2025 13th International Conference on Intelligent Embedded, MicroElectronics, Communication and Optical Networks (IEMECON) | 979-8-3315-7201-3/25/$31.00 ©2025 IEEE | DOI: 10.1109/IEMECON69302.2025.11365733

Towards Minimal-Telemetry VM Rightsizing: A Lifecycle-Anchored Morphology Approach

Sarnaavho Pal
Department of CSE(AIML)
Institute of Engineering & Management,
University of Engineering and Management, Kolkata, India
piyush30052004@gmail.com

Shirsendu Roy
Department of CSE(AIML)
Institute of Engineering & Management,
University of Engineering and Management, Kolkata, India
shirsenduroysr@gmail.com

Deepsubhra Guha Roy
IEM Centre of Excellence for Cloud Computing & IoT,
Department of CSE(AIML)
Institute of Engineering & Management,
University of Engineering and Management, Kolkata, India
roysubhraguha@gmail.com

Piyali Datta
IEM-IIT Mandi Centre for Joint Research on
Human Computer Interaction, Department of CSE(AIML)
Institute of Engineering & Management,
University of Engineering and Management, Kolkata, India
piyali.datta@iem.edu.in

***Abstract*—Cloud providers face high inefficiencies due to over-provisioned virtual machines (VMs) and churn of short-lived deployments. Existing rightsizing and autoscaling approaches rely on rich telemetry, but often overlook VM lifecycle patterns. We propose a minimal-telemetry, lifecycle-anchored taxonomy of VM morphologies, defined by simple statistics (average, p95, maximum CPU) and lifetime. Using Gaussian Mixture Models, we cluster workloads into operationally meaningful classes, and demonstrate predictive power for identifying short-lived VMs (recall $\approx 92\%$) using Random Forest and XGBoost. This taxonomy enables actionable policies such as early termination of toxic workloads, isolation of unstable clusters, and promotion of reliable ones. We present results on a large VM dataset and show cost, reliability, and sustainability benefits.**

***Index Terms*—Cloud computing, VM clustering, Rightsizing, Lifecycle analysis, Random Forest, XGBoost**

I. Introduction

Cloud platforms are the backbone of modern computing, offering flexibility and scalability through pay-as-you-go models. Yet, cost-effectiveness and reliability remain difficult due to inefficiencies in VM provisioning and high churn rates. Studies show that average CPU utilization across Virtual Machines (VMs) often remains below 40% [1], inflating both operational costs and carbon footprint. A further challenge is rapid VM turnover: many VMs are short-lived, terminated within minutes of creation before delivering meaningful work [2]. These ephemeral deployments waste resources and complicate autoscaling and monitoring systems that assume workload stability. This tension underscores the need for *rightsizing*—adjusting VM resources (vCPUs, memory) to match actual demand. Effective rightsizing avoids both over-provisioning, which drives up costs and emissions, and under-provisioning, which risks SLA violations [3]. Current solutions only partially address this. Cloud-native tools (e.g., Azure Advisor, AWS Compute Optimizer) rely on heuristics [4], while academic approaches often demand rich telemetry (high-frequency CPU, memory, I/O traces) [5]. Neither exploits lightweight lifecycle patterns to balance efficiency and reliability at scale.

We propose a lifecycle-anchored morphology taxonomy of VMs, built from minimal telemetry. Using coarse-grained statistics (average, 95th percentile, maximum CPU, and resource buckets), we apply Gaussian Mixture Models (GMM) to cluster workloads into interpretable categories such as "toxic idle," "spiky unstable," and "reliable workhorse."

Our contributions are threefold:

- A lifecycle-grounded taxonomy of VM morphologies via GMM clustering on lightweight features, enabling scalable classification.
- High-recall predictive modeling of short-lived VMs using Logistic Regression and tree-based methods (Random Forest, XGBoost), consistently achieving recall above 90%.
- An actionable decision framework: early termination of toxic VMs, isolation of unstable ones, and promotion of reliable workloads to premium infrastructure, thereby improving cost, reliability, and sustainability.

We evaluate our approach on $\sim$50,000 VMs from a real-world cloud provider, showing distinct cluster profiles and strong predictive power. The results demonstrate the practicality of lifecycle-aware rightsizing using minimal telemetry.

