Multi-Objective Hyperparameter Selection via Hypothesis Testing on Reliability Graphs

Amirmohammad Farzaneh **Osvaldo Simeone**
Centre for Intelligent Information Processing Systems
Department of Engineering
King's College London
London, United Kingdom
{amirmohammad.farzaneh,osvaldo.simeone}@kcl.ac.uk

Abstract

The selection of hyperparameters, such as prompt templates in large language models (LLMs), must often strike a balance between reliability and cost. In many cases, structural relationships between the expected reliability levels of the hyperparameters can be inferred from prior information and held-out data – e.g., longer prompt templates may be more detailed and thus more reliable. However, existing hyperparameter selection methods either do not provide formal reliability guarantees or are unable to incorporate structured knowledge in the hyperparameter space. This paper introduces reliability graph-based Pareto testing (RG-PT), a novel multi-objective hyperparameter selection framework that maintains formal reliability guarantees in terms of false discovery rate (FDR), while accounting for known relationships among hyperparameters via a directed acyclic graph. Edges in the graph reflect expected reliability and cost trade-offs among hyperparameters, which are inferred via the Bradley-Terry (BT) ranking model from prior information and held-out data. Experimental evaluations demonstrate that RG-PT significantly outperforms existing methods such as learn-then-test (LTT) and Pareto testing (PT) through a more efficient exploration of the hyperparameter space.

1 Introduction

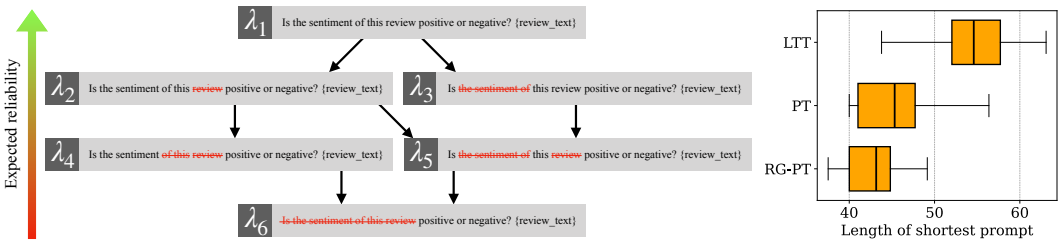
1.1 Context and Motivation

Consider the problem of selecting prompt templates for a large language model (LLM)-based sentiment analysis task [1]. In this setting, the LLM receives a natural language prompt template along with a movie review as input, and the goal is to determine whether the sentiment expressed in the review is positive or negative. As illustrated in Fig. 1, the prompt templates λ are chosen from a set of pre-determined choices Λ, and the objective is to identify prompt templates λ that elicit consistently accurate responses across inputs [1].

Longer prompts often yield more reliable outputs [2]. However, they are also more costly to the end user when pay-per-token billing schemes are applied. This is commonly the case for enterprise software incorporating AI-driven analytics or hosted LLM endpoints via the LLM application programming interface (API) [3]. As exemplified in Fig. 1a, this suggests that the reliability of different prompt templates follows a *directed acyclic graph* (DAG) structure with nodes closer to the roots corresponding to more costly prompt templates with a higher expected reliability.

An ideal prompt engineering scheme would apply *hyperparameter selection* methods capable of selecting prompt templates that are as short as possible while enduring formal reliability guarantees. Existing hyperparameter selection methods, however, either do not meet formal reliability require-

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) (b)

Figure 1: Illustrative example for prompt engineering in LLM-based sentiment analysis: (a) Prompt template candidates in set Λ have expected reliability levels that can be arranged on a reliability graph (RG), so that each parent prompt template is expected to be more reliable than its child prompts; (b) Distribution of the length of the shortest prompt templates identified by LTT [4], PT [5], and the proposed RG-PT for the Stanford Sentiment Treebank dataset [6] (see Sec. 4.1 for details).

ments [7, 8], or cannot incorporate the *structured knowledge* encoded in a graph like the DAG in Fig. 1a.

In particular, *Learn-Then-Test* (LTT) [4] pioneered the use of *multiple hypothesis testing* (MHT) for hyperparameter selection, providing formal guarantees on the reliability of the returned subset of hyperparameters. However, LTT cannot incorporate structured information about relative expected reliability levels of the hyperparameters. *Pareto Testing* (PT) [5] builds on LTT to address multi-objective optimization problems. Specifically, PT infers a global, *linear* ordering over hyperparameters from held-out data based on their expected relative reliability. Thus, PT cannot account for more complex structured relationships between candidate hyperparameters as in the example of Fig. 1a.

This paper proposes a novel framework, *reliability graph-based PT* (RG-PT), that systematically captures and exploits interdependencies between hyperparameter configurations for hyperparameter selection. RG-PT models the hyperparameter space as a DAG, termed the *reliability graph* (RG). In an RG, nodes correspond to candidate hyperparameter configurations, such as prompt templates, and edges encode reliability relationships. If there is an edge from a hyperparameter λ_i to another λ_j in the RG, then λ_i is expected to be more reliable than λ_j.

For the running example of prompt design, as shown in Fig. 1b (detailed in Sec. 4), RG-PT is seen experimentally to select shorter prompt templates than LTT and PT, while still satisfying formal guarantees in terms of *false discovery rate* (FDR). The FDR measures the fraction of unreliable prompt templates returned by the hyperparameter selection scheme. This advantage of RG-PT stems from its ability to encode rich structural relationships among prompt templates, so as to explore the hyperparameter space more efficiently during the MHT procedure.

1.2 Further Related Work

Hyperparameter Selection: State-of-the-art techniques for hyperparameter optimization, such as Bayesian optimization (BO) [9], multi-armed bandits (MAB) [8], and gradient-based optimization [10], provide satisfactory empirical performance, but they lack post-selection error control such as FDR guarantees. Under idealistic assumptions such as the correct specification of kernel models, BO and MAB can provide bounds on cumulative or simple average regret [11, 12, 13]. However, these bounds do not offer the type of assumption-free guarantees on the risk of the selected parameters that can be instead provided by FDR-controlling methods. LTT addresses this gap by incorporating MHT in the hyperparameter selection process [4]. Extensions of LTT are surveyed in [14]. Note that, while reference [4] mentions the possible use of graph-based approaches, these are limited to fixed user-defined graphs or linear directed graphs (chains).

Multi-Objective Optimization: Modern AI applications often require optimizing multiple objectives such as accuracy, efficiency, and cost. This can be formally done through Pareto optimization to identify all the feasible trade-off points among different objectives [15]. PT [5] extends LTT to settings with multiple objectives, inferring a linear testing order in the hyperparameter space based on held-out data.

1.3 Main Contributions

The main contributions of this paper are as follows.

Methodology: We propose RG-PT, a novel multi-objective hyperparameter selection framework that systematically infers and utilizes interdependencies among the expected reliability levels of candidate hyperparameter configurations. RG-PT first constructs a DAG, known as RG, based on prior information and held-out data via the Bradley-Terry (BT) ranking model [16] and the non-negative Lasso [17]. Then, it applies MHT-based hyperparameter selection on the RG by following DAGGER, a graphical testing method introduced in [18].

Applications: We demonstrate the effectiveness of RG-PT through experiments in LLM prompt engineering, sequence-to-sequence translation, object detection, image classification, and telecommunications, highlighting its advantages over existing methods.

The rest of this paper is organized as follows. In Sec. 2, we define the multi-objective hyperparameter selection problem. Sec. 3 details the proposed RG-PT framework. Experimental results are presented in Sec. 4. We conclude the paper in Sec. 5.

2 Multi-Objective Hyperparameter Selection

In this section, we define the problem of multi-objective hyperparameter selection, and we show how this problem can be formulated via MHT by following references [4, 5].

2.1 Problem Definition

Consider a predefined discrete and finite set Λ of hyperparameters λ, which govern the performance of a machine learning model such as an LLM (see, e.g., [19]). The discrete set Λ is populated in a preliminary pre-selection step [20, 21, 22] using methods including LLM judges [23] and continuous optimizers like Bayesian optimization [7] and Hyperband [8].

In a multi-objective setting with L risk functions, when tested on a data point Z, a hyperparameter λ attains risk values $r_l(Z, \lambda)$ for $l = 1, \ldots, L$. The risk functions $r_l(Z, \lambda)$ are negatively oriented, meaning that lower risk values correspond to better-performing hyperparameters. The risks are normalized within the range $0 \leq r_l(Z, \lambda) \leq 1$. Furthermore, for each performance criterion $l = 1, \ldots, L$, the *average risk function* is defined as

$$R_l(\lambda) = \mathbb{E}_Z \left[r_l(Z, \lambda) \right], \tag{1}$$

where the expectation is taken over the distribution P_Z of the data Z.

We partition the set of L risk functions into the following two groups:

1. *Reliability risk functions:* The first set of risk functions $\{R_l(\lambda)\}_{l=1}^{L_c}$ must be controlled via the choice of the hyperparameter λ. In particular, a hyperparameter configuration λ is said to be *reliable* if it guarantees the constraints

$$R_l(\lambda) \leq \alpha_l \quad \text{for all} \quad l = 1, \ldots, L_c. \tag{2}$$

2. *Auxiliary risk functions:* The second set of performance measures $\{R_l(\lambda)\}_{l=L_c+1}^{L}$ are unconstrained, and are optimized in a best-effort fashion via the selection of the hyperparameter λ.

Accordingly, the goal of hyperparameter selection is defined as the *multi-objective problem*

$$\begin{aligned} &\min_{\lambda \in \Lambda} \{R_{L_c+1}(\lambda), R_{L_c+2}(\lambda), \ldots, R_L(\lambda)\} \\ &\text{subject to } R_l(\lambda) < \alpha_l \text{ for all } 1 \leq l \leq L_c, \end{aligned} \tag{3}$$

which targets the minimization of the auxiliary risk functions $\{R_l(\lambda)\}_{l=L_c+1}^{L}$ under constraints on the reliability risk functions $\{R_l(\lambda)\}_{l=1}^{L_c}$. For example, in the setting of Fig. 1, we wish to minimize the prompt length, while ensuring a constraint on the accuracy of the LLM's outputs.

Solving a multi-objective optimization problem such as (3) ideally entails identifying the entire Pareto front of dominant solutions $\lambda \in \Lambda$, or at least obtaining specific solutions corresponding to scalar

criteria [24, 25]. However, the problem (3) cannot be directly addressed since the data distribution P_Z is assumed to be unknown. Instead, we assume to have access to i.i.d. data $\mathcal{Z} = \{Z_j\}_{j=1}^n$ drawn from the unknown data distribution P_Z. For any data subset $\widetilde{\mathcal{Z}} \subseteq \mathcal{Z}$, the empirical estimate of risk function $R_l(\lambda)$ can be obtained as

$$\hat{R}_l(\lambda|\widetilde{\mathcal{Z}}) = \frac{1}{|\widetilde{\mathcal{Z}}|} \sum_{Z \in \widetilde{\mathcal{Z}}} r_l(Z, \lambda). \tag{4}$$

2.2 Hyperparameter Selection as Multiple Hypothesis Testing

As proposed in [4], hyperparameter selection can be formally addressed as an MHT problem. Accordingly, for each hyperparameter $\lambda \in \Lambda$, we define the null hypothesis $\mathcal{H}_\lambda$ that hyperparameter λ violates the reliability constraints (2), i.e.,

$$\mathcal{H}_\lambda : \text{ there exists } l \in \{1, \dots, L_c\} \text{ such that } R_l(\lambda) > \alpha_l. \tag{5}$$

Thus, rejecting the null hypothesis $\mathcal{H}_\lambda$ implies that hyperparameter λ meets all the constraints in (2). A rejection is also referred to as a *discovery*. A discovery is *false* if the selected hyperparameter λ, i.e., the hyperparameter for which the corresponding null hypothesis $\mathcal{H}_\lambda$ is rejected, is actually unreliable, satisfying the null hypothesis $\mathcal{H}_\lambda$.

