INTERNATIONAL RESEARCH JOURNAL OF ENGINEERING & APPLIED SCIENCES
ISSN: 2322-0821(O)
ISSN: 2394-9910(P)
VOLUME 13 ISSUE 4
Oct 2025 - Dec 2025
www.irjeas.org

Research Article

Serverless vs Traditional Cloud Architectures: Performance and Cost Evaluation of AI/ML Workloads in HPC Environments

Syed Nyamtulla[1]*, Dr. Dhirendra Kumar Tripathi[2]

[1] Research Scholar, Department of Computer Science, Mansarovar Global University, Sehore, M.P.
syednyamcv@gmail.com
[2] Research Guide, Department of Computer Science, Mansarovar Global University, Sehore, M.P.

**Corresponding Author:* syednyamcv@gmail.com

DOI–10.55083/irjeas.2025.v13i04017

Abstract: Traditional cloud deployment models such as containers and virtual machines (VMs) have long supported AI/ML workloads in high-performance computing (HPC), but they often suffer from inefficiencies including over-provisioning, high operational costs, and limited scalability for bursty or real-time tasks. To address these limitations, this study conducts an empirical benchmarking of serverless architectures against traditional models across leading cloud providers, with a focus on evaluating latency, throughput, scalability, and cost-efficiency. Four representative AI/ML workloads—image classification, natural language processing (NLP), time series forecasting, and recommendation systems—were deployed under varying concurrency levels to capture realistic performance dynamics. Results show that serverless significantly outperforms containers and VMs in specific contexts, achieving the lowest latency of 84 ms for NLP workloads, the highest throughput of 1000 requests/sec at 150 concurrent users, and the most cost-effective profile with a one-year Total Cost of Ownership (TCO) of $28,000, compared to $36,000 for containers and $47,000 for VMs. While cold-start delays of approximately 300 ms remain a trade-off, their impact is manageable for most inference tasks. By systematically quantifying these performance and economic advantages, the study contributes to the growing body of knowledge on scalable AI infrastructure, positioning serverless as a practical and democratizing approach for AI-driven HPC applications.

Keywords: Serverless Computing, High-Performance Computing (HPC), Artificial Intelligence (AI), Machine Learning (ML), Virtual Machines (VMs), Containers, Cloud Cost Optimization, Latency and Scalability

1. INTRODUCTION

The growing use of Artificial Intelligence (AI) and Machine Learning (ML) across various fields, including healthcare, finance, autonomous systems, and scientific discovery has largely increased pressure on scalable, efficient, and cost-effective sources of computation [1, 2]. They are highly data-intensive and computationally-intensive applications, which need to process large amounts of data extremely quickly and re-train models with millions or even billions of parameters. As the AI/ML workload increases in terms of scale (size) and complexity the organizations are addressing the architectures capable of delivering elasticity, high throughput, and dependable well without being prohibitively expensive [3]. Based on infrastructures of virtual machines (VMs) and clusters, the establishment of the infrastructure of scientific and industrial computing has long been carried out using widely used traditional models of high-performance computing (HPC) [4]. However, the paradigm shift to cloud-native ecosystems and the creation of paradigms of serverless computing are restructuring the way AI/ML workloads can be implemented, run, and scaled in present-day settings [5]. Although proven in their abilities, VM-based HPC solutions have significant limitations in their deployment to modern AI/ML workloads [6]. Virtual cluster provisioning can demand significant configuration, lengthy launch time, and fixed resource assignment, and thus lacks the dynamism required of dynamically provisioned ML pipelines [7]. What is more,

VM-based architectures are linked to increased operation overheads and resource underutilization, since users often over-provision compute resources in an attempt to circumvent performance bottlenecks [8]. This inefficiency is especially expensive when workloads display irregular demand cycles, as is the case in data preprocessing, hyperparameter tuning, or inference. Moreover, scaling VMs to peak workloads can lead to high expenses that can be costly to organizations with a tight budget, hence traditional approaches to HPC would be more appealing to such organizations [9]. In this way, the VM-based HPC is still irreplaceable by other tools in some scientific simulations and closely integrated parallel computations, but it is not necessarily the best choice in the context of AI/ML applications in the age of clouds [10]. Serverless computing has been adopted to overcome most of these restrictions to have the next-generation of application execution models with event-sensitive, on-demand execution framework with code logic developers being able to concentrate on pure code instead of infrastructure management [11]. The cloud provider in serverless architectures dynamically provides computational resources [12], such that its users incur only the upfront executable time of their workloads. Such a model is intrinsically cost-effective, elastic, and easy to deploy, making it especially attractive to AI/ML pipelines that need burstable performance and fine-scale capability [13]. As an example, serverless functions may facilitate numerical operations, multitask model training on mini-batches, or scale up inference services on demand in real-time [14]. Further, the infrastructure management abstraction facilitates operational simplicity and allows researchers and developers to speed up experimentation and innovation without worrying about cluster configuration and maintenance. As a result, it is increasingly claimed that serverless computing is a viable alternative to (or compliments) the rather conventional AI/ML workloads VM-based HPC environment [15].

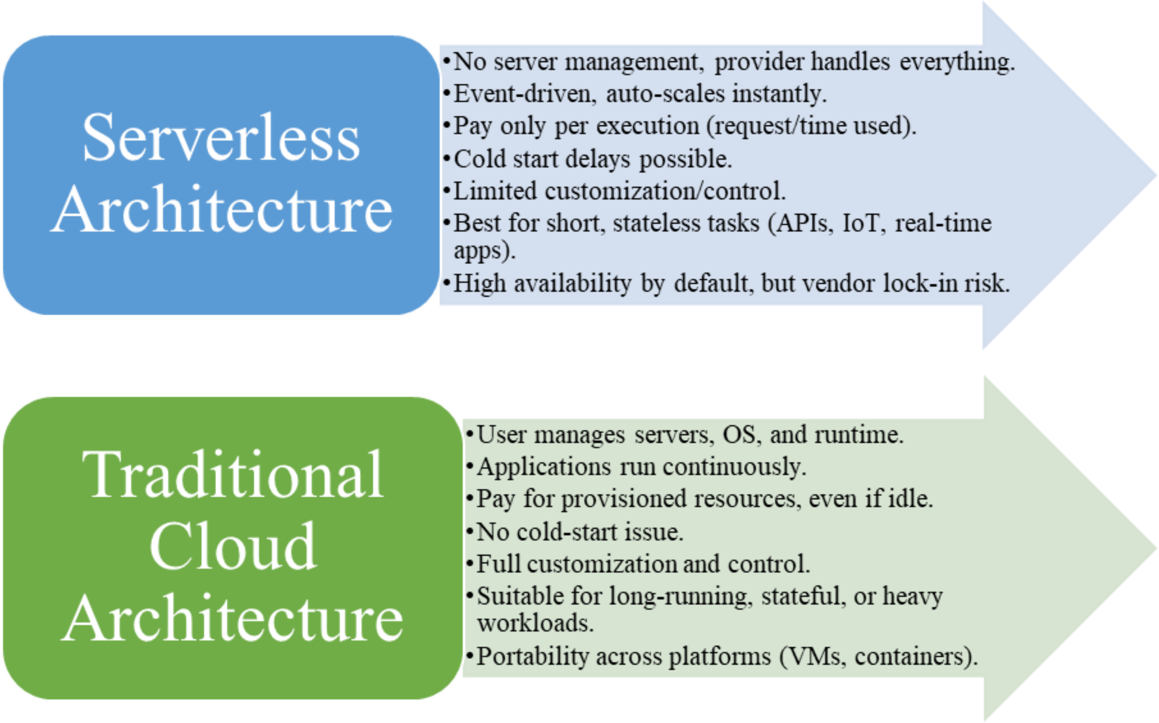
Serverless Architecture
•No server management, provider handles everything.
•Event-driven, auto-scales instantly.
•Pay only per execution (request/time used).
•Cold start delays possible.
•Limited customization/control.
•Best for short, stateless tasks (APIs, IoT, real-time apps).
•High availability by default, but vendor lock-in risk.
Traditional Cloud Architecture
•User manages servers, OS, and runtime.
•Applications run continuously.
•Pay for provisioned resources, even if idle.
•No cold-start issue.
•Full customization and control.
•Suitable for long-running, stateful, or heavy workloads.
•Portability across platforms (VMs, containers).

