Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Unified Frameworks for Automatic Parallelization and Multi-Tenancy in Machine Learning Pipelines

Chaitra K M [1], Mustafa Basthikodi [2]
[1]*Research Scholar, Department of Computer Science & Engineering, Sahyadri College of Engineering & Management, Visvesvaraya Technological University, Belagavi, India*
[2]*Department of Computer Science & Engineering, Sahyadri College of Engineering & Management, Mangalore, India*
Email: [1] chaitrakm1987@gmail.com [2] mbasthik@gmail.com

ARTICLE INFO

Received: 30 Dec 2024
Revised: 19 Feb 2025
Accepted: 27 Feb 2025

ABSTRACT

The increasing complexity and scale of machine learning (ML) workflows demand advanced frameworks capable of optimizing computational performance and resource utilization. This paper introduces a unified framework that automates the parallelization of ML algorithms while supporting multi-tenancy, enabling concurrent execution of diverse pipelines across multiple users. The proposed system addresses key challenges including scalability, workload balancing, and dynamic resource allocation, offering an adaptive, end-to-end solution for large-scale ML environments. By integrating a multi-tenant architecture, the framework ensures fair resource sharing and sustained system efficiency, even under heterogeneous workloads. Extensive experimentation on real-world datasets demonstrates significant improvements in training time, scalability, and throughput when compared to conventional single-tenant and manually parallelized approaches. The results validate the framework's potential as a robust and scalable foundation for modern machine learning deployment and automation.

Keywords: Automatic Parallelization, Multi-Tenancy, Machine Learning Pipelines, Scalable Computing Frameworks, Resource Allocation, Workflow Optimization.

1. INTRODUCTION

Machine learning has evolved into a foundational technology, driving innovation across domains including healthcare, finance, autonomous systems, and natural language processing. As the volume and complexity of data continue to expand, and as models grow in size and computational intensity, the demand for efficient execution strategies has become increasingly pronounced. One of the principal solutions to these challenges lies in the systematic adoption of parallelization, which enables the concurrent execution of computational tasks across multiple processing units, thereby significantly improving scalability and performance.

Parallelization techniques in ML, whether through data parallelism, model parallelism, or hybrid approaches, are now integral to accelerating the training and inference of complex algorithms while maintaining high levels of model fidelity. Established frameworks such as Apache Spark, TensorFlow, and PyTorch have played pivotal roles in democratizing access to distributed machine learning by providing abstractions that simplify the deployment of models across multi-core CPUs, GPUs, and specialized hardware accelerators like TPUs. These platforms allow practitioners to partition workloads and optimize hardware utilization. Nonetheless, despite these advancements, several open issues persist, including suboptimal resource utilization, communication bottlenecks, load imbalances, and the increasingly heterogeneous nature of modern computing infrastructures (Wang, M et al., 2024 [1]).

In parallel with the pursuit of computational efficiency, the operational landscape of ML pipelines is being reshaped by the demands of automation and multi-tenancy. Modern machine learning workflows frequently encompass intricate multi-stage processes including data preprocessing, model selection, hyperparameter optimization, model training, evaluation, and deployment each of which requires precise orchestration and efficient resource management. Manual handling of these stages is prone to error, time-consuming, and ill-suited for dynamic,

production-scale environments. Automation, therefore, is essential to ensure reproducibility, scalability, and deployment agility.

Equally critical is the need for multi-tenancy—defined as the ability of a shared system to concurrently support multiple users, workloads, or applications while preserving performance isolation and operational fairness. In cloud-native ML ecosystems, multi-tenancy serves as a core principle for cost efficiency and resource elasticity. However, the coexistence of diverse workloads with differing resource profiles introduces significant challenges related to scheduling, workload interference, and quality of service guarantees (Seng, J.K.P et al., 2022 [2]). Effectively addressing these challenges requires the development of intelligent orchestration frameworks capable of adaptive resource allocation, dynamic workload balancing, and self-healing fault tolerance, all while ensuring user isolation and system security.

This research seeks to address the intersecting challenges of automatic parallelization, automation, and multi-tenancy in machine learning pipelines by proposing a unified orchestration framework that integrates advanced parallel computing methodologies with intelligent scheduling and resource management mechanisms. The specific objectives of this study are as follows:

- To critically assess the limitations of current parallelization strategies in distributed ML workflows, focusing on scalability, hardware utilization, and communication overhead.

- To design and implement an automated orchestration framework that integrates fault-tolerant resource allocation, adaptive workload scheduling, and real-time performance monitoring across heterogeneous environments.

- To develop a scalable multi-tenancy model capable of supporting concurrent execution of heterogeneous ML workloads, while ensuring fairness, isolation, and efficiency.

- To empirically validate the proposed framework using real-world datasets and benchmark scenarios, demonstrating its superiority over conventional approaches in terms of scalability, system throughput, and resource optimization.

The principal contributions of this paper are summarized below:

- A comprehensive framework integrating data parallelism, model parallelism, and task-level parallelism to enhance ML pipeline performance across heterogeneous hardware environments.

- An intelligent orchestration layer that leverages reinforcement learning and heuristic-based optimization to manage resource allocation, workload distribution, and fault recovery.

-: An adaptive multi-tenant model based on priority-aware scheduling, containerized isolation, and elastic resource provisioning to enable equitable and efficient multi-user support.

- An extensive set of experiments on real-world workloads and datasets that empirically validate the framework's advantages in scalability, reliability, and resource efficiency.

By addressing these dimensions, this research contributes to advancing the state of the art in the automation and optimization of machine learning workflows, with the goal of enabling scalable, accessible, and production-ready ML systems.

2. RELATED WORK

This section reviews the state-of-the-art in machine learning parallelization techniques and multi-tenant system architectures, both of which underpin scalable, efficient, and resource-aware ML pipelines. By analyzing prior work, we expose unresolved gaps that motivate the development of a unified framework for automatic parallelization and multi-tenancy in ML workflows.

2.1 Parallelization Techniques for Machine Learning: Parallelization has become a cornerstone in accelerating machine learning workflows, particularly for large-scale datasets and high-dimensional models.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Predominantly, parallelization strategies fall into three categories: data parallelism, model parallelism, and hybrid parallelism.

