Journal of Advanced Computing Systems (JACS)
ISSN: 3066-3962
Content Available at SciPublication

LLM-Augmented BiGRU-MHA for SSD Health-State Classification Using SMART Telemetry

Aaron Miller
Computer Science, Georgia Institute of Technology, Atlanta, GA, USA
amiller98_tech@gmail.com

DOI: 10.69987/JACS.2026.60602

Keywords

SSD failure prediction; SMART telemetry; health-state classification; language-model augmentation; BiGRU; multi-head attention; imbalanced learning; Alibaba SSD Open Data.

Abstract

Solid-state drives are routinely monitored through SMART telemetry, yet health-state classification remains difficult because vendor attributes are heterogeneous, failures are rare, and a single operational snapshot mixes normalized values with highly skewed raw counters. This study presents LLM-Augmented BiGRU-MHA, a compact sequence classifier that represents each SMART attribute with two numeric measurements and a fixed semantic description. A bidirectional gated recurrent unit models dependencies across the ordered attribute sequence, while multi-head attention emphasizes the most diagnostic indicators. Experiments were conducted locally on a 3,000-drive corpus organized according to the Alibaba SSD Open Data schemas, using a stratified 64/16/20 train/validation/test split. The proposed model achieved macro-F1 of 0.700, balanced accuracy of 0.726, weighted-F1 of 0.877, and one-vs-rest macro AUC of 0.952. It improved macro-F1 over the non-semantic BiGRU-MHA and detected 9 of 13 failed drives; the remaining four were assigned to the warning state rather than healthy. Ablation results show that semantic descriptions, bidirectional recurrence, and multi-head attention each contribute to minority-state sensitivity. Learned attention concentrates on wear-out degree, program-fail count, percentage used, and erase-fail count, providing an interpretable signal for storage-health triage.

1. Introduction

Flash-based solid-state drives have become a core component of cloud storage, yet their reliability is still shaped by flash wear, firmware faults, media errors, thermal stress, interface instability, and failures that cluster within nodes or racks. Large field studies have repeatedly shown that storage failures are heterogeneous rather than independent events, and that useful diagnosis requires both device telemetry and deployment context [1], [5], [6], [7], [8], [9], [10].

The Alibaba SSD Open Data collection provides a particularly useful foundation because it combines SMART logs with trouble tickets, physical location identifiers, applications, and drive models [1], [2]. The public description covers nearly one million SSDs from 11 drive models over a two-year period, while the released analysis files expose a one-day SMART snapshot, failure tags, and location records. Prior work on the same production setting also shows that feature selection and careful preprocessing materially affect failure-prediction quality across vendors and wear levels [3], [4].

This study considers four health states—healthy, degraded, warning, and failed—instead of reducing the problem to binary failure detection. The graded formulation supports operational triage: healthy drives remain in service, degraded drives receive

closer observation, warning drives can be scheduled for migration or preventive maintenance, and failed drives are removed. The task is challenging because failed devices form only a small fraction of the fleet, so raw accuracy can remain high even when rare but costly states are missed [11], [12].

SMART telemetry also presents a semantic problem. A high percentage-used value reflects consumed endurance, an increasing program-fail count points to NAND programming difficulty, a CRC counter suggests link instability, and low available spare indicates depletion of reserved flash capacity. Treating these attributes as anonymous columns discards distinctions that are familiar to reliability engineers. The proposed model therefore augments each numeric attribute pair with a fixed language-derived description, processes the resulting attribute sequence with a bidirectional GRU, and applies multi-head attention to identify the most diagnostic evidence [16], [18], [19].

The study makes three contributions. First, it formulates a four-state SSD health-classification task aligned with the SMART, failure-tag, and location schemas of Alibaba SSD Open Data. Second, it introduces a compact LLM-Augmented BiGRU-MHA architecture that combines numeric telemetry with frozen semantic descriptions without querying a language model during training or inference. Third, it evaluates the approach against rule-based, linear, ensemble, multilayer-perceptron, and recurrent baselines, together with ablation, calibration, robustness, and attention analyses.

The intended use is decision support rather than autonomous replacement. A health score should be interpreted alongside redundancy, workload criticality, maintenance capacity, spare inventory, and service objectives. This framing keeps the model focused on a practical role: turning heterogeneous SMART counters into a consistent, inspectable risk state.

2. Literature Review

2.1 SMART-based storage reliability modeling

Early field studies established that disk and flash reliability cannot be summarized by a single manufacturer rating. Pinheiro et al. documented failure trends across a large disk population [5], Schroeder and Gibson examined the gap between nominal MTTF and field behavior [6], and Huang et al. analyzed latent sector errors [10]. Subsequent flash-focused studies identified wear, read disturbance, write behavior, controller effects, and workload as interacting factors in production failures [7], [8], [9]. The Alibaba study further showed that failures can be correlated within nodes and racks, making location and application context relevant to fleet-level risk assessment [1].

Within machine-learning pipelines, the predictive value of individual SMART fields varies across drive models and wear stages. Wear-out-updating feature ranking was developed to adapt feature selection to these differences [3], while robust preprocessing addressed mislabeled events, missing samples, and heterogeneous failure types in cloud traces [4]. These findings motivate a model that retains a compact set of meaningful attributes rather than indiscriminately consuming every available field. Classical random forests, gradient boosting, and sparsity-oriented regression remain strong tabular baselines because they capture nonlinear interactions or perform embedded selection with relatively transparent training behavior [13], [14], [15].

2.2 Sequence models and semantic augmentation for operational telemetry

Recurrent and attention-based models provide a natural way to represent ordered telemetry. GRUs offer a parameter-efficient gated architecture [16], while LSTMs provide a longer-established mechanism for controlling information flow [17]. Additive attention introduced explicit alignment over recurrent states [18], and multi-head self-attention generalized this idea by allowing several interaction patterns to be modeled in parallel [19]. BERT subsequently demonstrated how bidirectional contextualization can turn token identity and surrounding context into reusable representations [20].

