Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
DOI: https://doi.org/10.51903/jtie.v5i2.545

Profit-Aware Spot GPU Admission Control with Cost-Sensitive Loss and Evidence-Grounded Policy Memos for AI Workload Supply-Demand Matching

Siming Zhao*[1], Yuxuan Ren[2], Xiaohan Chang[3]
Email: sz2944@columbia.edu
[1]*Business Analytics, Columbia University, NY, USA*
[2]*Chemical Engineering, University of Washington, WA, USA*
[3]*Computer Science, University of Connecticut, CT, USA*
*Corresponding Author

Abstract
AI clusters increasingly operate with heterogeneous GPU resources where production workloads and opportunistic spot jobs compete for limited accelerator capacity. This study presents a trace-driven admission-control framework using the Alibaba cluster-trace-v2026-spot-gpu dataset, consisting of 466,867 job records and 4,278 GPU-node records. The experiment evaluates GPU demand forecasting, profit-aware spot admission control, and evidence-grounded operational policy generation using chronological training, validation, and test splits. Hourly spot GPU demand forecasting was evaluated across six GPU models, where Ridge regression achieved the best test performance with an RMSE of 38.50 requested GPUs per hour, improving over both last-hour and seasonal naive baselines. The admission-control evaluation compared FIFO, greedy packing, classifier-based acceptance, utility ranking, and the proposed cost-sensitive policy. The proposed approach achieved a test profit of 67,278.96, improving 1.97% over the accuracy-oriented classifier while increasing spot success rate and reducing costly false acceptances by 13.17%. Sensitivity analysis showed that the optimal policy depends on the protection cost assigned to high-priority workloads. A deterministic evidence-grounded explanation layer generated 500 policy memos and passed numeric, policy, and evidence consistency checks. The findings suggest that profit-aware admission control can serve as a practical scheduling guardrail before detailed GPU placement and resource allocation decisions.

Keywords: *Spot GPU; Admission Control; Cost-Sensitive Learning; Workload Forecasting; GPU Scheduling.*

I. INTRODUCTION

GPU clusters have become the production substrate for modern AI training, inference, and experimentation. Their operating challenge is not only that accelerator supply is scarce, but also that the same fleet must satisfy heterogeneous priorities. High-priority workloads require strong service guarantees, while spot workloads can use spare supply if they do not materially harm protected jobs. Cluster-management systems such as Borg, Mesos, and Kubernetes established that datacenter efficiency depends on explicit resource sharing, placement, and queueing policy rather than isolated machine-level optimization (Burns et al., 2016; Hindman et al., 2011; Verma et al., 2015). GPU fleets intensify this problem because accelerators are scarce, model-specific, and fragmented across nodes.

The spot-GPU setting adds a direct business trade-off. A spot job admitted during idle capacity creates useful GPU-hours and improves fleet monetization. The same decision can destroy value when the job is long, large, or likely to overlap with future high-priority load. In that case, the

Received on Juni 2026; Revised on Juni 2026; Accepted on July 2026; Published on August 2026

system either interrupts spot work, delays protected work, or increases operational risk. Conventional admission models often target accuracy, F1, or queueing delay, yet these metrics do not encode the asymmetric cost of false accepts and false rejects. Cost-sensitive learning provides the appropriate framing because real losses depend on job size, duration, resource scarcity, and priority impact (Elkan, 2001; Zadrozny et al., 2003).

This paper studies profit-aware spot-GPU admission control as an AI workload supply-demand matching problem. The supply side is static node capacity by GPU model. The demand side is a chronological sequence of high-priority and spot job requests with GPU model, GPU request, worker count, submit time, and duration. The proposed policy uses demand-forecast context and a cost-sensitive admission objective: reward successful spot GPU-hours, penalize incremental high-priority impact, penalize idle residual supply, and penalize erroneous acceptance of long risky jobs. Figure 1 summarizes the evaluation pipeline.

The paper also evaluates an evidence-grounded policy memo layer. The purpose is not to let a language model make scheduling decisions. The scheduler makes each decision from numerical features and cost-sensitive scores; the memo layer converts those facts into an auditable accept, defer, or reject explanation. This framing is deliberately narrower than a full LLM evaluation. It treats explanation as a consistency and traceability problem, which is appropriate for an operations interface that may later be connected to an LLM assistant.

The research question is empirical and operational: when the same spot demand can be described by prediction accuracy, ranking utility, packing feasibility, and economic value, which objective produces the most useful admission behavior on a production trace? The answer is obtained by holding the trace, split boundaries, coefficient settings, and simulator constant, then changing the decision rule. This design makes the resulting differences interpretable as policy effects rather than artifacts of different data windows.

II. LITERATURE REVIEW

Large-scale cluster-management research has long emphasized that workload traces reveal scheduling problems that synthetic benchmarks miss. Google trace analyses documented heterogeneity and time-varying cluster behavior at scale (Reiss et al., 2012), while Borg showed how production clusters mix different workload classes under shared resource control (Verma et al., 2015). Mesos and dominant resource fairness formalized multi-resource sharing across frameworks (Ghodsi et al., 2011; Hindman et al., 2011). Alibaba trace studies extended this empirical tradition by exposing co-located datacenter behavior and resource-efficiency bottlenecks in production traces (Cheng et al., 2018; Guo et al., 2019).

Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

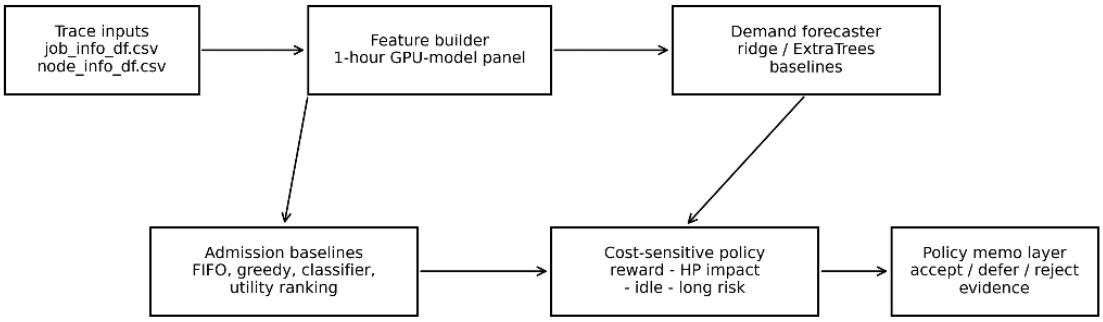

Figure 1. Evaluation pipeline for profit-aware spot GPU admission control and evidence-grounded policy memos.

GPU scheduling has become a separate systems problem because deep-learning jobs have large accelerator requirements, uncertain durations, and different sensitivity to placement and sharing. Gandiva uses introspection, time-slicing, and migration to improve deep-learning cluster efficiency (Xiao et al., 2018). Tiresias targets distributed deep-learning completion time under limited prior knowledge (Gu et al., 2019). Optimus, Pollux, and Themis study resource adaptation, goodput, fairness, and efficiency in GPU clusters (Mahajan et al., 2020; Peng et al., 2018; Qiao et al., 2021). Alibaba GPU trace work reports production heterogeneity, temporal patterns, heavy-tailed job behavior, and fragmentation challenges that motivate policies beyond simple packing (Weng et al., 2022; Weng et al., 2023).

Forecasting is useful because admission control depends on near-term supply-demand balance. Time-series forecast evaluation should use chronological splits and interpretable error measures. MAE and RMSE describe absolute error, while SMAPE normalizes relative error for demand panels with different scales (Hyndman & Koehler, 2006). In this paper, forecasts are not treated as hard reservations. They are context features that help the admission policy estimate local pressure.

Cost-sensitive learning changes the training objective from average correctness to expected utility. Elkan (2001) showed that class probabilities and decision costs should be separated, and Zadrozny et al. (2003) proposed cost-proportionate weighting as a practical method. For spot GPU admission, errors have asymmetric value: false accepts can create high-priority impact and interruption exposure, while false rejects create unused spot opportunity. The proposed weighted loss encodes this asymmetry directly.

Ranking is also relevant because admission controllers often sort candidate jobs competing for limited residual capacity. Learning-to-rank methods such as RankNet and Bayesian personalized ranking learn preferences from pairwise comparisons (Burges et al., 2005; Rendle et al., 2009). The utility-ranking baseline in this paper represents a ranking-oriented policy that favors high

nominal opportunity under capacity constraints. The cost-sensitive policy differs by tying the score to explicit profit and risk terms.

Explanation methods are necessary when automated systems affect operational capacity. LIME and SHAP established local-evidence and feature-attribution principles (Lundberg & Lee, 2017; Ribeiro et al., 2016). Modern language models can express policy rationales in natural language, but ungrounded explanations can be inconsistent with source evidence (Brown et al., 2020; Lewis et al., 2020). Therefore, the memo layer in this study is constrained by scheduler outputs and audited for numeric, policy, and evidence consistency.

III. RESEARCH METHOD

The experiment used the official Alibaba cluster-trace-v2026-spot-gpu files, job_info_df.csv and node_info_df.csv (Alibaba Cluster Data, 2026). The trace was treated as an official released trace rather than a reconstructed dataset. After checking required fields, the workload table contained 466,867 jobs and the node table contained 4,278 GPU nodes. The job table includes job_name, organization, gpu_model, cpu_request, gpu_request, worker_num, submit_time, duration, and job_type. The node table includes gpu_model, gpu_capacity_num, cpu_num, and node_name. The job_type field was used to distinguish high-priority and spot jobs as released. Requested GPUs were computed as gpu_request multiplied by worker_num, and requested GPU-hours were computed as requested GPUs multiplied by duration in hours.

Table 1. Dataset files, fields, data types, and non-null counts.

file	field	dtype	non-null
job_info_df.csv	job_name	int64	466,867
job_info_df.csv	organization	int64	466,867
job_info_df.csv	gpu_model	object	466,867
job_info_df.csv	cpu_request	float64	466,867
job_info_df.csv	gpu_request	float64	466,867
job_info_df.csv	worker_num	int64	466,867
job_info_df.csv	submit_time	float64	466,867
job_info_df.csv	duration	float64	466,867
job_info_df.csv	job_type	object	466,867
node_info_df.csv	gpu_model	object	4,278
node_info_df.csv	gpu_capacity_num	int64	4,278
node_info_df.csv	cpu_num	int64	4,278
node_info_df.csv	node_name	int64	4,278

The trace covers 4,419 hourly submission bins, equal to 184.13 days. The chronological split used 60% training hours, 15% validation hours, and 25% test hours for admission control. Demand forecasting used the first 75% of hourly rows for training and the final 25% for testing. Table 1 reports the data fields and non-null counts. Table 2 reports GPU-node capacity aggregated from node_info_df.csv. Tables 3 and 4 summarize workload scale, job size, duration, and GPU-hour tails.

Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

Table 2. GPU-node capacity aggregated from node_info_df.csv.

GPU model	Nodes	Total GPUs	GPUs/node	Total vCPUs
A10	2,494	2,494	1	319,228
A100-SXM4-80GB	432	3,456	8	55,296
A800-SXM4-80GB	22	176	8	2,816
GPU-series-1	989	1,558	1.575	189,824
GPU-series-2	122	976	8	23,424
H800	219	1,752	8	42,048

Node capacity was aggregated by GPU model. High-priority active demand was reconstructed with event arrays: each high-priority job adds requested GPUs at floor(submit_time/3600) and subtracts them at ceil((submit_time + duration)/3600). This creates an hourly high-priority load curve for each GPU model. Residual capacity available to spot jobs at hour t is max(capacity minus high-priority load, 0). If high-priority load already exceeds static capacity, residual capacity is zero, and any additional spot demand is charged as incremental high-priority impact.