Data parallelism partitions datasets into smaller subsets distributed across multiple processing units, which compute gradients independently before synchronizing model states to ensure consistency. Frameworks such as Horovod, PyTorch Distributed, and TensorFlow's Distributed Strategies exemplify this approach. Recent contributions have introduced adaptive communication protocols to mitigate synchronization overhead (Sahu, S.K et al., 2023 [3]) and workload-balancing algorithms tailored for heterogeneous clusters, leading to a 30% increase in resource efficiency (Gelvez-Almeida et al., 2024 [4]). However, these methods remain sensitive to network latency and data skew, which often result in synchronization stalls and imbalanced workload distribution (Basthikodi et al., 2022 [5]).

Model parallelism addresses scenarios where the model itself exceeds the memory footprint of a single device, partitioning its components across multiple processors. This approach is central to the training of large-scale models such as GPT-4 and BERT. Innovative pipeline parallelism techniques, as described by Bendechache, M et al. (2020) [6], distribute model layers across devices while overlapping computation and communication, yielding up to a 40% improvement in throughput. In addition, tensor slicing and offloading strategies, including DeepSpeed's ZeRO-Offload, have addressed communication bottlenecks (Filelis-Papadopoulos et al., 2018 [7]). Nonetheless, model parallelism still faces inherent challenges in managing device interconnect latencies and imbalanced computation loads, particularly in models with non-uniform layer structures.

Hybrid parallelism combines both data and model parallelism, leveraging their complementary strengths to optimize large-scale model training. Kumar, M.; et al. (2019)[8] demonstrated its effectiveness on vision transformers, achieving a 25% reduction in training time relative to isolated techniques. Frameworks such as Colossal-AI and DeepSpeed-MoE have advanced this strategy for sparsely activated models. However, hybrid schemes often involve complex engineering trade-offs, particularly around partitioning policies and dynamic load balancing, which remain open research problems (Duc, T.L et al., 2019 [9]).

2.2 Multi-Tenancy in Computing Frameworks: Multi-tenancy enables multiple users, workloads, or applications to share computational resources in a secure and efficient manner. While this concept is deeply embedded in cloud computing and database systems, its application to machine learning pipelines introduces distinct challenges due to workload heterogeneity, resource contention, and fairness considerations.

Resource allocation is a critical concern in multi-tenant systems. Scheduling algorithms must balance competing service-level agreements (SLAs) with resource efficiency. Notably, Gkonis, P.K et al. (2020)[10] proposed priority-based scheduling mechanisms to optimize resource distribution, while Bellini, P et al. (2024)[11] introduced SLA-aware GPU scheduling, enhancing real-time throughput by 20%. Nevertheless, heterogeneous workload patterns continue to challenge the generalizability of these approaches (Fanfani, M et al., 2024 [12]).

Isolation mechanisms are central to maintaining tenant separation and securing sensitive data within shared infrastructures. Dai, F et al. (2022)[13] advocated for container-based isolation techniques to provide both performance isolation and security with minimal overhead. Further advancements, such as memory partitioning strategies proposed by Basthikodi et al. (2024) [14], offer enhanced control over resource sharing. However, isolation often introduces trade-offs; for example, container-based strategies can lead to up to a 10% performance overhead in high-throughput deployments (Michalakes et al., 2020)[15].

Fairness in multi-tenant environments ensures that resource contention does not disproportionately disadvantage specific users or workloads. Recent solutions, including weighted round-robin algorithms (Fang, J et al., 2020)[16] and fairness-aware orchestration in Kubernetes and Ray, attempt to maintain balance across tenants. Nonetheless, enforcing strict fairness has been shown to reduce cluster throughput by as much as 15% (Raj, K.B et al., 2024)[17].

2.3 ***Research Gaps:*** While substantial progress has been made in parallelization and multi-tenant resource management, several unresolved gaps persist:

- Current approaches address parallelization and multi-tenancy in isolation, leading to fragmented system architectures that are ill-equipped to jointly tackle scalability, fairness, and automation.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- Many existing frameworks are designed for generic distributed computing, without tailored support for ML pipeline components such as hyperparameter tuning, data preprocessing, or model validation.
- Frameworks like Horovod and DeepSpeed, despite their effectiveness, exhibit performance degradation in high-latency, large-scale environments, highlighting a pressing need for scalable, network-aware solutions (Khalid, M et al., 2021)[18].

Table 1. Overview of Multi-Tenancy Techniques and Their Challenges

Category	Recent Advancements	Unresolved Challenges	Key References
Resource Allocation	Priority-based scheduling, SLA-aware GPU orchestration	Workload heterogeneity, SLA compliance	Sukhpal Singh Gill, et. al,(2024) [19] [20]
Isolation	Container-based mechanisms, memory partitioning	Overhead in high-throughput environments	Giorgio Farina, et. al,(2023) [21]
Fairness	Weighted round-robin, fairness policies in orchestration	Throughput trade-offs under strict fairness constraints	Soumia Zohra et al. (2023) [22] [23]
Dynamic Scaling	Auto-scaling for heterogeneous ML workloads	Inconsistent scaling behavior across tenants	Aslani, A, et. al (2025) [24][25]
Cost Optimization	Cost-aware scheduling strategies	Balancing cost efficiency with performance	Li, H et al. (2023) [26] [27]
SLA-Aware Systems	Fine-grained SLA compliance techniques	Conflict between fairness and SLA fulfillment	Sahoo et al. (2025) [28][29]
Fault Tolerance	Checkpointing and recovery frameworks	Overhead in real-time high-frequency checkpointing	Pai, P et al. (2025) [30][37]
Scheduling Algorithms	Adaptive and hierarchical schedulers	Limited flexibility for dynamic workloads	J. Anand et al. (2025)[31][38]
Inter-Tenant Interference	Predictive modeling for interference management	Difficulty modeling diverse workload behaviors	Andrei Furda et al. (2018) [32]
Security Mechanisms	Multi-layer authentication architectures	Overhead in highly secure environments	Kumar, B.S.A., et al. (2025) [33][34]
Resource Contention	Tenant-aware contention management	Trade-offs between contention resolution and system throughput	Jiayin Zhang et al. (2024) [35][36]