Given a dataset $\widetilde{\mathcal{Z}} \subseteq \mathcal{Z}$, evidence against the reliability of each candidate hyperparameter $\lambda \in \Lambda$ can be measured by a p-value derived by applying Hoeffding's inequality [26] to the empirical mean of bounded losses. Specifically, for each reliability risk function $l = 1, \dots, L_c$, we define the statistic [4]

$$p_{\lambda,l}(\widetilde{\mathcal{Z}}) = \exp(-2|\widetilde{\mathcal{Z}}|(\alpha_l - \hat{R}_l(\lambda|\widetilde{\mathcal{Z}}))_+^2). \tag{6}$$

Combining the statistic (6) across all L_c risk functions via the maximum

$$p_\lambda(\widetilde{\mathcal{Z}}) = \max_{1 \le l \le L_c} p_{\lambda,l}(\widetilde{\mathcal{Z}}). \tag{7}$$

yields a valid p-value for the null hypothesis $\mathcal{H}_\lambda$ in (5) [5, Appendix A.2]. Therefore, thresholding the p-value $p_\lambda(\widetilde{\mathcal{Z}})$ yields a reliability test that controls type-I error with finite-sample guarantees.

Furthermore, by formulating hyperparameter selection as an MHT problem, we can leverage statistical tools that guarantee *false discovery rate* (FDR) requirements [27]. To elaborate, define as $\hat{\Lambda}_\mathcal{Z}$ the subset of hyperparameters selected by an MHT mechanism. The FDR is defined as the expected proportion of unreliable hyperparameters in set $\hat{\Lambda}_\mathcal{Z}$. Therefore, controlling the FDR amounts to finding a subset $\hat{\Lambda}_\mathcal{Z} \subseteq \Lambda$ that satisfies the inequality

$$\mathbb{E}_Z \left[\frac{\sum_{\lambda \in \hat{\Lambda}_\mathcal{Z}} \mathbf{1}\{R_l(\lambda) > \alpha_l \text{ for any } l = 1, \dots, L_c\}}{\max(|\hat{\Lambda}_\mathcal{Z}|, 1)} \right] \le \delta, \tag{8}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function, and the expectation is taken over the unknown data distribution P_Z. The FDR constraint (8) ensures that the average fraction of unreliable hyperparameters in set $\hat{\Lambda}_\mathcal{Z}$ is upper bounded by δ.

3 Reliability Graph-Based Pareto Testing

In this section, we introduce RG-PT, a novel hyperparameter selection strategy based on MHT that adopts a testing schedule based on the novel concept of RG.

3.1 Overview

The design of RG-PT starts from the observation that the reliability of some hyperparameters can be highly predictive of the reliability of other hyperparameters, and that this structure can be encoded by a DAG as in Fig. 1a. By incorporating the DAG structure in the MHT process of hyperparameter selection, RG-PT supports a more efficient hyperparameter selection procedure, while meeting formal reliability constraints in terms of the FDR (8).

As illustrated in Fig. 2, using a partition $\mathcal{Z} = \{\mathcal{Z}_{\text{OPT}}, \mathcal{Z}_{\text{MHT}}\}$ of the data set $\mathcal{Z}$, RG-PT applies the following steps:

① *Estimating the Pareto front for all risk measures:* Following PT [5], RG-PT uses the dataset $\mathcal{Z}_{\text{OPT}}$ to identify the subset $\Lambda_{\text{OPT}} \subseteq \Lambda$ of hyperparameters that are on the Pareto front of the space of estimated risk measures $\{\hat{R}_l(\lambda|\mathcal{Z}_{\text{OPT}})\}_{l=1}^L$. This is done by addressing the multi-objective optimization problem (3) with the estimates $\{\hat{R}_l(\lambda|\mathcal{Z}_{\text{OPT}})\}_{l=1}^L$ in lieu of the true risks $\{R_l(\lambda|\mathcal{Z}_{\text{OPT}})\}_{l=1}^L$ using any suitable multi-objective optimization algorithm [5].

② *Learning the reliability graph:* Rather than ordering the hyperparameters in subset Λ_{OPT} in a linear sequence as done by PT [5], RG-PT creates an RG, with nodes given by the hyperparameters in subset Λ_{OPT}. This is done by following the principle that hyperparameters $\lambda \in \Lambda_{\text{OPT}}$ whose reliability levels are predictive of the reliability levels of other hyperparameters $\Lambda' \subset \Lambda_{\text{OPT}}$ should be tested before the hyperparameters Λ'. As detailed in Sec. 3.2, the RG construction leverages the BT model to incorporate prior information and the non-negative Lasso to determine the links in the graph.

③ *FDR-controlling MHT:* Using the data set $\mathcal{Z}_{\text{MHT}}$, FDR-controlling MHT is carried out by incorporating the structure encoded by the RG. As explained in Sec. 3.2, this is done by using DAGGER [18], returning the subset $\hat{\Lambda}_{\mathcal{Z}} \subseteq \Lambda_{\text{OPT}}$.

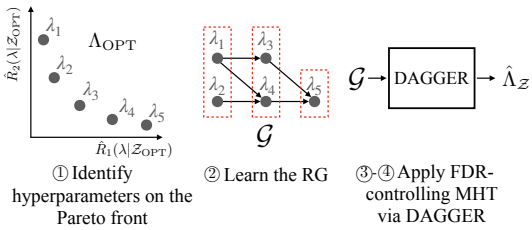


Figure 2: Illustration of the main steps of RG-PT: ① Estimate the hyperparameters Λ_{OPT} lying on the Pareto front pf problem (2); ② Build the RG over the selected hyperparameters Λ_{OPT}; ③ Apply an FDR-controlling MHT procedure, DAGGER, to the RG to obtain the selected set $\hat{\Lambda}_{\mathcal{Z}} \subseteq \Lambda_{\text{OPT}}$.

④ *Addressing the multi-objective optimization problem*: Given the subset $\hat{\Lambda}_{\mathcal{Z}}$, RG-PT addresses the problem

$$\min_{\lambda \in \hat{\Lambda}_{\mathcal{Z}}} \{\hat{R}_{L_c+1}(\lambda|\mathcal{Z}_{\text{OPT}}), \ldots, \hat{R}_L(\lambda|\mathcal{Z}_{\text{OPT}})\}, \tag{9}$$

where the auxiliary risk functions $R_l(\lambda)$ in (3) are replaced with the corresponding empirical estimates (4) obtained with data set $\mathcal{Z}_{\text{OPT}}$.

3.2 Learning the Reliability Graph

After obtaining the estimated Pareto front Λ_{OPT}, RG-PT constructs an RG to encode the expected relationships between the reliability levels attained by the candidate hyperparameters in the set Λ_{OPT}. The RG is a DAG in which each node represents a hyperparameter $\lambda \in \Lambda_{\text{OPT}}$, and edges are directed to describe a reliability hierarchy. Specifically, edges encode the expectation that parent nodes are predictive of the reliability of their child nodes.

Accordingly, starting from the nodes with no parents and following the direction of the edges in the RG, one encounters hyperparameters that are estimated to be increasingly unreliable. Generalizing the linear ordering assumed by PT, the DAG structure adopted by RG-PT can thus assign the same expected reliability ranking to multiple hyperparameters. Specifically, all the hyperparameters at the same depth in the DAG are deemed to have the same relative reliability level. This partial ordering enables a more efficient exploration of the hyperparameter space.

In order to learn the RG, RG-PT leverages the data set $\mathcal{Z}_{\text{OPT}}$, as well as, possibly, prior information about the relative reliability of pairs of hyperparameters. This is done via the following two steps:

1. *Depth assignment:* The hyperparameters in set Λ_{OPT} are ranked in terms of their expected reliability, allowing for multiple hyperparameters to be ranked equally. This step thus assigns a depth level d in the DAG to each hyperparameter $\lambda \in \Lambda_{\text{OPT}}$. As explained in Sec. 3.2.1, this is done by leveraging the BT ranking model [16].

2. *Learning the directed edges:* Given any hyperparameter λ at some depth d, RG-PT selects a subset of hyperparameters at the previous depth level $d-1$ to serve as parents of the hyperparameter λ. As

specified in Sec. 3.2.2, this is done by choosing the hyperparameters at depth $d-1$ that are most predictive of the reliability level of hyperparameter λ via the non-negative Lasso [17].

3.2.1 Depth Assignment

Fix a number $D \leq |\Lambda_{\text{OPT}}|$ of levels for the DAG. With $D = |\Lambda_{\text{OPT}}|$, one can assign each hyperparameter $\lambda \in \Lambda_{\text{OPT}}$ a distinct level, yielding a global ordering and recovering PT. Conversely, with $D = 1$, all hyperparameters $\lambda \in \Lambda_{\text{OPT}}$ are assigned to the same level. The setting of interest is thus $1 < D < |\Lambda_{\text{OPT}}|$, which is assumed from now on. A procedure for choosing a suitable value for depth D is presented in Appendix F.

Depth assignment is carried out by first obtaining a score $s(\lambda)$ for all hyperparameters $\lambda \in \Lambda_{\text{OPT}}$ using the data set $\mathcal{Z}_{\text{OPT}}$, and then partitioning the set Λ_{OPT} into D clusters according to the obtained scores.

To compute the scores $s(\lambda)$ for hyperparameters $\lambda \in \Lambda_{\text{OPT}}$, we use the BT model [16]. The BT model converts pairwise counts w_{ij} for all pairs of hyperparameters λ_i and λ_j in subset Λ_{OPT} into per-hyperparameter scores $s(\lambda)$ for all $\lambda \in \Lambda_{\text{OPT}}$. The pairwise count w_{ij} measures the number of times that hyperparameter λ_i was found to be more reliable than hyperparameter λ_j. In RG-PT, we propose to evaluate the pairwise counts w_{ij} by leveraging two sources of information:

• *Prior information:* Prior information is encoded by pairwise probabilities $0 \leq \eta_{ij} \leq 1$ for each pair of hyperparameters (λ_i, λ_j) in Λ_{OPT}. This probability reflects the expected rate at which hyperparameter λ_i is observed to be more reliable than hyperparameter λ_j. Note that we have $\eta_{ji} = 1 - \eta_{ij}$. The strength of the prior information is determined via a pseudocount variable n_p as in the standard categorical-Dirichlet model [28]. A larger pseudocount n_p indicates a stronger trust in the prior information. Importantly, the statistical guarantees of RG-PT do not depend on the choice of the prior probabilities $\{\eta_{ij}\}$ and pseudocount n_p, which can, however, improve the capacity of RG-PT to optimize the auxiliary risk functions. Note that in the absence of prior information, one can set $n_p = 0$, and a principled approach to choosing n_p is discussed in Appendix F.

