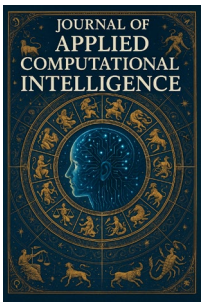
UNIVERSITY OF WEST ATTICA

eISSN: 3136-6937

Journal of Applied Computational Intelligence

Vol. 4, No. 1, January, 2026



Evaluating Pairwise Resource Contention Features in a CatBoost Model for GPU Colocation Slowdown Prediction

Barbara Danijela Galić[1,*], Bojana Tomašević[1], Adrijana Budimir[2], Karla Klarić[2]

[1] *Faculty of Computer Science, University of Dubrovnik, Dubrovnik, 20000, Croatia*

[2] *Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, 10000, Croatia*

Received 29 November 2025; Revised 16 March 2026; Accepted 21 March 2026; Published 24 March 2026

Abstract

Predicting interference before colocating graphics processing unit workloads requires a representation of how their resource demands interact. Isolated utilization measurements describe individual workloads, while shared bottlenecks and directional interference require interaction descriptors. This study develops a CatBoost regression framework that augments isolated profiles with resource products, capacity exceedance terms, and directional differences. The target is the ratio between colocated and isolated completion times for a designated victim workload. A reproducible benchmark contains 36 workload profiles, 1,260 directed workload pairs, and three aggressor activity levels, producing 3,780 observations. Unordered pair grouping prevents reversed pairs and repeated activity levels from crossing the training and test boundary. Five regressors are compared using identical partitions, while a separate workload holdout examines transfer beyond familiar profiles. The pairwise CatBoost model obtains a test mean absolute error of 0.02834 and a mean absolute percentage error of 1.631%, compared with 0.02615 mean absolute error for CatBoost using isolated descriptors alone. Its coefficient of determination is 0.97571, and its error on the workload holdout is 0.03381. Six data figures examine prediction dispersion, residual structure, feature dependence, sample efficiency, and decision sensitivity alongside aggregate accuracy. The findings characterize the proposed representation under a fixed evaluation protocol and identify the resource and response regions that merit closer assessment before deployment. The ablation does not establish an accuracy advantage for the pairwise feature expansion under the reported settings.

Keywords: GPU Colocation, Pairwise Resource Contention, Slowdown Prediction, CatBoost Regression, Workload Generalization

1. Introduction

GPU resource sharing seeks to improve cluster utilization while addressing interference, fairness, and isolation among workloads [1]. Shared-cache and memory-bandwidth contention can degrade application performance in heterogeneous CPU–GPU systems [2]. Concurrent submission mechanisms also affect individual progress through scheduling and resource sharing behavior [3]. Partition-based real-time GPU sharing can provide predictable execution for isolated software components [4]. For applications combining throughput targets with completion deadlines, admission decisions require both victim specific performance estimates and aggregate device load information. Chen et al. [5] modeled slowdown as a response to both workload characteristics and execution conditions. Workloads with similar isolated utilization can exhibit different responses to service delays because their latency hiding capacities differ. GPU resource-management studies vary computational allocations and operating frequencies to characterize workload-dependent performance and energy responses [6]. This distinction is particularly relevant when a resource intensive

* Corresponding author: galic.bd@unidu.hr

http://dx.doi.org/10.68406/jaci.2026.vol4iss1n12:160-173

➢ This is an open access article under the CC BY-NC-SA 4.0 license (https://creativecommons.org/licenses/by/4.0/).

© Authors retain all copyrights.

workload is colocated with a competitor whose demand changes over time. Average occupancy describes aggregate activity, whereas individual progress depends on workload specific sensitivity to contention. A prospective predictor therefore requires a designated victim, descriptors of its competitor, and a specified execution condition. Predicting the completion time ratio of a specified workload within a pair provides a quantitative interface between isolated profiling and placement decisions.

Existing approaches to this prediction task differ in the information used to characterize interference. Riahi et al. [7] developed an analytical GPU performance model that represents instruction processing and memory communication through explicit architectural assumptions. Their calibration becomes more demanding as interactions among memory behavior, occupancy restrictions, and workload phases increase. Performance-prediction surveys distinguish analytical, simulation-based, and machine-learning approaches, each with different information requirements [8]. Contextual optimization research distinguishes predicting uncertain parameters from optimizing the decisions that use those predictions [9]. Learned interference models also require evaluation partitions that reflect their intended use: predicting a new activity level for a familiar pair is less demanding than predicting an unseen combination of familiar workloads. Information leakage between dependent training and test observations can produce overly optimistic performance estimates [10]. Repeated configurations, reversed workload orders, and derived pair identifiers can introduce dependencies that favor memorization across evaluation partitions. Out-of-distribution detection research distinguishes familiar inputs from observations that differ from the training distribution [11]. These issues motivate a model whose feature construction, prediction target, and evaluation boundary are specified together. The research question concerns the predictive value of explicit pairwise resource features under dependency-controlled evaluation and the interpretation of errors across contention conditions.

The present study addresses this question through a controlled benchmark that combines isolated victim and aggressor descriptors with resource products, capacity exceedance terms, and signed differences. Products express simultaneous pressure; exceedance terms distinguish subcapacity combinations from combinations that cross a normalized capacity boundary; directional differences retain the identity of the workload whose slowdown is predicted. CatBoost provides the tree boosting implementation used to examine this representation under a fixed training budget. The contribution lies primarily in the representation and the evaluation design. The experimental design links resource representations to distinct evaluation questions: recombination of familiar workloads, transfer to unfamiliar profiles, and sensitivity to profile uncertainty. The paired ablation holds the learning settings constant, allowing the error change associated with the feature block to be examined directly. Comparisons with alternative regressors then assess whether that behavior depends on the learner. Residual and admission analyses extend the assessment to conditional bias and threshold decisions. The accompanying data and code document each evaluation stage through partition masks, saved predictions, and feature dependence statistics. The evaluation also retains workload direction throughout feature construction, allowing the designated victim and its competitor to contribute differently to the prediction interface and subsequent error analysis. The contribution is the joint specification of directional feature construction and dependency-controlled evaluation, rather than a new boosting algorithm or a demonstrated accuracy improvement. Section 2 reviews related work. Section 3 defines the prediction target, feature map, and regression protocol. Section 4 reports model comparisons, residual diagnostics, workload generalization, and admission sensitivity. Section 5 summarizes the findings and limitations.

2. Related Work

2.1. Resource Interference Representations

Resource based GPU performance models differ in the level at which they describe interference. Counter based approaches use utilization or throughput measurements to identify resources that constrain execution progress. GPU workload classification can group applications with similar power and performance responses using low-cost profiling [12]. These distinctions account for differences in contention sensitivity among workloads with similar utilization values. Available parallelism determines how effectively a workload can tolerate service delays, influencing its sensitivity relative to latency constrained kernels. A descriptor should therefore state the resource being observed, the normalization reference, and whether the value expresses occupancy, traffic, or a performance response. Explicit definitions preserve the physical interpretation of numerically comparable counters.

Pairwise interference characterization represents relationships between workload descriptions through joint descriptors [13]. Simultaneous demand for the same service path can be represented by products or joint occupancy terms, whereas differences can describe complementary use or asymmetric sensitivity. These features summarize demand relationships; shared service time additionally depends on arbitration behavior, and a capacity boundary estimated from isolated execution may shift under concurrency. The representation is most useful when these limitations are explicit and the fitted response is evaluated across workloads with different resource balances.

Multi-stage co-location models account for changes in interference as concurrent training workloads finish [14]. Workloads with similar average memory demand can exhibit different temporal overlap between high pressure intervals. This creates a distinction between a static compatibility prediction and a temporal interference model. The former can be suitable for coarse admission decisions, provided that the observation interval and execution scope are controlled. Work on controlled resource perturbation further motivates supplementing demand proxies with sensitivity measurements [15]. Such measurements enrich the feature representation at a profiling cost determined partly by the device's available resource controls. The present method retains a static interface so that its information requirements are transparent and its limitations can be tested independently of phase detection.