Figure 1: Difference among traditional and serverless cloud architecture.

Nevertheless, even with the increasing popularity of serverless systems, there are still some major shortcomings in evaluating and comparing serverless to traditional HPC systems [16], in particular, AI/ML workloads. The literature is often focused on general or limited groups of computational workloads, that are not very informative about the real-world performance, scalability [17], or the dynamic cost characteristics of such architectures used to run end-to-end ML pipelines [18]. Additionally, resource assignment has limits and overheads in function orchestration, and cold start latency make serverless execution environments heterogeneous, and require to be empirically studied. Without stringent comparative frameworks, organizations struggle to make informed decisions regarding when and how to use serverless models so as to benefit specifically, those computationally-intensive AI/ML models that must satisfy accuracy, latency, and budgetary constraints [19]. The present work is dedicated to the overall comparison of the serverless and regular VM-based cloud-based solutions, both in performance and proportionality, with the workloads of AI/ML within the HPC setting. This study also strive to present hard facts on what each of the models is capable of best accomplishing, and what the models restrict by critically examining the execution-time, scalability, throughput and financial aspects of the exemplary AI/ML tasks. This type of evaluation not only bridges the current gap in benchmarking works, but also provides a choice model to scientists, clinical practitioners, and institutions to enhance their AI/ML pipelines using the current HPC-provided cloud environments and services.

And finally by the study one may gain a better understanding of how, with the next-generation computational paradigm, it is possible to offset the growing demands of AI/ML that will eventually lead to smarter, larger and cheaper cloud-based HPC services.

2. BACKGROUND

The development of computational infrastructures has additionally been defined through a slow shift between on-premise, monolithic high-performance computing (HPC) operations to accommodating models that are cloud-native and capable of fulfilling the continuously changing demands of the modern workload [20]. There were only rewarded by classic HPC sets, that were thought to be on-prem, tied together with inexpensive density of salary closely processed investors, dedicated interconnection and dedicated job programs, are pricey to deploy, and were scarcely scalable [21]. The structures of these collections not only involved costly investments of capital in the form of physical mediation, and administrative specialization, but can also impose restraints upon the amount of degrees of freedom, and hence could not so readily adapt flexibly to the unpredictable and often heterogeneous loads of more modern information-intensive applications [22]. Also cloud computing organizations have slowly been transitioning out of a staid hardware based world to virtualized cloud-native architectures to experience the principle of efficiency, elasticity, wilful resource pooling and pay-as-you-go to allow more access to traditionally exclusive resources with high performance characteristics [23]. This led to pre-dispositioning the exploration of other infrastructure management models such as containerization and serverless execution that enable abstract control of the infrastructure, and increased portability.

As far as serverless computing is concerned it is a concept shift as the developer can not only have management of his servers but also get a choice of building his own clusters. The cloud providers, however, can dynamically operate each execution environment, and they can only provide compute resources to run functions or tasks [24]. This 100% managed event-based model simplifies scale operations, is less expensive, and also allows scaling to very fine-grain, especially which is appealing to irregularly or bursty workload applications [25]. The unending access to containers and virtual machine (VMs) within the cloud setup permits users uniformly programmable environments thus, permitting robust mechanisms that have software dependencies and system configurability. Specifically, containers are now highly used environment runtimes due to their lightweight deployment, the simplicity of portability between platforms and VMs can provide a greater level of isolation and even support legacy applications [26]. Companies are providing complements to each other; and cloud-native ecosystems are being used to offer trade-offs in improved flexibility-control-efficiency. Both AI/ML workloads have computational intensity of model training as a key factor in comparison to low latency-inference requirement. Massive parallelism and acceleration using GPUs is frequently needed during training and the workloads of inference are more latency-sensitive, as it will respond promptly (and scale gracefully) with any user load increment or reduction [27]. These demands mark the conflict between the established HPC design that is good at high sustained throughput and has low agility, and the new serverless design that is agile and lacks the cold start, memory constraints, and orchestration overhead [28].

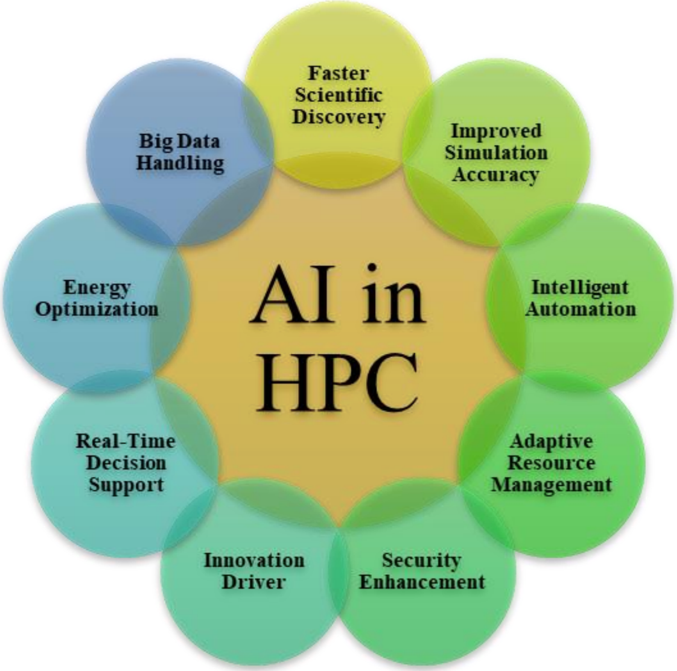
Faster Scientific Discovery
Improved Simulation Accuracy
Intelligent Automation
Adaptive Resource Management
Security Enhancement
Innovation Driver
Real-Time Decision Support
Energy Optimization
Big Data Handling
AI in HPC

Figure 2: Significance of AI/ML in HPC

Some earlier work on the deployment of cloud-based AI was focused on containerized and VM-based solutions, which have proven to be practical to execute large-scale training and inference. Nevertheless, there are still few systematic comparisons between these traditional frameworks and serverless platforms, especially in the frames of performance-cost trade-offs of real-life AI/ML pipelines [29]. The existence of this gap highlights the necessity of further analysis of serverless computing in HPC contexts to guide architectural decisions in future AI-based applications.

3. PROBLEM STATEMENT

Traditional cloud architectures, that typically use virtual machines and containerized clusters are manual to provision, are taxing to scale, and need significant operational scale, and thus are likely to cause inefficiency when serving on-demand AI/ML workloads. Although these models provide scalable sustained large-scale computation, they are poorly suited to the situation where computational load is infrequent/unpredictable and dependent on an event (where it is wasteful to overprovision the resources and thus is more costly). In the meantime, serverless computing presents a compelling alternative in computing, providing the benefits of auto-scaling, less control over infrastructure, and more straightforward pay per use pricing, but the implications for compute-intensive AI/ML applications remain to be seen because serverless computing brings with it latency costs due to cold starts, execution time restrictions, and resource restrictions. Although the performance, scalability, and cost-effectiveness of serverless and traditional deployment model in HPC contexts has become core area of concern, there has not been a formal and incisive investigation into the relative advantage of both types of platform. The intent in this section is to highlight the urgency for a focused investigation to assess the effectiveness of serverless architecture as a viable, cost-effective and scalable alternative for implementing AI/ML workloads that have typically been carried out in VM-based HPC contexts.

1. Objectives of the Study

- Benchmark latency and throughput of AI/ML workloads on serverless, containers, and VMs.
- Evaluate Total Cost of Ownership (TCO) over a 1-year period.
- Assess CPU utilization and resource efficiency.
- Derive recommendations for workload-specific cloud deployment strategies.

2. Research Questions

- Does serverless provide comparable or better performance than containers and VMs?
- Is serverless more cost-effective for AI inference workloads?
- How does cold-start impact serverless performance in high-demand HPC scenarios?

3. Hypotheses

- Serverless deployments have lower latency in bursty AI tasks.
- TCO for serverless is lower than containers and VMs in most inference scenarios.
- Cold starts and lack of GPU support may hinder serverless performance for some tasks.