Table 3. Workload distribution by job type and GPU model.

Type	GPU model	Jobs	Requested GPUs	GPU-hours	Median duration h
HP	A10	226,881	197,826.440	14,585,967.255	0.114
HP	A100-SXM4-80GB	137,396	688,319	10,940,601.712	0.523
HP	A800-SXM4-80GB	4,690	39,037	499,125.614	0.393
HP	GPU-series-1	16,054	16,427.500	5,083,539.248	1.099
HP	GPU-series-2	22,340	380,222	2,648,458.634	0.373
HP	H800	8,352	701,500	4,814,736.317	0.261
Spot	A10	1,762	12,499	86,248.031	1.331
Spot	A100-SXM4-80GB	31,624	140,449	387,673.603	0.479
Spot	A800-SXM4-80GB	4,109	14,924	27,013.434	0.388
Spot	GPU-series-1	2,857	24,416	133,492.032	2.605
Spot	GPU-series-2	8,057	62,349	228,976.889	1.422
Spot	H800	2,745	19,998	95,864.201	0.315

The forecasting task predicted hourly spot requested-GPU demand by GPU model. Features included lagged spot demand, lagged high-priority arrivals, rolling means, GPU model identity, capacity, hour-of-day terms, and day-of-week terms. Four forecasting models were evaluated: last-hour naive, 24-hour seasonal naive, ridge regression, and an ExtraTrees ensemble. Metrics were MAE, RMSE, and SMAPE on the held-out chronological test period.

For each spot job, an oracle safety label was computed from the high-priority residual capacity curve under the one-hour discretization. The oracle success fraction is the fraction of the job's requested hourly GPU allocation that can be served if the job runs alone against the high-priority load curve. A job is labeled safe when this success fraction is at least 0.90 and its individual oracle profit is positive before idle-cost accounting. The label is used for supervised training and diagnostics; final policy quality is still measured by the online simulator.

Table 4. Spot job size, duration, and GPU-hour quantiles by GPU model.

GPU model	Metric	P50	P75	P90	P95	P99
A10	gpus	3	8	18	24	71.780
A10	duration_h	1.331	4.327	9.927	10.073	351.729
A10	gpu_hours	3.448	22.355	129.240	209.130	695.296
A100-SXM4-80GB	gpus	2	4	8	14	32
A100-SXM4-80GB	duration_h	0.479	1.846	5.672	9.964	64.884
A100-SXM4-80GB	gpu_hours	1.148	5.702	19.572	41.332	158.497
A800-SXM4-80GB	gpus	2	4	8	9	16
A800-SXM4-80GB	duration_h	0.388	1.391	3.935	7.049	22.599
A800-SXM4-80GB	gpu_hours	0.858	4.508	16.091	28.306	79.289
GPU-series-1	gpus	3	7	17	35	94.220
GPU-series-1	duration_h	2.605	4.888	7.960	10.634	31.350
GPU-series-1	gpu_hours	7.306	24.150	57.748	116.794	810.605
GPU-series-2	gpus	3	8	17	25	77.880
GPU-series-2	duration_h	1.422	3.555	6.087	8.633	24.047
GPU-series-2	gpu_hours	3.444	14.064	48.961	94.743	472.562
H800	gpus	4	8	16	32	39.560
H800	duration_h	0.315	1.504	5.778	44.093	191.983
H800	gpu_hours	0.884	6.173	48.839	186.388	782.172

The baseline profit function is profit = successful spot GPU-hours - 4.00 x incremental high-priority impact GPU-hours - 0.03 x idle GPU-hours - 0.35 x long-risk GPU-hours. Successful spot GPU-hours are accepted spot active GPU-hours that fit into residual capacity. Incremental high-priority impact is max(HP + Spot - Capacity, 0) minus max(HP - Capacity, 0), which isolates the extra impact caused by admitted spot jobs. Long-risk GPU-hours are accepted jobs longer than 24 hours whose oracle success fraction is below 0.90.

Table 5. Admission thresholds selected on the validation period.

Policy	Threshold
Utility-ranking baseline	-0.009
Cost-sensitive admission	0.534
Cost-sensitive value-only	0.752
Cost-sensitive risk-only	0.053

The coefficient vector is treated as a normalized governance setting rather than an observed Alibaba price schedule, because the trace does not release internal spot revenue, service-level penalties, or preemption costs. To give the scale a practical reference, the current Amazon EC2 price-list file was processed for Linux, shared-tenancy GPU compute instances. The filtered prices ranged from $0.202 to $142.416 per instance-hour, with a median of $2.874 and a 75th percentile of $8.180. These prices are not mapped one-to-one to the anonymized Alibaba GPU models; they motivate why coefficient sensitivity is necessary. The sensitivity check in Table 11 varies the high-priority, idle, and long-risk penalties around the baseline.

The simulator is online with respect to spot admission. Within each hour, candidate spot jobs are processed according to the policy order, current high-priority load, and spot jobs already accepted by that policy. When a job is admitted, its requested GPU quantity is added to a policy-specific

Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

event array until its rounded end hour. FIFO processes admitted candidates by arrival. Greedy packing sorts hourly candidates by requested GPUs and duration. The accuracy-oriented classifier is an unweighted logistic model and applies a probability threshold of 0.50 as a gate. The utility-ranking baseline ranks candidates by predicted safe probability and nominal GPU-hour opportunity. The proposed cost-sensitive policy uses weighted logistic loss and ranks eligible candidates by the cost-trained safe probability. Thresholds for ranking-style policies were selected on the validation period using a conservative rule: choose the highest threshold whose validation profit is within 2% of the best validation profit, avoiding an over-accepting threshold when a safer threshold has nearly identical validation utility. Table 5 lists the resulting thresholds.

The policy memo layer receives the final decision, job ID, GPU model, requested GPUs, duration, residual ratio, safe probability, and score. It outputs a concise accept, defer, or reject memo. The generator is deterministic and uses only the evidence fields passed by the scheduler. This design supports later LLM-facing use, but the reported experiment evaluates rule-constrained memo consistency rather than unrestricted LLM generation.