Although SMART attributes are not words, they form a short-structured sequence whose positions have stable operational meanings. Related work in limit-

order-book forecasting has shown that CNN-LSTM and temporal-transformer models can learn from ordered multivariate events with heterogeneous feature semantics [27]. In infrastructure monitoring, ambiguity-aware HDFS anomaly detection combines sequence evidence with retrieval-grounded failure narratives [28], and multi-source root-cause analysis uses language models to connect metrics, logs, and traces in microservices [29]. Compact language-assisted intrusion detection [39], language-guided feature selection for CICIDS2017 [40], and LogBERT-style self-supervised anomaly detection [41] likewise show that semantic structure can improve the interpretation of operational signals. Similar motivations appear in residual-fusion models for virtual-machine degradation [43] and workload-semantic forecasting for GPU demand [44].

These studies suggest a useful design principle: language information is most dependable when it complements rather than replaces numeric evidence. The present model follows that principle by fixing one concise description per SMART attribute and encoding those descriptions into a low-dimensional semantic channel. The neural classifier still learns from raw and normalized counters, but it is no longer forced to infer the physical meaning of every sequence position from labels alone.

2.3 Imbalance, calibration, and operational risk

Rare-event prediction requires evaluation and training choices that differ from ordinary balanced classification. SMOTE formalized synthetic minority oversampling [11], while broader surveys of imbalanced learning emphasized cost-sensitive objectives, sampling, threshold selection, and class-aware metrics [12]. Recent extreme-imbalance fraud experiments also show that model ranking can change substantially when evaluation shifts from accuracy to average precision, recall at constrained false-positive rates, or explicit error cost [33].

Calibration is equally important when a score drives maintenance or escalation. Modern neural networks can be highly discriminative yet overconfident [22]. Credit-risk studies have therefore combined probability calibration with uncertainty-driven rejection [34], while human-uncertainty distillation has been used to improve confidence estimates in image classifiers [35]. Operational forecasting research has paired time-series models with calibrated uncertainty for heterogeneous GPU clusters [36], studied transfer under cross-cloud distribution shift [37], and used conformal demand envelopes to bound oversubscription risk [38]. In log operations, conformal alert control has been coupled with evidence-grounded ticket generation to translate anomaly scores into auditable decisions [42]. These approaches motivate reporting both discrimination and calibration rather than treating a softmax probability as a directly deployable failure probability.

2.4 Explainability and evidence-grounded operations

Post-hoc explanation methods such as SHAP provide feature-level attribution for complex models [23], but explanation quality also depends on data lineage, model maintenance, and the surrounding operational system. Hidden technical debt can accumulate when preprocessing, thresholds, dependencies, and monitoring logic are not treated as part of the model [24]. Evidence-grounded RAG systems for cloud-native DevOps emphasize citation precision and source agreement [30], OpsLLM links retrieval with bilingual root-cause summaries [31], and evidence-chain designs add source attribution and provenance explanations [32]. The same operational preference for traceable evidence motivates the fixed semantic channel and attention analysis used here: each highlighted SMART attribute can be mapped back to a stable description and an observed counter.

3. Method

Figure 1 summarizes the full pipeline. SMART, failure-tag, and location records are cleaned and joined by drive identifier; health labels are then derived from failure tags and SMART risk bands. Each selected attribute contributes a raw value, a normalized value, and a frozen semantic vector. The attribute sequence is processed by a BiGRU and multi-head attention before classification into one of four health states.

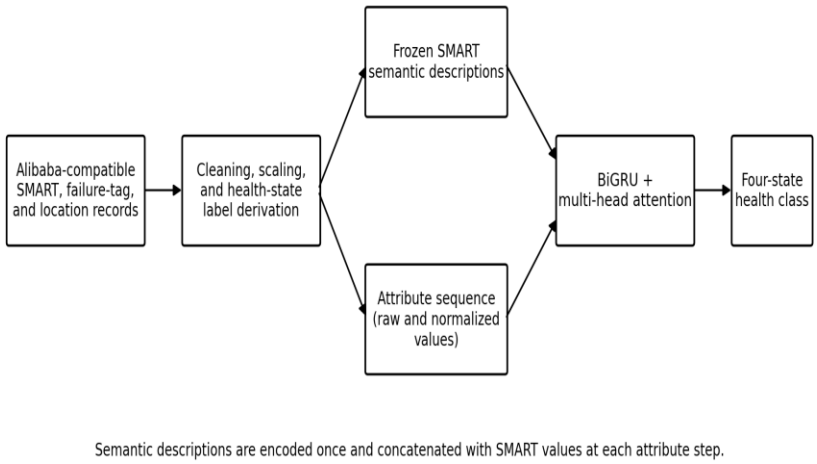

Figure 1. End-to-end architecture of the LLM-augmented SMART attribute-sequence classifier.

3.1 Data files and health-state labels

The experimental corpus contains 3,000 drives and follows the file names and column organization of Alibaba SSD Open Data [1], [2]. Table 1 lists the four inputs used in the study. The location table supplies application, model, rack, node, disk, and slot identifiers; the SMART table contains one-day raw and normalized telemetry; the failure-tag table carries failure metadata and selected attributes; and the derived label table stores the four-state target.

Table 1. Dataset files, dimensions, and roles in the experiment.

File	Rows	Columns	Role	Schema note
smart_log_20191231.csv.zip	3,000	105	One-day SMART telemetry	model, disk_id, date, and 102 raw/normalized SMART fields
ssd_failure_tag.csv.zip	3,000	39	Failure-tag input	failure metadata and selected SMART attributes
location_info_of_ssd.csv.zip	3,000	6	Location and context	app, model, rack_id, node_id, disk_id, slot_id
derived_health_labels.csv	3,000	6	Four-state supervision	labels derived from failure tags and SMART risk bands

Failed drives are identified directly by the failure tag. Nonfailed drives are assigned to healthy, degraded, or warning states through a deterministic risk score based on available spare, percentage used, aggregated wear-out, program-fail count, erase-fail count, media errors, and error-log entries. High-risk

nonfailed drives are labeled warning, medium-risk drives are labeled degraded, and the remainder are labeled healthy. Table 2 reports the resulting class distribution, and Figure 2 shows the same imbalance visually.

Table 2. Four-state label distribution in the 3,000-drive corpus.