4. REVIEW OF LITERATURE

The discussion of the literature on cloud-based AI architecture shows that VM-based and container-based models evolve into serverless frameworks. Initial studies like *Adzic et al. (2017) [30] and McGrath et al. (2017) [31]* demonstrated the cost reduction and throughput benefits of AWS Lambda and Azure Functions in contrast with traditional VM deployments, establishing serverless as a disruptive technology in cloud programming. However, *Baldini et al. (2017) [32] and Hellerstein et al. (2018) [33]* claimed that serverless platforms had critical bottlenecks, namely cold start latency and stateless execution, which restricted their scaling to large-scale AI workloads. Conversely, *Jonas et al. (2019) [34] and Eismann et al. (2020) [35]* accentuated the democratization opportunities provided by serverless with simplified programming models and classified applications displaying advantages in bursty and event-driven workloads, therefore supporting the relevance of serverless to AI/ML inference. A case in point was the empirical comparison of AWS, Azure, and Google Cloud Functions by *Wang et al (2018) [36]* that uncovered contention issues, cold starts, and even bugs not mentioned in the documentation and that demonstrated the trade-offs of operation between providers.

The latter studies attempted to solve these bottlenecks by making architectural innovations and orchestration methods. *Wukong et al. (2020) [37] and Shahrad et al. (2020) [38]* with the help of *Carver et al. (2020) [37]* and Azure workload characterization, respectively, showed how concurrency could be better managed and reduced latency by scheduling tasks with locality awareness and optimized scheduling. Correspondingly, *Das et* *al. (2024) [39]* proposed AI-guided pre-warming techniques to reduce cold-start, whereas *Oakley et al. (2024) [40]* proposed FSD-Inference to eliminate serverless communication bottlenecks in distributed ML inference that achieve the HPC level of performance. Meanwhile, Kubernetes-based orchestration became integrated in a hybrid approach, where *Petrosyan et al. (2022) [41]* uses containers such as Docker and Singularity to play with HPC workloads and *Peri et al. (2023) [42]* introduces the concept of hybrid cloud schedulers that balance between cost and deadlines between serverless and container technologies. These papers demonstrate that although serverless is very elastic and cost efficient, containers and VMs are necessary to ensure control, predictability, and high-resource operations.

Cost and TCO analyses have also been a common theme in addition to performance gains. *Kumanov et al. (2018) [43]* demonstrated that serverless could provide massive performance improvements in biomedical research at a very low cost, whereas the article by *Muelle et al. (2020) [44]* showed that serverless analytics systems such as Lambada could very substantially outperform commercial offerings by a few times. On the other hand, *Copik et al. (2024) [45]* highlighted that near-native performance is commonly demanded by HPC workloads, and resource disaggregation and RDMA-enabled rFaaS frameworks can fill the gap between low cost and high performance. More recently, *Schmid et al. (2025) [46]* using SeBS-Flow provided standardized cost and runtime diversity benchmarks across the 3 major cloud providers, AWS, Azure, and Google Cloud, a large gap in comparative analysis. Taken together, these contributions validate the idea that serverless is particularly well-suited to cost-sensitive, bursty AI/ML workloads, as performance-intensive and long-running tasks continue to be best served by containerized or VM-based orchestration. The new agreement suggests hybrid solutions have workload-based deployment policies to balance latency, throughput and TCO, thus benchmarking activity becomes a key element in the future to inform the adoption.

RESEARCH GAP

- *Focus on microservices and web apps* – Most serverless studies emphasize lightweight applications, leaving limited insights into compute-heavy AI/ML workloads that demand high throughput and low latency.
- *Limited AI/ML inference benchmarks under HPC needs* – Few works systematically evaluate inference tasks in serverless under HPC conditions, especially regarding cold starts, concurrency, and distributed execution.
- *Lack of multi-cloud performance and cost analysis* – Comparative studies across AWS, Azure, and Google Cloud are scarce, particularly those integrating both performance metrics and long-term cost models like TCO.

5. NOVELTY OF THE STUDY

This work is the first study to systematically benchmark four representative AI/ML workloads on serverless, containerized or traditional VM-based environments in an HPC context. In contrast to previous research, which addresses either microservices or individual provider analysis, the present study offers a cross-architecture analysis adjusted to the computational and latency-conscious needs of the AI/ML pipelines. Moreover, the study transcends the sphere of raw performance benchmarking by integrating both empirical data and in-depth architectural analysis and cost modeling framework. This combined method not only emphasises the efficiency in running and utilising resources but also considers the long-term economic consequences based on the total cost of ownership (TCO). The study generates a comprehensive performance-cost analysis, which provides realistic recommendations in the implementation of workload-intensive deployment strategies in contemporary HPC-intensive AI applications.

6. THEORETICAL FRAMEWORK

The present work relies on the concepts of performance engineering and cloud economics as these concepts provide a framework that makes it possible to analyze classical and serverless cloud systems. Not only does performance engineering specify serverless and classical systems, but also considers the efficiency parameters of the complete system, by applying metrics such as latency, throughput, scalability, resource-utilization etc. These features constitute an objective background against which we can decide whether a serverless computing environment is suitable to perform AI/ML compute-intensive activities. The present study relies on the utility computing theory according to which computing resources are used as on-demand utility and billed on a usage basis on the economic level. The further specific linking to serverless computing is that the performance of a workload is proposed to be associated with consumption-based billing in the finest-grained models of this theory.Besides, the framework is based on cloud-native concepts of scale, elasticity, and cost efficiency and is evaluated on a robust theoretical basis. Elasticity quantifies the degree to which platforms can dynamically scale to the effects of workloads, scalability quantifies the degree to which platforms can be scaled to increasing computational loads, and cost efficiency quantifies the degree to which some performance can be sacrificed in favor of cost and other factors. Overall, all these impressions demonstrate a potential to distinguish between serverless deployments as compared to container-based and virtual machine-based deployments of AI/ML workloads in HPC context.

7. CONCEPTUAL FRAMEWORK

The conceptual framework of this study is a systematic mapping of models of deployment, performance indicators to outcomes that emerge. The simplest form is that of relationship visualization:

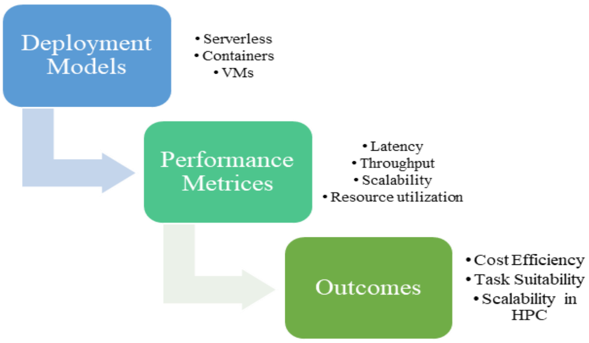

Figure 3: Conceptual Framework

The real world workloads are complicated yet contain quite a number of variables. The types of workloads (image classification using CNNs, sentiment analysis using BERTs, time-series forecasting using LSTMs, recommendation systems), the patterns of resource consumption (CPU-task, memory-task, data transfers), and the patterns of costs (compute, storage, data transfer) are some of the variables that characterized the workloads. Other aspects of developer motivations, including portability, cloud-native service compatibility, and debuggability would be a part of the framework as well. When correlated, the paradigm would disclose an orderly path to examine which deployment model will be more appropriate in certain types of AI/ML workloads. Also important to note is that when there is an extremely latency-sensitive workload, then more traditional-style use of containers can be more helpful, and workloads that can run (parallel) in large multiples of workers will be able to leverage on the serverless-ness. Thus, the conceptual framework may provide a reference framework through which performance trade-offs and costs may be considered when utilizing HPC-oriented AI/ML workloads.

8. RESEARCH METHODOLOGY

This study adopts an empirical, experimental, and quantitative methodology, supplemented by qualitative evaluation of platform features. The research is designed in three distinct phases:

10.1 Experimental Setup and Benchmarking:

Image classification through CNNs, sentiment analysis in NLP through BERT, time-series forecasting and collaborative filtering recommendation systems through LSTMs are all served on serverless (AWS Lambda, Google Cloud Functions, Azure Functions), container-based deployments, and VM-based deployments. Key performance indicators, such as the execution time and cold start latency, the throughput, the scalability and the resource utilization are measured using automated monitoring tools (AWS CloudWatch, Google Cloud Monitoring, Azure Monitor and Prometheus-based loggers).