2.2. Learning Based Slowdown Prediction

Tree based regressors provide a flexible mapping from resource descriptors to continuous performance targets. Boosted tree research describes how successive trees fit residual structure while regularization constrains model complexity [16]. This model family accommodates interactions and thresholds whose functional forms require estimation from data. CatBoost uses ordered target statistics and ordered boosting to mitigate biased prediction during learning [17]. Appropriate partitioning remains necessary to control dependencies across training and test observations. Assigning repeated observations of the same pair to training and test sets can inflate transfer estimates through memorization of familiar combinations. Algorithmic safeguards and evaluation safeguards address different problems. The use of CatBoost does not by itself establish that ordered boosting was active in this numerical-only CPU experiment; the resolved training configuration must be checked separately.

GPU performance-prediction research distinguishes platform-dependent profiling from platform-independent program representations for modeling execution across devices [18]. These alternatives motivate separate evaluation of familiar configurations, unfamiliar workloads, and unfamiliar devices. Holding out pair identities tests whether known workload profiles can be recombined; holding out workloads tests whether the model can operate on unfamiliar profiles. Holding out an architecture adds a further shift in counter meaning and capacity normalization. Each question consequently requires a corresponding partitioning unit. Reported performance should also identify whether profiling and preprocessing are repeated inside each training partition, particularly when category statistics or data dependent normalizers are used.

Real-time GPU inference systems coordinate latency-critical and best-effort tasks to combine low latency with efficient resource use [19]. A symmetric error metric can obscure asymmetric decision consequences: underestimating a harmful slowdown can admit an unsafe pair, while overestimating it can reject a useful pairing. Threshold sweeps, residual distributions, and explicit workload grouping expose the decision implications of a regression model. The proposed method therefore combines a conventional regression comparison with residual and admission analyses. Its narrower objective is to examine whether explicit contention features improve the information available to a CatBoost predictor under a controlled evaluation protocol.

3. Pairwise Contention CatBoost Method

3.1. Prediction Target and Profiling Interface

A prediction instance contains one designated victim workload, one aggressor workload, and an aggressor activity setting. The victim identity determines whose completion time is modeled. Reversing the workload roles produces a distinct directed observation; the two directions share an unordered pair identifier for partitioning. The isolated and colocated completion times must correspond to the same victim input, precision, warmup policy, and timing boundary. The slowdown equation normalizes the colocated completion time by the isolated baseline:

$$s(i,j,q) = \frac{T_{shared}(i,j,q)}{T_{solo}(i)} \tag{1}$$

In the slowdown equation, i denotes the victim, j denotes the aggressor, and q specifies the aggressor activity setting. A consistent timing protocol requires synchronized boundaries and repeated trials for both completion times. A target above unity denotes a longer completion time under colocation. The evaluation retains the learner's continuous output, including predictions near the lower target boundary, so that extrapolation behavior remains observable.

The isolated descriptor contains five normalized components: compute activity, memory demand, cache pressure, occupancy pressure, and phase burstiness. Each component is dimensionless and bounded within the benchmark profile range. Operational profiling requires a documented counter definition for compute activity, a calibrated throughput reference for memory demand, and a specified miss or traffic statistic for cache pressure. Occupancy pressure distinguishes achieved occupancy from sensitivity to reduced residency. Burstiness requires a fixed sampling window and a consistent rule for summarizing temporal variation. Every component is available through the isolated profiling

interface. These quantities define a benchmark interface; without device-specific counter mappings and acquisition records, they do not establish that the descriptors have been validated as operational GPU measurements.

Figure 1 shows the information boundary from isolated profiling through pair formation, grouped training, and prospective slowdown prediction. The offline path stores profile descriptors and training labels separately; the online path reads descriptors and the requested activity setting. A blocked connector identifies the separation between outcome labels and online features. This separation supports prediction before placement, while colocated performance counters remain available for subsequent diagnosis and model updates.

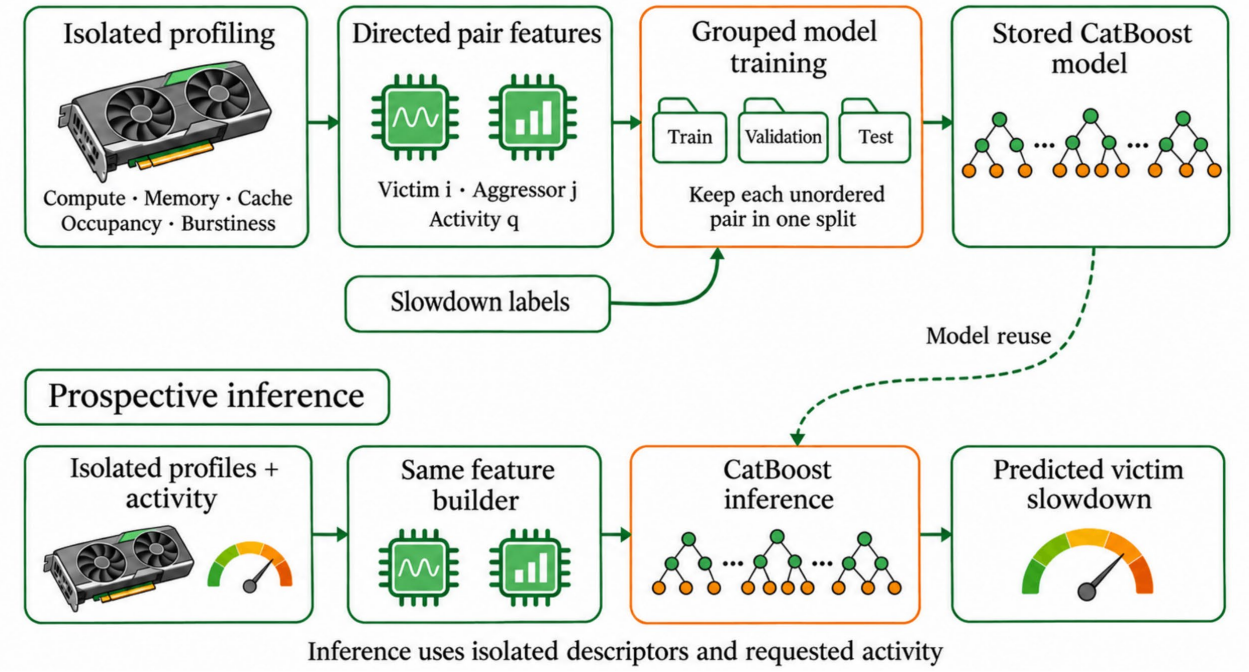


Figure 1. Offline profiling and prospective prediction workflow with an explicit boundary between outcome labels and inference features

3.2. Pairwise Contention Feature Construction

For every resource component, the feature builder produces three interaction descriptors. The product represents simultaneous demand, the exceedance term describes demand above a normalized capacity reference, and the signed difference preserves the imbalance between victim and aggressor. For resource component r, these transformations are defined jointly: The unit boundary is a feature-construction reference, not a verified physical saturation threshold for every descriptor, particularly occupancy pressure and burstiness.

$$p_r = u_{ir}u_{jr}, e_r = max(0, u_{ir} + u_{jr} - 1), d_r = u_{ir} - u_{jr} \quad (2)$$

In this equation, u_{ir} and u_{jr} are the normalized resource descriptors of the victim and aggressor. The product increases smoothly as both workloads demand the same resource. The exceedance term remains zero until their combined normalized demand passes unity, after which it increases linearly. The signed difference distinguishes a resource intensive victim paired with a lighter aggressor from the reverse arrangement. Retaining the isolated descriptors preserves the absolute demand levels associated with these relationships. The complete input contains one activity setting, ten isolated descriptors, and fifteen pairwise descriptors. All comparison models except the isolated CatBoost ablation receive this complete feature vector. The feature equations use the isolated descriptors directly, while the activity setting enters as a separate numerical input; the equations do not specify activity-dependent rescaling of the aggressor profile.

Figure 2 depicts the feature fusion and regression architecture, including the ordered victim and aggressor inputs, three parallel interaction operators, concatenated numerical vector, and boosted tree output. A separate training branch supplies labels to the fitting objective, while the inference branch produces a continuous slowdown estimate. Resource names remain consistent across the isolated and interaction blocks, allowing individual counters to be mapped to their corresponding transformations. Workload identifiers determine evaluation groups, while resource descriptors supply the model inputs.

