Intelligent Surrogates Pay Attention to Data, Improving Multi-Objective HPC Optimization

Ashna Nawar Ahmed[1] *[qvu10@txstate.edu]*, ***Banooqa Banday***[1] *[bhb37@txstate.edu]*, **Terry Jones**[2] (advisor) *[trjones@ornl.gov]*, **Tanzima Islam**[1] (advisor) *[tanzima@txstate.edu]*

Texas State University[1], Oak Ridge National Laboratory[2]

Abstract

High-performance computing (HPC) schedulers must balance runtime and power. We present a surrogate-assisted multi-objective Bayesian optimization (MOBO) framework using TabNet regressors and models trained on attention-based embeddings, coupled with active-learning sample selection. The surrogates predict runtime and power, enabling MOBO to efficiently discover Pareto-optimal node allocations. We quantify trade-offs with Pareto fronts, Hypervolume (HV), and Spread across PM100 and Adastra production traces. MOBO improves HV over single-objective baselines by 24% (PM100) and 37% (Adastra) and attains lower Spread in 75% of surrogate families. Active learning reduces evaluations by ~53–70%. To our knowledge, this is the first demonstration of embedding-informed surrogates for MOBO applied to HPC scheduling, optimizing runtime–power trade-offs on production datasets.

Hypotheses

Hypothesis 1:

Attention-based embedding learning improves surrogate models compared to directly using complex transformer-based models.

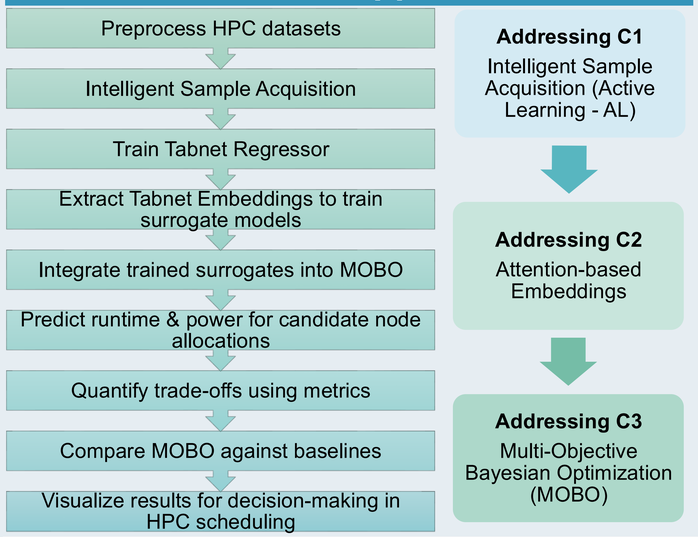
Hypothesis 2:

MOBO is more accurate in modelling multiple conflicting objectives compared to SOBO or Random Baseline.

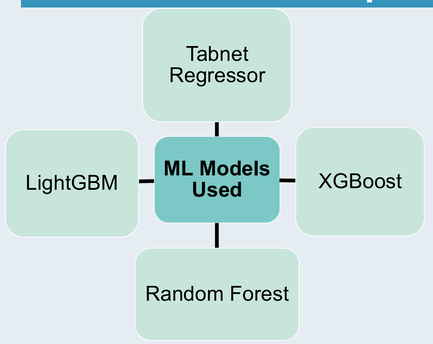
Motivation

HPC Systems' Challenges	Growing demand for fast results under strict energy limits; even small savings scale to huge gains at exascale
Current Scheduling Approaches	Often manual, heuristic, or single-objective, failing to capture runtime–power trade-offs
Multi Objective Optimization	Surrogate-based optimization shows promise but remains underused in HPC scheduling
Novelty	MOBO has been applied elsewhere, but combining it with attention-based embeddings to jointly balance runtime and power in HPC scheduling is new to this work

Solution Approach



Experimental Setup



Datasets Used

PM100: Public dataset comprising ~230K jobs executed on the Marconi100 Tier-0 supercomputer at CINECA in Italy

Adastra: Dataset collected from large-scale HPE-Cray EX supercomputer located at the CINES Tier-1 computing center in France

Metrics Used

Pareto Front: Represents the best trade-offs available.

Hypervolume (HV): Represents volume of space covered by the Pareto front; higher is better.

Spread: Represents diversity of Pareto solutions; lower means more uniform coverage.

Timing Overhead: Represents extra runtime from the pipeline; lower means faster scheduling.

Baselines Used

Random Baseline: Selects configurations uniformly from the design space, without guided search

SOBO (Single-Objective Bayesian Optimization): Optimizes a single objective (runtime or power), ignoring trade-offs.

NSGA-II: A traditional genetic algorithm baseline that evolves a population to approximate the Pareto front.

Goals

Goal 1 Develop deep surrogate models to accurately predict runtime and power for HPC jobs.

Goal 2 Incorporate intelligent sample selection and multi-objective Bayesian optimization (MOBO) to efficiently explore Pareto-optimal resource allocations

Goal 3 Quantify trade-offs using Pareto fronts and hypervolume metrics to guide scheduling decisions.

Goal 4 Demonstrate improvements over single-objective and random baselines in realistic HPC workloads.

Challenges

Telemetry Sizes with Data Irregularities (C1)	HPC telemetry is massive, but also irregular . Direct models trained on such data often collapse or fail to generalize.
Noisy HPC Job Logs (C2)	Logs combine categorical, numeric, and time-series data, with noise, that hinder conventional models from learning useful patterns
Inherently Conflicting Objectives (C3)	Minimizing runtime often increases power consumption, while saving energy typically slows jobs.