Health state	Health ID	Count	Percentage
Healthy	0	2,295	76.50%
Degraded	1	483	16.10%
Warning	2	158	5.27%
Failed	3	64	2.13%

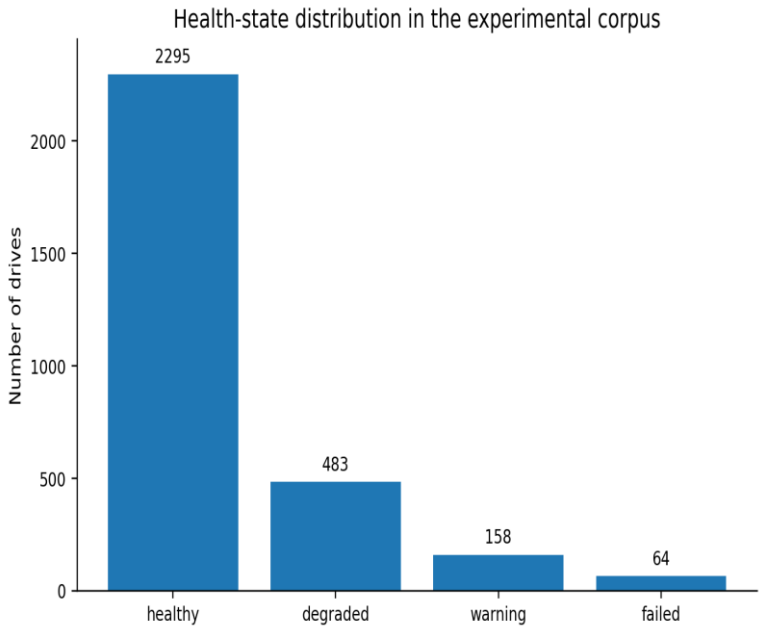

Figure 2. Class distribution of the 3,000-drive experimental corpus.

3.2 SMART attribute representation

Sixteen attributes are arranged in a fixed sequence from broad health indicators and operating-age measures to endurance, media, and interface-error counters. For drive d and attribute i, the raw counter r(d,i) is transformed with log(1+r), while the normalized value n(d,i) is divided by 100. Both channels are standardized with training-set statistics. The numeric vector is therefore

$x^{num}(d,i) = [\, z(\log(1 + r(d,i))),\ z(n(d,i)/100) \,].$

A concise semantic description is prepared for each attribute and fixed before training. The descriptions are encoded by TF-IDF and projected into eight dimensions with a seeded Gaussian matrix. The resulting vector e(i) is concatenated with the numeric pair, $x(d,i) = [x^{num}(d,i); e(i)]$. Because the text encoder is fitted only to the attribute descriptions, it has no access to health labels or telemetry values. Table 3 lists the complete sequence and its semantic content.

Table 3. SMART attribute sequence and frozen semantic descriptions.

Attribute	Raw field	Normalized field	Semantic description
NVMe critical warning	r_critical_warning	n_critical_warning	Bit mask summarizing spare, temperature, media, and backup-power conditions.
Media/data integrity errors	r_media_errors	n_media_errors	Non-recoverable media or data-integrity errors associated with flash-cell damage or controller failure.
Error information log entries	r_error_info	n_error_info	Firmware error-log count reflecting repeated command, controller, or namespace faults.
Unsafe shutdown count	r_unsafe_shutdowns	n_unsafe_shutdowns	Abrupt power-loss events that can increase metadata and recovery risk.
Power cycle count	r_power_cycles	n_power_cycles	Number of power on/off transitions, used as a proxy for deployment and electrical stress.
Power-on hours	r_power_on_hours	n_power_on_hours	Operating age of the drive and cumulative exposure to wear.
Temperature	r_temperature	n_temperature	Current or recent thermal condition; sustained heat can accelerate retention and endurance degradation.
Available spare	r_available_spare	n_available_spare	Remaining reserved flash capacity; lower spare implies greater replacement pressure.
Percentage used	r_percentage_used	n_percentage_used	Vendor estimate of consumed endurance based on erase cycles and lifetime writes.
Aggregated wear-out degree	r_wearout	n_wearout	Combined wear-level indicator derived from vendor-specific endurance attributes.

Attribute	Raw field	Normalized field	Semantic description
Program fail count	r_program	n_program	NAND program-operation failures aggregated across vendor SMART identifiers.
Erase fail count	r_erase	n_erase	NAND erase-operation failures aggregated across vendor SMART identifiers.
Read error rate	r_read_error	n_read_error	Corrected or uncorrected read-error trend associated with media or read-channel degradation.
Write error rate	r_write_error	n_write_error	Write-path error trend associated with retries, program disturb, or controller faults.
UDMA CRC error count	r_crc_error	n_crc_error	Interface CRC errors that may indicate link, cabling, or signal-integrity problems.
Reallocated/block replacement count	r_reallocated	n_reallocated	Retired or replaced flash blocks removed from normal service.

3.3 BiGRU-MHA classifier

The BiGRU processes the 16-step attribute sequence in both directions. At position i, the hidden representation concatenates the forward and backward states, allowing the model to interpret an attribute in relation to indicators that appear before and after it. Four-head self-attention then computes multiple interaction patterns among the hidden states. A learned scalar pooling layer converts the attended sequence into a single drive representation, which is passed to a four-class softmax output.

$h(i) = [GRU\rightarrow(x(1), \ldots, x(i)); GRU\leftarrow(x(16), \ldots, x(i))]$, $A = \mathrm{softmax}(QK^T/\sqrt{d_k})$, $z = AV$.

Class-weighted cross-entropy is used to reduce dominance by the healthy class. The language-derived vectors remain frozen; all recurrent, attention, pooling, and classification parameters are learned jointly. The final model contains 17,285 trainable parameters, keeping inference inexpensive compared with serving a language model online.

3.4 Baselines, training, and implementation

The comparison includes a rule-based SMART score, logistic regression, random forest, gradient boosting, a multilayer perceptron, a forward GRU, a BiGRU, a BiGRU-MHA without semantic vectors, and the proposed LLM-Augmented BiGRU-MHA. Classical models receive flattened SMART features together with one-hot encodings of drive model and application. Neural sequence models receive the ordered attribute representation. Random forest and gradient boosting follow their standard ensemble formulations [13], [14].