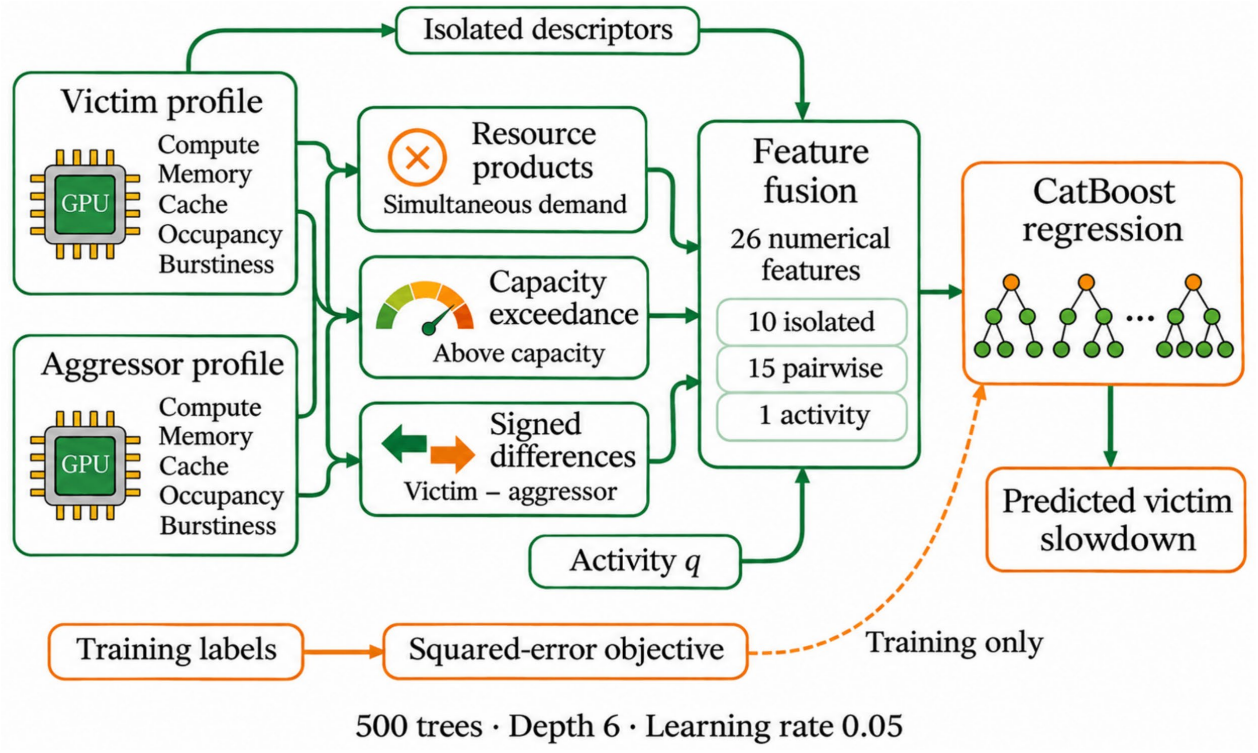


Figure 2. Directional resource feature fusion and CatBoost regression architecture with separate label and inference paths

The descriptor interface fixes feature scaling before partitioning, preserving the separation between calibration and held out observations. When descriptors require calibration, the reference must be established independently or estimated from training data. The feature builder operates exclusively on numerical inputs. Extending the representation to categorical workload attributes would require a dedicated assessment of encoding choices and transfer behavior. An independently fixed reference differs from fitting a normalizer on all profiles; data-dependent normalization must be fitted on training data and then applied unchanged to held-out observations.

3.3. Regression and Evaluation Safeguards

The predictor combines fixed depth regression trees through an additive ensemble. Its prediction equation expresses the estimated slowdown as an initial response plus the accumulated tree contributions:

$$\hat{s}(\boldsymbol{x}) = b + \eta \sum_{m=1}^{M} h_m(\boldsymbol{x}) \tag{3}$$

The input vector is denoted by $\mathbf{x}$, the initial response by b, the learning rate by η, and the contribution of tree m by h_m. The benchmark implementation uses 500 trees, depth six, a learning rate of 0.05, and random seed 42. Training runs on the CPU with four worker threads. The boosting type follows the CPU library default recorded in the reproducibility environment, and all model inputs are numerical. These settings are fixed before inspecting the test response. A reserved validation partition remains available for subsequent tuning.

Training combines squared prediction deviations with regularization of leaf values. The following equation provides a schematic description of this fitting criterion:

$$\mathcal{L} = \frac{1}{N}\sum_{n=1}^{N} (s_n - \hat{s}_n)^2 + \lambda \sum_{\ell \subset \mathcal{A}} w_\ell^2 \tag{4}$$

The training sample count is N, and s_n and $\hat{s}_n$ denote the target and prediction for observation n. The set $\mathcal{A}$ contains the regularized leaf values w_ℓ, with λ controlling their penalty. This formulation describes the relationship between fitting error and model regularization.

The baseline set comprises ridge regression, a random forest, histogram gradient boosting, and CatBoost using the activity setting and isolated descriptors. Comparing the isolated CatBoost configuration with the complete model examines the effect of the pairwise feature block while holding tree depth, training iterations, learning rate, and random seed constant. Comparisons among models receiving the complete representation assess the contribution of estimator

choice under a common input interface. Training time is recorded as a reproducibility statistic; prospective scheduling overhead requires separate inference and profiling measurements.

The primary evaluation groups both workload directions and all activity settings belonging to an unordered pair. Training, validation, and test sets contain disjoint pair groups. A second partition holds out entire workload identities to assess transfer to unfamiliar profiles. Prediction accuracy is evaluated principally through the mean absolute error equation: Pair grouping prevents the same pair from crossing partitions, but shared workload identities still create dependence across distinct pair groups; it is not a test of fully independent workload populations.

$$MAE = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} /s_n - \hat{s_n}/ \tag{5}$$

In the mean absolute error equation, N_{test} is the number of test observations. Mean absolute error retains the units of the slowdown ratio, supporting interpretation of individual prediction deviations. Root mean squared error emphasizes larger deviations, percentage error expresses deviations relative to the target, and the coefficient of determination compares residual variation with target variation. For example, slowdown 1.50 means a 50% increase in completion time relative to isolation; an absolute ratio error of 0.05 is distinct from a 5% relative prediction error.

Placement assessment additionally requires a decision threshold and an evaluation of the consequences of individual errors. The subsequent analysis therefore reports accepted coverage and the proportion of accepted observations whose reference slowdown exceeds the selected limit. Admission decisions depend on model predictions; reference responses are consulted afterward to evaluate those decisions.

4. Experimental Evaluation and Analysis

4.1. Benchmark Characteristics and Model Comparison

The benchmark contains 36 five component workload profiles and 3,780 directed observations across activity settings of 0.25, 0.50, and 0.75. Reporting checklists for machine learning recommend documenting data sources, reference standards, and validation procedures [20]. Figure 3(a) displays the reference slowdown distribution, including its concentration around moderate interference and its upper tail. On the 756-observation pair test set, slowdown ranges from 1.182 to 2.327, with a median of 1.700 and an interquartile interval of 1.553–1.894. These values establish the response scale against which prediction errors should be interpreted. The 1.50 reference intersects the lower part of the distribution, so an admission rule at that value evaluates a relatively selective operating region. The benchmark statistics alone do not establish hardware validity: the reported scope lacks the device-specific acquisition and timing evidence needed to verify deployment performance.

