runs with higher feasible precision are precisely those with lower realized feasible rank, and the gap between best and learned $\rho_K^{\mathcal{F}}$ predicts remaining headroom in exploration.

6 Discussion and Extensions

We conclude by discussing several directions in which the present framework can be extended, and by clarifying the limits of what our rank-style guarantees can (and cannot) certify. Throughout, we keep the central viewpoint that the proxy family $\{S_w\}$ is used to *induce a short list* inside a feasible region, after which expensive evaluations resolve the remaining uncertainty. In this sense, our theory is deliberately "list-centric": it treats the proxy as a retrieval mechanism whose quality is summarized by $\rho_K^{\mathcal{F}}(S_w, f)$ (or its noisy-feasibility analogue), rather than as a calibrated predictor of f.

Multi-constraint tradeoffs and Pareto-front variants. Although our definition of feasibility already allows $d > 1$ constraints (via the componentwise budget $c(A) \le b$), in practice one often cares about *families* of budgets or about Pareto-efficient tradeoffs among resource dimensions. A direct extension is to run C-RoBoT on a grid $\{b^{(m)}\}$ and return a set of architectures

$\{\widehat{A}^{(m)}\}$ approximating the Pareto front of f subject to $c \le b^{(m)}$. The analysis for each fixed $b^{(m)}$ is unchanged, since it only depends on the induced feasible set $\mathcal{F}(b^{(m)})$. More interesting is a "budget-adaptive" variant where we treat the budget as a context and learn $w(b)$, aiming to maximize $\rho_K^{\mathcal{F}(b)}(S_{w(b)}, f)$ uniformly over a budget range. One may also incorporate constraint preferences directly into the proxy, e.g. by augmenting the metric suite with normalized resource signals and learning w so that the proxy ranking is aligned with a chosen scalarization of $(f, -c)$; this converts Pareto-search into the problem of learning a proxy aligned with a user-specified utility while still enforcing hard feasibility at output time.

Beyond hard feasibility: near-feasible and robust feasibility. In deployment, constraints can be stochastic (e.g. latency varying with input and system load), so a hard threshold $c(A) \le b$ may be replaced by a chance constraint $\mathbb{P}(c(A) \le b) \ge 1 - \eta$ or a robust constraint $\sup_{u \in \mathcal{U}} c_u(A) \le b$. Our current protocol (verify feasibility upon query) extends by defining feasibility with respect to an empirical decision rule, such as a high quantile of repeated measurements. Theorem 3 then applies with (α, β) interpreted relative to that rule, but the meaningful choice of the rule becomes part of the design: more conservative rules decrease false-feasible events at the cost of increasing false-infeasible exclusions. A principled direction is to allocate a small measurement budget to reduce feasibility uncertainty, analogously to repeated evaluation for noisy objectives in best-arm identification, and to couple this with a guarantee on the probability of returning an infeasible architecture.

Distribution-free certificates for alignment. Our theory uses $\rho_K^{\mathcal{F}}(S_w, f)$ as a latent quantity, and Theorem 4 bounds the *gap* between the best achievable value and the value achieved by exploration under partial-monitoring conditions. A natural next step is a distribution-free, finite-sample *certificate* for proxy alignment that does not rely on symmetry assumptions and that can be computed online from queried data. Concretely, after observing a set of queried feasible architectures with their f values, one can form lower confidence bounds on the fraction of truly top-K feasible items captured by the proxy top-K list, using standard concentration tools for sampling without replacement or, more simply, conservative binomial bounds when modeling inclusion as randomized. Such a certificate would yield an end-to-end statement of the form: with probability at least $1 - \delta$, exploitation returns an architecture of feasible rank at most $\widehat{r}$, where $\widehat{r}$ is computed from observed data. Analogous certificates can be developed for noisy feasibility by calibrating $\widehat{c}$ and controlling (α, β) via conformal prediction or abstention, thereby turning Theorem 3 into an operational decision procedure.

Implicit and extremely large search spaces. Our exposition assumes a finite set $\mathcal{A}$ with cheap access to all $\widetilde{M}_i(A)$, which is appropriate for tabular NAS benchmarks but not for implicit spaces where N is astronomically large. In such settings, the computational bottleneck is no longer the expensive oracle alone: one must also approximately solve $\arg\max_{A \in \mathcal{F}} S_w(A)$ (or its predicted-feasible counterpart) and identify the proxy top-K. This suggests replacing exact top-K retrieval by approximate maximization using, e.g., beam search, evolutionary operators, or gradient-based optimization over a continuous relaxation, together with a learned feasibility filter. The immediate theoretical change is that $\rho_K^{\mathcal{F}}$ should be defined relative to the *retrieval algorithm* (which may return an approximate top-K), and additional error terms appear that depend on the approximation quality of the retrieval procedure. We view this as unavoidable: when the proxy itself is used to propose candidates, any inability to optimize S_w translates directly into missed feasible opportunities.

Limitations and failure modes. The rank guarantees hinge on two substantive requirements. First, if $\rho_K^{\mathcal{F}}(S_w, f)$ is small for every w in the chosen proxy class, then no query-efficient method restricted to proxy-induced shortlists can reliably find top feasible architectures; this is precisely the content of the minimax lower bound perspective. Second, Theorem 1 uses a symmetry assumption to obtain an exact expectation for the best feasible rank recovered from the intersection set; while this assumption is mild as a modeling device (it encodes lack of additional structure inside the proxy-selected set), it can be violated when proxy scores correlate with f in a highly non-uniform manner within the top-K. In that regime, one should treat the bound as a diagnostic baseline and seek refined analyses that exploit stronger modeling assumptions (e.g. monotone likelihood ratio conditions) or incorporate richer feedback (e.g. multi-fidelity observations) during exploration. Finally, the partial-monitoring condition underlying Theorem 4 is a sufficient condition for regret control in the exploration over w; understanding when it holds for concrete metric suites and weight parameterizations remains an open modeling question, and it motivates empirical observability checks as part of deployment.