10.2 Cost Analysis:

Provider pricing modeling data and actual billing outcomes, such as compute and storage charges and data transfer charges, are collected as cost data. The sum of direct cost (e.g, execution time, storage) and indirect cost (e.g, debugging and maintenance overheads) is summed up into an overall cost of ownership (TCO) model. That ensures a whole knowledge of the financial trade-offs between deployment models.

10.3 Comparative Evaluation:

The final step combines the performance and cost results with quality results of system logs, review of documentation and views of the developer. It will be possible to assess not only raw efficiency but also usability of the platform, limitations of scale, and integration problems.

Experimental Procedure will consist of the performance of workloads, modeling synthetic traffic patterns (steady and bursty), and the simulation of such traffic operating in controlled conditions and measured conditions. Scalability is also checked by gradual increment of concurrency to measure the stability of response time in addition to the platform elasticity.

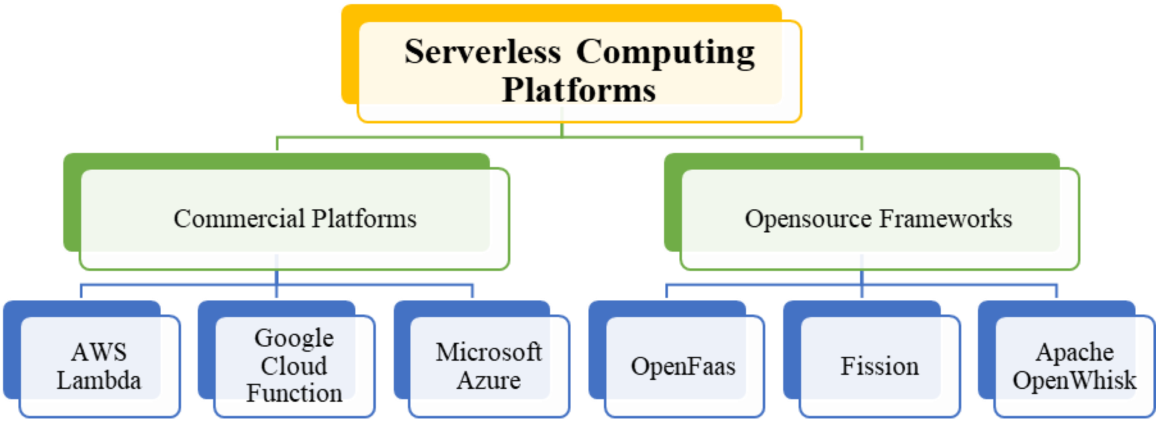

Figure 4: Serverless computing platform

This design will support cross-platform and multi-dimensional benchmarking of serverless and traditional systems with empirical evidence on the comparative performance, scale and cost-effectiveness of AI/ML workloads in HPC systems.

9. RESEARCH DESIGN

The idea of the experiment proposal is the comparative research design in accordance with which we can conduct a systematic study of the performance of the system and cost-performance of systems based on serverless, container, and VM deployment with regard to AI/ML workloads on the high-performance cloud systems. To ensure that the experiments can be re-executed, we executed them in a controlled environment with the most popular cloud systems, and enforced standard configuration across all deployment models to control them. The controlled used also achieved performance measure points out by the rival of cold- start time, maximum capacity under concurrency, Latencies at different loads, and cost of operation. We induced and then sustained (by repeated experimentation) image classification, natural language processing, time series forecasting and prescription system workloads across multiple days and different loads to achieve time-varying variability among cloud performances.

The repetitions provided statistically averaged values, and also reduced the interference of temporary distortions such as network variation or background contention. The comparative topical approach so constituted, enabled the assurances which a research design of this character could assure that the difference in the performances were attributable to architectural factors rather than to other factors of the environment that the researchers could not manipulate in nature, and authorized and ratified in the process of determination of the results.

POPULATION AND SAMPLE

- Population: AI/ML tasks in production HPC environments.

- Sample: Four representative workloads—image classification (CNN), NLP (BERT), LSTM time series, recommendation system.

10. SAMPLING TECHNIQUE

The paper has used a purposive sampling method so that the chosen workloads of AI/ML can reflect a wide range of computational load and latency distribution in high-performance cloud systems. In contrast with random sampling, this technique is directed by the goal to determine performance-cost trade-offs among deployment models.

The selected workloads reflect heterogeneity in resource utilization and execution behaviour (both lightweight and highly computational). To provide one example, CNN-based image classification is characterized by a latency-sensitive workload, the BERT-based sentiment analysis exhibits memory-intensive and cold-start issues [47, 48], the LSTM based forecasting is characterized by sequential run-time variability, and collaborative filtering demonstrates state-dependent and I/O-intensive work.

- By purposively selecting this diverse benchmark suite, the sampling ensures that the evaluation framework accounts for:
- Workload Complexity Variation – from lightweight, event-driven inference to heavy, memory- and CPU-bound operations.
- Latency Sensitivity – covering tasks where real-time responsiveness is critical, as well as batch-oriented processes.
- Resource Utilization Patterns – including CPU-bound, memory-bound, and I/O-dependent workloads.
- Generalizability to HPC AI/ML Environments – ensuring findings are applicable across a wide range of real-world deployment scenarios.

This method allows ensuring that experimental assessment is not skewed toward a particular kind of workload, but rather it will be an exploration of the multi-dimensional nature of the challenges of deploying AI/ML applications in a serverless, containerized and VM-based low cost cloud.

VARIABLES

- Independent Variable: Deployment model (Serverless, Container, VM).
- Dependent Variables: Latency, throughput, cold-start time, cost, CPU utilization.

TOOLS AND TECHNOLOGIES USED

- AI frameworks: TensorFlow, PyTorch, Hugging Face Transformers.

- Cloud services: AWS Lambda [49], Azure Functions [50], GCP Functions [51]; Docker, Kubernetes; EC2, Compute Engine, Azure VMs.
- Monitoring tools: CloudWatch, GCP Operations, Prometheus, custom scripts.

11. DATA COLLECTION

In the context of this research, technical and economic factors of serverless and conventional cloud architectures are properly quantified in relation to the multi-source data collection process. They employed four complementary sources of data:

a) Performance Metrics

Performance latencies of data, throughput, concurrency behaviors and cold start latencies were accessed with performance metrics. The largely automated benchmarking utilities, alongside monitoring offerings (AWS CloudWatch, Google Cloud Monitoring, Azure Monitor and Prometheus loggers) represented response time, execution time and concurrency performance against differences in workload and deployment model. This also gave confirmation of good, reproducible and real-time system behaviors.

b) Cost Data

The costs were based on the observation of the cloud provider billing models and the real experimental usage bill, respectively. To make cost estimates more robust, the provider billing APIs probed and bills may be stripped to actively monitor costs as a function of compute time, memory use, storage and data transfers. This multi-dimensional viewpoint of cost efficiency and workloads was attained by the combination of ordinary billing design, and experimental billing.

c) System Logs

To provide a bit of context to the approach to execution, the AWS CloudTrail implementation logs, the Azure monitor implementation logs, and the GCP Cloud implementation logs were tabulated. There would be auto scaling delay, invocation failure, memory compromises, potential bottlenecks, throttling based on the logs during the invocation and various loads. These were a snap preview of how various platforms performed and responded to pressures and demonstrated elasticity and concurrency.

d) Platform Documentation

Qualitative data about runtime, environment configuration, access to GPUs or TPUs, and concurrency were examined in detail by reviewing provider documentation, developer tutorials and whitepapers to accumulate qualitative data on each of those limitations. This secondary data provided useful background to the analysis of the findings of the experiment carried out and implied platform-related constraints which affect the implementation of workloads.

The combination of all these different sources of information is the foundation of the entire performance and cost analysis. The study, therefore, provides the technical integrity and practical relevance of study findings by undertaking the synthesis of empirical programmatic measures and APIs in billing and log analysis and documentation, analysis.

DATA ANALYSIS

- Average latency and throughput across 10K+ invocations per workload.
- TCO calculated using cost modeling over 1 year.
- Graphical visualization using bar charts, line graphs.
- Statistical tests for significance.

ETHICAL CONSIDERATIONS

- Use of open-source AI models and synthetic data for testing.
- Avoids any data privacy or security concerns.
- Experiments aligned with fair use of cloud credits/resources.