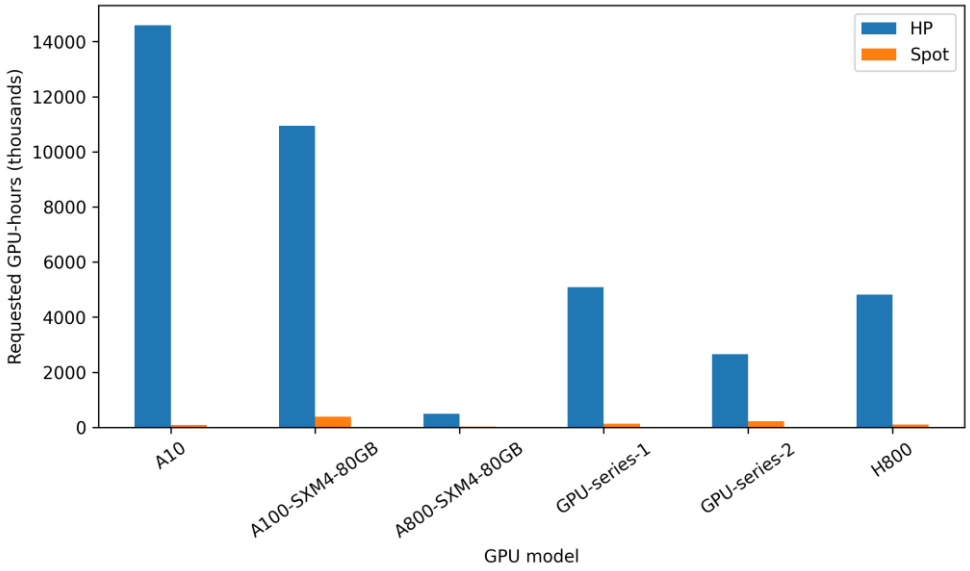

Figure 2. High-priority and spot requested GPU-hour distribution by GPU model.

IV. RESULT AND DUSCUSSION

The workload is strongly imbalanced between high-priority and spot jobs. High-priority jobs account for 415,713 records, while spot jobs account for 51,154 records. The imbalance matters because a naive accuracy objective can accept many apparently safe jobs while still producing high-cost false accepts. GPU demand is also heterogeneous across models. A10 contributes 2,494 GPUs across 2,494 nodes, while A100-SXM4-80GB contributes 3,456 GPUs across 432 nodes. A800-SXM4-80GB has only 176 GPUs, making decisions for that model sensitive to bursts.

Figure 2 shows the high-priority and spot GPU-hour distribution by model, and Figure 3 shows daily spot-demand nonstationarity across the trace.

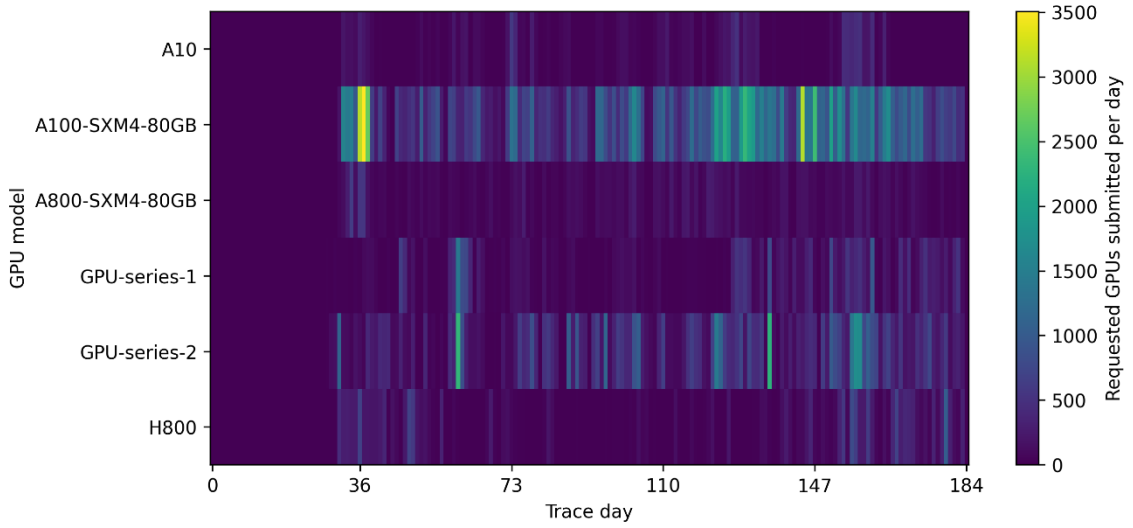

Figure 3. Daily spot requested-GPU demand heatmap across GPU models.

Spot job size and duration are heavy-tailed, as shown in Table 4. For A100-SXM4-80GB, the median spot request is 2 GPUs and the P99 request is 32 GPUs; for GPU-series-1, the median request is 3 GPUs and the P99 request exceeds 94 GPUs. Duration tails are also material. H800 has a median duration of 0.315 hours but a P99 duration of 191.98 hours, and A10 has a P99 duration of 351.73 hours. These tails justify the long-risk term because a feasible admission at the current hour can become costly when a long job overlaps future protected load.

Table 6. Demand forecasting results on the chronological test split.

Model	MAE	RMSE	SMAPE %
Ridge regression	16.320	38.499	132.815
ExtraTrees ensemble	17.515	38.705	147.509
Last-hour naive	18.231	49.260	78.098
Seasonal-24h naive	20.626	53.836	90.175

Forecasting results are reported in Table 6 and visualized in Figure 4. Ridge regression achieved the best RMSE, 38.50 requested GPUs per hour, and MAE of 16.32. It improved RMSE by 21.84% over the last-hour naive baseline and by 28.49% over the 24-hour seasonal naive baseline. ExtraTrees produced a similar RMSE but did not dominate the regularized linear model on this hourly panel. Table 7 shows that A100-SXM4-80GB is the hardest model in absolute terms, while smaller models have lower absolute error but high relative volatility.