• *Data:* Using the p-values $p_{\lambda_i}(\mathcal{Z}_{\text{OPT}})$ and $p_{\lambda_j}(\mathcal{Z}_{\text{OPT}})$ in (7), we construct the pairwise comparison score

$$p_{ij}(\mathcal{Z}_{\text{OPT}}) = \frac{p_{\lambda_i}(\mathcal{Z}_{\text{OPT}})}{p_{\lambda_i}(\mathcal{Z}_{\text{OPT}}) + p_{\lambda_j}(\mathcal{Z}_{\text{OPT}})}, \tag{10}$$

which satisfies $p_{ij}(\mathcal{Z}_{\text{OPT}}) = 1 - p_{ji}(\mathcal{Z}_{\text{OPT}})$. This construction of a comparison score is an instance of the broader class of models used for paired comparison data [29]. Other possible choices include the Thurstone model [30] $p_{ij}(\mathcal{Z}_{\text{OPT}}) = \Phi(p_{\lambda_i}(\mathcal{Z}_{\text{OPT}}) - p_{\lambda_j}(\mathcal{Z}_{\text{OPT}}))$, where $\Phi(\cdot)$ is the normal CDF, and the BT model [31] $p_{ij}(\mathcal{Z}_{\text{OPT}}) = 1/(1 + e^{-(p_{\lambda_i}(\mathcal{Z}_{\text{OPT}}) - p_{\lambda_j}(\mathcal{Z}_{\text{OPT}}))})$. While we adopt (10), alternative comparison models could be used without affecting the theoretical guarantees of RG-PT.

Overall, the pairwise count w_{ij} is obtained by combining prior information and data as

$$w_{ij} = |\mathcal{Z}_{\text{OPT}}| p_{ij}(\mathcal{Z}_{\text{OPT}}) + n_p \eta_{ij}, \tag{11}$$

so that the relative strength of the prior information in (11) depends on the ratio $n_p/|\mathcal{Z}_{\text{OPT}}|$ between the pseudocount n_p and the number of data points $|\mathcal{Z}_{\text{OPT}}|$. The pairwise count w_{ij} in (11) can be viewed as a smoothed estimate of the reliability of hyperparameter λ_i relative to hyperparameter λ_j, where prior and empirical counts combine additively to form the effective posterior sufficient statistic. The weighting mimics a Bayesian update, ensuring that both prior knowledge and observed evidence contribute to the final comparison score [32, 33, 34].

Using the BT model, the scores $s(\lambda_i)$ for all hyperparameters $\lambda_i \in \Lambda_{\text{OPT}}$ are obtained by maximizing the log-likelihood [16]

$$\sum_{i=1}^{|\Lambda_{\text{OPT}}|} \sum_{j=1}^{|\Lambda_{\text{OPT}}|} \left(w_{ij} \ln \left(\frac{s(\lambda_i)}{s(\lambda_i) + s(\lambda_j)} \right) \right), \tag{12}$$

with $w_{ii} = 0$ for all $1 \leq i \leq |\Lambda_{\text{OPT}}|$. With this design, in the absence of prior information ($n_p = 0$), the BT model reduces to assigning scores directly proportional to the p-values $p_\lambda(\mathcal{Z}_{\text{OPT}})$.

After obtaining the scores $s(\lambda_i)$ for all $1 \leq i \leq |\Lambda_{\text{OPT}}|$, depth assignment is done via clustering, producing disjoint subsets $\Lambda_1, \ldots, \Lambda_D$. The cluster $\Lambda_1 \subseteq \Lambda_{\text{OPT}}$ contains the hyperparameters with

the highest expected reliability, and the remaining clusters $\Lambda_2, \ldots, \Lambda_D$ are sorted in descending order of expected reliability. All hyperparameters in cluster Λ_d are assigned depth level d.

Clustering can be implemented by using methods such as K-means or hierarchical clustering. We recommend using agglomerative hierarchical clustering, which begins with each hyperparameter in its own cluster and iteratively merges clusters [35].

3.2.2 Learning the Directed Edges

Having obtained the clusters $\Lambda_1, \ldots, \Lambda_D$, the RG is constructed by: (*i*) including one node for each hyperparameter $\lambda \in \Lambda_{\text{OPT}}$; and (*ii*) selecting for each hyperparameter $\lambda \in \Lambda_d$ at depth level d a subset of hyperparameters in cluster Λ_{d-1} to serve as parents of λ for all depth levels $2 \leq d \leq K$. The resulting directed edges are intended to represent inferred reliability dependencies.

To this end, we implement feature selection via the non-negative Lasso [17]. Further discussions on the choice of this standard algorithm can be found in Appendix G. Specifically, given hyperparameter $\lambda \in \Lambda_d$, we consider the problem of predicting the risks $\{r_l(Z, \lambda)\}_{l=1}^L$ from the risks $\{r_l(Z, \lambda')\}_{l=1}^L$ attained by the hyperparameters $\lambda \in \Lambda_{d-1}$ at the previous depth level. The use of non-negative Lasso regression ensures that only positive correlations are represented in the DAG, preserving hierarchical reliability relationships between parent and child nodes.

Formally, using the data set $\mathcal{Z}_{\text{OPT}}$, for each hyperparameter $\lambda \in \Lambda_d$ we address the problem

$$\min_{\beta \geq 0} \sum_{Z \in \mathcal{Z}_{\text{OPT}}} \left\| r(Z, \lambda) - \sum_{\lambda' \in \Lambda_{d-1}} \beta_{\lambda'} r(Z, \lambda') \right\|_2^2 + \tau \sum_{\lambda' \in \Lambda_{d-1}} \beta_{\lambda'}, \tag{13}$$

where $\beta = \{\beta_{\lambda'}\}_{\lambda' \in \Lambda_{d-1}}$ is the vector of non-negative regression coefficients corresponding to each potential parent node $\lambda' \in \Lambda_{d-1}$; $r(Z, \lambda)$ is the vector containing the values $\{r_l(Z, \lambda)\}_{l=1}^{L_c}$; $\|\cdot\|_2$ represents the ℓ_2 norm; and $\tau > 0$ is a regularization parameter that controls the degree of sparsity in the solution. The procedure for choosing a suitable value for the regularization parameter τ is discussed in Appendix F. After solving the convex problem (13), only the hyperparameters $\lambda' \in \Lambda_{j-1}$ for which the corresponding coefficient $\beta_{\lambda'}$ are positive are selected as parent nodes of hyperparameter λ. As for the variables $(n_p, \{\eta_{ij}\})$ in the BT likelihood (12), the choice of the parameter τ does not affect the validity properties of RG-PT.

3.3 FDR-Controlling Multiple Hypothesis Testing

Given the obtained RG, RG-PT performs MHT via DAGGER [18], an FDR-controlling algorithm that operates on DAGs. DAGGER begins testing at the root nodes of the RG, i.e., at the hyperparameters in cluster Λ_1, and proceeds with the clusters $\Lambda_2, \Lambda_3, \ldots, \Lambda_D$, guided by the outcomes of prior tests. If a hyperparameter is deemed unreliable, none of its descendants are tested.

The test for each hyperparameter $\lambda_i \in \Lambda_{\text{OPT}}$ detects λ_i as reliable if the p-value $p_{\lambda_i}(\mathcal{Z}_{\text{MHT}})$ in (7) is no larger than a threshold δ_i, i.e., if $p_{\lambda_i}(\mathcal{Z}_{\text{MHT}}) \leq \delta_i$. The testing level δ_i is determined by DAGGER based on several factors, including the overall target FDR level δ in constraint (8), the number of reliable hyperparameters identified among those tested prior to λ_i, and the structure of the graph rooted at λ_i. We refer the reader to Appendix B and to [18] for details.

An algorithmic overview of RG-PT is provided in Appendix D. The following proposition states the theoretical guarantees provided by RG-PT.

Proposition 3.1. *The set $\hat{\Lambda}_{\mathcal{Z}}$ of hyperparameters returned by RG-PT controls the FDR below the pre-specified threshold δ as in (8).*

Proof. RG-PT applies the DAGGER algorithm [18] to a DAG over the candidate hyperparameters, using valid p-values defined in (7). Since DAGGER controls the FDR at level δ for *any* DAG structure when supplied with valid p-values under arbitrary dependence (see Appendix B), the result follows directly. □

4 Experiments

In this section, we evaluate the proposed RG-PT hyperparameter selection strategy on a prompt engineering problem [1] and a sequence-to-sequence translation task [36]. Additional experiments including on object detection [4] and telecommunications [37] can be found in Appendix H[1].

Throughout the experiments, we adopt as benchmarks LTT [4] and PT [5]. To the best of our knowledge, LTT and PT are the only existing hyperparameter selection methods that guarantee statistical validity in the sense of the FDR constraint (8), justifying this choice. LTT is implemented by applying Benjamini-Hochberg (BH) [38] as the FDR-controlling algorithm, while PT follows [5], with the caveat that FDR-controlling FST [39] is used in lieu of an FWER-controlling scheme. LTT uses the entire calibration data set $\mathcal{Z}$ to evaluate the p-values used in BH, while PT and RG-PT partition $\mathcal{Z}$ into data sets $\mathcal{Z}_{\text{OPT}}$ and $\mathcal{Z}_{\text{MHT}}$.

Across the experiments, following common practice in the MHT literature [38] including the LTT and PT papers, we set $\delta = 0.1$ in (8), corresponding to a confidence level of 90% for the reliability of the selected hyperparameters. Additionally, the choice for the threshold values α_l, $1 \leq l \leq L_C$, in (8) depends on the downstream application. In our experiments, we selected these thresholds via preliminary validation sweeps to identify values that effectively separate low-risk configurations, while maintaining adequate statistical power.

4.1 Reliable Prompt Engineering

Problem Setup. In this experiment, we focus on prompt engineering for the following three tasks from the instruction induction data set [1]: 1. *Sentiment analysis:* In this task, based on the Stanford Sentiment Treebank dataset [6], each data point $Z = (X, Y)$ encompasses a movie review X, and the corresponding sentiment $Y \in \{\text{positive}, \text{negative}\}$. 2. *Sentence similarity:* In this task, based on the Semantic Textual Similarity Benchmark dataset [40], each data point $Z = (X, Y)$ comprises two sentences as input X, along with a semantic similarity label $Y \in \{0, \ldots, 5\}$. 3. *Word in context:* In this task, based on the Word-in-Context dataset [41], each data point $Z = (X, Y)$ consists of a target word and two context sentences as input X, paired with a binary label $Y \in \{\text{same}, \text{not same}\}$ indicating whether the target word shares the same meaning across both contexts.

For each task, we use 1000 examples each for the data sets $\mathcal{Z}_{\text{OPT}}$ and $\mathcal{Z}_{\text{MHT}}$, as well as for the test data set. Furthermore, following the forward generation mode detailed in [42], we use the LLaMA3-70B-Instruct model [43] to generate a set $\Lambda = \{\lambda_1, \ldots, \lambda_{100}\}$ of distinct instruction-style prompt templates for each task.

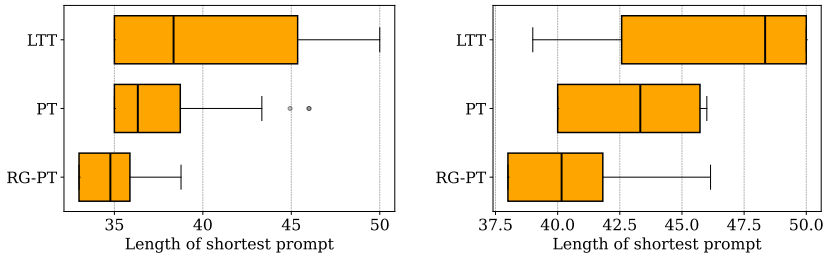
Given a prompt λ and an input X, the smaller LLaMA3-8B-Instruct model [44] f is queried with the concatenated input $[\lambda, X]$, producing the output $f([\lambda, X])$. For each input-output pair $Z = (X, Y)$, a task-specific 0-1 prompt loss is calculated as $r_{\text{prompt}}(Z, \lambda) = l(f([\lambda, X]), Y) \in \{0, 1\}$, indicating whether the task was performed correctly. The objective is to find prompts in set Λ that control the average prompt loss $R_{\text{prompt}}(\lambda) = \mathbb{E}_{\mathcal{Z}}\left[r_{\text{prompt}}(Z, \lambda)\right]$ below a target level of $\alpha = 0.2$, while minimizing the average prompt length. For this selection, we wish to control the FDR in (8) at level $\delta = 0.1$.