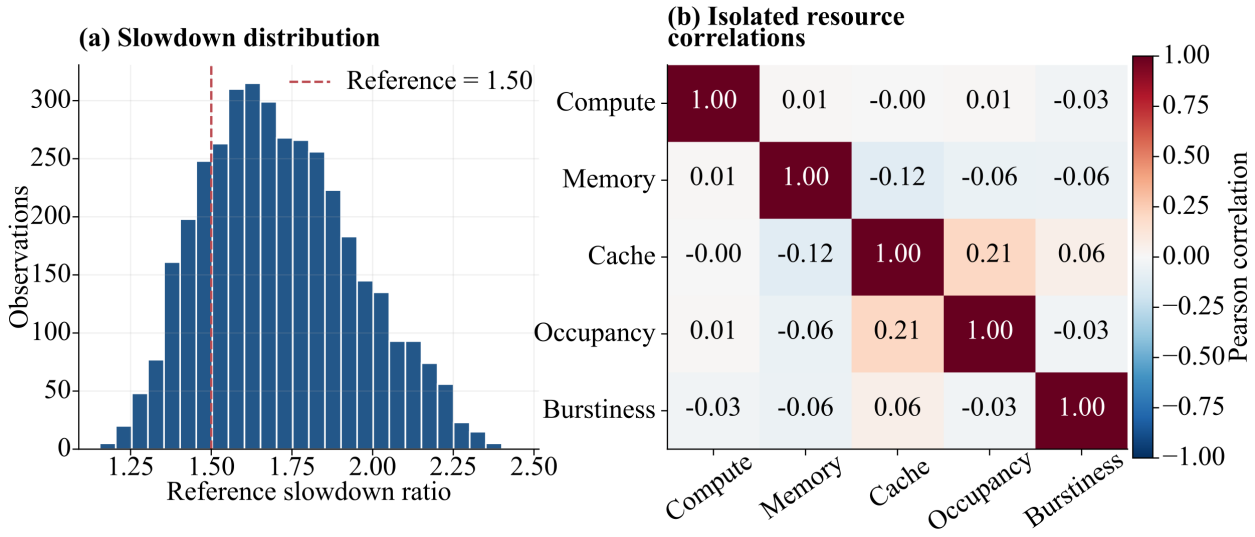


Figure 3. Benchmark characteristics. (a) Slowdown distribution with a 1.50 reference. (b) Correlation of the five isolated resource descriptors

Figure 3(b) shows correlations between the five isolated descriptors using the 36 profiles once each. Compute and memory demand have a correlation close to 0.01, while cache and occupancy pressure have a correlation of approximately 0.21. The weak profile correlations provide a setting in which several resource dimensions vary separately, helping distinguish the effects of complementary resource descriptions. Fine-grained GPU profiling distinguishes streaming-multiprocessor activity from cache and global-memory behavior and resolves their variation

over time [21]. The descriptors are dimensionless quantities representing compute activity, memory demand, cache pressure, occupancy pressure, and burstiness. Their joint distribution defines which contention combinations the predictor encounters and determines the coverage available for estimating conditional effects.

Dependent workload observations require grouping at the level of the intended generalization question [22]. The main partition assigns 378 unordered pairs to training and 126 pairs each to validation and testing, producing 2,268, 756, and 756 observations. Both directions and all activity settings remain within their assigned pair group. Familiar workload profiles consequently recur across partitions in different combinations, which makes the test an assessment of pair recombination. The validation partition is reserved for later tuning, and all reported models use predetermined settings. The static descriptor interface fixes the temporal resolution of the analysis. Transfer to a broader execution population requires consistent counter definitions, resource normalization, and timing boundaries. There are $36 \times 35 / 2 = 630$ unordered pairs, each contributing two directions and three activity levels, giving six observations per group and 3,780 observations overall.

Table 1 compares five regressors using the same pair test set. Figure 4(a) presents their absolute and root mean squared errors separately. The full CatBoost model achieves MAE 0.02834 and RMSE 0.03730, compared with 0.02615 and 0.03401 for the isolated descriptor ablation. The added feature block therefore increases MAE by 8.35% under the fixed learning settings. Ridge regression reaches MAE 0.02616, whereas histogram boosting and the random forest reach 0.03850 and 0.05136. Fair estimator comparisons require explicit control of input information and tuning budgets [23]. The ridge result indicates that the engineered representation approximates much of the response through a linear combination of the engineered features; CatBoost's additional partitioning flexibility offers limited benefit under this response variability and sample coverage. The MAE difference between ridge and isolated CatBoost is only 0.00001 at the reported precision, and no repeated-split uncertainty is reported; this comparison does not establish a statistically reliable ranking.

Table 1. Computed regression metrics on the held-out pair test set

Model	MAE	RMSE	MAPE (%)	R squared
Ridge	0.02616	0.03264	1.544	0.98139
Random Forest	0.05136	0.06801	2.933	0.91924
Histogram Boosting	0.03850	0.05203	2.195	0.95274
CatBoost Solo	0.02615	0.03401	1.525	0.97981
CatBoost Pair	0.02834	0.03730	1.631	0.97571

Figure 4(b) plots the 756 full model predictions against their reference targets, with an identity line and a band of ±0.05. The coefficient of determination is 0.97571, indicating that the predictor accounts for most variation across the response range. The upper response tail exhibits larger negative residuals, consistent with compression of extreme predictions toward more densely represented central values. Figure 4(c) quantifies the individual deviations: median absolute error is 0.02184, the 90th percentile is 0.05849, and the maximum is 0.17921. The difference between MAE and the maximum error is operationally relevant because placement decisions act on individual pairs. Strong aggregate fit can coexist with substantial prediction errors for sparsely represented combinations.

Figure 4(d) compares empirical cumulative absolute error for all five regressors. For the pairwise CatBoost model, 85.19% of observations fall within 0.05 error, while its 95th percentile error is 0.07265. The isolated CatBoost and ridge curves remain competitive throughout much of the central error range, consistent with their lower MAE values. The forest's broader tail corresponds to its larger RMSE and indicates occasional larger departures from the reference response. All complete feature baselines receive 26 numerical inputs; the isolated CatBoost receives the activity setting and ten isolated descriptors. The fixed hyperparameters define a controlled learning budget, and the observed ranking can change when training coverage, response characteristics, or estimator regularization changes. Predetermined settings control the reported comparison but do not establish equal tuning effort, equal computational cost, or the best achievable accuracy for each estimator.