Admission-control outcomes are reported in Table 8. The cost-sensitive admission policy achieved the highest baseline test profit, 67,278.96, compared with 65,981.96 for the accuracy-oriented classifier. The gain over the classifier is 1.97%, which should be interpreted as a modest

Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

but operationally meaningful improvement under the stated profit function. The accepted spot success rate rises from 0.8971 to 0.8988. The improvement comes mainly from lower high-priority impact and slightly fewer long-risk jobs, not from accepting the most total GPU-hours. FIFO and greedy packing create more accepted GPU-hours, but their additional high-priority impact and long-risk exposure lower profit.

Table 7. Ridge-regression forecasting results by GPU model.

Best model	GPU model	MAE	RMSE	SMAPE %	Mean true demand
Ridge regression	A10	5.202	15.526	123.735	3.367
Ridge regression	A100-SXM4-80GB	35.007	64.195	88.479	45.032
Ridge regression	A800-SXM4-80GB	3.784	6.031	126.138	3.841
Ridge regression	GPU-series-1	15.853	33.480	171.353	10.405
Ridge regression	GPU-series-2	26.272	47.307	155.101	20.112
Ridge regression	H800	11.801	33.703	132.086	10.035

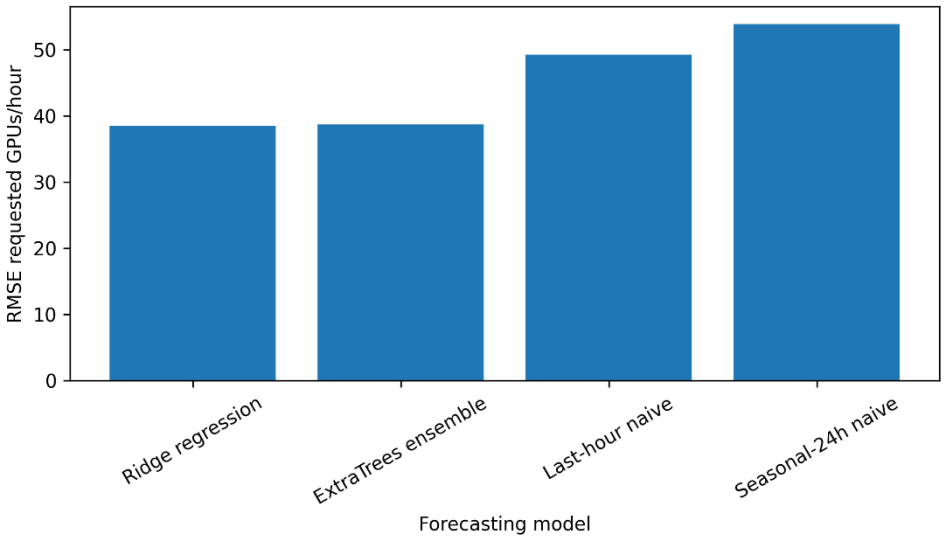

Figure 4. Forecasting RMSE comparison for hourly spot requested-GPU demand.

Classifier diagnostics appear in Table 9 and Figure 5. The cost-sensitive classifier reduced false accepts from 3,546 to 3,079, a 13.17% reduction. It increased accuracy from 0.7751 to 0.7919 and improved precision-safe from 0.7524 to 0.7750. This improvement is achieved by giving up a small amount of safe-label recall, which is consistent with the objective of reducing expensive false accepts.

Figure 6 explains why ranking and packing alone are not sufficient. Utility-ranking admits a narrower set of high-opportunity jobs, while FIFO and greedy packing have wider lower tails because they are guided by order or fit rather than expected impact. Figure 7 gives a concrete

A100-SXM4-80GB window: cost-sensitive admission fills available gaps, but it does not attempt to saturate every residual interval when protected demand is high.

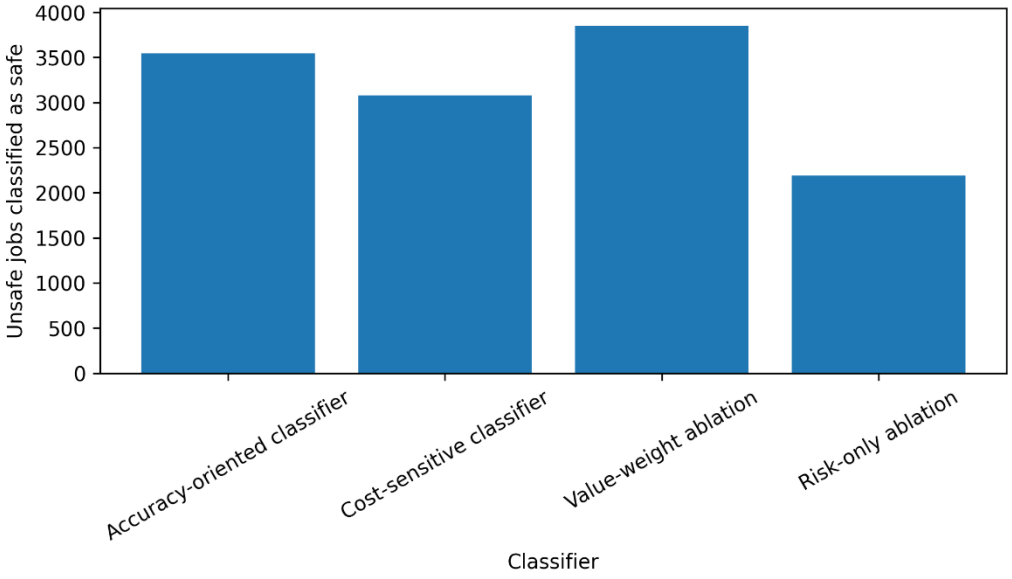

Figure 5. False accepts under accuracy-oriented and cost-sensitive losses.