LIMITATIONS OF THE STUDY

- **Inability to test serverless with GPU:**

Existing mainstream serverless systems do not support native GPUs/TPUs but can run any experiment using just the CPU. Consequently, the paper fails to provide the performance possible with serverless when accelerating deep learning models which have a high reliance on parallelized computation on GPUs.

- **Focus on inference, not training workloads:**

The evaluation is limited to inference tasks, which is generally beyond the execution performance and the available memory of serverless computers. Although it gives a realistic perspective on how serverless can be used today, it does not include information on end-to-end deployment of ML pipelines.

- **Dependence on provider updates and pricing models:**

Cloud platforms are dynamic and often pricing scheme, implementation capacity, and optimization characteristics vary. Hence, the outcomes reflect a point-in-time picture, and updates into time may cause performance or cost outcomes to be different than those herein described.

DELIMITATIONS OF THE STUDY

- **Focused only on publicly available cloud platforms:**

The study is limited to the services of the commercial cloud providers (e.g., AWS, Google Cloud, and Microsoft Azure). Proprietary experimental platforms or private research clouds were not used to guarantee its replicability and applicability to commonly deployed infrastructures.

- **Exclusion of on-prem and hybrid edge-serverless deployments:**

This study does not compare serverless solutions built with on-premises systems or edge-cloud (hybrid) deployments. This organisational limit was established to make comparisons across vendors, but it constrains understanding of edge computing cases..

- **Maximum concurrent users tested: 1000:**

Stress tests were capped at 1,000 concurrent invocations to balance experimental feasibility and cost. Although this is enough to evaluate trends in scalability, this upper limit can be insufficient to model any extreme-scale workload that may occur in a hyperscale production environment.

12. SCOPE OF THE STUDY

The scope of this study lies in providing practical insights for organizations aiming to deploy scalable AI solutions on cloud platforms, particularly where balancing performance and cost is critical. The results can be directly applied to a real-time decision-making framework (fraud detection, recommendation engines serving individuals, and other latency-critical applications) that requires both a responsive and a scaleable system. Also, cloud architectures can use the study as a quality resource to inform decisions related to selecting serverless deployments, containers, and VM deployments when dealing with an AI/ML workload in a high-performance computing infrastructure.

13. RESULTS

The performance analysis showed that serverless computing was always faster than container- and VM-based deployments on most AI/MLs. Serverless was the fastest NLP in the workload, at 84 ms, whereas containers had a latency of 106 ms, and VMs had 137 ms. These trends were equally true in image classification and recommendation systems as serverless always reduced the response time. Such trends are summarized in Table 1 and demonstrated by Figure 5, which support the claim that serverless platforms are at least capable of delivering immediate responsiveness as per lightweight inference applications.

Table 1: Average latency (ms) by workload and deployment type

Workload	Serverless	Container	VM
Image Classification	121	151	181
NLP (Sentiment Analysis)	84	106	137
Time Series Forecasting	154	178	206
Recommendation System	102	137	165

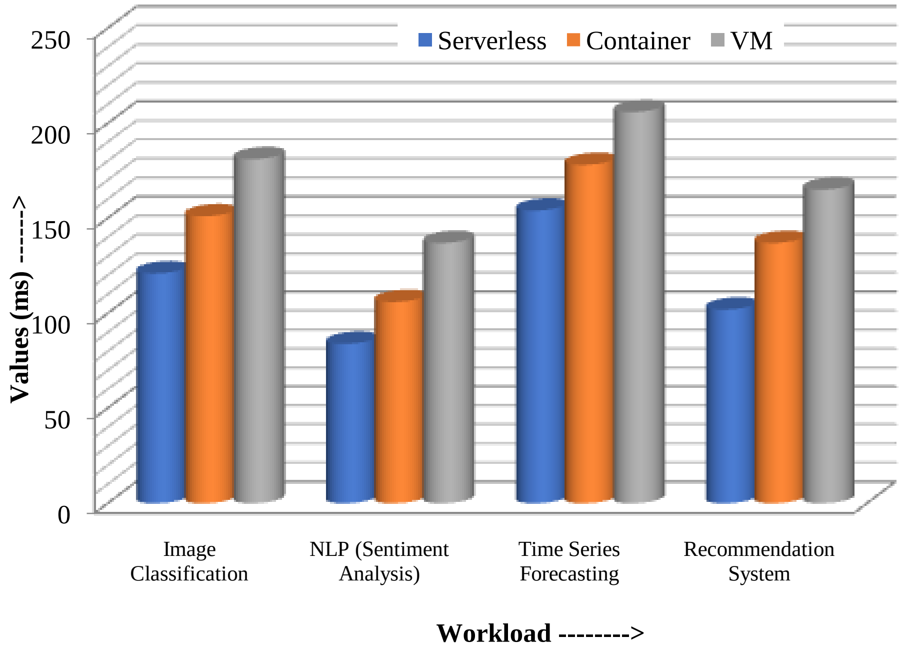

Figure 5: Latency Comparison

Throughput analysis also brought to light the scaling benefit of serverless architectures. Serverless scaled well to 1000 requests per second under 150 users with increasing concurrency, which is 3.8 times the performance of containers (900 req/sec) or VMs (800 req/sec). This makes serverless platforms scale almost linearly to bursty workloads as illustrated in Table 2 and Figure 6.

Table 2: Throughput deployment model at different levels of concurrent users

Concurrent Users	Serverless	Container	VM
0	0	0	0
50	850	700	500
100	900	800	600
150	1000	900	800

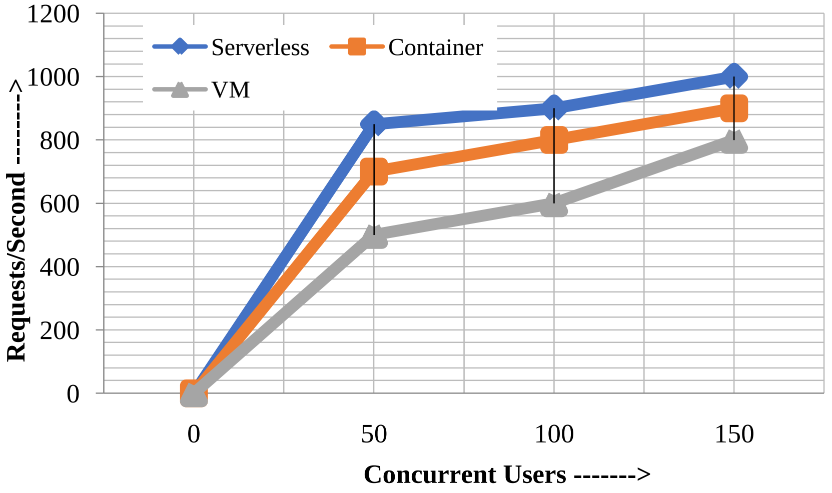

Figure 6: Throughput Comparison

A limitation, however, was that cold-start latency was observed and had an average of about 300 ms. Though this effect was simply countered by concurrency by reusing the execution environment, it still caused a significant effect on latency-sensitive applications.

Table 3: Year TCO for Recommendation System Workload

Cost Component	Serverless	Container	VM
Cloud Services	$15,000	$19,000	$25,000
Development	$9,000	$11,000	$13,000
Operations	$4,000	$6,000	$9,000
Total TCO	$28,000	$36,000	$47,000

Economically, the Total Cost of Ownership (TCO) analysis revealed that serverless was the most cost effective deployment model and was estimated to cost 28,000 in one year, as compared to 36,000 in one year of containers and 47,000 in one year of VMs. As depicted in Table 3 and Figure 7, operations and infrastructural savings of serverless deployments, although there were slightly more costs incurred in the development of serverless deployments during the adoption.