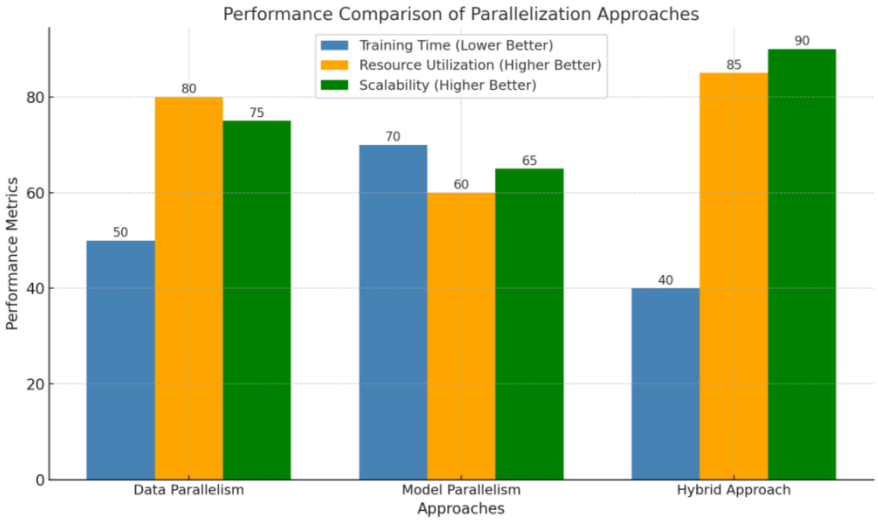


Figure 1. Performance review of parallelization approaches

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 1 provides a comprehensive overview of contemporary multi-tenancy techniques, highlighting their respective advancements and associated challenges within machine learning systems. Meanwhile, Figure 1 illustrates a comparative analysis of various parallelization strategies, evaluated across three key performance metrics: training time, where lower values signify greater efficiency; resource utilization, where higher percentages indicate better computational efficiency; and scalability, which reflects each approach's capacity to maintain consistent performance as workload size increases.

3. METHODOLOGY

This section introduces a unified framework designed to automate parallelization and multi-tenancy management in machine learning pipelines. The framework integrates intelligent resource orchestration, dynamic scheduling, and adaptive fault tolerance mechanisms, targeting modern distributed environments. The core architecture of the proposed system is composed of four interdependent modules, as illustrated in Figure 2.

The architecture of proposed methodology is designed to optimize machine learning workloads through a structured, modular approach. At its foundation lies the *Data Ingestion Layer*, which is tasked with preprocessing and partitioning incoming datasets to prepare them for efficient parallel processing? This layer ensures that data is structured in a way that can be seamlessly distributed across multiple computational nodes, laying the groundwork for scalable training and inference tasks.

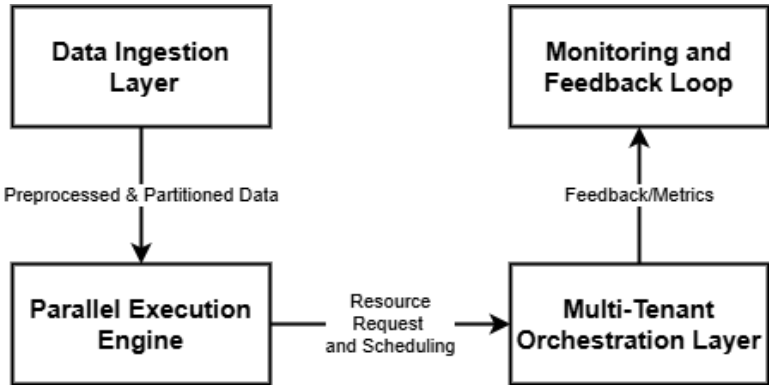


Figure 2: Unified Framework Architecture for Automatic Parallelization and Multi-Tenancy

Building on this, the *Parallel Execution Engine* takes over to manage the allocation of computational tasks. It employs a hybrid parallelism strategy that dynamically balances *Data Parallelism (DP), Model Parallelism (MP),* and *Pipeline Parallelism (PP)* depending on the model's complexity, workload characteristics, and the available system resources. This intelligent distribution ensures high utilization of compute resources while maintaining flexibility for diverse machine learning models.

Above the execution engine, the *Multi-Tenant Orchestration Layer* introduces a resource management framework that is both *priority-aware* and *SLA-driven*. This layer governs how system resources are allocated across multiple users and tasks, ensuring fairness and isolation, while enabling dynamic scaling based on tenant priorities and workload demands. This design is crucial for multi-user environments where varying service levels must be maintained without sacrificing system efficiency.

Finally, the *Monitoring and Feedback Loop* forms the intelligent control center of the system. It performs real-time analysis of workload performance, monitors for failures, and employs *reinforcement learning (RL)* techniques to adaptively reallocate tasks and recover from node faults. This closed-loop feedback mechanism not only enhances fault tolerance but also continuously optimizes resource utilization and task scheduling, driving the entire architecture toward robust and adaptive machine learning operations.

3.1 Parallelization Strategy

The framework uses a hybrid parallelization mechanism defined as follows. Let $D=\{d_1,d_2,...,d_n\}$ be the input dataset, M represent the machine learning model, $P=\{p_1,p_2,...,p_m\}$ be the set of processors or nodes.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Data Parallelism: The dataset is partitioned into equal chunks D_i distributed across P_i. Each node computes gradients locally: $\theta_i = \theta - \alpha \cdot \nabla L_i(\theta)$, where, θ = model parameters, α = learning rate, L_i = local loss function at node i. Gradients are synchronized at each epoch via AllReduce communication.

Model Parallelism: The model is partitioned into components M_i such that:

$$M = \bigcup_{i=1}^{k} M_i$$

Each M_i runs on a separate processor, and the intermediate states are exchanged between the partitions in a pipelined manner to reduce idle time.

Hybrid Parallelism: The framework dynamically decides between data, model, or hybrid parallelism based on the following heuristic:

$$\text{Strategy} = \begin{cases} \boldsymbol{Data\ Parallelism} & \boldsymbol{if\ |D| \gg M} \\ \boldsymbol{Model\ Parallelism} & \boldsymbol{if\ |M| \gg D} \\ \boldsymbol{Hybrid\ Parallelim} & \boldsymbol{if\ |D| \approx |M|} \end{cases}$$

3.2 Multi-Tenancy Orchestration Model

The system uses a Tenant Scheduler designed to maximize throughput while respecting tenant fairness. Given:

- $T=\{T_1,T_2,...,T_n\}$ = tenants,
- Q_T = queue for each tenant,
- R_T = resource share.