Table 10 reports ablations of the cost-sensitive objective. The full policy outperforms value-only and risk-only variants under the baseline coefficient vector. Value-only weighting leaves more high-priority impact, while risk-only weighting lowers some diagnostic risk but accepts a portfolio that is less profitable after capacity interactions are applied. This result supports the use of a combined reward-and-risk objective rather than a single-sided weighting scheme.

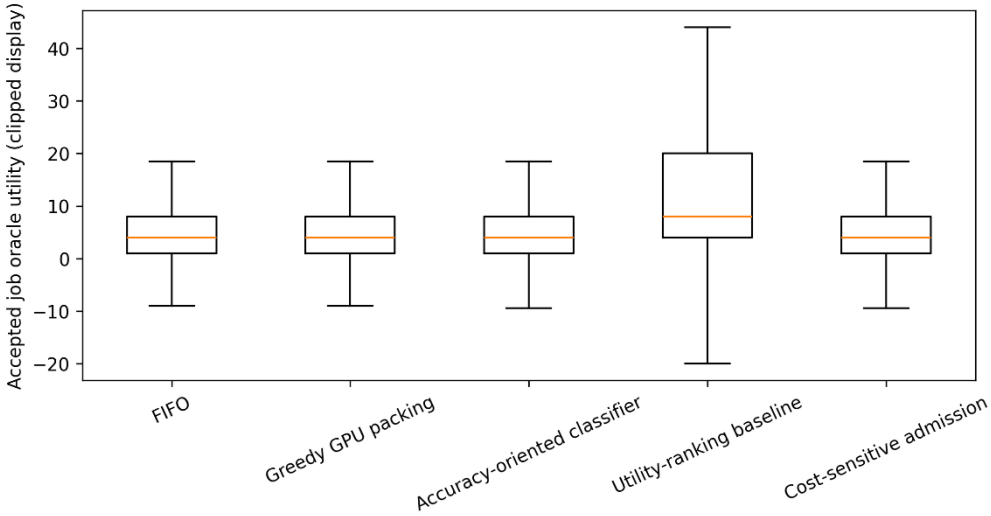

Figure 6. Accepted-job utility distribution by admission policy.

Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
DOI: https://doi.org/10.51903/jtie.v5i2.545

Table 8. Admission-control policy comparison on the held-out test period.

Policy	Accepted jobs	Accepted GPU-h	Successful GPU-h	HP impact GPU-h	Idle GPU-h	Long-risk jobs	Profit	Success rate	Avail. util.
Cost-sensitive admission	10,426	266,153.500	239,216	26,937.500	1,528,467.850	162	67,278.965	0.899	0.135
Accuracy-oriented classifier	10,542	268,709.500	241,046	27,663.500	1,526,637.850	164	65,981.965	0.897	0.136
Greedy GPU packing	10,978	285,191.500	253,517	31,674.500	1,514,166.850	192	59,438.495	0.889	0.143
Utility-ranking baseline	6,002	276,428	246,040	30,388	1,521,643.850	172	58,903.035	0.890	0.139
FIFO	10,678	293,030.500	259,540.500	33,490	1,508,143.350	183	58,302.649	0.886	0.147

Table 9. Classifier diagnostics for safe-admission labels on the test period.

Classifier	Accuracy	Precision safe	Recall safe	F1 safe	ROC AUC	False accepts	False rejects
Accuracy-oriented classifier	0.775	0.752	0.965	0.846	0.875	3,546	390
Cost-sensitive classifier	0.792	0.775	0.950	0.853	0.885	3,079	563
Value-weight ablation	0.768	0.740	0.981	0.844	0.878	3,849	210
Risk-only ablation	0.805	0.820	0.890	0.854	0.885	2,188	1,223

Table 10. Ablation of the cost-sensitive admission objective.

Policy	Accepted jobs	Accepted GPU-h	Successful GPU-h	HP impact GPU-h	Idle GPU-h	Long-risk jobs	Profit	Success rate	Avail. util.
Cost-sensitive admission	10,426	266,153.500	239,216	26,937.500	1,528,467.850	162	67,278.965	0.899	0.135
Cost-sensitive risk-only	10,818	280,146.500	250,464.500	29,682	1,517,219.350	170	66,608.719	0.894	0.142
Cost-sensitive value-only	10,267	273,286.500	244,541	28,745.500	1,523,142.850	168	63,634.365	0.895	0.138

Table 11. Profit sensitivity under alternate coefficient settings.

Coefficient setting	FIFO	Greedy GPU packing	Accuracy-oriented classifier	Utility-ranking baseline	Cost-sensitive admission
Baseline	58,302.65	59,438.49	65,981.96	58,903.03	67,278.96
High protection cost	-127,853.77	-119,498.34	-93,815.69	-113,017.49	-89,373.39
Low protection cost	171,183.77	168,753.33	165,135.22	164,351.66	164,818.32

Received on Juni 2026; Revised on Juni 2026; Accepted on July 2026; Published on August 2026

Journal of Technology Informatics and Engineering (JTIE)
Vol. 5 No. 2 August 2026
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 45-59
DOI: https://doi.org/10.51903/jtie.v5i2.545

