ISSN–2278-5655

AMIERJ Aarhat Multidisciplinary International Education Research Journal

Volume–XV, Special Issues– I(b) Jan – Feb, 2026

OPEN ACCESS Original Research Article

SERVERLESS AI: ARCHITECTURE AND EXECUTION OF AI MODELS WITHOUT SERVERS

*** Aadesh Patil & ** Samiksha Kadam**

Students, K S D's Model College (Empowered Autonomous), Khambalpada Rd, Thakurli, Dombivli East, Maharashtra

Abstract:

The deployment of artificial intelligence (AI) models traditionally relies on continuously running servers or virtual machines, leading to high operational costs, inefficient resource utilization, and limited scalability under dynamic workloads. Although cloud computing has improved accessibility, challenges such as manual infrastructure management and over- provisioning persist. Serverless AI has emerged as a promising paradigm that enables event-driven and on-demand execution of AI models without explicit server management.

This paper presents a structured analysis of Serverless AI, focusing on its system architecture, execution workflow, and operational characteristics. A comparative discussion highlights the advantages of Serverless AI over traditional deployments in terms of scalability, cost efficiency, fault tolerance, and reduced operational complexity. Key application domains including image processing, conversational systems, healthcare analytics, and fraud detection are examined. Despite its benefits, Serverless AI introduces challenges such as cold-start latency, execution time limits, hardware constraints, and model size overheads. These limitations and future research directions are discussed. The paper concludes that Serverless AI provides a flexible and scalable deployment model for AI inference workloads with unpredictable demand patterns.

Keywords: *Serverless Computing, Artificial Intelligence, Cloud Computing, AI Deployment, Function-as-a-Service*

Introduction:

Artificial Intelligence (AI) has emerged as a foundational technology across diverse application domains, including healthcare analytics, financial services, recommendation engines, and conversational systems. As the adoption of AI- driven services continues to grow, the efficient deployment of inference workloads has become a significant technical challenge. Conventional AI deployment approaches rely on continuously running physical servers or cloud-based virtual machines (VMs), which require manual scaling, infrastructure maintenance, and careful capacity planning. These requirements frequently result in elevated operational costs and inefficient resource utilization, particularly during periods of low demand [1].

Although cloud computing introduced elasticity through virtual machines and container-based platforms, most AI systems remain server-centric, with computational resources remaining active irrespective of workload intensity. This execution model often leads to resource over-provisioning during idle periods and performance degradation during sudden traffic spikes [2]. Serverless computing addresses these limitations by abstracting server management from developers and enabling stateless, event-driven execution models.

When this paradigm is applied to AI inference workloads, it is referred to as Serverless AI. Serverless AI enables on- demand execution, fine-grained scalability, and a pay-per-use cost model, making it particularly suitable for AI applications with highly variable or unpredictable workloads [3]. However,

ISSN–2278-5655
AMIERJ Aarhat Multidisciplinary International Education Research Journal
Volume–XV, Special Issues– I(b) Jan – Feb, 2026
OPEN ACCESS Original Research Article

despite its advantages, serverless AI introduces new challenges, including cold-start latency, execution time constraints, and limited access to specialized hardware. This paper presents a comprehensive analysis of Serverless AI, examining its system architecture, execution workflow, benefits, limitations, and future research directions.

Related Work:

Early research on serverless computing focused on web applications and event-driven backend services, highlighting benefits such as reduced operational overhead and automatic scaling [1]. Parallel research on AI deployment emphasized VM- and container-based inference pipelines that offer control over hardware resources but incur idle costs and management overhead [4].