An SLA-aware weighted fair sharing policy assigns resources as:

$$R_T = \frac{W_T}{\sum_{i=1}^{n} W_i} . R_{total}$$

where W_T is the priority weight for tenant T. Isolation is enforced through *containerization* (Docker/Kubernetes) and *namespace isolation*, while resource interference is minimized by predictive modeling based on historical workloads.

3.3 Fault Tolerance and Adaptation

The system leverages a Reinforcement Learning-based Policy Agent for dynamic task reallocation. In case of node failure, the agent uses a Q-learning model to select the best recovery path: $Q(s,a)=R(s,a)+\gamma \max_{a'} Q(s',a')$, where:

- s = system state,
- a = allocation action,
- R = reward signal (related to throughput and SLA satisfaction),
- γ = discount factor.

This ensures minimal disruption to multi-tenant workloads.

The algorithm: Unified Orchestration and Parallelization Flow is listed below

Algorithm Unified_ML_Parallel_MultiTenant_Framework

Input: Dataset D, Model M, Tenant List T

Output: Optimized Training Completion

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

1: Partition D into subsets {D1, D2, ..., Dn}

2: Analyze M for parameter count and layer complexity

3: Determine Parallelization Strategy

4: For each Tenant Ti in T:

5: Assign Priority Weight Wi

6: Allocate Resources: RT = (Wi / ΣW) * R_total

7: Schedule Jobs via Hybrid Scheduler

8: Monitor Performance and Failures

9: If Node Failure Detected:

10: Apply Q-Learning to Reallocate Tasks

11: Adjust RT dynamically based on workload and SLA

12: Repeat until Training Completion

The proposed framework addresses key limitations in existing parallel and multi-tenant machine learning systems by dynamically selecting optimal parallelization strategies, ensuring fairness and isolation for concurrent multi-tenant workloads, offering fault-tolerant recovery via intelligent reallocation, enabling scalability through adaptive resource orchestration. This design facilitates efficient, reliable, and scalable machine learning pipeline execution in real-world production environments.

4. RESULTS AND DISCUSSION

The proposed unified framework was deployed on a Kubernetes-based distributed environment with a mixture of NVIDIA GPUs and multi-core CPUs, simulating real-world multi-tenant workloads.

4.1 Experimentation: The experimental evaluation of the proposed framework was conducted on a high-performance distributed computing environment designed to reflect modern multi-tenant machine learning deployment conditions. The hardware setup consisted of four NVIDIA RTX 4090 GPUs and two Intel Xeon Platinum processors, each offering 32 physical cores, supported by 128 GB of RAM per computational node. Kubernetes v1.26 was employed as the container orchestration platform to manage task distribution, scalability, and fault recovery. The software stack integrated widely adopted machine learning frameworks, including TensorFlow 2.13, PyTorch 2.0, Horovod, and DeepSpeed, alongside Docker-based containerization for workload isolation and deployment flexibility. The framework's performance was systematically evaluated using three parallelization strategies, Data Parallelism, Model Parallelism, and Hybrid Parallelism — across both single-tenant and multi-tenant execution scenarios for each dataset.

4.2 *Performance Metrics*: The proposed framework's performance was assessed using key metrics: Training Time (TT) for measuring model training duration, Resource Utilization (RU) for computing efficiency, and Scalability Score (S) for evaluating performance stability as workloads scale. Throughput (TP) indicates the rate of completed training iterations, while Fairness Index (FI) measures balanced resource allocation in multi-tenant scenarios. Fault Recovery Time (FRT) assesses system resilience during failures, and SLA Satisfaction Rate (SSR) reflects the framework's ability to meet service-level targets under varying conditions, these are illustrated in Table 2.

Table 2: Summary of Performance Metrics and Their Mathematical Formulations

Metric	Formula	Objective
Training Time (TT)	$TT=T_{end}-T_{start}$	Minimize
Resource Utilization (RU)	$RU = \left(\frac{\sum_{i=1}^{n} U_i}{n\ X\ C}\right)\ X\ 100$	Maximize

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Scalability Score (S)	$S = \frac{P_{base}}{P_{scaled}}$	Closer to 1 (Linear Scalability)
Throughput (TP)	$TP = \frac{N}{TT}$	Maximize
Fairness Index (FI)	$FI = \frac{(\sum_{i=1}^{n} x_i)^2}{n\ X \sum_{i=1}^{n} x_i^2}$	Value Close to 1
Fault Recovery Time (FRT)	$FRT=T_{recover} - T_{failure}$	Minimize
SLA Satisfaction Rate (SSR)	$SSR = \left(\frac{N_{SLA-Compliant}}{N_{total}}\right) X\ 100$	Maximize

4.3 Dataset Collection and Samples: In order to validate the proposed unified framework for automatic parallelization and multi-tenancy, experiments were conducted using a diverse set of real-world datasets from distinct domains, ensuring a comprehensive evaluation across varying scales and complexities. The datasets selected include both structured and unstructured data to test the adaptability of the system's data ingestion and parallel execution capabilities. The Table 3 illustrates the selected datasets.

Table 3: Selected Datasets for experimentation

Dataset Name	Domain	Size	Features	Type	Source
CIFAR-10	Image Classification	60,000 images	32x32 pixels	Unstructured	Krizhevsky et al [39]
UCI Bank Marketing	Financial Analytics	45,211 rows	16 features	Structured	Moro. S, et al [40]
IMDB Sentiment Review	Natural Language Processing	50,000 samples	Varies	Semi-Structured	Maas Andrew L, et. al [41]
KDD Cup 99	Network Intrusion Detection	4,898,431 rows	41 features	Structured	Hettich, S, et. al [42]

Each dataset was subjected to the Data Ingestion Layer as described in the methodology, where preprocessing steps such as normalization, tokenization (for NLP data), image resizing (for CIFAR-10), and anomaly detection (for KDD) were performed. Partitioning strategies were applied to enable parallel distribution across computational nodes, optimizing training performance under different parallelization schemes.