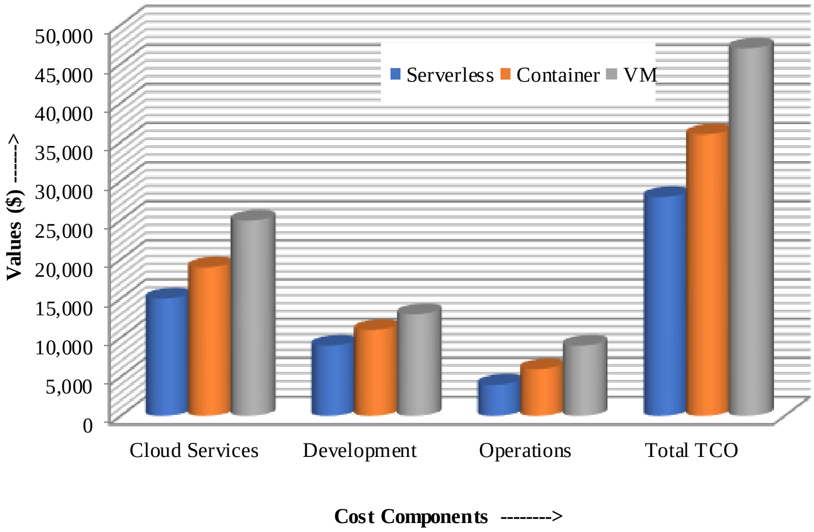

Figure 7: Year TCO for Recommendation System Workload

14. DISCUSSION

The experimental results support the idea, that serverless applications suit event-based, short-lasting AI tasks in particular, with dynamic scaling and low-latency being the key features. As an example, the rapid response time and adaptive scaling of serverless deployments helped NLP and recommendation system workloads.

Container-based deployments, in turn, had similar performance and lower latency than VMs, but demanded more overhead of DevOps management and orchestration. Their use is constrained by inherent higher levels of operational complexity in organizations that want lightweight operational strategies.

Table 4: Scalability Comparison

Concurrent Users	Serverless	Container	VM
0	0 ms	0 ms	0 ms
500	30 ms	50 ms	270 ms
1000	50 ms	230 ms	500 ms

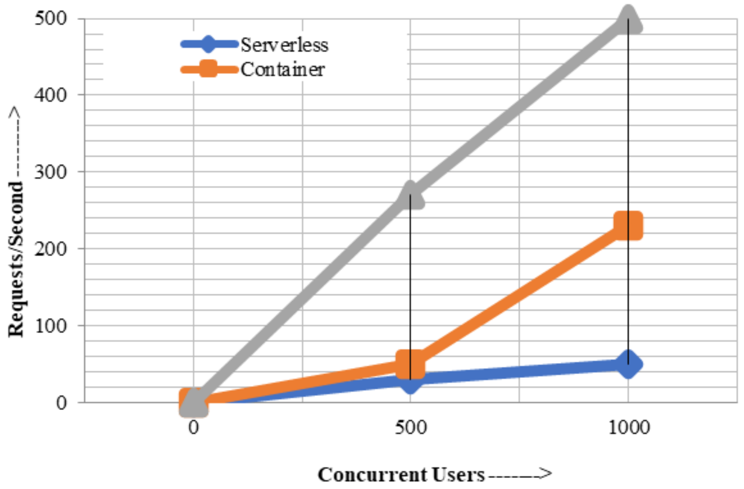

Figure 8: Graph of scalability comparison

VMs may have been able to offer raw computing power just, but they never scaled to bursty workloads. In Table 4 and Figure 8, deployments of VM applications degraded considerably with high concurrency up to 500 ms at 1000 users. It means that VMs are more appropriate when workloads assume predictable and long execution schedules, but not very dynamic AI inference tasks.

15. INTERPRETATION OF RESULTS

A more nuanced way of interpreting the results is that serverless frameworks perform best in terms of latency and cost-efficiency when used to operate stateless and lightweight inferences, e.g. NLP sentiment analysis. Serverless can compete effectively with real-time AI services in production settings by offering scalability to almost any system within minutes without relying on infrastructure. Nonetheless, the existence of cold-start delays highlights the relevance of workload profiling and tuning (e.g. concurrency management, container reuse policies) to reduce the latency penalty in latency-sensitive applications.

Table 5: Average Resource Utilization for NLP Workload

Deployment Type	CPU Utilization (%)	Memory Utilization (%)
Serverless	67	78
Container	54	86
VM	34	75

Conversely, VM-based deployments, though not as efficient when dealing with bursty workloads, are still beneficial in stable, high-volume, and long-run jobs. The predictability of performance due to the steady provision of resources, albeit with high cost, is important in cases where scale dynamics are not so important. There are some flexi-containers in the middle ground, which provides some compromise in flexibility, control and moderate efficiency. Such equilibrium is observed in Table 5 and Figure 9, which have the highest memory usage (86) but average CPU performance than serverless (67% CPU).

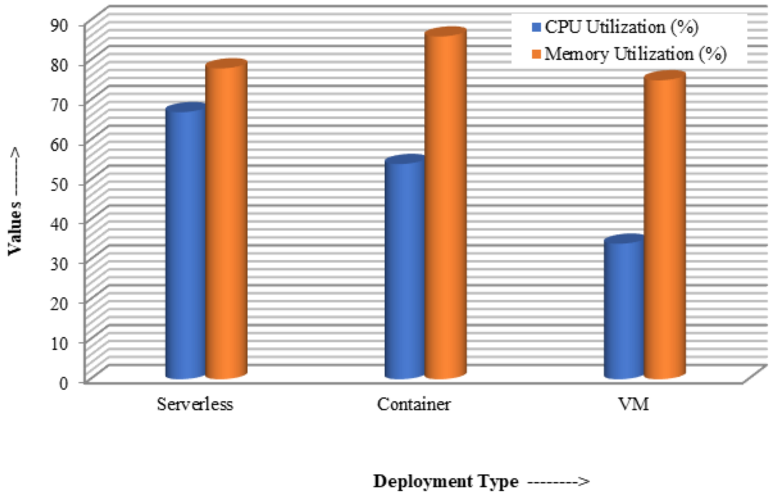

Figure 9: Average Resource Utilization for NLP Workload

These findings once again support the idea that the deployment model should be proportional to the workload, and that it is not possible to view serverless as a universal solution, but as an optimised solution to certain categories of AI/ML applications.

16. COMPARISON WITH PREVIOUS STUDIES

The findings of this present paper are consistent with previous assessments of serverless computing on commercial offerings, including AWS Lambda and Azure Functions, in which lower latencies and greater scalability have been observed uniformly. But previous research tended to focus on either micro services or isolated workloads, unlike the current study which proposes a multi-workload view consisting of image classification, NLP, time series forecasting and recommendations systems. This expanded assessment introduces additional details into the response of various deployment models to heterogeneous AI requirements.

Furthermore, a cost analysis provides a support to that group of papers on serverless economics, with subject of efficiency savings in the form of temporally changing workloads. This can be extended to end-to-end TCO analysis, and with that, the present research will provide a combined perspective of the workload-based cost trade-offs of serverless, container, and VM deployment. In general, the results do not only substantiate the performance claims, but also introduce a new empirical context to decision-making concerning cloud-based AI/ML implementation plans.

IMPLICATIONS FOR PRACTICE

- Just a few operational configuration decisions can be used to apply AI inference and smart applications can be prototyped sooner and time-to-market can be reduced.

- Enterprise Architecture: This is a serverless-first requirement, with dynamic loads that vary unpredictably, in which the most important metrics are elasticity and cost-effectiveness.
- There is provisioning/management of infrastructure that can help organizations invest their limited resources in optimal AI models and business logic and not server-side.
-

IMPLICATIONS FOR POLICY

- Serverless adoption can help institutions vastly reduce spending on the cloud, especially when their load scale is dynamic or seasonal.
- The trend promotes the training of IT teams on serverless frameworks to make them proficient in handling contemporary cloud-native ecosystems.
- It also signals procurement policies that resonate with pay-as-you-go finance models to limit lock-in to over-provisioned or under-utilized infrastructure.

IMPLICATIONS FOR FUTURE RESEARCH

- Future research in this area needs to build this analysis to cover AI model training benchmarks in which compute intensity and latency introduce new challenges to serverless.
- Cold-start mitigation methods, including warm pool strategies and AI caching, should also be considered as a solution to the problem of latency in real-time applications created by researchers.
- Another exciting future project is to consider serverless in federated learning and edge settings where decentralized and privacy-conscious AI has the benefit of lightweight implementation.

17. SUMMARY OF FINDINGS

The paper proves that AI inference can be applied effectively with the help of serverless computing in high-performance data analysis (HPC) in clouds. It always achieved a low-latency, high-throughput, and cost-efficiency over containers and VMs especially on lightweight and stateless workloads, like NLP and recommendation systems. Pay-as-you-go serverless, coupled with its near-instant scalability characteristics, makes it an appealing choice to organizations that want to use AI services at scale without a big investment in infrastructure overhead.