Prior Information via LLM-as-a-Judge. To incorporate prior structure into the reliability graph, we adopt the LLM-as-a-judge framework [23]. For each pair of prompts $\lambda_i, \lambda_j \in \Lambda_{\text{OPT}}$, we query the GPT-4 Turbo (gpt-4-0125-preview, temperature $= 0$, max tokens $= 10$) model [45] with a task-specific prompt template to assess which instruction is more likely to elicit a correct or helpful response. We perform this comparison once per hyperparameter pair $\lambda_i, \lambda_j \in \Lambda_{\text{OPT}}$, and define the binary pairwise preference $\eta_{ij} = 1$ if GPT-4 Turbo selects η_{ij} as more reliable, and $\eta_{ij} = 0$ otherwise. For each reliable set of prompts returned by each method, we choose the shortest prompt as the final hyperparameter choice. We set the pseudocount n_p to 1,000.

Results. For each task, we plot the distribution of the length of the shortest reliable prompt for 100 independent runs over random splits of the data set, for LTT, PT, and RG-PT in Fig. 1b, Fig. 3a, and Fig. 3b for the sentiment analysis task, the sentence similarity task, and the word in context task, respectively. The figures demonstrate that RG-PT identifies more concise instructions compared to

[1]The code for the experiments can be found at the anonymous Github repository https://anonymous.4open.science/r/RG-PT-EF3A/

LTT and PT by leveraging the prior information provided by the LLM judge. Note that all schemes satisfy the FDR constraint (not shown). For instance, RG-PT achieved an average FDR of 0.089, 0.095, and 0.092 for the the sentiment analysis, the sentence similarity, and the word in context tasks, respectively.



(a) (b)

Figure 3: Distribution of the length of the shortest prompt templates identified by LTT [4], PT [5], and the proposed RG-PT for (a) the Semantic Textual Similarity Benchmark dataset [40] and (b) the Word-in-Context dataset [41].

An ablation study on the effect of the RG depth D, as well as the effect of a misspecified prior information for this experiment can be found in Appendix H.3.

4.2 Sequence-to-Sequence Language Translation

We consider a sequence-to-sequence language translation task on the WMT16 Romanian-English dataset [46], using BLEU [47] and ROUGE-L [48] as the objectives. Following [36], the dataset is preprocessed with SentencePiece tokenization [49], and an LSTM-based encoder-decoder is trained. Two key hyperparameters are considered:

1. The hyperparameter ρ controls the *sparsity* of the output distribution using Entmax [50], transitioning between a dense output with softmax ($\rho = 1$) and sparsemax ($\rho = 2$) [51].

2. The *Fenchel-Young label smoothing strength* ϵ is a training regularization hyperparameter that determines the extent to which one-hot targets are mixed with uniform noise based on Fenchel-Young losses [36]. Accordingly, the original one-hot targets are assigned weight $1 - \epsilon$, while the uniform distribution over all possible classes is assigned the weight ϵ.

To create the initial candidate set Λ, we selected hyperparameters over a grid of 32 combinations, using 8 logarithmically spaced values in the interval $[1, 2]$ for ρ, and values in the set $\{0.0, 0.01, 0.05, 0.1\}$ for ϵ. This selection is in line with reference [36].

To set up RG-PT, we leveraged the prior knowledge that less sparse settings may be more reliable than their sparser counterparts. Specifically, for any two hyperparameters $\lambda_i = (\rho_i, \epsilon_i)$ and $\lambda_j = (\rho_j, \epsilon_j)$ where $\rho_i < \rho_j$, we assigned a prior probability $\eta_{ij} = 1$, reflecting this prior reliability assumption. Furthermore, the pseudocount parameter n_p, which determines the weight of prior information, was set to be equal to $|\mathcal{Z}_{\text{OPT}}|$.

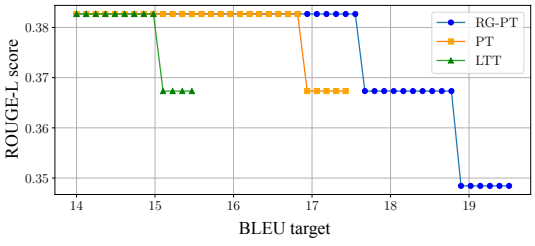


Figure 4: Test ROUGE-L scores achieved by LTT, PT, and RG-PT methods as a function of the target reliability value for the BLEU score.

Denote as $R_{\text{BLEU}}(\lambda)$ and $R_{\text{ROUGE}}(\lambda)$ the average BLEU and ROUGE-L scores, respectively, obtained for a given hyperparameter configuration $\lambda = (\rho, \epsilon)$. The goal is to guarantee the BLEU score to be above a threshold α, while maximizing the ROUGE-L score. This amounts to an instance of problem (3), with $L = 2$, $L_c = 1$, $R_1(\lambda) = -R_{\text{BLEU}}(\lambda)$, $R_2(\lambda) = -R_{\text{ROUGE}}(\lambda)$, and $\delta = 0.1$. After MHT, all the schemes choose the hyperparameter $\lambda \in \hat{\Lambda}_{\mathcal{Z}}$ with the maximum

estimated value $R_{\text{ROUGE}}(\lambda)$, i.e., minimum $\hat{R}_2(\lambda|\mathcal{Z})$ for LTT, and minimum $\hat{R}_2(\lambda|\mathcal{Z}_{\text{OPT}})$ for PT and RG-PT. The data set sizes are $|\mathcal{Z}| = 400$, $|\mathcal{Z}_{\text{OPT}}| = 200$, and $|\mathcal{Z}_{\text{MHT}}| = 200$.

Fig. 4 illustrates the ROUGE-L score achieved on the test data by each calibration method, plotted against the target value for the BLEU score. The results demonstrate that RG-PT consistently maintains higher ROUGE-L scores, even under stricter requirements for the BLEU score. This highlights RG-PT's advantage in effectively exploring the hyperparameter space, enabling a more efficient testing procedure and identifying superior hyperparameter configurations that still statistically satisfy the desired conditions on the risk functions.

To assess the practical implications of the different theoretical guarantees offered by BO [9] and MAB [8] methods, we implemented both approaches on this task, and compared their achieved FDR with LTT, PT, and RG-PT. For BO, we employed the `scikit-optimize` implementation with the Upper Confidence Bound (UCB) acquisition function [9, 11]. For the MAB baseline, we used Thompson Sampling, a canonical stochastic bandit algorithm with Bayesian regret guarantees [52, 53]. We observed that only RG-PT, LTT, and PT satisfied the target FDR level $\alpha = 0.1$, while the achieved FDR on the test dataset for BO and MAB was 0.14 and 0.21, respectively.

5 Conclusion, Limitations, and Future Work

In this paper, we have introduced RG-PT, a novel framework for multi-objective hyperparameter selection that integrates MHT with the concept of RGs to capture interdependencies among candidate hyperparameters. By leveraging a DAG structure informed by prior knowledge and data, RG-PT enables a more powerful parallel testing of hyperparameters compared to the state-of-the-art methods LTT and PT. RG-PT provides statistical guarantees through FDR control, while expanding the space of reliable hyperparameter configurations, leading to a superior optimization of auxiliary objectives.

Limitations of this work include the exclusive applicability to settings with discrete hyperparameter spaces and the lack of theoretical results on the power and sample efficiency of the method. Future work may focus on optimizing the RG structure to maximize power, on the use of synthetic data for the derivation of an RG, as well as on the integration with sequential testing methods based on e-processes [54].

References

[1] Or Honovich, Uri Shaham, Samuel R Bowman, and Omer Levy. Instruction induction: From few examples to natural language task descriptions. *arXiv preprint arXiv:2205.10782*, 2022.

[2] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.

[3] Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024.

[4] Anastasios N Angelopoulos, Stephen Bates, Emmanuel J Candès, Michael I Jordan, and Lihua Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. *arXiv preprint arXiv:2110.01052*, 2021.

[5] Bracha Laufer-Goldshtein, Adam Fisch, Regina Barzilay, and Tommi S. Jaakkola. Efficiently controlling multiple risks with Pareto testing. In *Proc. International Conference on Learning Representations*, 2023.

[6] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[7] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[8] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

[9] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

[10] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*, pages 2113–2122. PMLR, 2015.

[11] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 1015–1022, Madison, WI, USA, 2010. Omnipress.

[12] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[13] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[14] Amirmohammad Farzaneh and Osvaldo Simeone. Ensuring reliability via hyperparameter selection: Review and advances. *arXiv preprint arXiv:2502.04206*, 2025.

[15] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[16] David R Hunter. Mm algorithms for generalized bradley-terry models. *The annals of statistics*, 32(1):384–406, 2004.

[17] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[18] Aaditya Ramdas, Jianbo Chen, Martin J Wainwright, and Michael I Jordan. A sequential algorithm for false discovery rate control on directed acyclic graphs. *Biometrika*, 106(1):69–86, 2019.

[19] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[20] Katharina Eggensperger, Philipp Müller, Neeratyoy Mallik, Matthias Feurer, René Sass, Aaron Klein, Noor Awad, Marius Lindauer, and Frank Hutter. Hpobench: A collection of reproducible multi-fidelity benchmark problems for hpo. *arXiv preprint arXiv:2109.06716*, 2021.

[21] Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *International conference on machine learning*, pages 1437–1446. PMLR, 2018.

[22] Noor Awad, Neeratyoy Mallik, and Frank Hutter. DEHB: evolutionary hyberband for scalable, robust and efficient hyperparameter optimization. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2147–2153. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track.

[23] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

[24] Kalyanmoy Deb, Karthik Sindhya, and Jussi Hakanen. Multi-objective optimization. In *Decision sciences*, pages 161–200. CRC Press, 2016.

[25] Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*, pages 240–248. PMLR, 2016.

[26] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[27] George Casella and Roger Berger. *Statistical inference*. CRC press, 2024.

[28] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[29] Manuela Cattelan. Models for paired comparison data: A review with emphasis on dependent data. 2012.

[30] Louis L Thurstone. A law of comparative judgment. In *Scaling*, pages 81–92. Routledge, 2017.

[31] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[32] José M Bernardo, Adrian FM Smith, and Mark Berliner. *Bayesian theory*, volume 586. Wiley Online Library, 1994.

[33] Persi Diaconis and Donald Ylvisaker. Conjugate priors for exponential families. *The Annals of statistics*, pages 269–281, 1979.

[34] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[35] Anil K Jain and Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[36] Ben Peters and André FT Martins. Smoothing and shrinking the sparse seq2seq search space. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2642–2654, 2021.

[37] Alvaro Valcarce. Wireless Suite: A collection of problems in wireless telecommunications. *https://github.com/nokia/wireless-suite*, 2020.

[38] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

[39] Gavin Lynch, Wenge Guo, Sanat K. Sarkar, and Helmut Finner. The control of the false discovery rate in fixed sequence multiple testing. *Electronic Journal of Statistics*, 11(2):4649 – 4673, 2017.

[40] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

[41] Mohammad Taher Pilehvar and Jose Camacho-Collados. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*, 2018.

[42] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*, 2022.

[43] meta. Llama. `https://LLaMA.meta.com/LLaMA3`, 2025. [Online; accessed 28-Jan-2025].

[44] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[45] OpenAI. GPT-4 technical report, 2023. `https://openai.com/gpt-4`.