II. Related Work

Optimizing cloud resource usage has been studied through rightsizing, workload clustering, lifecycle modeling, and minimal telemetry. Below we summarize the most relevant directions.

*1) **VM Rightsizing and Autoscaling**:* Rightsizing dynamically adjusts VM capacity to workload demand. Commercial tools such as AWS *Compute Optimizer* and Azure *Advisor*

979-8-3315-7201-3/25/$31.00 ©2025 IEEE

provide utilization-based recommendations but rely on heuristics [4]. Recent studies confirm that overprovisioning remains widespread in production clouds, with average CPU utilization often below 40% [1], [3]. Academic work explores adaptive and learning-based strategies, but many approaches require rich telemetry.

*2) **Workload Clustering and Lifecycle Modeling**:* Clustering has been used to identify recurring workload patterns and bursty behaviors [6], [7]. More recently, lifecycle-aware studies model VM longevity to guide checkpointing and scheduling. Wu et al. [2] developed probabilistic lifetime models for preemptible VMs, while Xu et al. [8] proposed lightweight morphology proxies for lifecycle-aware management. These highlight the value of combining clustering with lifecycle modeling.

*3) **Minimal Telemetry Approaches**:* Collecting fine-grained telemetry at scale incurs high overhead. Instead, recent work has emphasized lightweight signals such as coarse utilization summaries or start/stop events. Klasnja et al. [9] showed that minimal telemetry can still support carbon-aware scheduling, while Aloufi et al. [5] used reduced features for predictive autoscaling. These works support our claim that effective rightsizing is possible without intrusive monitoring.

Prior efforts advanced rightsizing, clustering, and telemetry-aware scheduling, but gaps remain. Rightsizing often depends on heavy telemetry; clustering rarely links morphologies to lifecycle outcomes; and lifecycle analyses lack predictive integration. Our work addresses these gaps by combining clustering, regression-based explanation, and tree-based prediction into a lifecycle-aware framework.

III. Problem Statement

Cloud virtual machines (VMs) exhibit highly heterogeneous utilization and lifecycles. To optimize rightsizing and reduce waste, we require a compact representation of VM morphologies and a predictive model for identifying short-lived instances.

A. Feature Representation

Let each VM $v \in \mathcal{V}$ be characterized by its utilization profile, from which we extract statistical features:

$$\begin{aligned} \text{avg}_v &= \text{mean CPU utilization}, \\ p95_v &= 95^{th}\text{-percentile utilization}, \\ \text{max}_v &= \max \text{ utilization}. \end{aligned}$$

From these, we define two normalized indicators of workload variability:

$$\text{burst}_v = \frac{p95_v}{\text{avg}_v + \varepsilon}, \quad \text{spike}_v = \frac{\text{max}_v}{p95_v + \varepsilon},$$

where $\varepsilon > 0$ avoids division by zero. The *burstiness* captures sustained high-demand phases relative to average load, while *spikiness* captures transient peaks beyond the 95^{th} percentile.

B. Lifecycle and Label Definition

The lifecycle of VM v is defined as:

$$t_{life}(v) = t_{deleted}(v) - t_{created}(v),$$

where $t_{created}$ and $t_{deleted}$ are timestamps of creation and deletion events. We classify VMs into *short-lived* and *long-lived* using the empirical distribution of lifetimes. Specifically, let Q_{25} denote the 25^{th} percentile of VM lifetimes. We then define the binary indicator:

$$y_v = \mathbb{I}(t_{life}(v) \leq Q_{25}),$$

where $y_v = 1$ indicates a short-lived VM. This operational definition captures the minority population of ephemeral VMs that contribute disproportionately to churn and resource inefficiency.

C. Objectives

Given the above formalization, our study aims to achieve three objectives:

1) **Morphology Discovery:** Cluster VMs into morphological groups based on utilization features $(\text{avg}_v, p95_v, \text{max}_v, \text{burst}_v, \text{spike}_v)$, thereby deriving a lifecycle-anchored taxonomy.
2) **Predictive Modeling:** Train supervised models to predict y_v with emphasis on high recall, ensuring reliable identification of short-lived VMs.
3) **Operational Policies:** Map morphological clusters and predictive outcomes to actionable resource management policies (e.g., proactive termination, deferred provisioning, or monitoring intensification).

This formulation integrates unsupervised discovery, supervised prediction, and operational decision-making into a unified framework for lifecycle-aware VM management.