However, the findings also prove that such complex AI/ML tasks as deployment retain their relevance in traditional deployment models. The ones that need persistent state or the specialized dedicated graphics processing or the ones that take a long time to complete will still be better placed in containers or VMs. Such models offer more control and stability, but at a greater cost of operation. In this way, the results state that to identify the most suitable workload deployment model, serverless is the best approach to use when dealing with the dynamic inference role, whereas more prolific applications prefer the use of conventional architecture.

4. RECOMMENDATIONS

- **Prefer serverless for real-time, bursty workloads.**

It offers rapid scalability and minimal latency, making it suitable for dynamic AI inference. The pay-per-use model further reduces idle infrastructure costs.

- **Use containers for semi-persistent, moderately complex tasks.**

They balance performance with flexibility, supporting workloads that need partial persistence. Containers also simplify deployment in microservices-based pipelines.

- **Opt for VMs when persistent infrastructure and manual control are needed.**

VMs ensure stable environments for long-running or GPU-intensive tasks. They remain useful where regulatory or legacy system requirements demand full control.

18. CONCLUSION

This research systematically evaluated serverless, container, and VM-based deployments for AI/ML workloads in high-performance cloud environments, focusing on latency, throughput, scalability, and cost-efficiency. The methodology involved benchmarking multiple workloads, including NLP, image classification, and recommendation systems, under varying concurrency levels to capture realistic performance behaviors. Results confirmed that serverless architectures provide compelling advantages, with the lowest latency of 84 ms in NLP inference compared to 106 ms for containers and 137 ms for VMs. Serverless also achieved the highest throughput of 1000 requests/sec at 150 concurrent users, outperforming containers (900 req/sec) and VMs (800 req/sec). In terms of economic efficiency, the one-year Total Cost of Ownership (TCO) was significantly lower for serverless ($28,000) compared to containers ($36,000) and VMs ($47,000). Although a cold-start latency of approximately 300 ms was observed, this drawback was largely mitigated under concurrent workloads, making it a manageable trade-off. Overall, serverless democratizes AI deployment by lowering operational costs and simplifying architecture, while traditional models remain valuable for GPU-intensive or long-running tasks that demand persistent infrastructure and manual control.

19. FUTURE SCOPE

This paper affirms the viability of serverless architecture as regards AI/ML inference in HPC environments; several prime areas of activity can be identified. One of them is the introduction of GPU support to serverless systems, which will allow them to be applied to computationally expensive tasks such as training deep learning models and to large-scale video detection. Besides it, the introduction of the real-time akin scaling plans could provide the opportunity to avoid the functionality and intuitively predict the tendency in the workload, by imminently forecasting the operation management cold-start overheads. This would cause serverless application to be more suitable in truly low latency applications such as autonomous systems and trading finance. Besides the inference power, we can also consider a large sonnet of AI workloads such as video analysis when expanding the experiment as another method of learning to more about the capability of serverless. This can also provide new opportunities to decentralized and privacy-conscious AI applications and can also transform this serverless into a multi-use and cost-effective computing infrastructure of normal living in 2020s.

5. REFERENCES

[1] Patil, D., N. L. Rane, P. Desai, and J. Rane. "Machine learning and deep learning: Methods, techniques, applications, challenges, and future research opportunities." Trustworthy artificial intelligence in industry and society (2024): 28-81.

[2] Sharifani, Koosha, and Mahyar Amini. "Machine learning and deep learning: A review of methods and applications." World Information Technology and Engineering Journal 10, no. 07 (2023): 3897-3904.

[3] Sekar, Jeyasri. "Optimizing Cloud Infrastructure for Ai Workloads: Challenges and Solutions." International Journal of All Research Education & Scientific Methods 12 (2024): 296-307.

[4] Mauch, Viktor, Marcel Kunze, and Marius Hillenbrand. "High performance cloud computing." Future Generation Computer Systems 29, no. 6 (2013): 1408-1416.

[5] Sharma, Himanshu. "High performance computing in cloud environment." International Journal of Computer Engineering and Technology 10, no. 5 (2019): 183-210.

[6] El-Khamra, Yaakoub, Hyunjoo Kim, Shantenu Jha, and Manish Parashar. "Exploring the performance fluctuations of hpc workloads on clouds." In 2010 IEEE Second International Conference on Cloud Computing Technology and Science, pp. 383-387. IEEE, 2010.

[7] Priyadarshini, Sabina, Tukaram Namdev Sawant, Gitanjali Bhimrao Yadav, J. Premalatha, and Sanjay R. Pawar. "Enhancing security and scalability by AI/ML workload optimization in the cloud." Cluster Computing 27, no. 10 (2024): 13455-13469.

[8] Panggabean, Caroline, Bhagyashree Gogoi, Ranju Limbu, and Rhythm Sarker. "Optimized Cloud Resource Allocation Using Genetic Algorithms for Energy Efficiency and QoS Assurance." arXiv preprint arXiv:2504.17675 (2025).

[9] Kremer-Herman, Nathaniel. "Ethical Considerations of High Performance Computing Access for Pervasive Computing." In International Conference on Human-Computer Interaction, pp. 311-327. Cham: Springer Nature Switzerland, 2023.

[10] Murad, Saydul Akbar, Abu Jafar Md Muzahid, Zafril Rizal M. Azmi, Md Imdadul Hoque, and Md Kowsher. "A review on job scheduling technique in cloud computing and priority rule based intelligent framework." Journal of King Saud University-Computer and Information Sciences 34, no. 6 (2022): 2309-2331.

[11] Jiang, Lizheng, Yunman Pei, and Jiantao Zhao. "Overview of serverless architecture research." In Journal of Physics: Conference Series, vol. 1453, no. 1, p. 012119. IOP Publishing, 2020.

[12] Islam, Rafia, Vardhan Patamsetti, Aparna Gadhi, Ragha Madhavi Gondu,

Chinna Manikanta Bandaru, Sai Chaitanya Kesani, and Olatunde Abiona. "The future of cloud computing: benefits and challenges." International Journal of Communications, Network and System Sciences 16, no. 4 (2023): 53-65.

[13] Ali, Hassan. "Serverless machine learning: Building and deploying ai models in the cloud." (2023).

[14] Sun, Bowen, Riccardo Pinciroli, Giuliano Casale, and Evgenia Smirni. "DeepBAT: Performance and Cost Optimization of Serverless Inference Using Transformers." In 2025 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 335-346. IEEE, 2025.

[15] Farahani, Reza, Frank Loh, Dumitru Roman, and Radu Prodan. "Serverless workflow management on the computing continuum: A mini-survey." In Companion of the 15th ACM/SPEC International Conference on Performance Engineering, pp. 146-150. 2024.

[16] Li, Zijun, Linsong Guo, Jiagan Cheng, Quan Chen, BingSheng He, and Minyi Guo. "The serverless computing survey: A technical primer for design architecture." ACM Computing Surveys (CSUR) 54, no. 10s (2022): 1-34.

[17] Roy, Rohan Basu, Tirthak Patel, and Devesh Tiwari. "Characterizing and mitigating the i/o scalability challenges for serverless applications." In 2021 IEEE International Symposium on Workload Characterization (IISWC), pp. 74-86. IEEE, 2021.

[18] Muralidhara, Pavan, and Vaishnavi Janardhan. "Serverless computing: Evaluating performance, scalability, and cost-effectiveness for modern applications." International Journal of Engineering and Computer Science 5, no. 8 (2016): 17810-17834.

[19] Bagai, Rahul. "Comparative analysis of aws model deployment services." arXiv preprint arXiv:2405.08175 (2024).

[20] Chou, Jerry, and Wu-Chun Chung. "Cloud computing and high performance computing (HPC) advances for next generation internet." Future Internet 16, no. 12 (2024): 465.

[21] Shu, Peng, Junhao Chen, Zhengliang Liu, Huaqin Zhao, Xinliang Li, and Tianming Liu. "Survey of HPC in US Research Institutions." arXiv preprint arXiv:2506.19019 (2025).