[46] Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 conference on machine translation. In Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Liane Guillou, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Aurélie Névéol, Mariana Neves, Pavel Pecina, Martin Popel, Philipp Koehn, Christof Monz, Matteo Negri, Matt Post, Lucia Specia, Karin Verspoor, Jörg Tiedemann, and Marco Turchi, editors, *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany, August 2016. Association for Computational Linguistics.

[47] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[48] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[49] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, 2018.

[50] Ben Peters, Vlad Niculae, and André FT Martins. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, 2019.

[51] Andre Martins and Ramon Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pages 1614–1623. PMLR, 2016.

[52] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[53] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

[54] Matteo Zecchin, Sangwoo Park, and Osvaldo Simeone. Adaptive learn-then-test: Statistically valid and efficient hyperparameter selection. *arXiv preprint arXiv:2409.15844*, 2024.

[55] John A. Rice. *Mathematical Statistics and Data Analysis.* Belmont, CA: Duxbury Press., third edition, 2006.

[56] Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.

[57] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

[58] Jerome H Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33:1–22, 2010.

[59] Aaditya K Ramdas, Rina F Barber, Martin J Wainwright, and Michael I Jordan. A unified treatment of multiple testing with prior knowledge using the p-filter. *The Annals of Statistics*, 47(5):2790–2821, 2019.

[60] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[61] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[62] Bradley Efron and Carl Morris. Stein's estimation rule and its competitors—an empirical bayes approach. *Journal of the American Statistical Association*, 68(341):117–130, 1973.

[63] Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 72(4):417–473, 2010.

[64] Rajen D Shah and Richard J Samworth. Variable selection with error control: another look at stability selection. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 75(1):55–80, 2013.

[65] Martin J Wainwright and Max Chickering. Estimating the" wrong" graphical model: Benefits in the computation-limited setting. *Journal of Machine Learning Research*, 7(9), 2006.

[66] Luca Franceschi, Michele Donini, Valerio Perrone, Aaron Klein, Cédric Archambeau, Matthias Seeger, Massimiliano Pontil, and Paolo Frasconi. Hyperparameter optimization in machine learning. *arXiv preprint arXiv:2410.22854*, 2024.

[67] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[68] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. Accessed: 2024-11-19.

[69] DM Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2:37, 2011.

[70] Dani Yogatama, Lingpeng Kong, and Noah A Smith. Bayesian optimization of text representations. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[71] Pedro M de Sant Ana and Nikolaj Marchenko. Radio Access Scheduling using CMA-ES for optimized QoS in wireless networks. In *2020 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the contributions of the paper, including the proposal of the RG-PT method, its theoretical guarantees (e.g., FDR control), and empirical improvements over the state-of-the-art PT and LTT. These claims are substantiated through theoretical analysis and experiments, as discussed throughout the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the main limitations of the proposed method in Sec. 1 and Sec. 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The main theoretical result of the paper, the FDR control guarantee of RG-PT, is formally stated and proven in Proposition 3.1, along with all necessary assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The algorithm is described in detail in Appendix D, and the experimental setup, including datasets, objectives, model architectures, and hyperparameter grids, is fully documented in Sec. 4 and Appendix H. This information is sufficient to reproduce the main results without access to the code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Anonymized code and data are presented as an anonymous Github repository. All datasets used are publicly accessible, and instructions for reproduction are included in Sec. 4 and Appendix H.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All training/test details (e.g., data splits, priors, objectives, and hyperparameters) are described in Sec. 4 and Appendix H, with key elements summarized in Appendix H.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Where appropriate, such as in Fig. 1b, Fig. 3, and Fig. 9, the variability due to random data splits is visualized using boxplots.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computer resources used to run the experiments are clearly stated in Appendix H.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the authors have read the NeurIPS Code of Ethics, and confirm that the research conducted in this paper conforms with the guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The potential positive societal impact is discussed in Sec. 1, where the authors highlight that reliable hyperparameter selection with statistical guarantees can improve the safety and trustworthiness of machine learning in high-stakes domains.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All external code and datasets used in the paper are properly cited with references to their original sources, and their licenses and terms of use are respected and followed as specified by the original creators.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code for the proposed algorithm, RG-PT, is linked in the paper as an anonymous Github repository.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Fixed Sequence Testing

In this section, we provide a brief overview of FST for controlling the FDR, which we used in our simulations for PT. While PT, as outlined in [5], is designed to support control of the FWER, our focus in this paper is on FDR control.

MHT methods such as the Bonferroni correction for FWER control [55] and the Benjamini-Yekutieli (BY) procedure for FDR control [56] do not leverage any side information about the relative reliability of the hyperparameters. When such information is available during calibration, FST can be used to test hyperparameters in order of expected reliability. When the ordering information is accurate, FST can be beneficial to reduce the FDR [39]. In this section, we briefly describe the FST procedure for FDR control.

With FST, the candidate hyperparameters are ordered as $\lambda_{(1)}, \ldots, \lambda_{(|\Lambda|)}$ using side information. The ordering ideally lists the hyperparameters from the most to the least likely to meet the reliability criterion (2).

Starting with $i = 1$, each hyperparameter $\lambda_{(i)}$ is tested sequentially based on its p-value $p_{\lambda_{(i)}}$ against an adjusted critical value δ_i that decreases with the index $i = 1, 2, \ldots, |\Lambda|$. At each step i, the hyperparameter $\lambda_{(i)}$ is deemed to be reliable if $p_{\lambda_{(i)}} \leq \delta_i$. Testing continues until k hyperparameters are deemed to be unreliable, at which point testing stops. The choice of the integer k is typically set as a small proportion, often around 5-10%, of the total number of hypotheses, $|\Lambda|$.

The critical values δ_i are adapted to account for the position of each hypothesis in the testing sequence. These values are specifically designed to control the FDR under various dependency structures among the p-values. For the case of interest here, which is arbitrary dependence of the p-values, the critical levels can be set as [39]

$$\delta_i = \begin{cases} \frac{\delta}{k} & \text{if} \quad i \leq k \\ \frac{(|\Lambda|-k+1)\delta}{(|\Lambda|-i+1)k} & \text{if} \quad i > k, \end{cases} \tag{14}$$

where δ is the target FDR level and k is the number of unreliable hyperparameters allowed before testing stops.

The final set of reliable hyperparameters is $\hat{\Lambda}_{\mathcal{Z}} = \{\lambda_{(1)}, \ldots, \lambda_{(j)}\}$, where j corresponds to the index of the last hyperparameter tested before stopping. This ensures that the FDR is rigorously maintained below level δ.

B Summary of DAGGER

This section outlines the step-up procedure used in DAGGER [18] to dynamically adjust the testing thresholds. DAGGER determines the testing thresholds adaptively based on the structure of the DAG and on the outcomes of previously tested hypotheses. At each depth level of the DAG, thresholds are updated dynamically to control the FDR, while respecting the hierarchical dependencies encoded by the DAG.

At each depth d, only the hyperparameters with no unreliable parents are considered for testing. If any parent of a hyperparameter λ is deemed unreliable by DAGGER, all of its descendants are also automatically deemed unreliable. The threshold for testing the i-th hypothesis at depth d is given by

$$\delta_i(r) = \frac{v_i}{V} \cdot \frac{\delta}{\beta(m_i + r + R_{1:d-1} - 1)}, \tag{15}$$

where $r \in [1, |\Lambda_d|]$ is a parameter set as detailed below; v_i is the effective number of leaves in the subgraph rooted at the current node; V is the total number of leaves in the DAG; m_i is the effective number of nodes in the subgraph rooted at the current node; $R_{1:d-1}$ is the total number of rejections at depths 1 through $d-1$; and $\beta(\cdot)$ is a reshaping function, such as the Benjamini-Yekutieli [56] function $\beta_{BY}(x) = x / \sum_{k=1}^{V} 1/k$, which is designed to ensure FDR control under arbitrary dependence.

The effective number of leaves v_i and the effective number of nodes m_i for node i are calculated as follows. If i is a leaf, then we have $v_i = m_i = 1$. Otherwise, the values v_i and m_i are calculated recursively from leaves to roots as

$$v_i = \sum_{j \in \text{children}(i)} \frac{v_j}{|\text{parents}(j)|}, \tag{16}$$

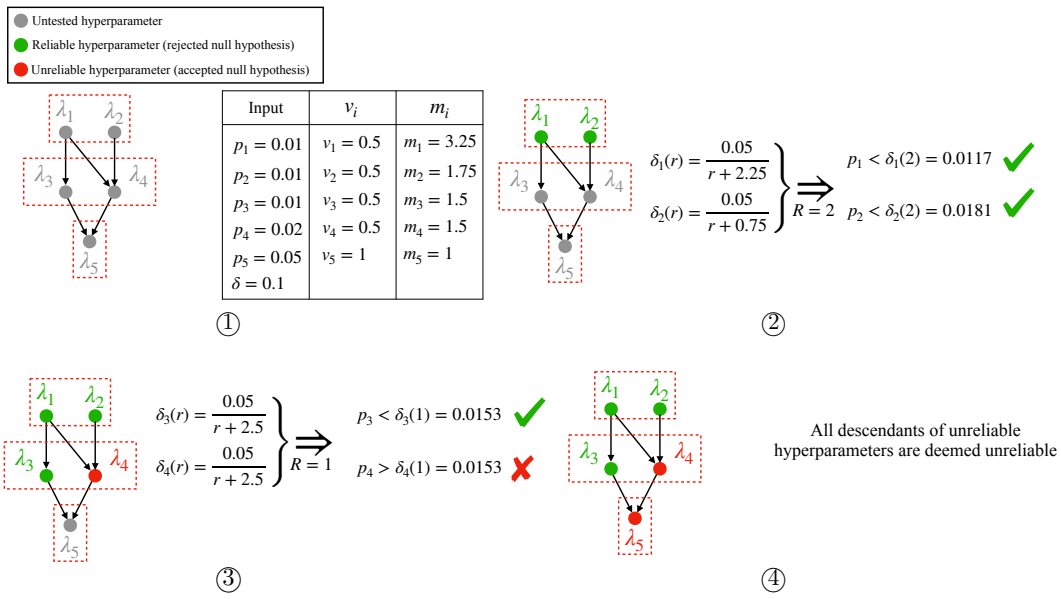


Figure 5: Illustration of the DAGGER algorithm's operation to control the FDR at $\delta = 0.1$. At each step, a hyperparameter is tested, starting from the root nodes and progressing level by level through the DAG. The testing thresholds δ_i are computed for each hyperparameter λ_i using the step-up procedure in (15) and (18), using the identity function as $\beta(\cdot)$. The p-value of each hyperparameter is compared against its respective threshold δ_i to assess the reliability of λ_i.

and

$$m_i = 1 + \sum_{j \in \text{children}(i)} \frac{m_j}{|\text{parents}(j)|}, \tag{17}$$

where $\text{children}(i)$ and $\text{parents}(i)$ denote the sets of the children and the parents of node i, respectively.

The parameter r at depth d needs to be determined before testing can begin. To maximize the number of rejections while ensuring FDR control, DAGGER calculates the value

$$R = \arg \max_{r=1,\ldots,|\Lambda_d|} \left[\sum_{\lambda_i \in \Lambda_d} \mathbf{1}\{p_{\lambda_i}(\mathcal{Z}_{\text{MHT}}) \leq \delta_i(r)\} \geq r \right]. \tag{18}$$

The threshold $\delta_i(R)$ is then used to perform the testing for hyperparameter λ_i.