IV. System Model and Dataset

A. System Model

We consider a cloud environment where each virtual machine (VM) $v \in \mathcal{V}$ is characterized by both performance telemetry and metadata. The system model captures three categories of information:

- **CPU Utilization Statistics:** Aggregated metrics including mean utilization (avg_v), 95th percentile utilization ($p95_v$), and maximum utilization (max_v).
- **Derived Morphology Indicators:** Burstiness and spikiness ratios (as defined in Section III), providing normalized measures of workload variability.
- **Resource Metadata:** VM category (e.g., general-purpose, compute-optimized), virtual core count bucket, and memory allocation bucket.

This representation balances expressiveness with parsimony: only minimal aggregated telemetry is required, avoiding the need for fine-grained traces while retaining the ability to distinguish lifecycle morphologies.

TABLE I: Descriptive statistics of VM dataset (illustrative values).

Metric	Mean	Median	Q1	Q3
Avg CPU (%)	22.3	15.4	5.1	34.7
p95 CPU (%)	61.2	55.9	32.8	84.5
Max CPU (%)	88.1	92.4	75.3	99.9
Lifetime (hours)	26.5	7.4	0.5	32.1

B. Dataset Characteristics

We analyze a real-world dataset of approximately $N \approx 50{,}000$ VMs, each annotated with lifecycle events (creation and deletion timestamps) and utilization statistics. The dataset exhibits strong heterogeneity: some VMs persist for days with stable usage, while others terminate within minutes, exhibiting bursty or spiky demand. Table I summarizes the key descriptive statistics of CPU utilization and lifetime distributions. The lifetime distribution is particularly skewed: the first quartile ($Q1 = 0.5$ hours) indicates that at least 25% of VMs terminate within 30 minutes, highlighting the prevalence of short-lived instances. This motivates the need for both clustering (to categorize morphologies) and predictive modeling (to anticipate early termination).

V. Proposed Methodology

Our methodology integrates three complementary components—discovery, explanation, and prediction—to construct a lifecycle-anchored taxonomy of VM morphologies and operationalize it for cloud rightsizing and reliability optimization. Figure 1 summarizes the workflow.

A. Discovery: Clustering-based Taxonomy

We first employ Gaussian Mixture Models (GMMs) to identify latent VM morphologies. Each VM instance is modeled as coming from one of K Gaussian components [10]:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k), \tag{1}$$

where π_k are mixture weights ($\sum_k \pi_k = 1$), and $\mathcal{N}(x \mid \mu_k, \Sigma_k)$ is a Gaussian density with mean μ_k and covariance Σ_k.

The optimal number of clusters K is selected using the Bayesian Information Criterion (BIC):

$$\text{BIC} = -2 \ln \hat{L} + p \ln(n), \tag{2}$$

where $\hat{L}$ is the maximized likelihood of the model, p is the number of estimated parameters, and n is the number of data points. Lower BIC values indicate better models.

Each VM instance is represented by features derived from its lifecycle and telemetry traces, including:

- *Lifecycle features*: creation and deletion timestamps, lifespan duration.
- *Resource features*: average CPU utilization, maximum CPU, 95^{th} percentile CPU.
- *Configuration features*: vCore bucket, memory bucket, VM category.

The resulting clusters form a morphology-based taxonomy of cloud VMs, capturing short-lived, long-lived, bursty, and steady-state behaviors.

B. Explanation: Regression-based Interpretability

While clustering provides an unsupervised taxonomy, we further employ logistic regression to interpret and validate the predictive significance of cluster membership and configuration features on the likelihood of a VM being short-lived [11].

The regression model is formulated as:

$$\Pr(Y = 1 \mid X) = \frac{1}{1 + \exp\left(-(\beta_0 + \sum_{i=1}^{k} \beta_i X_i)\right)}, \tag{3}$$

where $Y = 1$ denotes a short-lived VM and X_i are explanatory features. The coefficients β_i are transformed into odds ratios, providing interpretable insights into which morphologies are most associated with short-lived behavior.