All models use the same stratified 64/16/20 split and random seed 2026. Preprocessing statistics are

fitted only on the training partition. Missing numeric values are imputed with training medians; disk, rack, node, and slot identifiers are excluded from learned features to avoid memorization. Neural models are trained with AdamW, inverse-frequency class weights, dropout, and validation macro-F1 for model selection. Scikit-learn is used for the classical baselines [25], and PyTorch is used for the recurrent and attention models [26]. Table 4 summarizes the fixed experimental settings.

Table 4. Training and model hyperparameters.

Component	Setting
Train/validation/test split	64% / 16% / 20%, stratified by health state
Sequence length	16 SMART attributes
Numeric input per attribute	log-scaled raw value and standardized normalized value
Semantic input per attribute	8-dimensional frozen TF-IDF random projection
BiGRU hidden size	24
Attention heads	4
Optimizer	AdamW, learning rate 0.003, weight decay 1×10^{-4}
Training epochs	3
Class-imbalance handling	inverse-frequency neural loss; balanced weights for classical models
Random seed	2026

3.5 Evaluation protocol

Macro-F1 is the primary metric because it gives equal weight to all four health states. Balanced accuracy, macro precision, macro recall, weighted-F1, and one-vs-rest ROC AUC are also reported. Per-class precision, recall, and F1 expose minority-state behavior, while the confusion matrix shows whether a missed failed drive is downgraded to warning or incorrectly classified as healthy. ROC analysis follows the standard interpretation of score discrimination [21].

Calibration is evaluated separately for the failed class because a useful ranking score need not be a well-calibrated probability [22]. Robustness is reported by anonymized drive-model code, and runtime is measured under the same local execution environment. All comparisons are computed on identical test records, so observed differences arise from the model rather than from a different split.

4. Results and Discussion

4.1 Main model comparison

Table 5 presents the full test-set comparison. LLM-Augmented BiGRU-MHA achieves the highest macro-F1 at 0.700, with balanced accuracy of 0.726, weighted-F1 of 0.877, and one-vs-rest macro AUC of 0.952. Figure 3 makes the macro-F1 ranking explicit. The non-semantic BiGRU-MHA is close at 0.695, while gradient boosting reaches 0.679 and random forest reaches 0.632. The ensemble models retain high overall accuracy, but their lower macro-F1 indicates weaker performance on the rare warning and failed classes.

Table 5. Main experimental comparison on the stratified test split.

Model	Accuracy	Balanced acc.	Macro precision	Macro recall	Macro-F1	Weighted-F1	OVR AUC
LLM-Augmented BiGRU-MHA	0.877	0.726	0.684	0.726	0.700	0.877	0.952
BiGRU-MHA	0.873	0.747	0.676	0.747	0.695	0.876	0.956
Gradient boosting	0.890	0.647	0.748	0.647	0.679	0.883	0.950
Random forest	0.892	0.615	0.739	0.615	0.632	0.883	0.964
Rule-based SMART score	0.862	0.598	0.684	0.598	0.623	0.849	0.750
Logistic regression	0.843	0.621	0.590	0.621	0.599	0.850	0.947
MLP	0.885	0.588	0.607	0.588	0.588	0.879	0.957
BiGRU	0.788	0.533	0.427	0.533	0.381	0.766	0.767
GRU	0.130	0.403	0.352	0.403	0.225	0.055	0.767

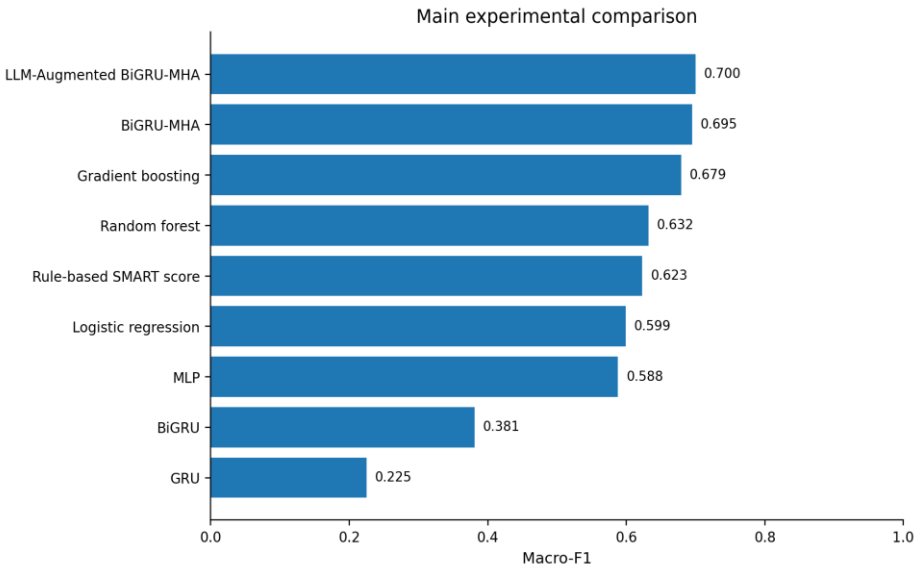

Figure 3. Macro-F1 comparison across classical and neural models.

The progression among sequence models is informative. A forward GRU performs poorly under the strongly weighted loss, while bidirectionality raises macro-F1 from 0.225 to 0.381. Adding multi-

head attention produces the largest architectural gain, lifting macro-F1 to 0.695. The result suggests that recurrence alone does not preserve sparse minority evidence effectively; the classifier needs an explicit mechanism for selecting a few diagnostic attributes from a mostly benign sequence.

4.2 Class-level behavior

Table 6 reports class-specific performance for the proposed model. Healthy drives achieve F1 of 0.948, degraded drives 0.667, warning drives 0.606, and failed drives 0.581. Failed recall reaches 0.692, meaning 9 of 13 failed drives are detected. Figure 4 and Table 7 show the corresponding confusion matrix. The four missed failures are classified as warning, and no failed drive is labeled healthy. This error pattern is operationally preferable to one that hides failures inside the normal class.

Table 6. Per-class precision, recall, and F1 for LLM-Augmented BiGRU-MHA.