The step-up procedure (15) ensures that thresholds are increasingly relaxed, i.e., increased, as we move further down the DAG. The overall algorithm is described in Algorithm 1, and an example illustrating DAGGER's operation is shown in Fig. 5. The figure highlights how thresholds are updated and decisions propagated through the DAG. For simplicity, the figure assumes the reshaping function $\beta(\cdot)$ to be the identity function $\beta(x) = x$. First, the values for the effective number of leaves v_i and the effective number of nodes m_i are calculated for each hyperparameter λ_i. Next, going level by level, the thresholds $\delta_i(R)$ are calculated using (15) and (18), and each hyperparameter λ_i is tested by comparing $p_{\lambda_i}(\mathcal{Z}_{\text{MHT}})$ with $\delta_i(R)$.

C Computational Complexity of Constructing the Reliability Graph

Constructing the RG involves three main steps: training the BT model, clustering, and Lasso regression. The training of the BT model has a time complexity $\mathcal{O}(n^2)$, because it involves a cyclic optimization over the model parameters associated to the pairs of configurations [16, Sec. 3]. Hierarchical clustering has a complexity $\mathcal{O}(n^2)$ [57, Sec. 3], while Lasso regression has a per-iteration complexity $\mathcal{O}(n)$ [58, Sec. 2.1]. Therefore, the overall complexity of RG construction is of the order $\mathcal{O}(n^2)$.

Algorithm 1 DAGGER [18]

Input: DAG structure, p-values $\{p_\lambda(\mathcal{Z}_{\text{MHT}})\}$, target FDR level δ
Output: Set of reliable hyperparameters $\hat{\Lambda}_{\mathcal{Z}}$
for depth $d = 1, \ldots, D$ **do**
  for each hyperparameter λ_i in cluster Λ_d **do**
    if all parent hyperparameters of λ_i are deemed as reliable **then**
      Evaluate threshold $\delta_i(R)$ using (15) and (18)
      if $p_{\lambda_i}(\mathcal{Z}_{\text{MHT}}) \leq \delta_i(R)$ **then**
        Detect λ_i as reliable
      else
        Detect λ_i as unreliable
      end if
    end if
  end for
  Update $\hat{\Lambda}_{\mathcal{Z}}$ with all the hyperparameters detected as reliable at depth d
end for
return $\hat{\Lambda}_{\mathcal{Z}}$

D RG-PT Algorithm

Algorithm 2 provides a summary of RG-PT.

Algorithm 2 Reliability Graph-Based Pareto Testing (RG-PT)

Input: Hyperparameter set Λ, calibration dataset $\mathcal{Z}$, FDR level δ, reliability thresholds $\{\alpha_l\}_{l=1}^{L_c}$, number of DAG levels D
Output: Reliable hyperparameter subset $\hat{\Lambda}_{\mathcal{Z}}$
Split calibration data: $\mathcal{Z} = \mathcal{Z}_{\text{OPT}} \cup \mathcal{Z}_{\text{MHT}}$
Estimate Pareto front $\Lambda_{\text{OPT}} \subseteq \Lambda$ using $\mathcal{Z}_{\text{OPT}}$
Construct Reliability Graph (RG):
Compute pairwise comparisons using $\mathcal{Z}_{\text{OPT}}$ and optional priors
Estimate scores $s(\lambda)$ via Bradley-Terry model
Cluster Λ_{OPT} into D levels using scores
for each level $d = 2$ to D **do**
  for each $\lambda \in \Lambda_d$ **do**
    Select parents in Λ_{d-1} via non-negative Lasso
  end for
end for
Run DAGGER on RG with data $\mathcal{Z}_{\text{MHT}}$:
for each node λ in topological order **do**
  if all parents of λ are reliable **then**
    Compute p-value p_λ
    if $p_\lambda \leq \delta_\lambda$ **then**
      Add λ to $\hat{\Lambda}_{\mathcal{Z}}$
    end if
  end if
end for
return $\hat{\Lambda}_{\mathcal{Z}}$

E Sample Efficiency and Finite-Sample Behavior

As demonstrated in [59], DAG-structured multiple testing yields higher statistical power compared to linear testing whenever the graph encodes true logical dependencies among hypotheses. We illustrate this through a simple analytical comparison based on FST, which can be seen as a special case of DAGGER with a chain graph.

Setup. Suppose we wish to test K hypotheses $\mathcal{H}_1, \ldots, \mathcal{H}_K$. Let $m < K$ of these hypotheses correspond to non-null hypotheses (reliable hyperparameters), and the remaining $K - m$ be true nulls (unreliable hyperparameters). Assume all K p-values $\{p_i\}_{i=1}^K$ are independent and identically distributed. Under the null, $p_i \sim \text{Unif}[0, 1]$, so the CDF is $F_0(p) = p$. Under the alternative, p_i is stochastically smaller, with CDF $F_A(p)$ satisfying $F_A(\alpha) = \beta > \alpha$.

At testing threshold α, the type-I error probability is α, while the power (probability of correctly rejecting a non-null) is β.

Consider an ideal ordering in which all m non-nulls appear first. FST proceeds sequentially, rejecting $\mathcal{H}_i$ if $p_i \leq \alpha$. The probability of correctly rejecting all m non-nulls is

$$\prod_{i=1}^{m} \Pr(p_i \leq \alpha) = \beta^m. \tag{19}$$

Now suppose $r \geq 1$ null hypotheses appear before the m non-nulls, i.e., among the first $m + r$ positions. Assume pessimistically that all r nulls come before any non-nulls. FST terminates if it encounters a null with $p_i > \alpha$ (probability $1 - \alpha$). To reach and reject all m non-nulls, the procedure must: 1. reject each of the r preceding nulls (probability α^r), and 2. reject all m non-nulls (probability β^m). Thus, the probability of correctly rejecting all m non-nulls is

$$\alpha^r \cdot \beta^m. \tag{20}$$

This calculation shows that even a single misplaced null reduces power by a factor of $\alpha < 1$. In the worst case, with r nulls preceding all non-nulls, the power degradation becomes exponential in r.

In a DAG-structured testing framework, when the DAG accurately reflects the logical dependencies among hypotheses, non-nulls tend to be concentrated in subtrees rather than arbitrarily interleaved with nulls. In this case, DAGGER avoids early termination caused by misplaced nulls, yielding higher power and improved sample efficiency compared to fixed linear orders. In the RG setting, this insight underscores the importance of incorporating structural information: well-specified edges increase the probability of detecting all reliable hyperparameters at a given sample size, while FDR control remains valid regardless of structure.

F Selection of RG-PT Hyperparameters

The theoretical FDR guarantee of RG-PT holds regardless of the choices of the hyperparameters D, n_p, and τ. These parameters affect the construction of the RG and hence influence the statistical power of the procedure, but not its validity. For practical use, however, guidance on their selection is essential. Below we describe the automated, data-driven procedures we employed in our experiments, which allow RG-PT to be applied in a reproducible manner without user intervention.

Graph Depth D. We determine the number of levels in the RG by computing the Silhouette score [60] for each candidate value $D \in \{2, 3, \ldots, D_{\max}\}$ and selecting the value that maximizes the average Silhouette score. The maximum depth can be set to $|\mathcal{Z}_{\text{OPT}}|$ or capped at a smaller constant (e.g., $D_{\max} = 20$) for computational efficiency. The Silhouette score quantifies intra-cluster cohesion and inter-cluster separation and is a standard criterion for cluster number selection [61]. This approach is fully data-driven, with each cluster naturally corresponding to a DAG level in RG-PT.

Prior Weight n_p. We set the prior weight n_p using a predictive empirical Bayes procedure [62], which calibrates the reliance on prior pairwise information η_{ij} according to its predictive value on held-out data. Specifically:

1. Define a grid of candidate values for n_p, e.g., $\{0.25|\mathcal{Z}_{\text{OPT}}|, 0.5|\mathcal{Z}_{\text{OPT}}|, 0.75|\mathcal{Z}_{\text{OPT}}|, |\mathcal{Z}_{\text{OPT}}|\}$.
2. For each n_p, compute BT scores $s(\lambda)$ for $\lambda \in \Lambda_{\text{OPT}}$.
3. Evaluate the predictive log-likelihood on a validation dataset $\mathcal{Z}_{\text{val}}$ as

$$\mathcal{L}_{\text{val}}(n_p) = \sum_{\lambda_i, \lambda_j \in \Lambda_{\text{OPT}}} \left[y_{ij} \log \frac{e^{s(\lambda_i)}}{e^{s(\lambda_i)} + e^{s(\lambda_j)}} + (1 - y_{ij}) \log \frac{e^{s(\lambda_j)}}{e^{s(\lambda_i)} + e^{s(\lambda_j)}} \right], \tag{21}$$

   where $y_{ij} = 1$ if λ_i outperforms λ_j in $\mathcal{Z}_{\text{val}}$.

4. Select the n_p maximizing $\mathcal{L}_{\text{val}}(n_p)$.

This ensures the prior contributes proportionally to its predictive usefulness.

Lasso Regularization τ. We tune the non-negative Lasso regularization parameter τ via stability selection [63], a principled approach to obtain reproducible sparse structures. The procedure is:

1. Generate multiple random subsamples of the calibration dataset $\mathcal{Z}_{\text{OPT}}$, each containing 50% of the data.
2. For each τ, fit the non-negative Lasso on each subsample and construct the RG.
3. For each edge $(\lambda_i \to \lambda_j)$, record its selection frequency across subsamples.
4. Compute the edge stability score for τ as the average selection frequency.
5. Choose the τ that maximizes stability, subject to a sparsity constraint on the graph (e.g., average node degree ≤ 3) to avoid overfitting and preserve power [64].

G Semantics of Edges Inferred via Non-Negative Lasso

The purpose of the RG is to encode reliability dominance relationships among candidate hyperparameters, rather than mere predictive correlations. We provide here the structural motivation for our use of non-negative Lasso to infer edges in the DAG.

Let $G = (V, E)$ denote the RG, where $V = \{\lambda_1, \ldots, \lambda_{|\Lambda_{\text{OPT}}|}\}$ is the set of candidate hyperparameters, and $E \subseteq V \times V$ the set of directed edges. An edge $(\lambda_i \to \lambda_j) \in E$ indicates that the reliability of λ_j is dependent on that of λ_i, i.e., λ_i is a "parent" of λ_j in the DAG.

We assume that each configuration $\lambda_j \in V$ has an unobserved binary reliability label $R_j \in \{0, 1\}$, where $R_j = 1$ indicates that λ_j is reliable. The relationship between reliability variables can be expressed as the stochastic Boolean structural equation

$$R_j = f_j\big(\{R_i : (\lambda_i \to \lambda_j) \in E\}\big) + \varepsilon_j, \tag{22}$$

where $f_j(\cdot)$ is an unknown binary-valued function, and ε_j is a noise term, possibly correlated with the output of $f_j(\cdot)$. For example, $f_j(\cdot)$ may encode that λ_j is likely to be reliable whenever a sufficient number of its parents are. The noise term allows for the possibility that a child configuration is unreliable ($R_j = 0$) due to random effects not captured by $f_j(\cdot)$.

While the true reliability labels $\{R_j\}$ are unobserved, we do observe scalar-valued performance vectors $Y(\lambda_j) \in \mathbb{R}^T$ (e.g., per-task or per-example risks), which act as continuous surrogates for reliability. We approximate the Boolean process with the linear regression model

$$Y(\lambda_j) \approx \sum_{i \neq j} \beta_{ij}\, Y(\lambda_i) + \varepsilon_j, \tag{23}$$

subject to non-negativity constraints $\beta_{ij} \geq 0$ to preserve monotonic influence. This yields the non-negative Lasso objective, which provides both sparsity (retaining only the most predictive and interpretable relationships) and monotonicity (reflecting the domain prior that improving a parent cannot reduce the reliability of the child).