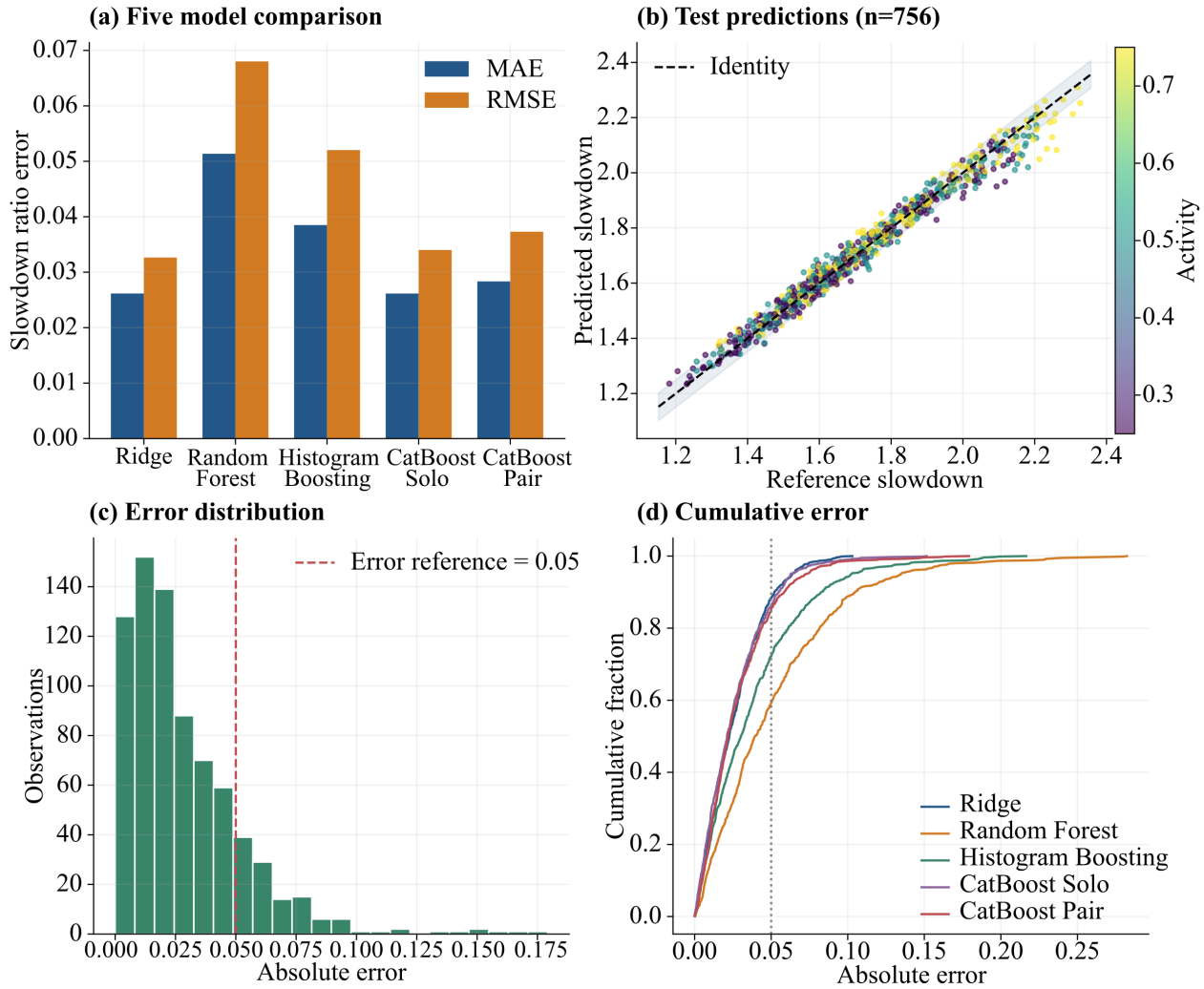


Figure 4. Model comparison and prediction fidelity. (a) MAE and RMSE by model. (b) Reference and predicted slowdown. (c) Absolute error distribution. (d) Cumulative absolute error for five regressors

4.2. Conditional Residuals and Feature Dependence

Figure 5(a) displays signed residuals against reference slowdown, with zero indicating exact agreement between an individual prediction and its target and a shaded interval marking ±0.05. Residual-plot diagnostics can identify model departures that conventional residual tests fail to detect [24]. The overall mean residual is −0.00164, whereas the mean residual in the highest slowdown quintile is −0.02918. The contrast between these residual means identifies systematic underprediction in the upper response region despite a near zero global mean. The same region also contains several of the largest absolute deviations. Tree predictions average outcomes within fitted leaves, so combinations near the response boundary can receive estimates influenced by more frequent interior observations. This leaf averaging behavior offers a plausible explanation of the fitted model's conditional residual pattern. Here, the signed residual is predicted slowdown minus reference slowdown, so negative residuals denote underprediction.

Figure 5(b) separates residuals across activity settings of 0.25, 0.50, and 0.75, with 252 observations in each group. Violin envelopes show distributional shape, jittered points display individual observations, and median and quartile markers locate the central mass. All three groups contain both positive and negative errors, while downward outliers extend farther than upward outliers. The lowest residual reaches approximately −0.18, compared with positive deviations below 0.10. Regression model validation can examine conditional means and conditional variances as distinct properties of predictive performance [25]. A common vertical scale supports comparison across activity groups, while equal group sizes control the sampling contribution to their density profiles.

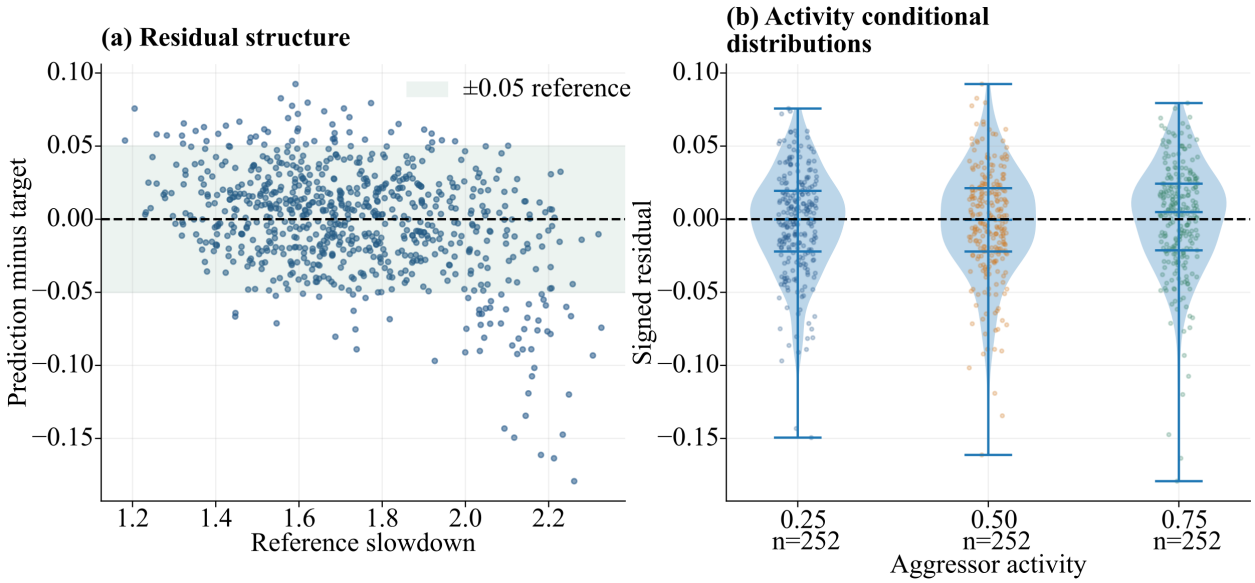


Figure 5. Residual diagnostics. (a) Signed error across reference slowdown. (b) Signed error distributions at three aggressor activity settings

Variation in resource combinations within each activity group motivates joint analysis of activity and resource dependent errors. The shared test design includes both directions for every pair at each activity level, supporting paired analysis of direction dependent errors. Such an analysis would compare the two residuals within the same pair and activity condition before aggregating across groups. The full observation distribution associates elevated errors more strongly with the upper response tail than with a uniform shift across activity settings. Horizontal jitter is determined by a separate fixed seed and preserves the residual values. For an admission threshold near the upper response range, negative residuals warrant particular attention because underprediction can cause admission of a combination whose slowdown exceeds the selected limit.

Figure 6(a) ranks the ten largest native CatBoost feature importance values. The memory product contributes 22.92%, cache exceedance 19.23%, memory exceedance 12.69%, activity 9.91%, and the cache product 8.99%. These five descriptors account for approximately 73.73% of the reported importance. Attribution methods describe dependence within a fitted predictor and require separate assumptions for causal interpretation [26]. The strong memory product ranking identifies joint memory demand as a prominent predictive input within the fitted model. The importance of memory exceedance, which is correlated with the product and isolated demands, also shows that predictive credit can be distributed across related transformations. The ranking therefore characterizes the learned approximation under this representation and training sample. These percentages describe attribution within the fitted predictor; they are neither measured shares of resource contention nor estimates of a causal hardware contribution.

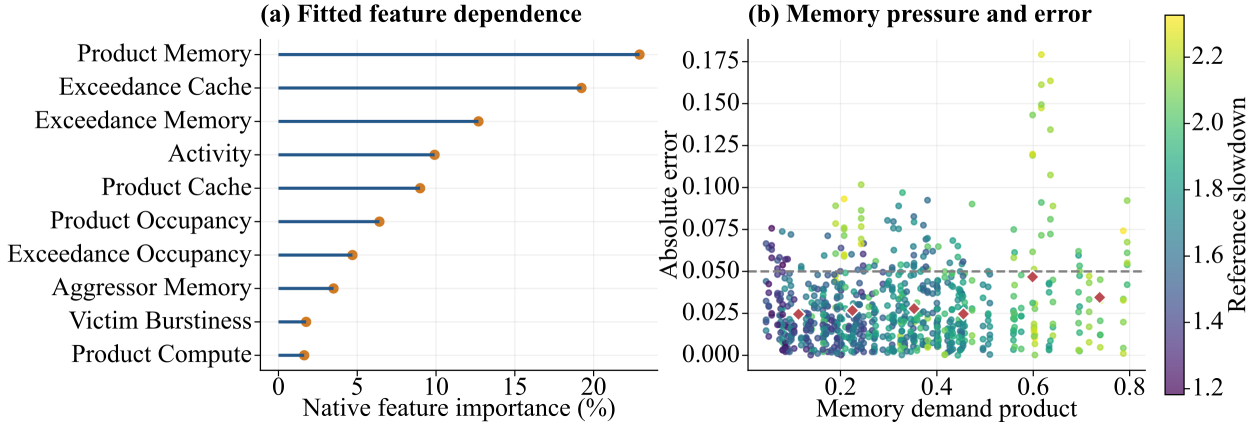


Figure 6. Feature dependence and error localization. (a) Ten highest CatBoost feature importance values. (b) Absolute error versus memory demand product with binned means

Figure 6(b) relates absolute error to the memory demand product, with reference slowdown encoded by color and six binned means marked by diamonds. Most errors remain below 0.05, with larger deviations concentrated among high slowdown observations at memory products near 0.60. Accurate predictions at the highest memory products indicate that error depends on the joint resource configuration. Feature interaction analysis in tree models examines how a

descriptor's predictive effect changes with the values of other inputs [27]. Cache exceedance, occupancy pressure, and the burstiness–memory interaction provides additional coordinates for describing response variation. Their combinations can produce different errors at similar memory product values.