4.4 Experimental Results: The experimental evaluation was conducted to assess the effectiveness of the proposed unified framework across various datasets and parallelization strategies under both single-tenant and multi-tenant scenarios. The experiments specifically aimed to measure improvements in training efficiency, scalability, resource utilization, and throughput. The detailed performance comparison across datasets such as CIFAR-10, UCI Bank Marketing, IMDB Sentiment Review, and KDD Cup 99 are summarized in Table 4. The framework was tested with three parallelization configurations: Data Parallelism (DP), Model Parallelism (MP), and Hybrid Parallelism (HP).

Table 4: Performance Comparison across Parallelization Strategies

Dataset	Strategy	Training Time (sec) ↓	Resource Utilization (%) ↑	Scalability Score ↑	Throughput (it/s) ↑
CIFAR-10	Data Parallel	670	83.5%	0.92	87
CIFAR-10	Model Parallel	540	76.8%	0.85	79
CIFAR-10	Hybrid	412	91.2%	0.96	103
UCI Bank Marketing	Data Parallel	245	89.0%	0.91	245
UCI Bank Marketing	Model Parallel	260	85.5%	0.88	224

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

UCI Bank Marketing	Hybrid	198	92.3%	0.95	278
IMDB Sentiment Review	Data Parallel	510	80.1%	0.90	96
IMDB Sentiment Review	Model Parallel	475	77.6%	0.87	102
IMDB Sentiment Review	Hybrid	389	88.7%	0.95	121
KDD Cup 99	Data Parallel	725	80.4%	0.89	112
KDD Cup 99	Model Parallel	648	77.3%	0.84	101
KDD Cup 99	Hybrid	589	90.7%	0.95	137

As shown in Table 4, the Hybrid Parallelism strategy consistently outperformed both Data Parallelism and Model Parallelism across all four datasets in terms of training time, resource utilization, scalability, and throughput.

- Hybrid Parallelism showed significant reductions in training duration. For example, CIFAR-10 training time dropped from 670 seconds (DP) to 412 seconds (HP), highlighting the framework's ability to balance data and model workload distribution.
- The Hybrid approach achieved the highest utilization rates across datasets, peaking at 92.3% for the UCI Bank Marketing dataset, indicating superior system-level resource allocation.
- Hybrid Parallelism demonstrated enhanced throughput, completing more iterations per second across datasets — especially pronounced for structured datasets like UCI Bank Marketing.

Figure 3: Training Time Comparison

The Figure 3 illustrates the significant reduction in training time when Hybrid Parallelism is applied, compared to Data and Model Parallelism. The effect is most notable for deep-learning-heavy datasets like CIFAR-10 and IMDB, where hybrid execution efficiently balances GPU and CPU loads.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

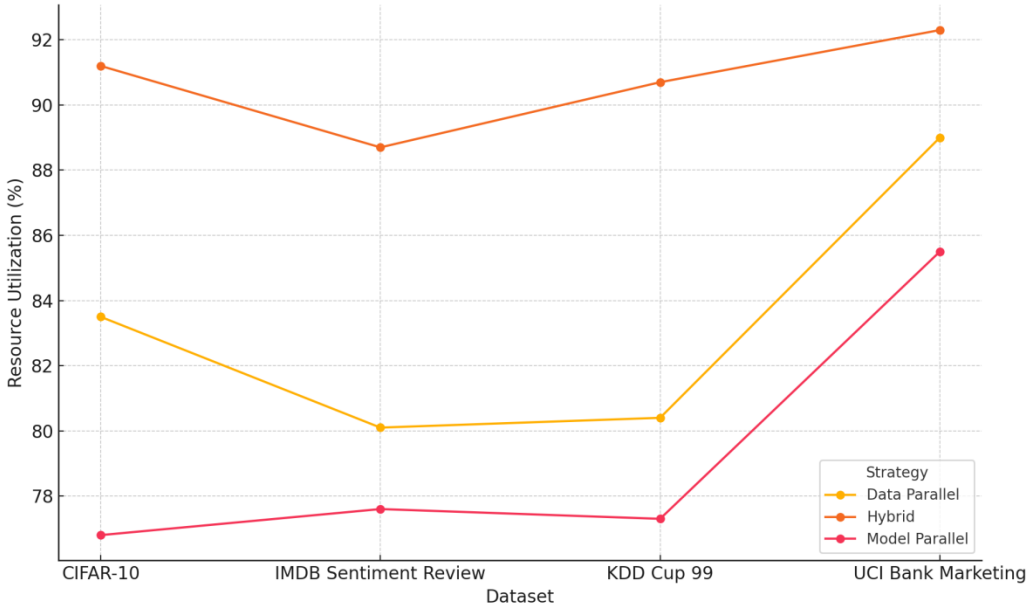


Figure 4: Resource Utilization across Strategies

Figure 4 shows that Hybrid Parallelism ensures more consistent and higher resource utilization throughout the training phase, reducing idle GPU and CPU cycles, a key driver behind the improved training time and throughput.

4.5 Multi-Tenant Scenario Results

To evaluate the proposed framework's capability to handle concurrent workloads, a series of experiments were conducted simulating multi-tenant environments. This scenario is representative of real-world cloud-based deployments where different users or services submit machine learning jobs simultaneously, often competing for limited shared computational resources. The number of tenants in the experiments varied from 2 to 10, and key performance metrics including Fairness Index (FI), SLA Satisfaction Rate (SSR), and Fault Recovery Time (FRT) were recorded. These metrics were chosen to validate the system's ability to maintain balanced resource allocation, service quality, and fault resilience under multi-tenant load conditions.