C. Prediction: Tree-based Models for Operational Deployment

For practical use in cloud platforms, predictive accuracy is essential. We therefore evaluate two tree-based ensemble methods: Random Forests [12] and XGBoost [13]. These models are particularly suited to the task due to:

- Their ability to capture non-linear interactions between VM category, lifecycle features, and resource configuration.
- Robustness to noise and imbalanced datasets when combined with rebalancing strategies such as SMOTE [14].
- High recall for minority classes, enabling proactive policies for short-lived VMs.

a) Random Forest (RF): RF constructs an ensemble of decision trees $\{T_b\}_{b=1}^{B}$, each trained on a bootstrap sample of the dataset [12]. The prediction is the majority vote (classification) or average (regression):

$$\hat{f}_{RF}(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x). \tag{4}$$

b) XGBoost (Extreme Gradient Boosting): XGBoost trains additive trees sequentially to minimize a regularized objective [13]:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{t=1}^{T} \Omega(f_t), \tag{5}$$

where l is a differentiable loss function, f_t is the t-th regression tree, and $\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \|w\|^2$ penalizes model complexity.

c) SMOTE (Synthetic Minority Oversampling Technique): To address class imbalance, we apply SMOTE [14], which generates synthetic minority samples by interpolating between neighbors:

$$x_{\text{new}} = x_i + \delta \cdot (x_{nn} - x_i), \tag{6}$$

where x_i is a minority class sample, x_{nn} is one of its k nearest neighbors, and $\delta \sim U(0, 1)$.

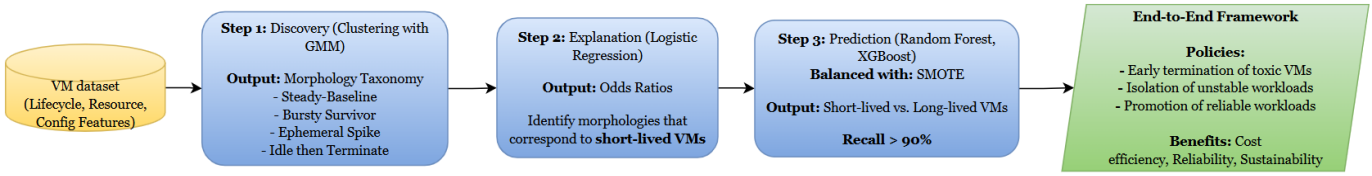

Fig. 1: Proposed end-to-end VM lifecycle framework. The process includes discovery of VM morphologies (via clustering), explanation (using logistic regression), prediction (with ensemble models), and policy enforcement to improve cost, reliability, and sustainability.

Algorithm 1: Lifecycle-Anchored VM Morphology Framework

Input: VM dataset D with lifecycle, resource, and configuration features
Output: Taxonomy clusters, regression insights, predictive model for short-lived VMs
Step 1: Discovery (Clustering)
Extract lifecycle, resource, and configuration features from D ;
Apply Gaussian Mixture Model (GMM) clustering ;
Obtain taxonomy clusters $C = \{c_1, c_2, ..., c_k\}$;
Step 2: Explanation (Interpretability)
Encode cluster membership as features ;
Fit logistic regression model: $\Pr(Y = 1|X) = \sigma(\beta_0 + \sum_{i=1}^{k} \beta_i X_i)$
Compute odds ratios for cluster effects ;
Identify clusters strongly associated with short-lived behavior ;
Step 3: Prediction (Operational Deployment)
Balance dataset using SMOTE ;
Train tree-based models (Random Forest, XGBoost) on D ;
Evaluate via Precision, Recall, F1, ROC-AUC ;
Select model with highest recall for deployment ;
return $\{C$, Logistic Regression Insights, Best Predictive Model$\}$;

Both RF and XGBoost are evaluated using cross-validation, with metrics including precision, recall, F1, ROC-AUC, and PR curves. Our experiments show recall > 0.90 for short-lived VMs, validating these models for operational deployment.

D. End-to-End Framework

Together, these three components form an end-to-end framework:

1) **Discovery:** Construct a lifecycle-anchored taxonomy of VM morphologies via GMM clustering.
2) **Explanation:** Interpret the taxonomy with logistic regression to identify statistically significant correlates of short-lived behavior.
3) **Prediction:** Deploy tree-based ensemble models (RF, XGBoost) with SMOTE balancing for real-time classification.

This methodology not only grounds the taxonomy in statistical evidence but also operationalizes it for predictive decision-making in large-scale cloud platforms.

VI. Results and Discussion

This section presents the experimental results of our lifecycle-anchored VM taxonomy and predictive modeling. We evaluate both *unsupervised clustering* for morphology discovery and *supervised classification* for predicting short-lived VMs. All experiments were conducted on a dataset of approximately 50,000 VMs, as described in Section IV.