This formulation does not assert that reliability is intrinsically linear. Instead, it provides a tractable linearization of an unknown monotonic Boolean process, in line with influence modeling approaches in graphical models [65]. Crucially, the formal FDR control of RG-PT remains valid regardless of the DAG learning strategy; the graph structure influences statistical power but not correctness of the guarantees.

H Additional Experiments

In this section, we present four new experiments across language model calibration [36], object detection [4], image classification [66], and telecommunications engineering [37], to demonstrate the advantages of our method over LTT and PT. We begin by detailing the hardware used for the simulations and the specific parameter settings chosen for each experiment.

H.1 Experimental Setups

All experiments were conducted using dedicated computational resources. Specifically, RG-PT, LTT, and PT runs, along with data generation for the object detection, image classification, and telecommunications engineering tasks, were executed on a machine equipped with an Apple M1 Pro chip (10-core CPU, 16-core GPU, 16 GB RAM). Data generation for the prompt engineering experiment (Section 4.1) and the sequence-to-sequence translation task was performed on an NVIDIA A100 GPU (40 GB VRAM), using CUDA 11.3 and 40 GB system memory.

The RG-PT parameter settings for each experiment are summarized in Table 1.

Table 1: RG-PT parameter settings for each experiment.

Experiment	D	n_p	τ
Prompt Engineering	17	1000	0.1
Sequence-to-sequence translation	10	200	0.1
Object Detection	20	0	0.1
Image Classification - Low Dimension	10	0	0.1
Image Classification - High Dimension	20	0	0.1
Telecommunications Engineering	10	0	0.1

H.2 FDR Analysis Across All Experiments

To begin with, we present a high-level summary of our experimental validation of Proposition 3.1. The specific details of each experiment, including their objectives, datasets, and task configurations, are provided in their respective sections. Briefly, for each task, we ran RG-PT 100 times using different random splits of the available dataset into $\mathcal{Z}_{\text{OPT}}$, $\mathcal{Z}_{\text{MHT}}$, and an independent test set. The target was to control the average FDR on the test set below a threshold of $\delta = 0.1$. We then measured the average FDR on the test set across runs, and the results, summarized in Table 2, confirm that RG-PT satisfies the FDR condition as theoretically established in Proposition 3.1. For completeness, we also evaluated the average FDR of LTT and PT under the same procedure, thereby validating that both methods also maintain FDR control in practice.

Table 2: Average FDR achieved by LTT, PT, RG-PT across tasks for a target FDR threshold of $\delta = 0.1$.

Task	Prompt Engineering	Object Detection	Language Translation	Image Classification	Radio Access Scheduling
RG-PT	0.089	0.093	0.095	0.084	0.093
LTT	0.073	0.091	0.097	0.071	0.090
PT	0.079	0.087	0.093	0.071	0.087

H.3 Ablation Study

In this section, we use the prompt engineering task in Sec. 4.1 to perform an ablation study over the RG depth D, as well as the effect of misspecified prior information.

H.3.1 Effect of DAG Depth

To assess the impact of the DAG depth D in RG-PT, we vary it from 1 (flat graph, equivalent to LTT) to 100 (fully linear, equivalent to PT), and measure both the length of the shortest prompt in $|\hat{\Lambda}_{\mathcal{Z}}|$ and the test FDR. Fig. 6 illustrates the average shortest reliable prompt length and average FDR over 100 runs for each depth, showing that FDR remains valid across depths. Additionally, it can be seen that there exists an intermediate depth that minimizes the average prompt length, indicating that the depth $1 < D < |\Lambda_{\text{OPT}}|$ can be chosen to optimize power.

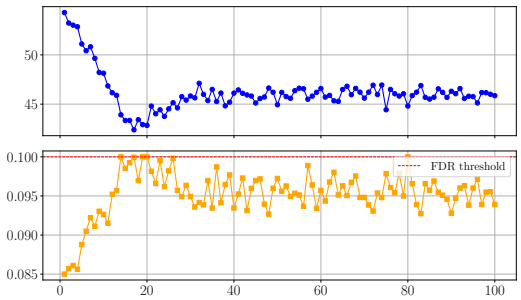


Figure 6: Average shortest prompt length in the returned prompt set $\hat{\Lambda}_{\mathcal{Z}}$ and the average FDR achieved by RG-PT as a function of RG depth D.

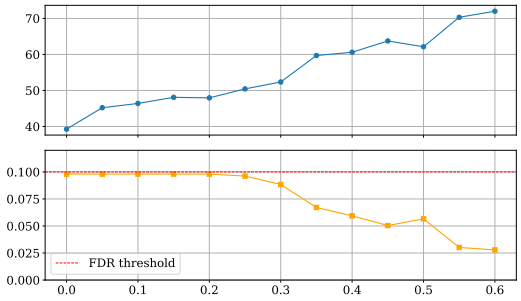


Figure 7: Average shortest prompt length in the returned prompt set $\hat{\Lambda}_{\mathcal{Z}}$ and the average FDR achieved by RG-PT as a function of fraction f of flipped pairwise prior probabilities.

H.3.2 Effect of Misspecified Priors

To simulate prior misspecification, we inject noise into the pairwise priors η_{ij}. For every pair $\lambda_i, \lambda_j \in \Lambda_{\text{OPT}}$, we independently swap the priors η_{ij} and η_{ji} with probability $f \in [0, 1]$. When $f = 0$ the prior remains intact, and when $f = 1$ every pairwise preference is completely reversed.

Fig. 7 illustrates the average shortest prompt length in the returned set $\hat{\Lambda}_{\mathcal{Z}}$ of reliable prompts and the average FDR, across 100 random data splits, as a function of the flipped fraction f of pairwise preferences η_{ij}. As f increases, the shortest prompt length grows, indicating the effect of reduced prior information quality. Despite this, the FDR remains controlled below the target level $\delta = 0.1$, demonstrating robustness to misspecified priors. Notably, for $f > 0.6$, RG-PT returns an empty set $\hat{\Lambda}_{\mathcal{Z}}$ across all runs, correctly avoiding any potentially unreliable hyperparameters in the face of highly corrupted priors.

H.4 Image Segmentation for Object Detection

We now evaluate the proposed RG-PT framework on a multi-objective image segmentation task for object detection, leveraging the MS-COCO dataset [67] and a pretrained detector from Detectron2 [68] as done in [4]. The task involves three distinct objectives: (*i*) detecting objects within an image (object detection); (*ii*) delineating object boundaries (image segmentation); and (*iii*) assigning correct labels to detected objects (object classification). These tasks are measured using recall, intersection-over-union (IoU), and classification accuracy, respectively. The goal is to control classification errors while optimizing recall and segmentation quality, addressing the trade-offs among these objectives.

The performance of the detection is determined by three hyperparameters:

1. The *object recall threshold* (λ_1) controls the threshold for selecting objects based on confidence scores. Reducing the value of λ_1 lowers the confidence threshold, which allows more objects to be selected at the cost of, potentially, increasing false positives.

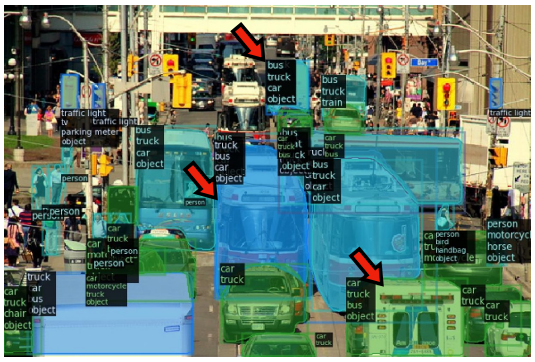


Figure 8: Illustration of the benefits of the proposed RG-PT hyperparameter selection scheme over the state-of-the-art LTT and PT for an object detection application [4]. The red arrows mark the objects not detected by an object recognition model calibrated using LTT or PT that are instead detected by the same model calibrated via RG-PT (see Appendix H.4 for details).

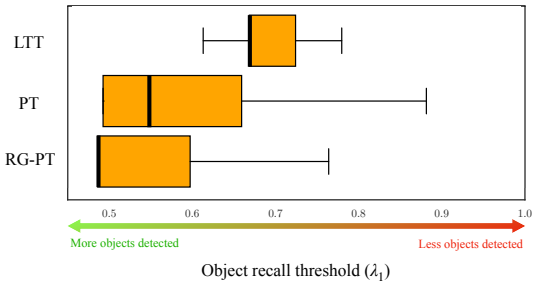


Figure 9: Example of of hyperparameter distributions for LTT, PT, and RG-PT methods. The box plots show the range (denoted by the horizontal black whiskers), median (represented by the thick black vertical lines), and interquartile range (depicted by the boxes) of the object recall threshold hyperparameter λ_1.

2. The *mask size threshold* (λ_2) tunes the size of the bounding masks used to segment objects, impacting the IoU score.

3. The *classification certainty level* (λ_3) controls the certainty level required for object classification, adjusting the tolerance for inclusion in the set of labels assigned to each detected object.

The candidate hyperparameter set Λ was constructed as per [4], by taking all combinations of 50 linearly spaced values in $[0.2, 0.5]$ for λ_1, 5 linearly spaced values in $[0.3, 0.7]$ for λ_2, and 25 logarithmically spaced values in $[-0.00436, 0]$ for λ_3. These discretization choices were optimized [4].

Denote as $R_1(\lambda)$, $R_2(\lambda)$, and $R_3(\lambda)$ the risks associated with recall, IOU, and coverage, respectively, for hyperparameter $\lambda = (\lambda_1, \lambda_2, \lambda_3)$. Controlling these risks in the context of problem (3) is equivalent to having $L = 3$ and $L_c = 3$. Additionally, as in [4], we set the targets as $\alpha_1 = 0.5$, $\alpha_2 = 0.5$, and $\alpha_3 = 0.75$ with $\delta = 0.1$. Within the set $\hat{\Lambda}_{\mathcal{Z}}$ of reliable hyperparameters returned by the algorithm of choice, we choose the hyperparameter in subset $\hat{\Lambda}_{\mathcal{Z}}$ with the lowest value of λ_1 in order to increase the number of detected objects as much as possible. No prior knowledge was leveraged in creating the RG in this experiment by setting $n_p = 0$.

We compare the distribution of the hyperparameters returned by LTT, PT, and RG-PT. The distribution is obtained by running 200 trials for each algorithm over different splits of calibration data $\mathcal{Z}$ into subsets $\mathcal{Z}_{\text{OPT}}$ and $\mathcal{Z}_{\text{MHT}}$ with $|\mathcal{Z}_{\text{OPT}}| = 1500$ and $|\mathcal{Z}_{\text{MHT}}| = 1500$. As shown in Fig. 9, the results demonstrate that RG-PT tends to return lower values for λ_1 than both LTT and PT. In particular, both the mean and dispersion for RG-PT are lower than those for LTT and PT. A lower threshold λ_1 allows the detector to select more objects, which directly enhances object recall, while still maintaining controlled levels of segmentation and classification accuracy (see also Fig. 8).

Notably, this experiment shows that RG-PT still exhibits a larger power compared to LTT and PT even when ordering among hypotheses are not hard-coded into the testing procedure. In particular, in this task, we did not assume access to any prior reliability information and set $n_p = 0$. Despite this, RG-PT still outperformed both LTT and PT, demonstrating its ability to discover and exploit latent structure purely from empirical data.

H.5 Image Classification

In this experiment, following [66], we consider the problem of hyperparameter selection for a support vector machine (SVM) model used to classify images from the Fashion MNIST dataset. The Fashion MNIST is a widely used benchmark for image classification, consisting of 70,000 grayscale images of 10 different clothing categories.