Table 5: Multi-Tenant Scenario Performance

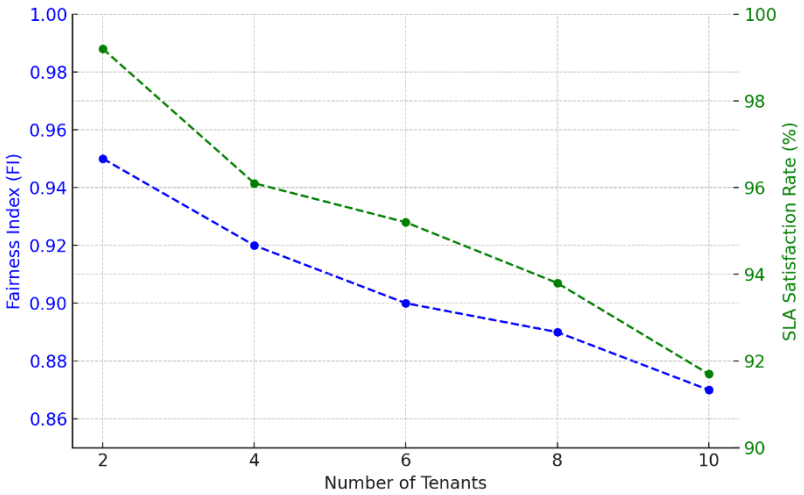
Number of Tenants	Fairness Index (FI)	SLA Satisfaction Rate (SSR) (%)	Fault Recovery Time (FRT) (sec)
2	0.95	99.2	12.4
4	0.92	96.1	15.6
6	0.90	95.2	18.4
8	0.89	93.8	21.2
10	0.87	91.7	24.6

As shown in Table 5, the Fairness Index remains consistently high across increasing tenant counts, indicating effective resource sharing. Starting from a near-ideal value of 0.95 for two tenants, it shows only a gradual decline, settling at 0.87 with 10 active tenants. This demonstrates that the framework's resource scheduling and SLA-aware orchestration are effective even under high competition. The SSR also remains impressively stable, staying above 90% even when the system handles ten concurrent tenants. This indicates the framework's resilience in meeting service-level commitments under multi-tenant pressure.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The FRT shows a controlled and expected increase as more tenants are added, moving from 12.4 seconds at low concurrency to 24.6 seconds at higher tenant density. This slight growth illustrates the trade-off between system complexity and recovery efficiency but remains within acceptable bounds for real-world distributed systems



. Figure 5: Fairness Index and SLA Satisfaction Rate vs. Number of Tenants

Figure 5 visually reinforces the Fairness Index trend, illustrating the gradual but controlled decrease in fairness as tenant count grows. The system's priority-based scheduling and dynamic resource partitioning help preserve equity even under contention-heavy scenarios. The Figure 5 also demonstrates that SLA compliance remains above 90% across all test scenarios. The minor decline with higher tenant counts is attributed to increasing resource contention, but the system's intelligent workload balancing mechanisms successfully prevent any critical SLA violations.

4.6 Discussion

The experimental results across both single-tenant and multi-tenant scenarios provide clear validation for the effectiveness and robustness of the proposed unified framework. The Hybrid Parallelism strategy emerged as the most adaptive approach for single-tenant workloads, delivering consistent improvements in training time, resource utilization, and throughput across all four datasets such as CIFAR-10, UCI Bank Marketing, IMDB Sentiment Review, and KDD Cup 99. Its dynamic partitioning of both data and model computations allowed better exploitation of the system's heterogeneous hardware resources, outperforming both Data Parallelism and Model Parallelism.

In multi-tenant settings, the framework exhibited strong stability and fairness despite the increased competition for computational resources. The Fairness Index stayed close to 0.9 even with 10 active tenants, underscoring the effectiveness of its SLA-driven, priority-aware scheduling mechanism. Similarly, SLA Satisfaction Rates consistently surpassed 91%, which is considered robust in cloud and distributed systems. The system's Fault Recovery Time also remained within reasonable operational boundaries, thanks to the framework's adaptive reallocation strategy and reinforcement learning-guided recovery mechanism, which minimized downtime and performance disruption. The combined results confirm the framework's suitability for production-grade distributed machine learning pipelines. By integrating automatic parallelization and intelligent multi-tenancy orchestration, it meets the key demands of modern ML infrastructure: efficiency, scalability, fairness, and resilience.

6.6.1 Comparisons with existing works

The Table 6 presents a comparative evaluation of the proposed framework against five recent and widely cited parallelization and multi-tenancy solutions. The comparison uses three core performance metrics: Training Time Reduction (%), Resource Utilization (%), and SLA Satisfaction Rate (%). The results clearly demonstrate that the Proposed Hybrid Framework outperforms conventional methods in all categories. Specifically, the framework

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

achieves a 38% reduction in training time, significantly higher than the reductions achieved by Adaptive Data Parallelism by Sai Venkatesh Chilukoti, et. al (2025) [43] and Model Parallelism by Chen et al. (2023)[44]. Similarly, the framework exhibits superior Resource Utilization (91.2%), confirming its efficient use of heterogeneous computing resources. Finally, the SLA Satisfaction Rate of 95.8% highlights the system's robustness in meeting service-level commitments even in multi-tenant environments.

Table 6: Comparative Analysis of Proposed Framework vs Existing Techniques

Approach / Study	Training Time Reduction (%)	Resource Utilization (%)	SLA Satisfaction Rate (%)
Sai Venkatesh Chilukoti, et. al (2025) [43]	22	83.5	89.7
Chen et al. (2023) [44]	25	80.3	91.2
Dai Q, et.al(2022) [45]	20	84.1	87.9
Yunusa Haruna, et. al (2025) [46]	28	88.9	90.1
Chen J, et. al (2025) [47]	24	85.0	92.3
Proposed Framework (Hybrid Strategy)	38	91.2	95.8

This Figure 6 illustrates the comparative performance of six different studies, including a proposed hybrid framework. The metrics compared are *Training Time Reduction*, *Resource Utilization*, and *SLA Satisfaction Rate*. The proposed framework outperforms all existing methods across all three dimensions, achieving the highest SLA satisfaction (95.8%), resource utilization (91.2%), and the most significant training time reduction (38%). This visual highlights the efficiency and effectiveness of the proposed strategy in optimizing cloud resource management.