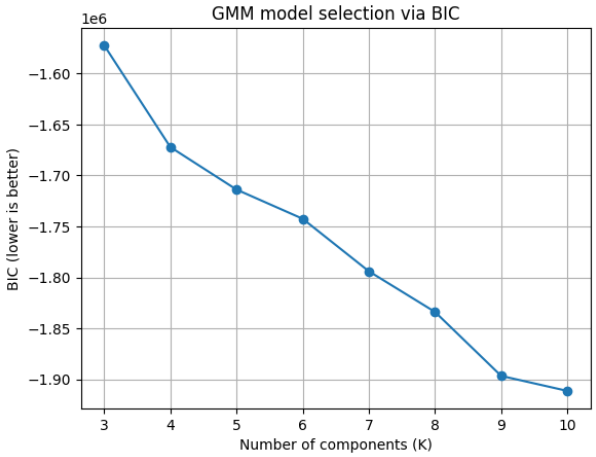

Fig. 2: Model selection for GMM via BIC. A lower BIC indicates better fit.

A. Clustering Results: Morphology Discovery

Gaussian Mixture Models (GMM) were applied to features derived from CPU utilization statistics (average, 95th percentile, maximum) and lifecycle length. The Bayesian Information Criterion (BIC) was used to select the optimal number of clusters, as shown in Figure 2. Once the cluster count was fixed, we visualized the results in the Avg CPU vs. P95 Max CPU feature space. Figure 3 shows that GMM produces well-separated clusters, corresponding to distinct workload morphologies. Empirically, four dominant morphologies emerged:

- **Steady-Baseline VMs:** Long-lived, low-burst workloads with stable utilization.
- **Bursty Survivors:** Long-lived but spiky workloads with intermittent bursts.
- **Ephemeral Spikes:** Short-lived VMs with sharp transient peaks.
- **Idle-then-Terminate:** VMs that remain underutilized and terminate quickly.

B. Operational Trade-offs Across Morphologies

While clustering reveals structural morphologies, operators require guidance on how these morphologies translate into efficiency and reliability trade-offs. We analyze two key dimensions: (i) median inefficiency, defined as $(100 - \text{avg\_cpu})$, capturing resource underutilization; and (ii) short-lived rate, defined as the fraction of VMs in a cluster with t_{life} below

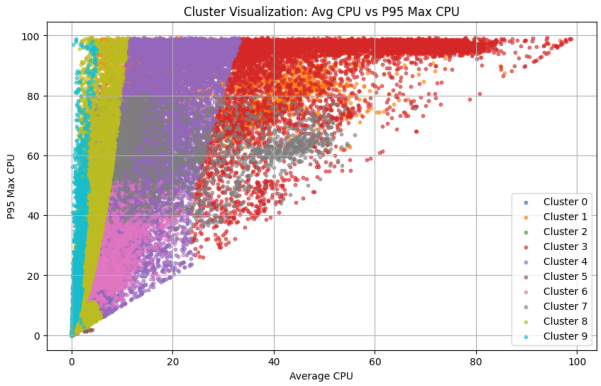

Fig. 3: Cluster visualization in the feature space of average CPU vs. 95th percentile maximum CPU. Each color represents a GMM-derived cluster, highlighting distinct workload morphologies.

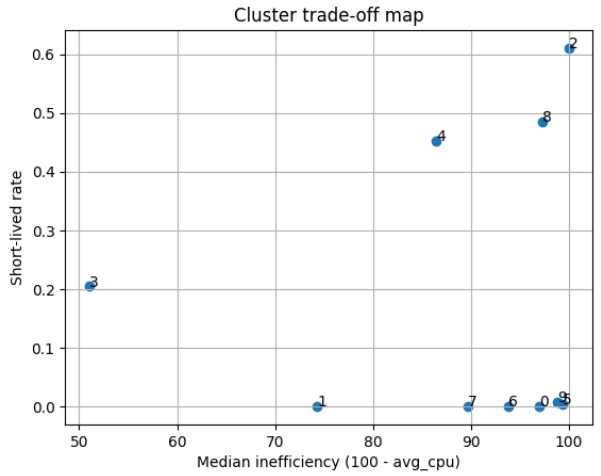

Fig. 4: Cluster trade-off map contrasting inefficiency (x-axis) and short-lived rate (y-axis). Clusters in the upper-right (e.g., 2, 4, 8) are highly inefficient and unstable, whereas clusters in the lower-left (e.g., 1, 3) exhibit more sustainable behavior.