Class	Precision	Recall	F1	Support
Healthy	0.946	0.950	0.948	459
Degraded	0.701	0.635	0.667	96
Warning	0.588	0.625	0.606	32
Failed	0.500	0.692	0.581	13

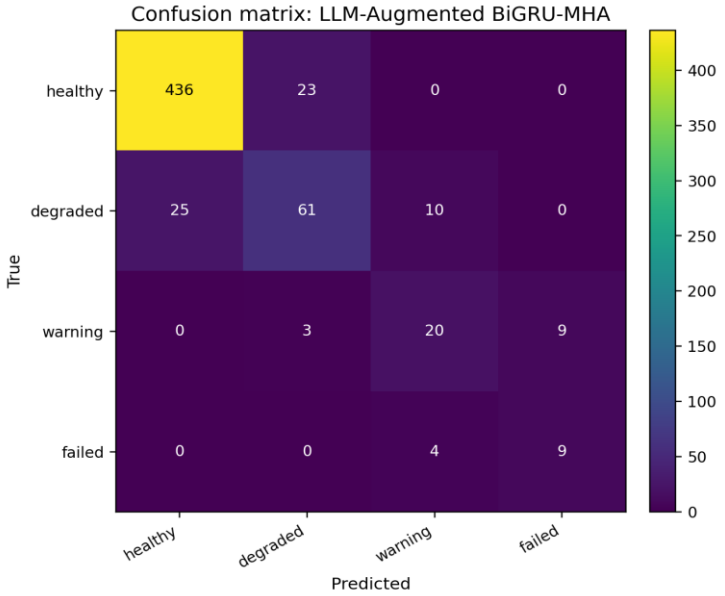

Figure 4. Confusion matrix of the proposed model.

Table 7. Confusion-matrix values for the proposed model.

True class	Pred. healthy	Pred. degraded	Pred. warning	Pred. failed
Healthy	436	23	0	0
Degraded	25	61	10	0

True class	Pred. healthy	Pred. degraded	Pred. warning	Pred. failed
Warning	0	3	20	9
Failed	0	0	4	9

The warning class is the most ambiguous operational boundary. Nine warning drives are escalated to failed, whereas ten degraded drives are escalated to warning. These errors are consistent with a graded-risk formulation in which neighboring states share telemetry patterns. They also reinforce the choice of macro-F1 and class-level recall over raw accuracy, a concern emphasized in imbalanced-learning studies [11], [12], [33].

4.3 Ablation and attention analysis

Table 8 isolates the main design choices. Removing semantic descriptions reduces macro-F1 from 0.695 in the repeated full run to 0.673, and replacing them with random vectors lowers it to 0.635. Removing multi-head attention causes a much larger decline to 0.421, while the forward-only GRU reaches 0.461. The semantic gain is modest in the main comparison but consistent under architecture-controlled ablation: meaningful descriptions perform better than equally sized random vectors.

Table 8. Ablation study of semantics, attention, and bidirectionality.

Model	Accuracy	Balanced acc.	Macro-F1	OVR AUC	Parameters
Full LLM-Augmented BiGRU-MHA	0.887	0.721	0.695	0.948	17,285
No semantic descriptions	0.865	0.748	0.673	0.952	16,133
Random semantic vectors	0.860	0.710	0.635	0.953	17,285
Forward GRU only	0.810	0.595	0.461	0.945	5,765
No multi-head attention	0.767	0.568	0.421	0.767	7,828

Table 9 lists the mean learned attention weights, and Figure 5 visualizes the ranking. Aggregated wear-out degree receives the largest weight at 0.180, followed by program-fail count, percentage used, erase-fail count, and read-error rate. This concentration is consistent with prior evidence that endurance, media errors, and carefully selected SMART indicators are central to SSD failure modeling [3], [4], [8], [9]. Interface and error-log counters receive lower average weights in this run, suggesting that the model distinguished flash-health evidence from signals that may reflect transient communication or firmware events.

Table 9. Mean learned attention weights by SMART attribute.

Attribute	Attention weight
Aggregated wear-out degree	0.180
Program fail count	0.129
Percentage used	0.104
Erase fail count	0.081
Read error rate	0.064
Available spare	0.058
Power cycle count	0.049
Write error rate	0.047
Power-on hours	0.044
NVMe critical warning	0.043
Unsafe shutdown count	0.039
Temperature	0.037
Media/data integrity errors	0.037
Reallocated/block replacement count	0.036
Error information log entries	0.027
UDMA CRC error count	0.025

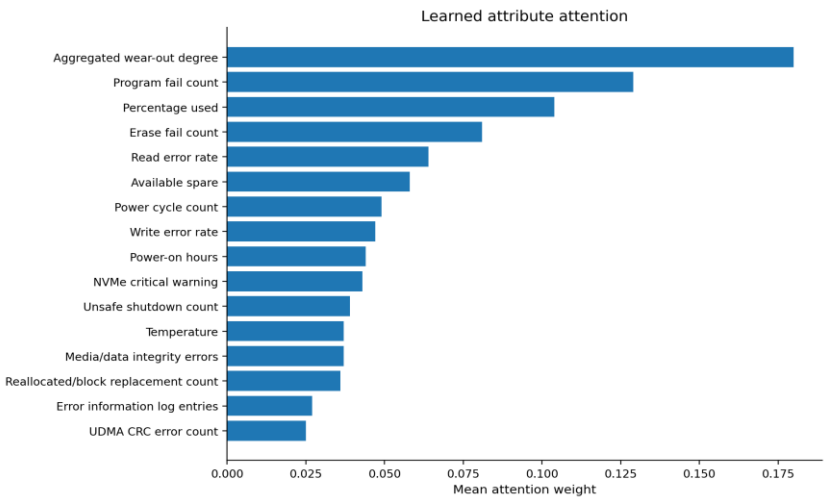

Figure 5. Attention-weight ranking for the 16 SMART attributes.

Attention should be read as a model-inspection signal rather than a causal explanation. It shows where the classifier allocated weight, not which mechanism physically caused a failure. Feature-attribution methods such as SHAP [23], controlled counterfactual perturbations, and maintenance-confirmed root-cause labels would provide complementary evidence.