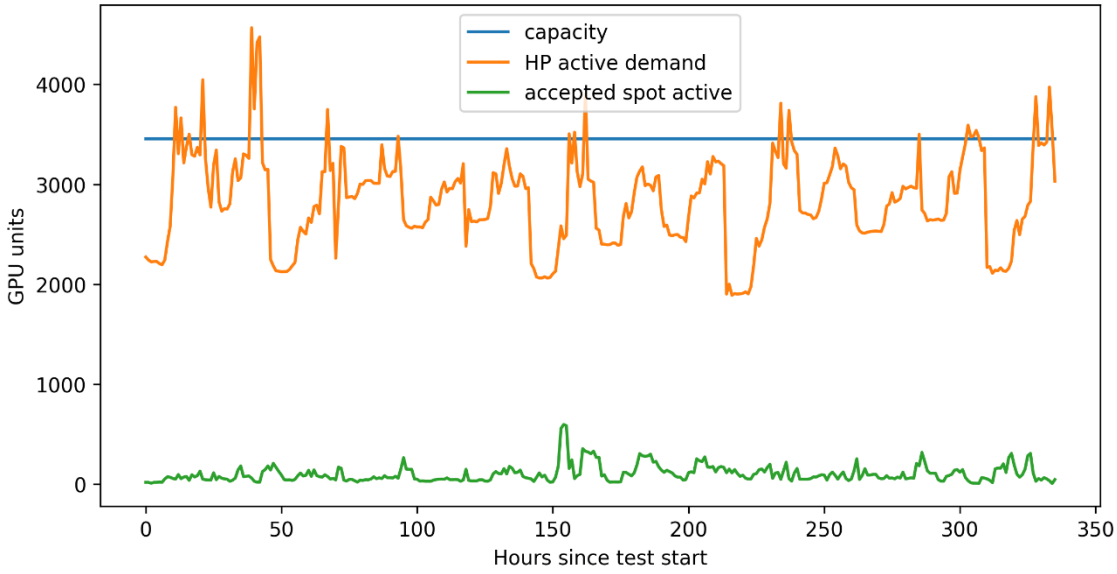

Figure 7. Supply-demand matching timeline for A100-SXM4-80GB during the first 14 test days.

ACCEPT | job 449366 | A100-SXM4-80GB | 8.0 GPUs | 8.46 h | safe-prob 0.999 | score 0.999. The decision follows the threshold 0.534; expected utility remains positive under current residual supply.

DEFER | job 449541 | A100-SXM4-80GB | 8.0 GPUs | 1.23 h | safe-prob 0.587 | score 0.587. The decision follows the threshold 0.534; score passes the policy threshold but available capacity is already reserved at arrival.

REJECT | job 449436 | A100-SXM4-80GB | 1.0 GPUs | 0.05 h | safe-prob 0.048 | score 0.048. The decision follows the threshold 0.534; policy score is below the validated admission threshold.

Figure 8. Example policy memo cards generated from scheduler evidence.

Table 11 reports coefficient sensitivity. The policy ranking is not invariant to all cost assumptions. When high-priority impact is assigned a low penalty, utilization-heavy FIFO obtains the largest profit because the extra accepted GPU-hours outweigh the smaller impact charge. Under the baseline and high-protection settings, cost-sensitive admission is better than the accuracy-oriented classifier. This sensitivity is important for interpretation: the result supports profit-aware admission under protection-oriented operating assumptions, but it does not imply that one coefficient vector is universally optimal.

The memo evaluation appears in Table 12. The deterministic evidence-grounded memo layer generated 500 held-out cards with 100% numeric consistency, 100% policy consistency, and

Received on Juni 2026; Revised on Juni 2026; Accepted on July 2026; Published on August 2026

100% evidence coverage. Figure 8 shows representative accept, defer, and reject cards, and Figure 9 shows the cost-sensitive confusion matrix used by the policy checker. These results do not claim that unrestricted LLM output is always reliable. They show that a rule-constrained memo interface can make numerical infrastructure decisions easier to audit.

Table 12. Evidence-grounded policy memo evaluation.

Memo count	Numeric consistency	Policy consistency	Evidence coverage	Mean words
500	1	1	1	32.830

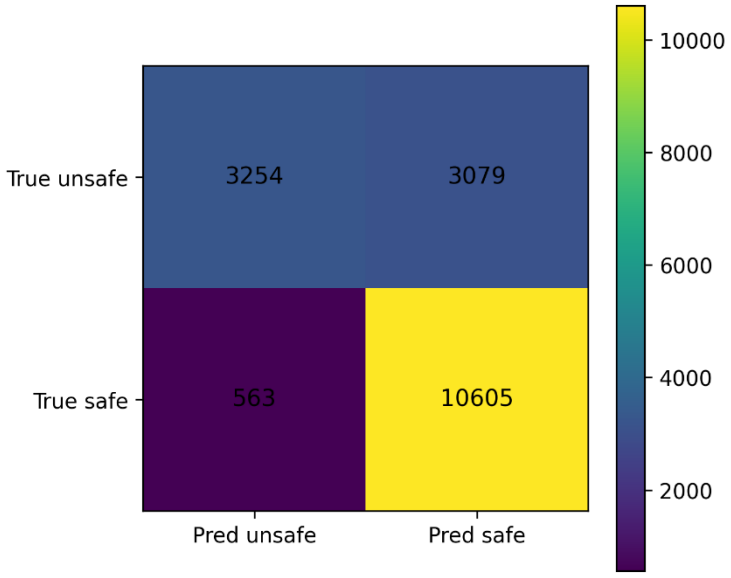

Figure 9. Cost-sensitive classifier confusion matrix for the safe-admission label.

V. CONCLUSION AND RECOMMENDATION

v2026-spot-gpu dataset. The trace contains heterogeneous GPU capacity, imbalanced high-priority and spot workloads, bursty model-specific demand, and heavy-tailed spot job sizes and durations. These characteristics make FIFO and simple packing policies insufficient when the operating objective includes both monetization and high-priority protection.

The main finding is that cost-sensitive learning can improve the business objective even when the improvement in conventional success rate is small. Compared with the accuracy-oriented classifier, the proposed policy increases baseline test profit by 1.97%, slightly improves accepted-job success rate, and reduces false accepts by 13.17%. The result should be read as evidence for utility-aware admission under the stated coefficient settings, not as a universal guarantee that the same policy will dominate under every price or penalty schedule.

For production adoption, three recommendations follow. First, infrastructure teams should define admission objectives in GPU-hour economic units rather than only in precision, recall, or

utilization terms. Second, spot-GPU policies should be model-specific because capacity and demand patterns differ substantially across A10, A100-SXM4-80GB, A800-SXM4-80GB, GPU-series-1, GPU-series-2, and H800. Third, operations explanations should be constrained by scheduler evidence. A memo generator or LLM-facing interface should explain decisions after the numerical policy has made them, and each memo should be checked for numeric and policy consistency.