The isolated inputs and pairwise descriptors encode the same underlying profile information through different coordinates. Product features explicitly represent smooth interactions, while exceedance terms identify capacity thresholds for shallow tree models. Their benefit therefore depends on sample coverage and the learner's capacity to approximate the same relationships from isolated values. The observed ablation shows a small error increase after this coordinate expansion under fixed settings. Additional deterministic features can introduce alternative split candidates and distribute observations across different leaves, changing bias and variance. Repeated grouped partitions and feature block ablations would help separate that training effect from a persistent representation advantage. Correlated profiling noise can propagate through products and exceedance terms, potentially amplifying uncertainty near capacity boundaries. The pairwise block is a deterministic transformation of the isolated profiles and therefore adds representational flexibility rather than additional measured information.

4.3. Workload Generalization and Admission Sensitivity

Figure 7(a) compares training sample requirements with pair and workload holdout reference errors. Training prefixes of 100, 300, 600, 1,200, and 2,268 observations produce MAE values of 0.15016, 0.09653, 0.04567, 0.03329, and 0.02834. Error decreases by approximately 81.1% between the smallest and largest budgets, with the largest absolute improvements occurring before 600 observations. Learning curves relate predictive performance to a resource budget, such as the number of training observations [28]. Because the prefixes follow workload enumeration, sample quantity and domain coverage vary jointly. The learning curve is therefore conditional on the specified acquisition order. Repeated sampling of complete pair groups would provide a more general estimate of expected sample efficiency and its variability. The prefix curve does not estimate an order-independent sample requirement, and the reduction in error cannot be attributed to sample count alone.

The workload holdout uses training pairs whose two identities are below 28 and test pairs containing an identity of at least 30. This identity separation evaluates unfamiliar profiles, whereas the primary split evaluates unfamiliar combinations of familiar profiles. The refitted CatBoost model reaches MAE 0.03381, approximately 19.3% above the primary pair holdout error. Figure 7(b) tests profile observation sensitivity using multiplicative noise amplitudes of 0%, 5%, 10%, 15%, and 20% on the memory descriptors. The fixed model's MAE rises from approximately 0.028 to 0.034, 0.045, 0.058, and 0.071. Causal-inference frameworks distinguish observational associations from interventions on the system under study [29]. Holding the target fixed while recomputing interaction features isolates the propagation of profiling error. Under this rule, identities 28 and 29 are excluded from training, while the test rule can include both familiar–unfamiliar and unfamiliar–unfamiliar pairs; aggregate MAE does not resolve these cases separately.

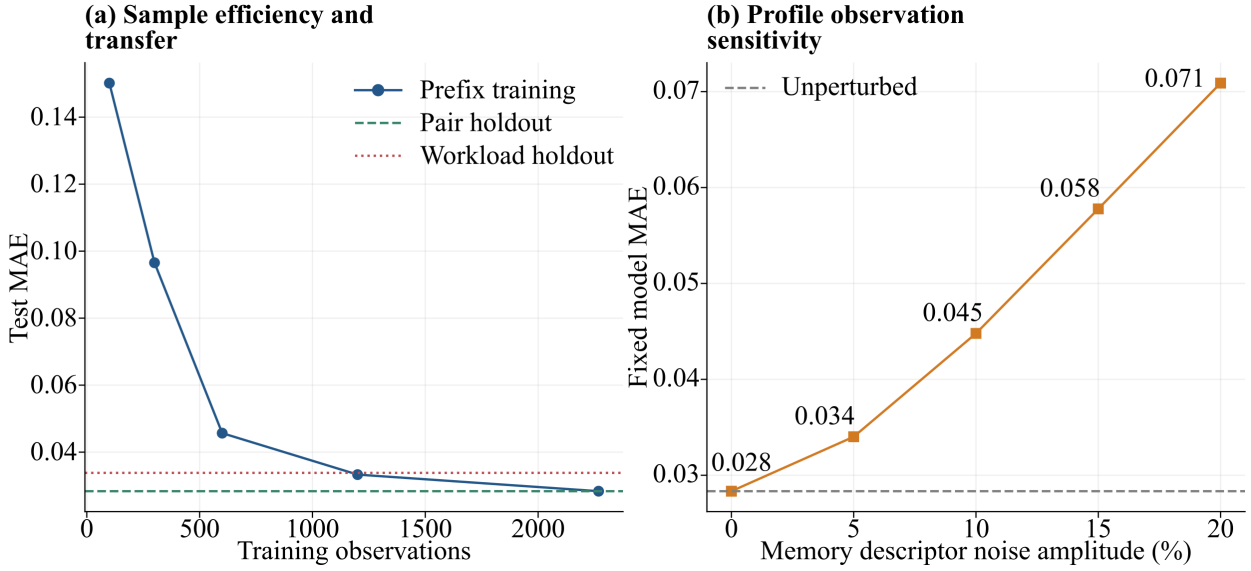


Figure 7. Generalization and profiling sensitivity. (a) Training sample learning curve with pair and workload holdout references. (b) Error under five memory descriptor perturbation amplitudes

The perturbation direction is fixed across amplitudes, which isolates noise magnitude within this stress test. The effect increases because perturbed memory values enter the isolated descriptors, the memory product, and the capacity exceedance calculation simultaneously. Capacity exceedance also changes piecewise when combined demand crosses unity, so measurement error can move an observation between distinct input regions. The workload holdout and

perturbation experiment address separate risks: unfamiliar profile combinations and inaccurate profile acquisition. A hardware protocol should accordingly report synchronized timing repeats, sampling windows, and resource normalization references. The single workload identity split provides an initial transfer assessment; repeated workload folds would quantify how strongly the score depends on the particular profiles assigned to the unfamiliar population. This fixed-direction input perturbation is a profiling-noise stress test, not a physical intervention on GPU demand or an estimate averaged over independent noise realizations.

Figure 8(a) sweeps ten admission thresholds from 1.20 to 2.10 and reports accepted coverage together with violations among accepted observations. Decision-focused learning evaluates prediction methods through the quality of the downstream decisions they support [30]. Coverage grows as the threshold increases, while the violation rate varies with the local prediction error near each boundary. Figure 8(b) resolves the decision counts at 1.50: 119 acceptable observations are admitted, 10 harmful observations are admitted, 11 acceptable observations are rejected, and 616 harmful observations are rejected. Accepted coverage is therefore 17.06%, and the violation rate among accepted observations is 7.75%. The admission counts quantify the decision consequences of regression errors for observations near the selected boundary. At this threshold, coverage is (119 + 10) / 756 = 17.06%, and the violation rate among admitted observations is 10 / 129 = 7.75%; the rejected acceptable fraction is 11 / 130 = 8.46%.

Figure 8(c) divides the reference response into five quantile groups and shows their mean absolute errors with one standard deviation intervals. The group means are 0.02494, 0.02581, 0.02581, 0.02350, and 0.04154. The highest response group has approximately 76.8% more mean error than the preceding group and also exhibits a wider dispersion. This pattern is consistent with the negative residuals at high slowdown and identifies a response region requiring closer assessment of admission uncertainty. Figure 8(d) relates error to victim burstiness and marks the 0.05 reference. Large deviations occur at some intermediate and higher burstiness values, while many observations at those same values remain accurate. Burstiness therefore acts through its interaction with aggressor demand and the remaining resource conditions. The one-standard-deviation intervals describe within-group error dispersion, not confidence intervals for the group means or uncertainty across retrained models.