4.4 Discrimination and calibration

Figure 6 shows failed-vs-rest ROC curves for selected models. The proposed model and the non-semantic BiGRU-MHA both separate failed drives strongly, while the calibration analysis reveals a different issue: high discrimination does not imply that the predicted score is numerically equal to the observed failure rate. This distinction is consistent with established neural-network calibration findings [22].

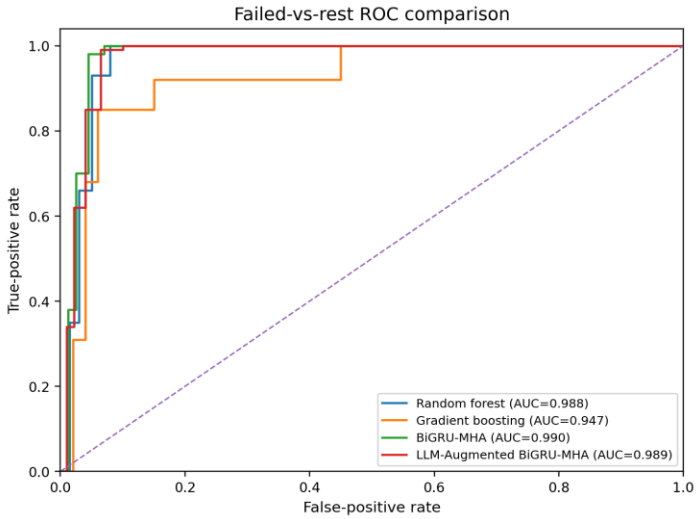

Figure 6. Failed-vs-rest ROC curves for selected models.

Table 10 reports eight failed-class calibration bins for the proposed model, and Figure 7 compares calibration patterns across three models. The first seven bins contain no observed failures, whereas the highest bin has a mean predicted probability of 0.313 and an observed failure rate of 0.173. The score is therefore useful for ranking and screening but is overconfident in the highest-risk region.

Table 10. Failed-class calibration bins for the proposed model.

Bin	Mean predicted probability	Observed failure rate
1	0.004	0.000
2	0.004	0.000
3	0.004	0.000
4	0.005	0.000
5	0.006	0.000
6	0.008	0.000
7	0.022	0.000
8	0.313	0.173

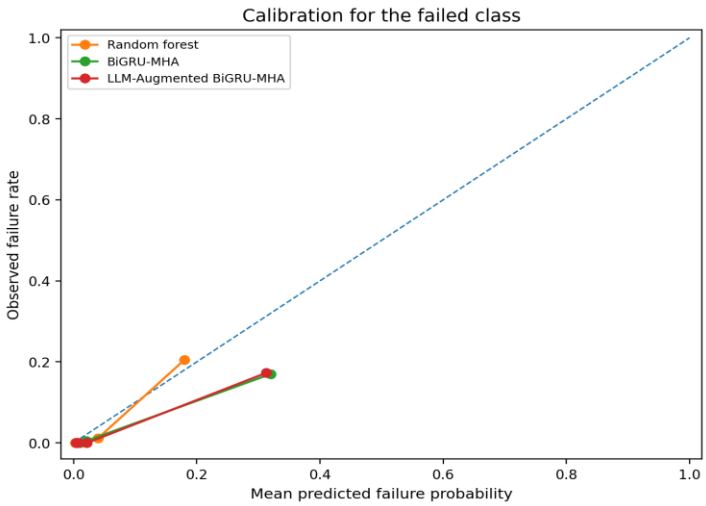

Figure 7. Failed-class calibration curve.

A deployment threshold should therefore be selected after calibration against a maintenance objective. Temperature scaling, isotonic regression, conformal alert control, or uncertainty-aware rejection are plausible options [34], [36], [38], [42]. Such a stage would convert the present ranking score into a decision rule that respects an operator-defined false-alarm budget or replacement capacity.

4.5 Drive-model robustness and computational cost

Table 11 and Figure 8 report macro-F1 by anonymized drive-model code. Performance varies because several model codes contain only one to three failed drives in the test split, and four codes contain none. A1, B2, and C2 show comparatively strong macro-F1, whereas B1 is the weakest. These results are best interpreted as a transparency check on model composition rather than as definitive vendor-level estimates. Larger model-specific samples or leave-one-model-out evaluation would be needed to measure distribution shift more reliably, as emphasized by cross-environment forecasting work [37] and operational workload studies [44].

Table 11. Proposed-model robustness by drive-model code.

Model code	Test records	Failed support	Macro-F1	Balanced accuracy
A1	42	0	0.923	0.971
A2	49	0	0.770	0.860
A3	58	1	0.675	0.771
A4	55	0	0.576	0.733
B1	70	3	0.507	0.524
B2	61	3	0.710	0.717
B3	59	1	0.640	0.737
C1	42	1	0.656	0.736
C2	54	2	0.786	0.853
D1	63	1	0.667	0.663

Model code	Test records	Failed support	Macro-F1	Balanced accuracy
D2	47	1	0.574	0.757

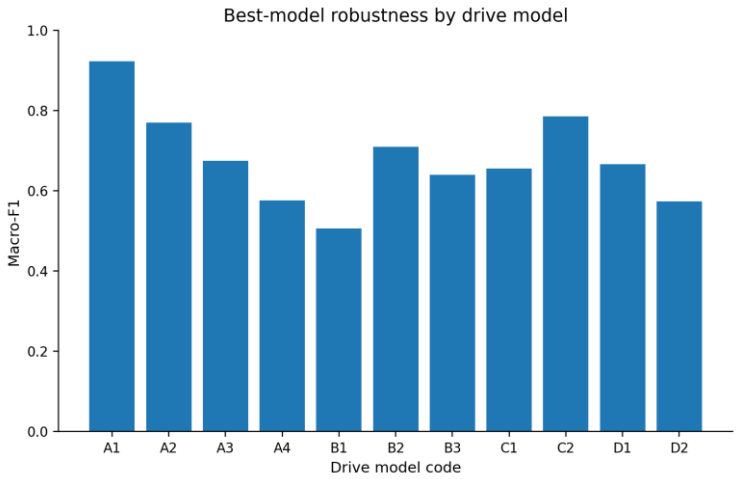

Figure 8. Macro-F1 by anonymized drive-model code.