Key Takeaways

Attention-based Embeddings
- Improves fidelity

Intelligent Data Acquisition
- Reduces Job Evaluations

Integration into Multi-Objective Optimization
- Improves Pareto front quality

Preliminary Results

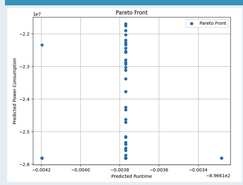


Fig. 1: Pareto Front generated for PM100 + Tabnet Regressor + MOBO

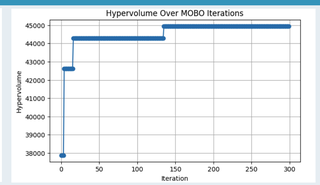


Fig. 2: HV progression generated for PM100 + Tabnet Regressor + MOBO

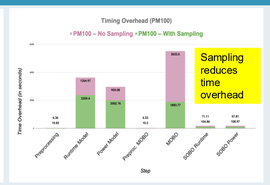


Fig. 3: Timing Overhead for each sub-step of the pipeline (PM100)

Observations:
Active learning with MOBO rapidly improves hypervolume by focusing on high-value regions (Fig. 2), producing compact, high-quality Pareto fronts (Fig. 1). This reduces timing overhead (Fig. 3) but comes at the cost of Spread, as the front is less evenly distributed across runtime–power trade-offs.

Dataset	Metric	Hypothesis 1	Hypothesis 2
PM100	HV	✅ Embeddings have orders of magnitude higher than Regressor in SOBO	✅ MOBO improved HV 24% vs. SOBO-Runtime (TabNet Regressor)
	Spread	✅ Embeddings have ~99% lower than Regressor	✅ MOBO has best in 3/4 families' (75%) cases
Adasta	HV	✅ Embeddings have 37% more than Regressor	✅ MOBO improved HV 37% vs SOBO-Runtime (TabNet Regressor)
	Spread	✅ Embeddings have ~90% lower than Regressor	✅ MOBO best in 3/4 families (75%)

Reduction In Samples

Dataset	Reduction Percentage
PM100	52.8%
Adastra	70.2%

← **Legend:** ✅ supports hypothesis; ✕ does not support hypothesis

Scan to access Google Drive for full results

Findings

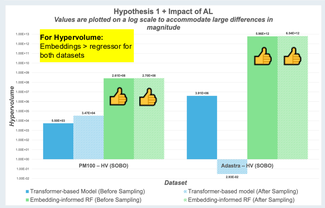


Fig. 4: Hypothesis 1 + AL Support

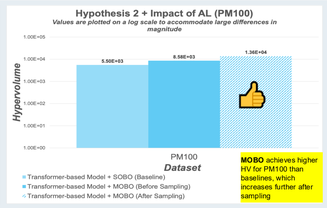


Fig. 5: Hypothesis 2 + AL Support (PM100)

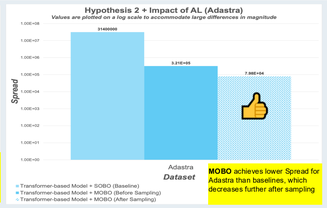


Fig. 6: Hypothesis 2 + AL Support (Adastra)

Support for Hypothesis 1:

PM100 – HV (SOBO): Embeddings reach much higher HV than TabNet.

Adastra – HV (SOBO): Embeddings raise HV by ~37% vs. TabNet, with AL steadying results. Transformer drops after sampling from skew, unlike embeddings.

Support for Hypothesis 2:

PM100 – HV (TabNet Regressor): MOBO achieves ~2.5× higher HV than SOBO, with AL further boosting performance.

Adastra – Spread (TabNet Regressor): MOBO reduces Spread by ~99.7% vs. SOBO, with AL improving stability.

Impact of Active Learning:

Across both datasets, **HV** improved for MOBO + surrogates; **Spread** gains were rarer, reflecting focus on high-value regions;

Timing overhead dropped in ~85% of sub-steps, enabling **faster, more efficient scheduling.**

MOBO Performance Variability: Possible Causes
- ❑ Surrogate-specific strengths making SOBO appear better for some single-objective metrics
- ❑ Acquisition tuning sensitivity affecting Pareto front coverage

Variability-Mitigation Possible Solutions
- ❑ Outlier-aware preprocessing
- ❑ Adaptive reference points
- ❑ Constraint & feasibility shaping
- ❑ Ensemble surrogates
- ❑ Acquisition diversification
- ❑ Stage-wise sampling budget

Observations: MOBO outperforms NSGA-II baseline in HV under similar evaluation budgets.

Future Plan

- ❑ Extend to additional datasets to test generality
- ❑ Integrate with live HPC schedulers such as RAPS to study the actual impact

References

[1] Antici, F., Seyedkazemi Ardebili, M., Bartolini, A., & Kiziltan, Z. (2023). PM100: A Job Power Consumption Dataset of a Large-Scale HPC System [Data set]. In Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis. Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W 23), Denver. Zenodo. https://doi.org/10.5281/zenodo.10127767

[2] Logs from the HPE-Cray EX system at CINES (France), including runtime, power consumption, and job metadata; ranked among the Green500 list (2024)

This material is based upon work supported by the U.S. Department of Energy, Office of Science under Award Number DE-SC0022843. Additional support was provided by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Next-Generation Scientific Software Technologies program under Contract Number DE-AC05-00OR22725.