the 25th percentile. Figure 4 presents a trade-off map across clusters, highlighting how morphologies differ in terms of inefficiency and stability. These findings highlight that different morphologies call for differentiated operational policies: clusters such as (2, 4, 8) require aggressive termination or preemption strategies due to their instability, while clusters such as (1, 3) are better candidates for reserved instances or steady-state optimizations.

Table II summarizes the trade-offs quantitatively. A Kruskal–Wallis test across clusters confirmed statistically significant differences ($H = 41164.3$, $p < 0.001$), reinforcing that morphologies are not only visually distinct but also statistically separable.

C. Classification Results: Predicting Short-Lived VMs

For predictive modeling, we balanced the dataset using SMOTE and undersampling, as described in Section V. Three

TABLE II: Trade-off summary of clusters: inefficiency and short-lived rate.

Cluster	Short-lived Rate	Median Inefficiency
0	0.000	97.0
1	0.000	74.3
2	0.610	100.0
3	0.204	51.0
4	0.452	86.4
5	0.004	99.3
6	0.000	93.9
7	0.000	89.6
8	0.485	97.3
9	0.007	98.8

TABLE III: Classification performance on short-lived VM prediction.

Model	**Precision**	**Recall**	**F1**	**ROC-AUC**
Logistic Regression	0.51	0.82	0.63	0.789
Random Forest	0.50	0.92	0.65	0.809
XGBoost	0.50	0.90	0.65	0.809

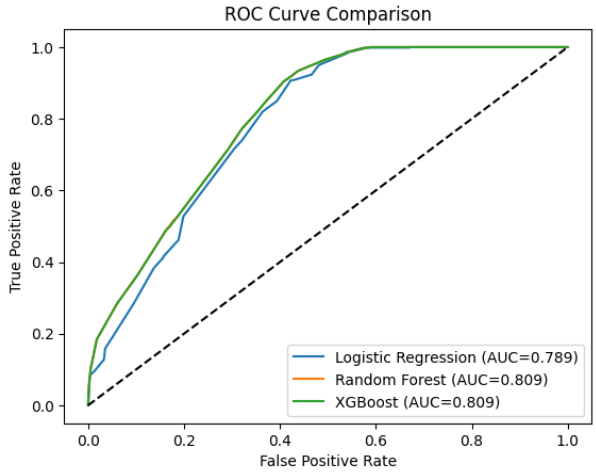

Fig. 5: ROC comparison of predictive models for short-lived VM classification. Random Forest and XGBoost achieve higher AUC ($\approx$ 0.81) compared to Logistic Regression (0.789).

classifiers were evaluated: Logistic Regression, Random Forest (RF), and XGBoost. Table III summarizes the classification metrics. The results show that tree-based methods (RF and XGBoost) consistently achieve higher recall compared to Logistic Regression, which is critical for identifying short-lived VMs. Although precision remains moderate, the high recall ensures that ephemeral workloads are rarely missed, making the models practical for proactive scaling policies. Figure 5 compares the Receiver Operating Characteristic (ROC) curves of the models. Tree-based models achieve superior separation, with ROC-AUC $\approx$ 0.81, while Logistic Regression trails slightly at 0.789.

D. Logistic Regression Interpretability: Odds Ratios

To provide interpretability, we analyzed the logistic regression coefficients and their odds ratios. Table IV reports

TABLE IV: Logistic regression odds ratios for cluster indicators. Odds ratio > 1 indicates higher likelihood of short-lived termination; < 1 indicates reduced likelihood.

Cluster	Coefficient (β)	Odds Ratio
Cluster 2	3.86	47.53
Cluster 8	3.17	23.92
Cluster 4	2.85	17.36
Cluster 3	1.67	5.33
Cluster 5	-1.02	0.36
Cluster 9	-1.69	0.19
Cluster 0	-2.32	0.10
Cluster 6	-2.54	0.08
Cluster 7	-3.56	0.03
Cluster 1	-3.98	0.02

the odds ratios for cluster indicators only, showing which morphologies increase or decrease the likelihood of a VM being short-lived.