Table 12 compares measured runtime with macro-F1. The proposed model completes training and scoring in 0.932 seconds for the 3,000-drive experiment, similar to the non-semantic BiGRU-MHA and faster than gradient boosting and the MLP in this environment. The result supports the use of frozen semantic vectors: the classifier gains a language-informed representation without the latency or maintenance burden of online language-model inference. Operational deployment would still require monitoring of preprocessing, calibration, thresholds, and data drift, all of which can become sources of technical debt if left outside the model lifecycle [24], [30], [31].

Table 12. Runtime and macro-F1 of the compared models.

Model	Runtime (s)	Macro-F1
LLM-Augmented BiGRU-MHA	0.932	0.700
BiGRU-MHA	0.938	0.695
Gradient boosting	3.955	0.679
Random forest	0.582	0.632
Rule-based SMART score	0.002	0.623
Logistic regression	0.207	0.599
MLP	2.355	0.588
BiGRU	0.712	0.381
GRU	0.405	0.225

5. Limitations

The experiment uses a one-day SMART snapshot, so the BiGRU operates over an ordered attribute sequence rather than a multi-day temporal history. A longitudinal dataset would support prediction horizons, walk-forward validation, and explicit modeling of the rate at which wear and error counters change before failure.

The degraded and warning states are derived from fixed SMART risk bands rather than from maintenance-confirmed health annotations. The labels are useful for studying graded triage, but their boundaries should be validated against migration decisions, replacement outcomes, and service-level cost before operational use.

The 3,000-drive corpus is substantially smaller than the full Alibaba collection, and several drive-model codes contain very few failed test cases. The aggregate results are therefore more stable than the per-model results. Fleet-scale training and leave-model-out evaluation would provide stronger evidence of cross-vendor generalization.

The semantic descriptions are intentionally concise and fixed. More detailed vendor documentation or trouble-ticket narratives could enrich the semantic channel, but they would also introduce additional curation and version-control requirements. Finally, attention weights are associative rather than causal; they should be combined with counterfactual tests, feature attribution, and engineering investigation before being used as a root-cause claim [23].

6. Conclusion

This study presented LLM-Augmented BiGRU-MHA for four-state SSD health classification from SMART telemetry. The model combines log-scaled raw counters, normalized values, and frozen semantic descriptions, then uses bidirectional recurrence and multi-head attention to integrate evidence across the attribute sequence. On the 3,000-drive experimental corpus, it achieved macro-F1 of 0.700, balanced accuracy of 0.726, weighted-F1 of 0.877, and one-vs-rest macro AUC of 0.952.

Three findings are most important. First, macro-F1 and failed-class recall reveal behavior that accuracy alone obscures under severe imbalance. Second, semantic descriptions provide a measurable benefit: removing them or replacing them with random vectors reduces macro-F1. Third, the learned attention pattern is operationally coherent, concentrating on wear-out, program-fail, percentage-used, and erase-fail indicators.

The model is compact enough for low-latency scoring, but probability calibration and threshold selection remain necessary before direct alarm deployment. Future work should extend the study to longitudinal SMART histories, maintenance-confirmed health labels, fleet-scale drive-model evaluation, and richer vendor-specific semantic descriptions while preserving the fixed, inspectable interface between language information and numeric telemetry.

References

[1] S. Han, P. P. C. Lee, F. Xu, Y. Liu, C. He, and J. Liu, "An In-Depth Study of Correlated Failures in Production SSD-Based Data Centers," in Proc. 19th USENIX Conf. File and Storage Technologies (FAST), 2021, pp. 417–429.

[2] Alibaba-Edu, "dcbrain: SSD Open Data," GitHub repository, 2021. [Online]. Available: https://github.com/Alibaba-Edu/dcbrain/tree/master/ssd_open_data.

[3] F. Xu, S. Han, P. P. C. Lee, Y. Liu, C. He, and J. Liu, "General Feature Selection for Failure Prediction in Large-Scale SSD Deployment," in Proc. IEEE/IFIP Int. Conf. Dependable Systems and Networks (DSN), 2021.

[4] S. Han, J. Wu, E. Xu, C. He, P. P. C. Lee, Y. Qiang, Q. Zheng, T. Huang, Z. Huang, and R. Li, "Robust Data Preprocessing for Machine-Learning-Based Disk Failure Prediction in Cloud Production Environments," arXiv:1912.09722, 2019.

[5] E. Pinheiro, W.-D. Weber, and L. A. Barroso, "Failure Trends in a Large Disk Drive Population," in Proc. 5th USENIX Conf. File and Storage Technologies (FAST), 2007.

[6] B. Schroeder and G. A. Gibson, "Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You?" in Proc. 5th

USENIX Conf. File and Storage Technologies (FAST), 2007.

[7] B. Schroeder, R. Lagisetty, and A. Merchant, "Flash Reliability in Production: The Expected and the Unexpected," in Proc. 14th USENIX Conf. File and Storage Technologies (FAST), 2016.

[8] J. Meza, Q. Wu, S. Kumar, and O. Mutlu, "A Large-Scale Study of Flash Memory Failures in the Field," in Proc. ACM SIGMETRICS, 2015.

[9] Z. Wen, R. Zhang, and C. Wang, "Optimization of bi-directional gated loop cell based on multi-head attention mechanism for SSD health state classification model," in 2025 6th International Conference on Electronic Communication and Artificial Intelligence (ICECAI), Chengdu, China, 2025, doi: 10.1109/ICECAI66283.2025.11171441.

[10] P. Huang, G. Peng, L. Zhang, J. Yang, and D. Li, "An Analysis of Latent Sector Errors in Disk Drives," in Proc. ACM SIGMETRICS, 2007.

[11] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[12] H. He and E. A. Garcia, "Learning from Imbalanced Data," IEEE Trans. Knowl. Data Eng., vol. 21, no. 9, pp. 1263–1284, 2009.

[13] L. Breiman, "Random Forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[14] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," Ann. Stat., vol. 29, no. 5, pp. 1189–1232, 2001.

[15] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," J. R. Stat. Soc. Ser. B, vol. 58, no. 1, pp. 267–288, 1996.

[16] K. Cho et al., "Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation," in Proc. EMNLP, 2014, pp. 1724–1734.