Recent studies explored the feasibility of running machine learning inference on serverless platforms, identifying cost benefits for sporadic workloads and highlighting cold-start latency as a key challenge [3]. Other works proposed optimization techniques such as function warming and model caching [5]. However, most existing studies focus on platform-specific evaluations and lack a holistic architectural and workflow-level analysis. This paper addresses this gap by analyzing Serverless AI as a general deployment paradigm

Background and Motivation:

The rapid growth of artificial intelligence (AI) applications has created a strong need for deployment models that are both scalable and cost-efficient. Many AI systems face highly variable workloads, from sporadic inference requests to sudden bursts of real-time traffic. Traditional deployments using always-on physical servers or cloud virtual machines (VMs) are inefficient in such dynamic environments, as they require pre-provisioning resources based on peak demand. This often leads to wasted capacity, higher costs, or performance degradation during spikes [1].

Even with cloud-based elasticity through VMs and container orchestration, most AI systems remain server-centric, keeping compute resources active regardless of actual workload. This over-provisioning increases operational complexity, including infrastructure management, auto scaling, monitoring, and fault tolerance, especially in systems deploying multiple models or frequently updating pipelines [2].

Serverless computing mitigates these issues by adopting an event-driven, stateless function model. Functions execute only in response to triggers, while cloud providers manage resource allocation, scaling, and availability. Users are billed based solely on execution time and resources consumed, enabling fine-grained scalability and cost-efficient operation for workloads with unpredictable demand [6].

When applied to AI inference workloads, this approach forms Serverless AI, where models run only on-demand. This eliminates idle compute resources and supports rapid, automatic scaling. Serverless AI is well suited for real-time applications such as image recognition, conversational agents, and event-driven analytics [3]. However, challenges like cold- start latency, model loading overhead, execution limits, and limited access to specialized hardware must be addressed to fully leverage this paradigm.

System Architecture of Serverless AI:

Serverless AI architectures are designed to support artificial intelligence inference workloads without relying on continuously running servers. Instead, they leverage event driven cloud services that dynamically allocate and release computational resources based on incoming requests. This architecture enables elastic scaling, reduced operational overhead, and cost efficient execution, particularly for workloads with highly variable demand [1], [6].

ISSN–2278-5655

AMIERJ Aarhat Multidisciplinary International Education Research Journal

Volume–XV, Special Issues– I(b) Jan – Feb, 2026

OPEN ACCESS Original Research Article

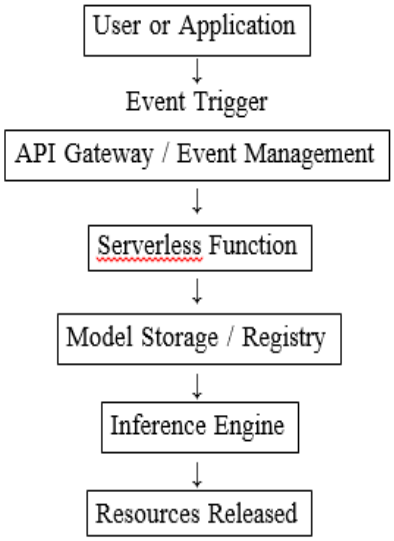

Fig. 1. Serverless AI Execution Flow

A. ***User or Application layer***

The User/Application layer serves as the entry point of Serverless AI, encompassing end users, web/mobile apps, IoT devices, and automated services. It handles requests such as image/video uploads, text classification, and real-time data streams, communicating via standard APIs (e.g., REST/HTTP) without directly interacting with AI models or infrastructure [7].

B. ***API Gateway***

The API Gateway manages incoming requests, performing authentication, rate limiting, request validation, and routing to serverless functions. It abstracts backend complexity, ensuring secure, scalable, and controlled access [2].

C. ***Serverless Function***

Serverless Functions are lightweight, on-demand execution units triggered by events from the API Gateway. They handle input preprocessing, inference logic, and interaction with storage services, automatically scaling to handle concurrent requests without persistent servers [1].

D. ***AI Model Storage***

AI models are stored in cloud storage or model registries, providing version control, high availability, and fault tolerance. Serverless functions fetch models at runtime, enabling easy updates, rollback, and reuse across multiple functions [8].

E. ***Inference Engine***

The Inference Engine executes loaded AI models on input data using CPUs, GPUs, or accelerators. It performs computations, generates predictions or classifications, and is optimized for low latency, short lived workloads suitable for real time applications [3].