The study has two main limitations. First, the simulator operates at the GPU-model admission level. It does not model node-level placement, topology, intra-node fragmentation, or preemption mechanics. The intended role of the proposed method is therefore a guardrail before lower-level placement, not a replacement for a full GPU scheduler. Second, the profit coefficients are governance parameters. A deployment should calibrate them from spot price, service-level penalties, customer priority, and interruption cost, then rerun validation before changing policy thresholds.

Overall, the results support profit-aware admission as a practical layer for AI workload supply-demand matching. The admission layer decides whether a spot job deserves scarce model-specific capacity; the placement layer can then solve node-level constraints. This division keeps the economic objective explicit while leaving detailed topology and fragmentation decisions to specialized GPU schedulers.

REFERENCES

Alibaba Cluster Data. (2026). cluster-trace-v2026-spot-gpu. Alibaba Cluster Trace Program. https://github.com/alibaba/clusterdata/tree/master/cluster-trace-v2026-spot-gpu

Amazon Web Services. (2026a). Getting price list files using the AWS Price List Bulk API. AWS Documentation.

Amazon Web Services. (2026b). View Spot Instance pricing history. Amazon EC2 User Guide.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877-1901.

Burges, C. J. C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005). Learning to rank using gradient descent. Proceedings of the 22nd International Conference on Machine Learning, 89-96.

Burns, B., Grant, B., Oppenheimer, D., Brewer, E., & Wilkes, J. (2016). Borg, Omega, and Kubernetes. Communications of the ACM, 59(5), 50-57.

Cheng, Y., Chai, Z., & Anwar, A. (2018). Characterizing co-located datacenter workloads: An Alibaba case study. Proceedings of the 9th Asia-Pacific Workshop on Systems.

Delimitrou, C., & Kozyrakis, C. (2014). Quasar: Resource-efficient and QoS-aware cluster management. Proceedings of ASPLOS, 127-144.

Elkan, C. (2001). The foundations of cost-sensitive learning. Proceedings of the 17th International Joint Conference on Artificial Intelligence, 973-978.

Ghodsi, A., Zaharia, M., Hindman, B., Konwinski, A., Shenker, S., & Stoica, I. (2011). Dominant resource fairness: Fair allocation of multiple resource types. Proceedings of NSDI, 323-336.

Grandl, R., Ananthanarayanan, G., Kandula, S., Rao, S., & Akella, A. (2014). Multi-resource packing for cluster schedulers. Proceedings of ACM SIGCOMM, 455-466.

Gu, J., Chowdhury, M., Shin, K. G., Zhu, Y., Jeon, M., Qian, J., Liu, H., & Guo, C. (2019). Tiresias: A GPU cluster manager for distributed deep learning. Proceedings of NSDI, 485-500.

Guo, J., Chang, Z., Wang, S., Ding, H., Feng, Y., Mao, L., & Bao, Y. (2019). Who limits the resource efficiency of my datacenter: An analysis of Alibaba datacenter traces. Proceedings of IWQoS.

Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A., Joseph, A. D., Katz, R., Shenker, S., & Stoica, I. (2011). Mesos: A platform for fine-grained resource sharing in the data center. Proceedings of NSDI, 295-308.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. International Journal of Forecasting, 22(4), 679-688.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Kuttler, H., Lewis, M., Yih, W.-t., Rocktaschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459-9474.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. Proceedings of the IEEE International Conference on Computer Vision, 2980-2988.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems, 30.

Mahajan, K., Singhvi, A., Balasubramanian, A., Lee, B., Venkataraman, S., Akella, A., & Phanishayee, A. (2020). Themis: Fair and efficient GPU cluster scheduling. Proceedings of NSDI, 289-304.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). Training language models

to follow instructions with human feedback. Advances in Neural Information Processing Systems, 35, 27730-27744.

Peng, Y., Bao, Y., Chen, Y., Wu, C., & Guo, C. (2018). Optimus: An efficient dynamic resource management system for deep learning clusters. Proceedings of EuroSys.

Qiao, A., Lee, B., Chandrashekar, P., Zhao, Y., Zhang, W., Li, X., Chen, M., Zhang, S., Mars, J., & Tang, L. (2021). Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning. Proceedings of OSDI, 1-18.

Reiss, C., Tumanov, A., Ganger, G. R., Katz, R. H., & Kozuch, M. A. (2012). Heterogeneity and dynamicity of clouds at scale: Google trace analysis. Proceedings of SoCC.

Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. Proceedings of UAI, 452-461.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Why should I trust you? Explaining the predictions of any classifier. Proceedings of KDD, 1135-1144.

Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E., & Wilkes, J. (2015). Large-scale cluster management at Google with Borg. Proceedings of EuroSys.

Weng, Q., Xiao, W., Yu, Y., Wang, W., Wang, C., He, J., Li, Y., Zhang, L., Lin, W., & Ding, Y. (2022). MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters. Proceedings of NSDI, 945-960.

Weng, Q., Yang, L., Yu, Y., Wang, W., Tang, X., Yang, G., & Zhang, L. (2023). Beware of fragmentation: Scheduling GPU-sharing workloads with fragmentation gradient descent. Proceedings of USENIX ATC, 995-1008.

Xiao, W., Bhardwaj, R., Ramjee, R., Sivathanu, M., Kwatra, N., Han, Z., Patel, P., Peng, X., Zhao, H., Zhang, Q., Yang, F., & Zhou, L. (2018). Gandiva: Introspective cluster scheduling for deep learning. Proceedings of OSDI, 595-610.

Zadrozny, B., Langford, J., & Abe, N. (2003). Cost-sensitive learning by cost-proportionate example weighting. Proceedings of ICDM, 435-442.