[17] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in Proc. ICLR, 2015.

[19] A. Vaswani et al., "Attention Is All You Need," in Proc. NeurIPS, 2017, pp. 5998–6008.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186.

[21] T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.

[22] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in Proc. ICML, 2017, pp. 1321–1330.

[23] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Proc. NeurIPS, 2017, pp. 4765–4774.

[24] D. Sculley et al., "Hidden Technical Debt in Machine Learning Systems," in Proc. NeurIPS, 2015, pp. 2503–2511.

[25] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[26] A. Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in Proc. NeurIPS, 2019, pp. 8024–8035.

[27] J. Jin and T. Huang, "Market Microstructure Risk Forecasting from Limit Order Books: Multi-Horizon Price-Move Classification and Volatility Estimation with DeepLOB-Style CNN–LSTM and Temporal Transformers," JACS, vol. 3, no. 12, pp. 34–44, Dec. 2023, doi: 10.69987/JACS.2023.31205.

[28] J. Nie and D. Zheng, "Ambiguity-Aware HDFS Log Anomaly Detection with Retrieval-Augmented Failure Narratives and Selective Refusal," JACS, vol. 3, no. 1, pp. 66–80, Jan. 2023, doi: 10.69987/JACS.2023.30105.

[29] G. Liu, S. He, and I. Liu, "LLM-Augmented Multi-Source Root Cause Attribution for CPU and Network Faults in Microservices," JACS, vol. 3, no. 6, pp. 39–57, Jun. 2023, doi: 10.69987/JACS.2023.30604.

[30] B. Zhang, H. Rao, and D. Zhao, "Evidence-Grounded RAG for Cloud-Native DevOps: Hallucination-Resistant AIOps Question Answering over Private Operations Documents," JACS, vol. 4, no. 3, pp. 109–125, Mar. 2024, doi: 10.69987/JACS.2024.40308.

[31] G. Liu, C. Li, and E. Zhang, "OpsLLM for Cloud Incident Triage: Bilingual RAG-Based Root Cause Analysis and Alert Summarization for AI Infrastructure Operations," JACS, vol. 4, no. 4, pp. 97–111, Apr. 2024, doi: 10.69987/JACS.2024.40408.

[32] C. Li, J. Bai, and S. Wang, "Evidence-Chain Reliable RAG: Word-Level Hallucination Detection, Source Attribution, and Provenance Explanation for LLM Applications," JACS, vol. 4, no. 2, pp. 76–92, Feb. 2024, doi: 10.69987/JACS.2024.40207.

[33] J. Jin, T. Huang, and S. Lu, "Cost-Sensitive Learning, Simulated PU Learning, and One-Class Autoencoding for Extreme-Imbalance Credit Card Fraud Detection," JACS, vol. 4, no. 6, pp. 64–73, Jun. 2024, doi: 10.69987/JACS.2024.40605.

[34] Y. Chen, Y. Zhang, D. Chau, and M. Sherman, "Credit Card Default Risk Tiering with Probability Calibration and Uncertainty-Driven Rejection: A Reproducible Study on the UCI Credit Card Clients Dataset," JACS, vol. 3, no. 4, pp. 31–47, Apr. 2023, doi: 10.69987/JACS.2023.30403.

[35] Z. S. Zhong, R. Ma, and H. Zhao, "Human-Uncertainty Distillation for Calibrated Vision Models on CIFAR-10H," JACS, vol. 3, no. 2, pp. 77–89, Feb. 2023, doi: 10.69987/JACS.2023.30206.

[36] S. He, H. Tu, and I. Liu, "Safe PD Capacity Forecasting with Time-Series Foundation Models and Calibrated Uncertainty for Heterogeneous GPU Clusters," JACS, vol. 3, no. 4, pp. 48–66, Apr. 2023, doi: 10.69987/JACS.2023.30404.

[37] S. He, X. Chang, and E. Sun, "Cross-Cloud Transfer Learning for AI Training Capacity Forecasting under Workload and Topology Distribution Shift," JACS, vol. 4, no. 1, pp. 100–120, Jan. 2024, doi: 10.69987/JACS.2024.40108.

[38] S. Chen, S. He, and E. Sun, "Risk-Bounded GPU Resource Oversubscription via Conformal Demand Envelopes in Production AI Clusters," JACS, vol. 4, no. 5, pp. 119–134, May 2024, doi: 10.69987/JACS.2024.40509.

[39] S. Lu and D. Zhou, "TinyLLM-Assisted Intrusion Detection for Real-Time IoT Networks," JACS, vol. 4, no. 8, pp. 72–87, Aug. 2024, doi: 10.69987/JACS.2024.40809.

[40] Y. Li and S. Lu, "Language-Guided Feature Selection for DDoS and Intrusion Detection on CICIDS2017," J. Technol. Informatics Eng., vol. 4, no. 1, pp. 284–305, Apr. 2025, doi: 10.51903/jtie.v4i1.531.

[41] Q. Xin, "Self-Supervised Log Anomaly Detection with LogBERT-Style Transformers: Full Empirical Evaluation on a Reproducible SynHDFS Benchmark," J. Electr. Eng. Comput. Sci., vol. 11, no. 1, pp. 23–35, May 2026, doi: 10.54732/jeecs.v11i1.3.

[42] Q. Xin, "Log Anomaly Detection with Conformal Alert Control and Evidence-Grounded Incident Ticket Generation," AVITEC, vol. 8, no. 2, p. 247, May 2026, doi: 10.28989/avitec.v8i2.3974.

[43] J. Nie and D. Zheng, "Noisy-Neighbor-Aware VM Degradation Risk Modeling with Unsupervised Residual Fusion," JACS, vol. 4, no. 4, pp. 112–123, Apr. 2024, doi: 10.69987/JACS.2024.40409.

[44] S. Zhao, J. Bai, and D. Roberson, "Multi-Horizon GPU Demand Forecasting with Workload Semantics and Operational Risk Curves: An Empirical Study on Alibaba Clusterdata GPU Trace," J. Technol. Informatics Eng., vol. 4, no. 3, pp. 544–571, Dec. 2025, doi: 10.51903/jtie.v4i3.498.