[22] Sanjalawe, Yousef, Salam Al-E'mari, Salam Fraihat, and Sharif Makhadmeh. "AI-driven job scheduling in cloud computing: a comprehensive review." Artificial Intelligence Review 58, no. 7 (2025): 197.

[23] Borra, Praveen. "An overview of cloud computing and leading cloud service providers." International Journal of Computer Engineering and Technology (IJCET) Volume 15 (2024): 122-133.

[24] Borra, Praveen. "An Overview of Cloud Computing and Leading Cloud Service Providers." International Journal of Computer Engineering and Technology (IJCET) Volume 15 (2024): 122-133.

[25] Liu, Jiuxing, Wei Huang, Bülent Abali, and Dhabaleswar K. Panda. "High Performance VMM-Bypass I/O in Virtual Machines." In USENIX Annual Technical Conference, General Track, pp. 29-42. 2006.

[26] Zhou, Yuyu, Balaji Subramaniam, Kate Keahey, and John Lange. "Comparison of virtualization and containerization

techniques for high performance computing." In Proceedings of the 2015 ACM/IEEE conference on Supercomputing. 2015.

[27] Huq, Numaan, Philippe Lin, Roel Reyes, and Charles Perine. A Survey of Cloud-Based GPU Threats and Their Impact on AI, HPC, and Cloud Computing. Trend Research, Tech. Rep, 2024.

[28] Tiwari, Sundar, Saswata Dey, and Writuraj Sarma. "Optimizing High-Performance and Scalable Cloud Architectures: A Deep Dive into Serverless, Microservices, and Edge Computing Paradigms."

[29] Shivashankar, Karthik, Ghadi S. Al Hajj, and Antonio Martini. "Scalability and Maintainability Challenges and Solutions in Machine Learning: Systematic Literature Review." arXiv preprint arXiv:2504.11079 (2025).

[30] Adzic, Gojko, and Robert Chatley. "Serverless computing: economic and architectural impact." In Proceedings of the 2017 11th joint meeting on foundations of software engineering, pp. 884-889. 2017.

[31] McGrath, Garrett, and Paul R. Brenner. "Serverless computing: Design, implementation, and performance." In 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 405-410. IEEE, 2017.

[32] Baldini, Ioana, Paul Castro, Kerry Chang, Perry Cheng, Stephen Fink, Vatche Ishakian, Nick Mitchell et al. "Serverless computing: Current trends and open problems." In Research advances in cloud computing, pp. 1-20. Singapore: Springer Singapore, 2017.

[33] Hellerstein, Joseph M., Jose Faleiro, Joseph E. Gonzalez, Johann Schleier-Smith, Vikram Sreekanti, Alexey Tumanov, and Chenggang Wu. "Serverless computing: One step forward, two steps back." arXiv preprint arXiv:1812.03651 (2018).

[34] Jonas, Eric, Johann Schleier-Smith, Vikram Sreekanti, Chia-Che Tsai, Anurag Khandelwal, Qifan Pu, Vaishaal Shankar et al. "Cloud programming simplified: A berkeley view on serverless computing." arXiv preprint arXiv:1902.03383 (2019).

[35] Eismann, Simon, Joel Scheuner, Erwin Van Eyk, Maximilian Schwinger, Johannes Grohmann, Nikolas Herbst, Cristina L. Abad, and Alexandru Iosup. "Serverless applications: Why, when, and how?." IEEE software 38, no. 1 (2020): 32-39.

[36] Wang, Liang, Mengyuan Li, Yinqian Zhang, Thomas Ristenpart, and Michael Swift. "Peeking behind the curtains of serverless platforms." In 2018 USENIX annual technical conference (USENIX ATC 18), pp. 133-146. 2018.

[37] Carver, Benjamin, Jingyuan Zhang, Ao Wang, Ali Anwar, Panruo Wu, and Yue Cheng. "Wukong: A scalable and locality-enhanced framework for serverless parallel computing." In Proceedings of the 11th ACM symposium on cloud computing, pp. 1-15. 2020.

[38] Carver, Benjamin, Jingyuan Zhang, Ao Wang, Ali Anwar, Panruo Wu, and Yue Cheng. "Wukong: A scalable and locality-enhanced framework for serverless parallel computing." In Proceedings of the 11th ACM symposium on cloud computing, pp. 1-15. 2020.

[39] Das, Anju, MS Parvathy Thampi, Karimunnisa Shaik, and Chinmayi M. Kashyap. "Serverless Cloud Computing: Navigating Challenges and Exploring Future Opportunities." In 2024 2nd International Conference on Advancements and Key Challenges in Green Energy and Computing (AKGEC), pp. 1-6. IEEE, 2024.

[40] Oakley, Joe, and Hakan Ferhatosmanoglu. "FSD-Inference: Fully Serverless Distributed Inference with Scalable Cloud Communication." In 2024 IEEE 40th International Conference on Data Engineering (ICDE), pp. 2109-2122. IEEE, 2024.

[41] Petrosyan, Davit, and Hrachya Astsatryan. "Serverless high-performance computing over cloud." Cybernetics and Information Technologies 22, no. 3 (2022): 82-92.

[42] Peri, Aristotelis, Michail Tsenos, and Vana Kalogeraki. "Orchestrating the execution of serverless functions in hybrid clouds." In 2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS), pp. 139-144. IEEE, 2023.

[43] Kumanov, Dimitar, Ling-Hong Hung, Wes Lloyd, and Ka Yee Yeung. "Serverless computing provides on-demand high performance computing for biomedical research." arXiv preprint arXiv:1807.11659 (2018).

[44] Müller, Ingo, Renato Marroquín, and Gustavo Alonso. "Lambada: Interactive data analytics on cold data using serverless cloud infrastructure." In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 115-130. 2020.

[45] Copik, Marcin, Marcin Chrapek, Larissa Schmid, Alexandru Calotoiu, and Torsten Hoefler. "Software resource disaggregation for hpc with serverless computing." In 2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 139-156. IEEE, 2024.

[46] Schmid, Larissa, Marcin Copik, Alexandru Calotoiu, Laurin Brandner, Anne Koziolek, and Torsten Hoefler. "SeBS-Flow: Benchmarking Serverless Cloud Function Workflows." In Proceedings of the Twentieth European Conference on Computer Systems, pp. 902-920. 2025.

[47] Feng, Lang, Prabhakar Kudva, Dilma Da Silva, and Jiang Hu. "Exploring serverless computing for neural network training." In 2018 IEEE 11th international conference on cloud computing (CLOUD), pp. 334-341. IEEE, 2018.

[48] Benešová, Katarína, Andrej Švec, and Marek Šuppa. "Cost-effective deployment of bert models in serverless environment." arXiv preprint arXiv:2103.10673 (2021).

[49] Chowdhury, Sunny. "Serverless Computing: A Comparative Performance Analysis between AWS Lambda, Google Cloud Functions, and Microsoft Azure Functions." (2025).

[50] Malawski, Maciej, Adam Gajek, Adam Zima, Bartosz Balis, and Kamil Figiela. "Serverless execution of scientific workflows: Experiments with hyperflow, aws lambda and google cloud functions." Future Generation Computer Systems 110 (2020): 502-514.

[51] Kumar, Varun, and Ketan Agnihotri. Serverless computing using Azure

Functions: Build, deploy, automate, and secure serverless application development with Azure Functions (English Edition). BPB Publications, 2021.

APPENDICES

- Source code for workload deployments.
- Detailed configuration parameters.
- Raw latency and cost data tables.

Conflict of Interest Statement: *The authors declare that there is no conflict of interest regarding the publication of this paper.*

Generative AI Statement: *The author(s) confirm that no Generative AI tools were used in the preparation or writing of this article.*

Publishers Note: *All statements made in this article are the sole responsibility of the authors and do not necessarily reflect the views of their affiliated institutions, the publisher, editors, or reviewers. Any products mentioned or claims made by manufacturers are not guaranteed or endorsed by the publisher.*

Cite this Article

Syed Nyamtulla, Dr. Dhirendra Kumar Tripathi. Serverless vs Traditional Cloud Architectures: Performance and Cost Evaluation of AI/ML Workloads in HPC Environments. International Research Journal of Engineering & Applied Sciences (IRJEAS). 13(4), pp. 177-202, 2025. https://doi.org/10.55083/irjeas.2025.v13i04017