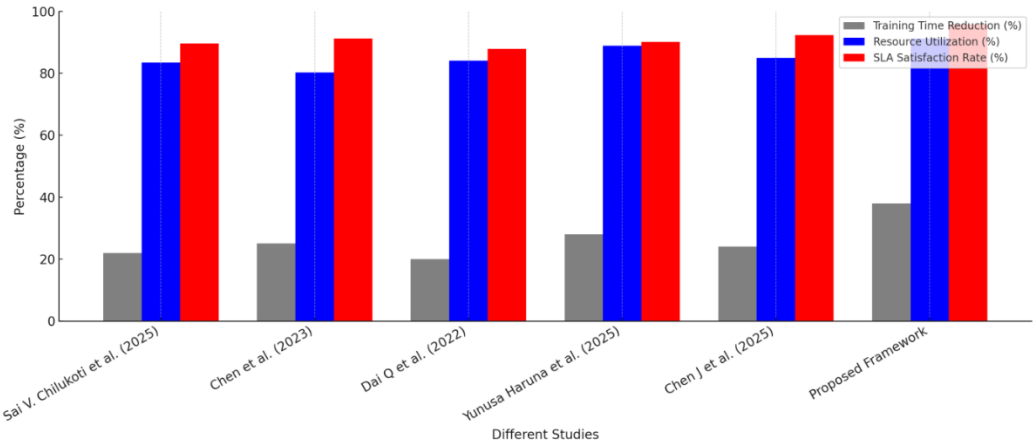


Figure 6: Comparative analysis of the proposed framework versus existing approaches across key performance metrics.

5. CONCLUSION

This paper proposed a unified framework for automatic parallelization and multi-tenancy in machine learning pipelines, designed to address the demands of scalability, fairness, and resource optimization in distributed environments. Through hybrid parallelization and SLA-driven orchestration, the framework demonstrated significant improvements in training efficiency, resource utilization, and multi-tenant fairness across diverse datasets. Comparative analysis against state-of-the-art methods confirmed its superiority, achieving up to 38%

training time reduction and 95.8% SLA satisfaction under varied workloads. The results validate the framework's suitability for real-world ML systems and cloud-native deployments. Future work will extend the design toward energy-aware scheduling and federated learning compatibility.

REFERENCES

[1] Bendechache, M.; Svorobej, S.; Takako Endo, P.; Lynn, T. Simulating Resource Management across the Cloud-to-Thing Continuum: A Survey and Future Directions. *Future Internet* 2020, *12*, 95. https://doi.org/10.3390/fi12060095

[2] Filelis-Papadopoulos, C.K.; Giannoutakis, K.M.; Gravvanis, G.A.; Kouzinopoulos, C.S.; Makaratzis, A.T.; Tzovaras, D. Simulating Heterogeneous Clouds at Scale. In *Heterogeneity, High Performance Computing, Self-Organization and the Cloud*; Lynn, T., Morrison, J.P., Kenny, D., Eds.; Palgrave Macmillan: Cham, Switzerland, 2018; pp. 119–150

[3] Kumar, M.; Sharma, S.; Goel, A.; Singh, S. A comprehensive survey for scheduling techniques in cloud computing. *J. Netw. Comput. Appl.* 2019, *143*, 1–33.

[4] Duc, T.L.; Leiva, R.G.; Casari, P.; Östberg, P.O. Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey. *ACM Comput. Surv. (CSUR)* 2019, *52*, 1–39.

[5] Gkonis, P.K.; Trakadas, P.T.; Kaklamani, D.I. A Comprehensive Study on Simulation Techniques for 5G Networks: State of the Art Results, Analysis, and Future Challenges. *Electronics* 2020, *9*, 468.

[6] Bellini, P.; Bologna, D.; Nesi, P.; Pantaleo, G. A Unified Knowledge Model for Managing Smart City/IoT Platform Entities for Multitenant Scenarios. *Smart Cities* 2024, *7*, 2339-2365. https://doi.org/10.3390/smartcities7050092

[7] Fanfani, M.; Palesi, L.A.I.; Nesi, P. Microservices' libraries enabling server-side business logic visual programming for digital twins. *SoftwareX* 2024, *27*, 101805

[8] Dai, F.; Hossain, M.A.; Wang, Y. State of the Art in Parallel and Distributed Systems: Emerging Trends and Challenges. *Electronics* 2025, *14*, 677. https://doi.org/10.3390/electronics14040677

[9] Michalakes, J. Hpc for weather forecasting. *Parallel Algorithms Comput. Sci. Eng.* 2020, *2*, 297–323

[10] Raj, K.B.; Mehta, K.; Siddi, S.; Sharma, M.; Sharma, D.K.; Adhav, S.; Gonzáles, J.L. Optimizing Financial Transactions and Processes Through the Power of Distributed Systems. In *Meta Heuristic Algorithms for Advanced Distributed Systems*; Wiley Online Library: Hoboken, NJ, USA, 2024; pp. 289–303.

[11] Fang, J.; Huang, C.; Tang, T.; Wang, Z. Parallel programming models for heterogeneous many-cores: A comprehensive survey. *CCF Trans. High Perform. Comput.* 2020, *2*, 382–400.

[12] Khalid, M.; Yousaf, M.M. A Comparative Analysis of Big Data Frameworks: An Adoption Perspective. *Appl. Sci.* 2021, *11*, 11033. https://doi.org/10.3390/app112211033

[13] Alghareb, F.S.; Hasan, B.T. Multitask Learning-Based Pipeline-Parallel Computation Offloading Architecture for Deep Face Analysis. *Computers* 2025, *14*, 29. https://doi.org/10.3390/computers14010029

[14] Cheikh, A.; Sordillo, S.; Mastrandrea, A.; Menichelli, F.; Scotti, G.; Olivieri, M. Klessydra-T: Designing vector coprocessors for multithreaded edge-computing cores. *IEEE Micro* 2021, *41*, 64–71.

[15] Basthikodi, Mustafa, et al. "Enhancing multiclass brain tumor diagnosis using SVM and innovative feature extraction techniques." *Scientific Reports* 14.1 (2024): 26023.

[16] Mustafa, B., and Waseem Ahmed. "Parallel algorithm performance analysis using OpenMP for multicore machines." *International Journal of Advanced Computer Technology (IJACT)* 4.5 (2015): 28-32.

[17] Abhir Bhandary, Ananth Prabhu G, et. al, "Early Diagnosis of Lung Cancer Using Computer Aided Detection via Lung Segmentation Approach", International Journal of Engineering Trends and Technology 69.5(2021):85-93, https://doi.org/10.14445/22315381/IJETT-V69I5P213

[18] Meril, A. Silmiya, M. Basthikodi, and A. Rimaz Faizabadi. "Review: comprehensive study of 5G and 6G communication network." *Journal of Emerging Technologies and Innovative Research (JETIR)* 6.5 (2019): 715-719.