E. Discussion

The combination of clustering and classification offers complementary benefits:

- Clustering provides an interpretable taxonomy of VM lifecycles, guiding human operators in policy design.
- Predictive modeling automates early detection of short-lived VMs, enabling proactive interventions.
- Logistic regression offers statistical interpretability through odds ratios, highlighting unstable vs. reliable morphologies.
- Together, they support hybrid strategies, where morphology clusters inform thresholds and predictors provide fine-grained detection.

A key finding is that minimal telemetry—CPU utilization aggregates and lifecycle timestamps—is sufficient to drive both taxonomy and prediction, suggesting that lightweight rightsizing frameworks are feasible without intrusive monitoring. This has important implications for operational scalability, privacy preservation, and cross-cloud portability.

VII. Conclusion and Future Work

In this paper, we proposed a *lifecycle-anchored morphology taxonomy* for virtual machines (VMs) in cloud environments. By leveraging minimal telemetry—aggregated CPU utilization and lifecycle metadata—we demonstrated that VMs can be effectively clustered into distinct workload morphologies using Gaussian Mixture Models (GMM). In parallel, we applied tree-based predictive models (Random Forest and XGBoost) to identify short-lived VMs with high recall, enabling proactive resource management strategies. Together, these contributions provide a foundation for data-driven policies that reduce overprovisioning, mitigate churn, and improve the operational efficiency of cloud platforms.

Future work can extend this research along several directions:

- **Richer Feature Sets:** Incorporating additional telemetry such as memory, disk, and network I/O usage could refine morphology definitions and improve predictive accuracy.
- **Real-Time Detection:** Designing online models that can identify VM morphologies within minutes of instantiation, thereby enabling early corrective actions.
- **Carbon- and Cost-Aware Scheduling:** Mapping morphologies to policies that optimize for energy efficiency and sustainability, in addition to traditional performance metrics.
- **Cross-Cloud Generalization:** Evaluating the taxonomy across different providers (AWS, Azure, GCP) and workload types to establish robustness and portability.
- **Operational Integration:** Embedding the proposed framework into autoscalers or advisory systems, bridging the gap between research prototypes and production deployments.

Overall, our results highlight that even with lightweight telemetry, lifecycle-aware clustering and prediction can jointly advance cloud rightsizing, improve reliability, and pave the way for sustainable, intelligent resource management.

Acknowledgments

The authors would like to express their sincere gratitude to the IEM Centre of Excellence for Cloud Computing and IoT for providing financial and infrastructural support through research grants IEMT(S)/2024/02-G19 and IEMT(S)/2023/02-G05 and to IEDC-CSE at the Institute of Engineering and Management, Kolkata, for offering technical resources and a collaborative environment.

References

[1] N. S. Islam and S. Bagchi, "A study of resource utilization in public cloud vms," *ACM Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1–25, 2021.
[2] X. Wu *et al.*, "Modeling and optimizing preemptible vm lifecycles," in *Proc. of ACM SoCC*, 2024.
[3] R. Prakash *et al.*, "Workload inefficiency in large-scale cloud environments," *IEEE Transactions on Cloud Computing*, 2025, early Access.
[4] M. Goli and R. Buyya, "A reinforcement learning-based auto-scaling system for cloud applications," in *Proc. of IEEE/ACM CCGrid*, 2018, pp. 492–501.
[5] S. Aloufi *et al.*, "Lightweight predictive autoscaling with minimal features," *Journal of Cloud Computing*, vol. 10, no. 1, pp. 1–15, 2021.
[6] J. Hare and S. Uhlig, "Characterizing cloud workloads via temporal clustering," in *Proc. of IEEE INFOCOM*, 2018, pp. 398–406.
[7] S. Song and D. Rossi, "Behavior-aware clustering of cloud workloads," in *Proc. of IEEE ICDCS*, 2013, pp. 765–774.
[8] L. Xu *et al.*, "Lightweight morphology proxies for cloud workload management," in *Proc. of IEEE ICAC*, 2020, pp. 45–54.
[9] M. Klasnja *et al.*, "Carbon-aware scheduling with minimal telemetry," in *Proc. of ACM EuroSys*, 2023.
[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
[11] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*, 3rd ed. Wiley, 2013.
[12] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
[13] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proc. of KDD*, 2016, pp. 785–794.
[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.