7 Conclusion

We have studied a constrained architecture selection problem in which the objective $f(A)$ is expensive to query, feasibility is defined by a budgeted constraint $c(A) \leq b$, and a suite of training-free metrics $\{M_i\}_{i=1}^M$ provides cheap but imperfect information. The central methodological choice is to treat the proxy family $S_w(A) = \sum_i w_i \widetilde{M}_i(A)$ not as a calibrated surrogate for f, but as a *ranking device* whose purpose is to propose a short list *within*

the feasible region; expensive queries are then spent to resolve uncertainty inside that list. This list-centric view leads to guarantees stated directly in terms of constrained ranks, thereby matching the operational reality of neural architecture search under tight evaluation budgets.

Our first contribution is to isolate a single alignment quantity, Feasible-Precision@K,

$$\rho_K^{\mathcal{F}}(S_w, f) = \frac{1}{K}\big|\{A \in \mathcal{F} : R_{S_w}^{\mathcal{F}}(A) \le K \wedge R_f^{\mathcal{F}}(A) \le K\}\big|,$$

which measures how many of the truly top-K feasible architectures are retrieved by the proxy top-K list restricted to $\mathcal{F}$. This quantity is invariant to monotone transformations of S_w and f, and it is agnostic to the scale of the objective; it is therefore an appropriate summary when the ultimate decision is made by evaluating a handful of architectures rather than by trusting proxy scores numerically. In particular, $\rho_K^{\mathcal{F}}$ directly controls the expected feasible rank of the best architecture recovered by a simple exploitation rule.

Concretely, for a fixed w and known feasible set $\mathcal{F}$, Theorem 1 shows that querying f on the proxy top-K feasible architectures and outputting the best queried feasible item yields

$$\mathbb{E}\Big[R_f^{\mathcal{F}}(\widehat{A})\Big] = \frac{K+1}{\rho_K^{\mathcal{F}}(S_w, f)\,K + 1},$$

under a mild symmetry assumption on the distribution of true ranks within the proxy–objective intersection. The form of the bound has an immediate deployment interpretation: when $\rho_K^{\mathcal{F}}(S_w, f)$ is a constant independent of K, the expected feasible rank is $O(1)$ even though we only evaluate K architectures; conversely, when $\rho_K^{\mathcal{F}}$ is small, the rank cannot improve appreciably without either increasing K or enriching the information available during selection.

This dependence is not an artifact of our analysis. Theorem 2 establishes a minimax lower bound (within a natural class of instances parameterized only by ρ) showing that no method restricted to querying within the proxy top-K feasible set can guarantee expected feasible rank $o(1/\rho)$ up to constants. Thus, within the retrieval-then-evaluate paradigm, $\rho_K^{\mathcal{F}}$ is not merely sufficient for controlling performance; it is, in an information-theoretic sense, the relevant obstruction. Practically, this clarifies when training-free metrics can be expected to help: their value is precisely in inducing a nontrivial intersection between proxy and objective top lists inside $\mathcal{F}$.

We further addressed the ubiquitous case in which feasibility is not known a priori and must be inferred from a predictor $\widehat{c}$. Theorem 3 provides a graceful-degradation statement under bounded misclassification rates: a false-feasible rate α limits wasted evaluations on infeasible candidates, and a false-infeasible rate β quantifies how many truly feasible high-performing

candidates are excluded before evaluation. By introducing an effective shortlist size $K_{\text{eff}} \approx (1-\alpha)K$ and an effective precision $\rho_{\text{eff}} \approx (1-\beta)\rho^{\mathcal{F}}_{K_{\text{eff}}}$, we obtain a bound of the same functional form as in the known-feasibility setting. This directly supports deployment workflows in which resource predictors are imperfect but inexpensive: one may tune the predictor (or an abstention rule) to trade off wasted queries against exclusion of feasible candidates, while retaining a quantitative rank guarantee after verification.

Finally, we connected exploitation to an exploration mechanism that learns the proxy combiner w from a small number of objective evaluations. By viewing the choice of w as an action in a constrained partial-monitoring problem and measuring reward via $\rho^{\mathcal{F}}_K(S_w, f)$, Theorem 4 shows that a RoBoT-style Bayesian optimization procedure can identify a weight vector whose feasible precision is near-optimal up to a term of order $q_K K^{-1/3}$ (with high probability under global observability), leading to an end-to-end expected feasible-rank bound of the same type as Theorem 1 but with $\rho^{\mathcal{F}}_K$ replaced by $\rho^{\mathcal{F},*}_K$ minus the exploration error. In operational terms, this provides a principled justification for the two-phase procedure implemented by C-RoBoT: use a limited portion of the query budget to learn which metric combination retrieves good feasible candidates, then spend the remaining budget evaluating the top of the resulting feasible shortlist.

Taken together, our results supply a coherent set of guarantees that map cleanly onto practice. They explain when proxy-guided candidate generation should succeed (large $\rho^{\mathcal{F}}_K$ for some w), how success scales with the evaluation budget K and constraint-prediction errors (α, β), and why the resulting dependence cannot be substantially improved without additional structure or richer feedback. In this sense, the framework provides both a method and a diagnostic: it motivates designing metric suites and weight parameterizations that increase feasible precision, and it clarifies the conditions under which query-efficient constrained architecture selection is provably attainable.