We consider two risk functions ($L = 2$) in problem (3), namely the classification error $R_{\text{err}}(\lambda)$ and the recall, $R_{\text{rec}}(\lambda)$. The classification error $R_{\text{err}}(\lambda)$ measures the proportion of incorrectly classified images out of the total number of samples. The recall measures the ability of a model to correctly identify all the relevant instances of each class. Accordingly, for each class, the recall is computed as the ratio of correctly identified instances of that class to the total number of actual instances of the same class in the dataset. The recall $R_{\text{rec}}(\lambda)$ represents the average of the recall values across all classes.

With reference to problem (3), we aim at minimizing recall, i.e. $R_2(\lambda) = R_{\text{rec}}(\lambda)$, while keeping the classification error rate below 0.3, i.e. $R_1(\lambda) = R_{\text{acc}}(\lambda)$, $L_c = 1$, and $\alpha_1 = 0.3$. The goal is therefore defined as

$$\min_{\lambda \in \Lambda} R_{\text{rec}}(\lambda) \quad \text{subject to} \quad R_{\text{acc}}(\lambda) < 0.3. \tag{24}$$

This is a non-trivial problem since the accuracy maximizing model may not also optimize the recall [69].

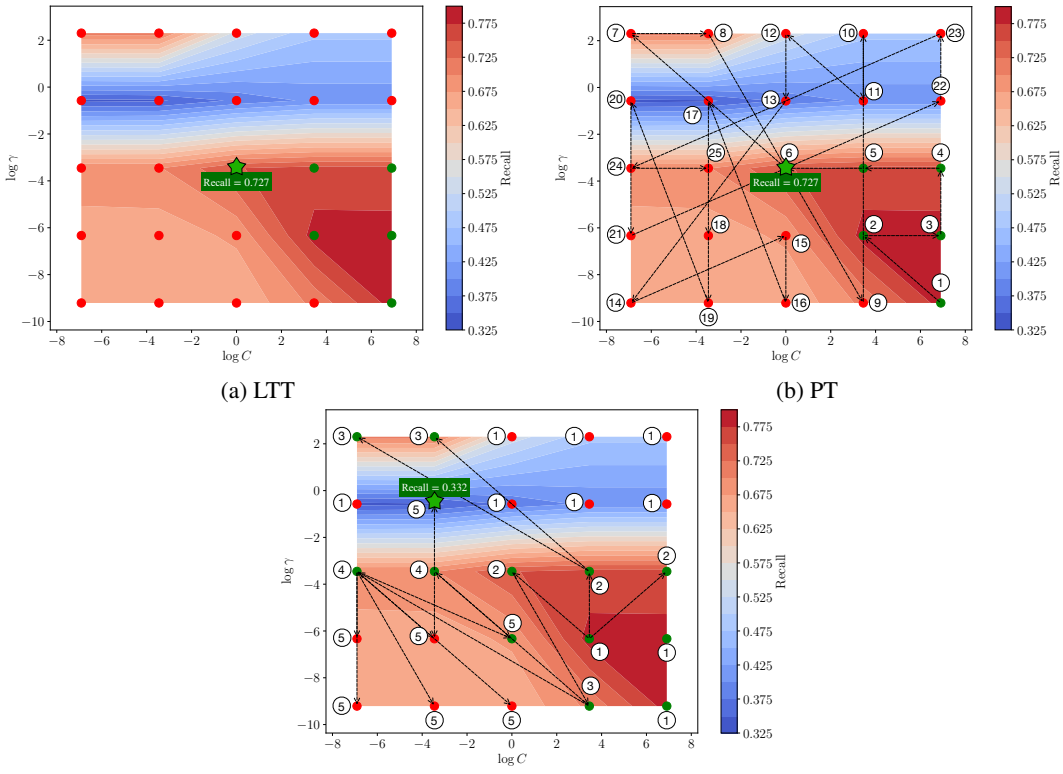
The SVM model requires the selection of two hyperparameters [66]. The regularization parameter, C, controls the desired trade-off between maximizing the margin and minimizing the classification error. Lower values of C allow for a softer margin that can overlook some misclassification errors, while higher values enforce stricter classification error requirements. The kernel coefficient, γ, determines the impact of a single training example on the decision boundary, with higher values capturing finer details in the data set but risking overfitting. To create the initial candidate set Λ, we selected hyperparameters over a grid of 25 combinations, using five logarithmically spaced values in the intervals $[-3, 3]$ and $[-4, 1]$ for C and γ, respectively. This selection is in line with approaches such as [70] for SVM hyperparameter selection.

We used 5,000 data points for training the SVM, and used an additional 5,000 data points as calibration data $\mathcal{Z}$. The calibration data set $\mathcal{Z}$ was in turn divided into two groups of size 2,500, for the data sets $\mathcal{Z}_{\text{OPT}}$ and $\mathcal{Z}_{\text{MHT}}$, respectively.

Fig. 10 illustrates the testing procedures of LTT [4], PT [5], and RG-PT. The x- and y-axes represent the logarithmic scales of the two hyperparameters C and γ, while the contours indicate levels of recall $R_{\text{rec}}(\lambda)$, evaluated on the test data set, as a function of the hyperparameters $\lambda = (c, \gamma)$. The numbers illustrate the testing order for each testing method. Note that than LTT, which uses BY, does not follow any inferred order on the hyperparameters, and thus does not have the order labels in the figures. Furthermore, while LTT and PT test hyperparameters one by one, following a linear trajectory, RG-PT proceeds along a DAG, testing at the same time all hyperparameters at the same depth in the DAG.

LTT and PT are seen to stop at the sixth tested hyperparameter, yielding the set of reliable hyperparameters $\hat{\Lambda}_{\mathcal{Z}}$ marked as green dots. In contrast, RG-PT returns a much larger set $\hat{\Lambda}_{\mathcal{Z}}$ of reliable hyperparameters, also marked as green dots. Choosing within these sets the hyperparameter that minimizes the estimated recall as per problem (24) yields the solutions indicated as green stars, corresponding to a test recall of 0.727 for LTT and PT, and 0.332 for RG-PT.

It is important to note that all three methods yield test accuracies below the 0.3 threshold in (24). Specifically, the hyperparameters selected by LTT and PT result in a test accuracy error of 0.267, while those chosen by RG-PT achieve a slightly higher accuracy error of 0.286. Although the accuracy of RG-PT is closer to the threshold, it remains consistent with the statistical guarantee outlined in (24). In fact, RG-PT achieves a lower recall while maintaining the desired accuracy constraint, whereas



(a) LTT

(b) PT

(c) RG-PT

Figure 10: Illustration of the hyperparameter selection procedure followed by LTT (a), PT (b), and RG-PT (c) for the setting studied in Sec. H.5. Each node represents a hyperparameter $\lambda = (C, \gamma)$, with the numbers representing the testing order. Green nodes show the hyperparameters included in the reliable set $\hat{\Lambda}_{\mathcal{Z}}$, and the star node shows the hyperparameter in set $\hat{\Lambda}_{\mathcal{Z}}$ with the lowest recall rate.

LTT and PT follow a more conservative approach, leading to a reliable hyperparameter with higher recall.

To demonstrate the scalability of RG-PT to high-dimensional hyperparameter spaces, we repeated the previous experiment over a grid of 10,000 hyperparameter configurations instead of 25. This grid was constructed using 100 logarithmically spaced values for C and γ over the same intervals as before. The average FDR across 100 runs is reported in Table 2, highlighting RG-PT's robustness and effectiveness even in high-dimensional settings.

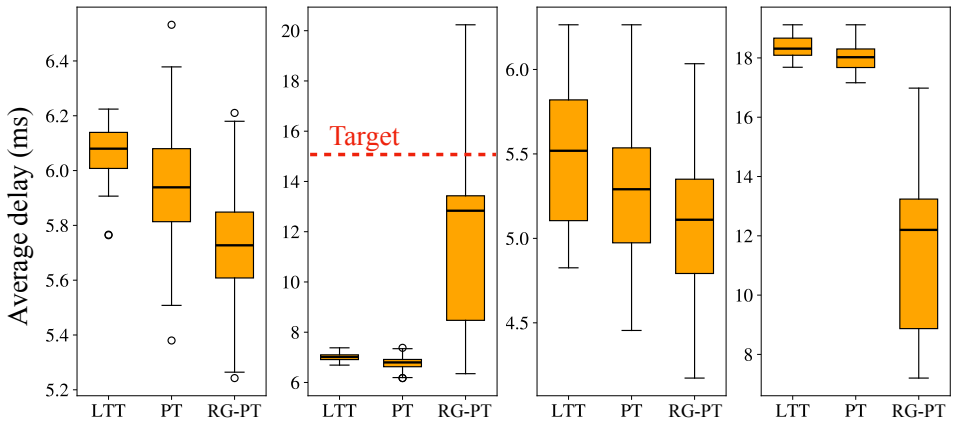
H.6 Radio Access Scheduling

In this section, we study a telecommunications engineering problem, namely the optimization of a radio access scheduler [37]. In this setup, each user equipment (UE) belongs to one of four quality-of-service (QoS) classes, assigned at random, each with its own delay and bit rate requirements [71]. The goal is to control the delay of UEs in a given QoS class, while simultaneously minimizing the delays for UEs in the other three QoS classes.

Accordingly, in the context of problem (3), we choose $L = 4$ and $L_c = 1$, and we set the risk $R_i(\lambda)$ to be equal to the average delay of QoS class i for $1 \leq i \leq 4$. We aim to keep $R_2(\lambda)$ below 15 ms, while minimizing $R_1(\lambda)$, $R_3(\lambda)$, and $R_4(\lambda)$. Formally, the problem is stated as

$$\min_{\lambda \in \Lambda} \{R_1(\lambda), R_3(\lambda), R_4(\lambda)\} \text{ subject to } R_2(\lambda) < 15 \text{ ms}. \tag{25}$$

The scheduling algorithm at the base station allocates spectral resources to the UEs. As in [37], the UEs are randomly distributed within a $1\,\text{km}^2$ area containing a centrally located base station. Each UE has an initial buffer of 100 packets, and moves at random speeds and directions. Resource



(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

Figure 11: Distribution of the average delay for the four QoS classes using hyperparameters optimized by PT (left column) and RG-PT (right column). The dashed red line indicates the target threshold for the average delay in QoS class 2.

allocation is carried out in intervals of 1 ms, called transmission time intervals (TTIs), over 10,000 TTIs per episode.

The resource allocation algorithm is controlled by a set of hyperparameters $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4) \in \Lambda$, where each λ_i adjusts a specific criterion in the reward model as detailed in [37]. Hyperparameters λ_1, λ_2, λ_3, and λ_4 determine respectively the channel quality for each UE, the total queue sizes at the UEs, the age of the oldest packet in each UE's buffer, and the fairness in resource block allocation among UEs.

Calibration and test data were generated using the Nokia wireless suite [37]. For each run, we used 100 episodes for calibration and 100 episodes for testing.

The candidate hyperparameter set Λ was generated by keeping λ_1 and λ_2 at the values λ_1^* and λ_2^* recommended by [71], and linearly sweeping hyperparameters λ_3 and λ_4 in [0.02, 0.2] and [-0.1, 0.1], respectively, with 10 steps each, resulting in a total of 100 combinations.

Fig. 11 presents the results of using PT and RG-PT to optimize the hyperparameter $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)$. Both methods successfully meet the statistical guarantee of $R_2(\lambda) < 15$ ms for class 2. However, RG-PT demonstrates a greater ability to explore the hyperparameter space Λ, identifying configurations that more effectively minimize the average delay across the other three classes.