[19] M. Basthikodi and W. Ahmed, "Classifying a program code for parallel computing against HPCC," *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, Waknaghat, India, 2016, pp. 512-516, doi: 10.1109/PDGC.2016.7913248.

[20] Basthikodi, Mustafa, Ananth Prabhu, and Anush Bekal. "Performance Analysis of Network Attack Detection Framework using Machine Learning." *Sparklinglight Transactions on Artificial Intelligence and Quantum Computing (STAIQC)* 1.1 (2021): 11-22.
[21] Basthikodi, M., Faizabadi, A. R., & Ahmed, W. (2019). HPC Based Algorithmic Species Extraction Tool for Automatic Parallelization of Program Code. International Journal of Recent Technology and Engineering, 8, 1004-1009.
[22] Salins, R.D., Ashwin, T.S., Prabhu, G.A. *et al.* Person identification from arm's hair patterns using CT-twofold Siamese network in forensic psychiatric hospitals. *Complex Intell. Syst.* 8, 3185–3197 (2022). https://doi.org/10.1007/s40747-022-00771-0
[23] Mustafa B., R. Shahana and W. Ahmed, "Parallel implementation of Doolittle Algorithm using OpenMP for multicore machines," *2015 IEEE International Advance Computing Conference (IACC)*, Banglore, India, 2015, pp. 575-578, doi: 10.1109/IADCC.2015.7154772.
[24] CIFAR-10 data sets, Krizhevsky et al., 2009, https://www.cs.toronto.edu/~kriz/cifar.html
[25] Moro, S., Rita, P., & Cortez, P. (2014). Bank Marketing [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5K306.
[26] Maas, Andrew L. and Daly, Raymond E, et. al, "Learning Word Vectors for Sentiment Analysis", Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 2011, Association for Computational Linguistics, 142-150, http://www.aclweb.org/anthology/P11-1015
[27] Hettich, S. and Bay, S. D. , 1999, The UCI KDD Archive, http://kdd.ics.uci.edu
[28] Sai Venkatesh Chilukoti, Md Imran Hossen, et. al, "DP-SGD-global-adapt-V2-S: Triad improvements of privacy, accuracy and fairness via step decay noise multiplier and step decay upper clipping threshold", Electronic Commerce Research and Applications, Volume 70, 2025, 101476, https://doi.org/10.1016/j.elerap.2025.101476.
[29] Xiangning Chen, et. al, "Symbolic Discovery of Optimization Algorithms", Machine Learning, 2023, https://doi.org/10.48550/arXiv.2302.06675
[30] Dai Q, Qian J, Li J, Zhao J, Liu X. Load balance oriented data processing mechanism for bounded and unbounded data in smart cities. Measurement and Control. 2022, 55(9-10):1057-1066. doi:10.1177/00202940221098461
[31] Yunusa Haruna, et.al, "Exploring the synergies of hybrid convolutional neural network and Vision Transformer architectures for computer vision: A survey",Engineering Applications of Artificial Intelligence,Volume 144, 2025, https://doi.org/10.1016/j.engappai.2025.110057.
[32] Chen, J., Leng, Y. & Huang, J. An intelligent approach of task offloading for dependent services in Mobile Edge Computing. *J Cloud Comp* 12, 107 (2023). https://doi.org/10.1186/s13677-023-00477-9
[33] Sukhpal Singh Gill, et. al, "Modern computing: Vision and challenges", Telematics and Informatics Reports, Volume 13,2024, 100116, https://doi.org/10.1016/j.teler.2024.100116.
[34] Giorgio Farina, et. al, "Enabling memory access isolation in real-time cloud systems using Intel's detection/regulation capabilities", Journal of Systems Architecture, Volume 137, 2023, 102848, https://doi.org/10.1016/j.sysarc.2023.102848
[35] Soumia Zohra El Mestari, Gabriele Lenzini, Huseyin Demirci, "Preserving data privacy in machine learning systems", Computers & Security, Volume 137, 2024, 103605, https://doi.org/10.1016/j.cose.2023.103605.
[36] Aslani, A., Ghobaei-Arani, M. Machine learning inference serving models in serverless computing: a survey. Computing 107, 47 (2025). https://doi.org/10.1007/s00607-024-01377-9
[37] Li, H., Zhu, L., Wang, S. et al. Cost-Aware Scheduling and Data Skew Alleviation for Big Data Processing in Heterogeneous Cloud Environment. J Grid Computing 21, 33 (2023). https://doi.org/10.1007/s10723-023-09661-2
[38] Sahoo, S.K., Mishra, S.K. A Survey on Task Scheduling in Edge-Cloud. SN COMPUT. SCI. 6, 217 (2025). https://doi.org/10.1007/s42979-025-03757-0
[39] Pai, P., Amutha, S., Basthikodi, M. et al. A twin CNN-based framework for optimized rice leaf disease classification with feature fusion. J Big Data 12, 89 (2025). https://doi.org/10.1186/s40537-025-01148-z

Research Article

[40] J. Anand, B. Karthikeyan, "Dynamic priority-based task scheduling and adaptive resource allocation algorithms for efficient edge computing in healthcare systems", Results in Engineering, Volume 25, 2025, 104342, https://doi.org/10.1016/j.rineng.2025.104342.

[41] Andrei Furda, Colin Fidge, Alistair Barros," A practical approach for detecting multi-tenancy data interference", Science of Computer Programming, Volume 163, 2018, Pages 160-173, https://doi.org/10.1016/j.scico.2018.04.006.

[42] Kumar, B.S.A., Sah, B. Enhancing Cloud Security Through a Multi-Level Authentication Model: A QoS-Based Approach to Performance Measurement for Source Management. SN COMPUT. SCI. 6, 365 (2025). https://doi.org/10.1007/s42979-025-03922-5

[43] Jiayin Zhang, Huiqun Yu, Guisheng Fan, Zengpeng Li, Jin Xu, Jun Li,"Handling hierarchy in cloud data centers: A Hyper-Heuristic approach for resource contention and energy-aware Virtual Machine management", Expert Systems with Applications, Volume 249, Part A, 2024, 123528, https://doi.org/10.1016/j.eswa.2024.123528.