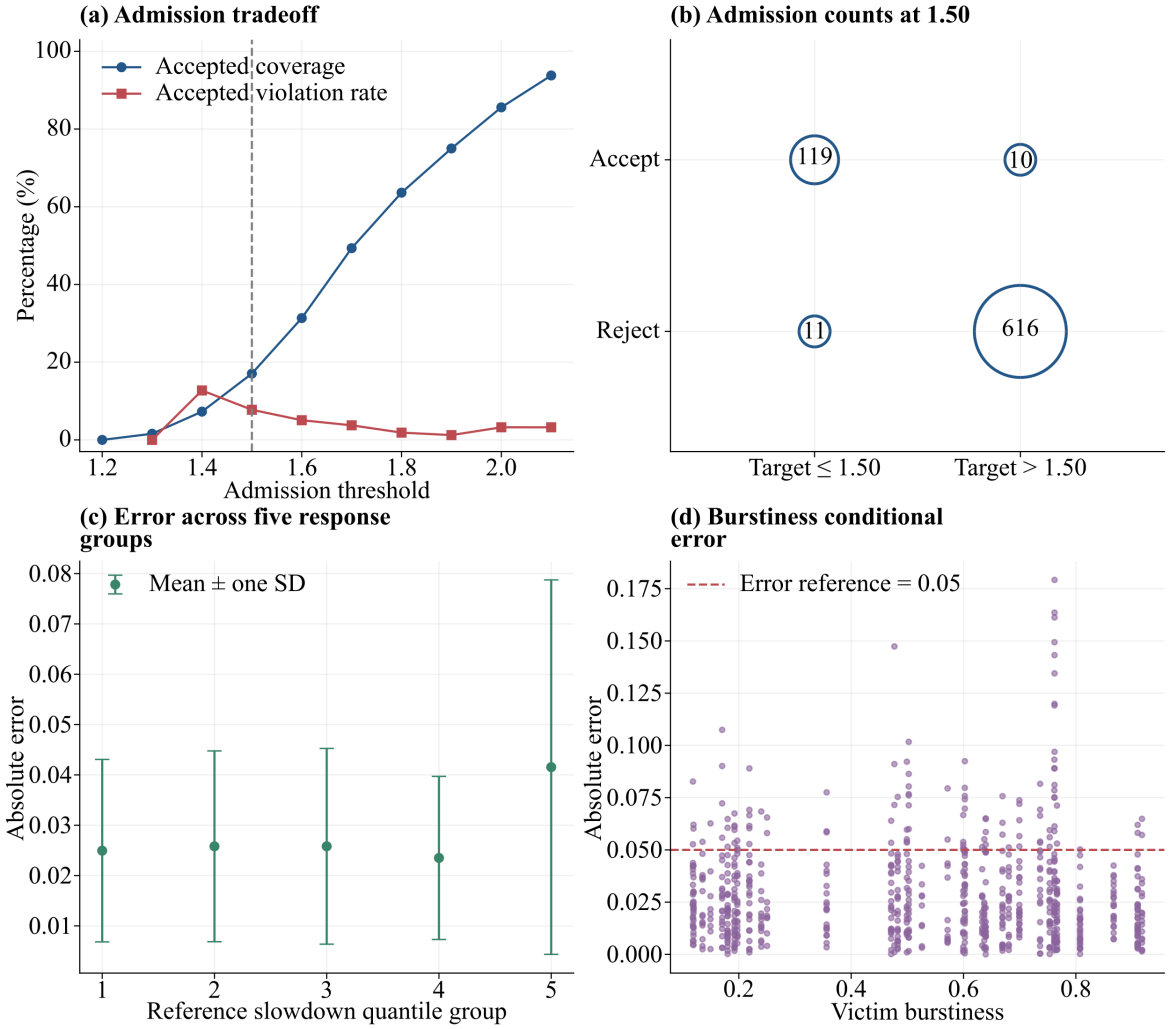


Figure 8. Decision and subgroup behavior. (a) Coverage and violation tradeoff. (b) Admission confusion matrix at 1.50. (c) Error in five slowdown quantile groups. (d) Error versus victim burstiness

At the 1.50 boundary, the ten harmful admissions and eleven conservative rejections have comparable frequencies and application dependent costs. Latency constrained services may assign a greater penalty to harmful admission, whereas throughput-oriented batch systems may permit a specified rate of bounded violations. Threshold choice should therefore incorporate the magnitude of excess slowdown, rejection cost, and uncertainty in the isolated profiles. The present decision analysis uses saved predictions from one fixed model, allowing all threshold comparisons to share the same evaluation population. A deployment protocol could add a margin to the predicted slowdown and estimate its calibration on a separate validation population. Profiles outside the calibrated resource range would require additional measurement before the margin could support a reliable admission decision. No profiling or inference latency measurement is reported here, so the admission analysis does not establish a net scheduling benefit or a deadline guarantee.

5. Conclusion

This study developed and evaluated a CatBoost model incorporating pairwise resource contention features for GPU colocation slowdown prediction, characterizing benchmark prediction accuracy and the limitations of explicit interaction features.

1. The proposed model achieved a mean absolute error of 0.02834, a root mean squared error of 0.03730, and a mean absolute percentage error of 1.631%, with a coefficient of determination of 0.97571. Absolute prediction error remained within 0.05 for 85.19% of the pair holdout observations.

2. The isolated descriptor CatBoost baseline achieved a lower mean absolute error of 0.02615; adding pairwise features increased this error by 8.35% under identical learning settings. Workload holdout error reached 0.03381, approximately 19.3% above pair holdout error. These comparisons indicate that feature expansion and transfer to unfamiliar workloads require separate assessment, across models with high aggregate predictive accuracy.

3. The framework combines directional resource descriptors with grouped evaluation that keeps reversed pairs and repeated activity settings within the same partition. Its applicability depends on workload coverage and profiling consistency. The highest slowdown quintile exhibited a mean signed residual of −0.02918, identifying systematic underprediction in the region most relevant to harmful admission decisions.

The framework provides a reproducible basis for evaluating contention representations and placement sensitivity; further work should broaden workload coverage, incorporate phase dependent descriptors, and calibrate prediction uncertainty.

6. Declarations

6.1 Author Contributions

Barbara Danijela Galić contributes to conceptualization, methodology, software, validation, analysis, investigation, data collection, draft preparation, manuscript editing, visualization, supervision, project administration, and funding acquisition. Karla Klarić contributes to conceptualization, methodology, software, validation, analysis. Bojana Tomašević contributes to validation, analysis, investigation, data collection, draft preparation. Adrijana Budimir contributes to conceptualization, validation, analysis. All authors have read and agreed with the manuscript before its submission and publication.

6.2 Funding and Acknowledgments

This research received no specific financial support from any funding agency.

6.3 Institutional Review Board Statement

Not applicable.

6.4 Informed Consent Statement

Not applicable.

6.5 Date Availability

Not applicable.

6.6 Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

6.7 Declaration of generative AI and AI-assisted technologies

During the preparation of this manuscript, the authors utilized Grammarly for linguistic polishing, logical organization and draft revision. The generative AI tool was not involved in research design, data collection, data analysis, or the equation of core conclusions. All outputs generated by AI were reviewed, revised and verified manually by the authors. The authors take full responsibility for all content of this manuscript.

7. References

[1] Huang, J., Lin, W., Wu, W., Wang, Y., Zhong, H., Wang, X., & Li, K. (2025). On efficiency, fairness and security in AI accelerator resource sharing: A survey. ACM Computing Surveys, 57(9), 1–35. https://doi.org/10.1145/3721427

[2] Bechtel, M. G., & Yun, H. (2024). Analysis and mitigation of shared resource contention on heterogeneous multicore: An industrial case study. IEEE Transactions on Computers, 73(7), 1753–1766. https://doi.org/10.1109/tc.2024.3386059

[3] Han, Z., Zhou, R., Xu, C., Zeng, Y., & Zhang, R. (2024). InSS: An intelligent scheduling orchestrator for multi-GPU inference with spatio-temporal sharing. IEEE Transactions on Parallel and Distributed Systems, 35(10), 1735–1748. https://doi.org/10.1109/tpds.2024.3430063

[4] Ali, S. W., Tong, Z., Goh, J., & Anderson, J. H. (2024). Predictable GPU sharing in component-based real-time systems. In R. Pellizzoni (Ed.), 36th Euromicro Conference on Real-Time Systems (ECRTS 2024) (Leibniz International Proceedings in Informatics, Vol. 298, pp. 15:1–15:22). Schloss Dagstuhl–Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPIcs.ECRTS.2024.15

[5] Chen, S., Wang, D., Zhang, W., Hao, M., Tian, Y.-C., & He, H. (2026). Performance prediction of concurrent DNN training tasks in GPU spatial sharing environments. ACM Transactions on Architecture and Code Optimization, 23(2), 1–25. https://doi.org/10.1145/3807454

[6] Jiang, Z., Cai, Z., Zhao, H., Zhang, B., Wu, T., Qiang, Y., Ma, R., Guan, H., & Buyya, R. (2026). HeShare: Energy-aware and efficient multi-task GPU sharing in heterogeneous GPU-based computing systems. IEEE Transactions on Computers, 75(3), 776–787. https://doi.org/10.1109/tc.2025.3628924

[7] Riahi, A., Savadi, A., & Naghibzadeh, M. (2024). Many-BSP: An analytical performance model for CUDA kernels. Computing, 106(5), 1519–1555. https://doi.org/10.1007/s00607-023-01255-w

[8] Gu, B., Zhao, L., Li, C., & Chi, X. (2026). Review and analysis of performance prediction methods and tools for heterogeneous parallel programs. CCF Transactions on High Performance Computing, 8(3), 331–351. https://doi.org/10.1007/s42514-025-00236-z

[9] Sadana, U., Chenreddy, A., Delage, E., Forel, A., Frejinger, E., & Vidal, T. (2025). A survey of contextual optimization methods for decision-making under uncertainty. European Journal of Operational Research, 320(2), 271–289. https://doi.org/10.1016/j.ejor.2024.03.020

[10] Bernett, J., Blumenthal, D. B., Grimm, D. G., Haselbeck, F., Joeres, R., Kalinina, O. V., & List, M. (2024). Guiding questions to avoid data leakage in biological machine learning applications. Nature Methods, 21(8), 1444–1453. https://doi.org/10.1038/s41592-024-02362-y

[11] Yang, J., Zhou, K., Li, Y., & Liu, Z. (2024). Generalized out-of-distribution detection: A survey. International Journal of Computer Vision, 132(12), 5635–5662. https://doi.org/10.1007/s11263-024-02117-4

[12] Jain, R., Jiang, Y., Sinclair, M. D., & Venkataraman, S. (2026). Minos: Systematically classifying performance and power characteristics of GPU workloads on HPC clusters. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 10(2), 1–27. https://doi.org/10.1145/3805644

[13] Liu, J., Cai, Z., Liu, Y., Li, H., Zhang, Z., Ma, R., & Buyya, R. (2025). SMore: Enhancing GPU utilization in deep learning clusters by serverless-based co-location scheduling. IEEE Transactions on Parallel and Distributed Systems, 36(5), 903–917. https://doi.org/10.1109/tpds.2025.3548320

[14] Liu, Z., Cheng, Y., Chen, C., Hu, J., Fu, R., & Zhang, D. (2025). ISACPP: Interference-aware scheduling approach for deep learning training workloads based on co-location performance prediction. IEEE Transactions on Parallel and Distributed Systems, 36(8), 1591–1607. https://doi.org/10.1109/tpds.2025.3577796

[15] Hao, M., Wu, Z., Tian, X., Yang, S., Guo, G., Yang, H., He, H., Wang, Y., Wang, F., Wang, D., & Zhang, W. (2026). PctoDL: Adaptive GPU throughput optimization for deep learning inference with power constraints. ACM Transactions on Architecture

and Code Optimization, 23(2), 1–25. https://doi.org/10.1145/3805802

[16] Emami, S., & Martínez-Muñoz, G. (2025). Condensed-gradient boosting. International Journal of Machine Learning and Cybernetics, 16(1), 687–701. https://doi.org/10.1007/s13042-024-02279-0

[17] Lin, S., Liang, Z., Dong, M., Guo, H., & Zheng, H. (2024). Imbalanced rock burst assessment using variational autoencoder-enhanced gradient boosting algorithms and explainability. Underground Space, 17, 226–245. https://doi.org/10.1016/j.undsp.2023.11.008

[18] Liu, Z., Huang, Y., & Liu, L. (2024). ShaderPerFormer: Platform-independent context-aware shader performance predictor. Proceedings of the ACM on Computer Graphics and Interactive Techniques, 7(1), 1–17. https://doi.org/10.1145/3651295

[19] Han, M., Chen, R., Shen, W., Zhang, H., Yang, J., & Chen, H. (2026). Real-time, work-conserving GPU scheduling for concurrent DNN inference. ACM Transactions on Computer Systems, 44(1), 1–42. https://doi.org/10.1145/3768622

[20] Maleki, F., Moy, L., Forghani, R., Ghosh, T., Ovens, K., Langer, S., Rouzrokh, P., Khosravi, B., Ganjizadeh, A., Warren, D., Daneshjou, R., Moassefi, M., Haddadi Avval, A., Sotardi, S., Tenenholtz, N., Kitamura, F., & Kline, T. (2025). RIDGE: Reproducibility, integrity, dependability, generalizability, and efficiency assessment of medical image segmentation models. Journal of Imaging Informatics in Medicine, 38(4), 2524–2536. https://doi.org/10.1007/s10278-024-01282-9

[21] Topcu, B., Karabacak, D., & Öz, I. (2025). Demystifying power and performance variations in GPU systems through microarchitectural analysis. Computer Science and Information Systems, 22(2), 533–561. https://doi.org/10.2298/csis240722021t

[22] Allgaier, J., & Pryss, R. (2024). Cross-validation visualized: A narrative guide to advanced methods. Machine Learning and Knowledge Extraction, 6(2), 1378–1388. https://doi.org/10.3390/make6020065

[23] Willemsen, F.-J., Schoonhoven, R., Filipovič, J., Tørring, J. O., van Nieuwpoort, R., & van Werkhoven, B. (2024). A methodology for comparing optimization algorithms for auto-tuning. Future Generation Computer Systems, 159, 489–504. https://doi.org/10.1016/j.future.2024.05.021

[24] Li, W., Cook, D., Tanaka, E., & VanderPlas, S. (2024). A plot is worth a thousand tests: Assessing residual diagnostics with the lineup protocol. Journal of Computational and Graphical Statistics, 33(4), 1497–1511. https://doi.org/10.1080/10618600.2024.2344612

[25] Delong, Ł., & Wüthrich, M. V. (2025). Isotonic regression for variance estimation and its role in mean estimation and model validation. North American Actuarial Journal, 29(3), 563–591. https://doi.org/10.1080/10920277.2024.2421221

[26] Verdinelli, I., & Wasserman, L. (2024). Feature importance: A closer look at Shapley values and LOCO. Statistical Science, 39(4), 623–636. https://doi.org/10.1214/24-sts937

[27] Muschalik, M., Fumagalli, F., Hammer, B., & Hüllermeier, E. (2024). Beyond TreeSHAP: Efficient computation of any-order Shapley interactions for tree ensembles. Proceedings of the AAAI Conference on Artificial Intelligence, 38(13), 14388–14396. https://doi.org/10.1609/aaai.v38i13.29352

[28] Mohr, F., & van Rijn, J. N. (2024). Learning curves for decision making in supervised machine learning: A survey. Machine Learning, 113(11–12), 8371–8425. https://doi.org/10.1007/s10994-024-06619-7

[29] Rawal, A., Raglin, A., Rawat, D. B., Sadler, B. M., & McCoy, J. (2025). Causality for trustworthy artificial intelligence: Status, challenges and perspectives. ACM Computing Surveys, 57(6), 1–30. https://doi.org/10.1145/3665494

[30] Mandi, J., Kotary, J., Berden, S., Mulamba, M., Bucarey, V., Guns, T., & Fioretto, F. (2024). Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. Journal of Artificial Intelligence Research, 80, 1623–1701. https://doi.org/10.1613/jair.1.15320