F. ***Result Returned***

After inference, results are sent back to the user/application via the API Gateway. Outputs can include structured data (JSON/XML), processed media, or prediction scores. The serverless execution environment terminates immediately afterward, releasing all allocated resources [9].

Execution Workflow of Serverless AI:

Serverless AI follows an event-driven execution lifecycle consisting of event triggering, function invocation, model loading, inference execution, result return, and resource deallocation. Resources are provisioned only for the duration of execution and released immediately afterward.

The total inference latency can be expressed as:

$$T_{total} = T_{trigger} + T_{init} + T_{load} + T_{infer} + T_{response}$$

Here, *Ttrigger:* Represents the delay associated with event generation and delivery. This includes the time taken for incoming requests, such as API calls or data uploads,

Tinit: Denotes the startup time of the serverless runtime environment. This phase involves provisioning compute resources, initializing the execution container, and preparing runtime dependencies,

Tload: Refers to the duration required to retrieve and load the AI model and related artifacts into memory from cloud storage or a model repository.

Tinfer: Indicates the computation time consumed by

ISSN–2278-5655

AMIERJ Aarhat Multidisciplinary International Education Research Journal

Volume–XV, Special Issues– I(b) Jan – Feb, 2026

OPEN ACCESS Original Research Article

the AI model to process input data and generate inference results.

Tresponse: Captures the time needed to format, transmit, and deliver the inference output back to the requesting client through the response interface

During cold-start scenarios, the initialization and model loading phases (Tinit and Tload) contribute most significantly to total latency. This formulation highlights why latency-sensitive applications require optimization techniques such as caching and execution environment reuse.

Comparative Analysis: Traditional Ai Vs Serverless AI

Serverless AI provides a flexible and efficient deployment paradigm compared to traditional AI systems that rely on continuously running servers or VMs. Key differences across resource utilization, scalability, cost, operational complexity, and reliability are summarized below.

A. ***Resource Utilization***

Traditional AI systems keep resources active regardless of workload, causing idle capacity and higher costs. Serverless AI allocates resources on-demand, releasing them immediately after execution, ensuring optimal utilization [1], [8].

B. ***Scalability and Elasticity***

Manual provisioning or predefined autoscaling in traditional AI may respond slowly to traffic spikes. Serverless AI automatically scales function instances in parallel based on incoming events, providing rapid elasticity for variable workloads [2], [3].

C. ***Cost Efficiency***

Traditional deployments incur fixed costs for always-on infrastructure. Serverless AI uses a pay-per-use model, charging only for execution time and resources, reducing cost for event- driven workloads [6], [1].

D. ***Operational Complexity***

Managing servers, dependencies, monitoring, and scaling increases operational overhead in traditional systems. Serverless AI offloads infrastructure management to the provider, allowing developers to focus on model and application logic [7], [2].

E. ***Fault Tolerance and Reliability***

Server-based AI requires explicit redundancy and recovery mechanisms. Serverless platforms offer built-in fault tolerance with automatic retries and distributed workload execution [8], [9].

F. ***Summary of Comparison***

Table I highlights the key differences between traditional AI deployment models and Serverless AI.

TABLE I

TRADITIONAL AI VS. SERVERLESS AI

Aspect	Traditional AI	Serverless AI
Resource Utilization	Compute resources remain active regardless of workload	Resources are allocated only during execution
Scalability	Manual scaling or delayed auto-scaling	Automatic and fine-grained scaling
Cost Model	Fixed cost due to always-on infrastructure	Pay-per-use based on execution
Maintenance	High operational overhead	Infrastructure managed by provider
Fault Tolerance	Requires explicit configuration	Built-in fault tolerance

ISSN–2278-5655

AMIERJ Aarhat Multidisciplinary International Education Research Journal

Volume–XV, Special Issues– I(b) Jan – Feb, 2026

OPEN ACCESS Original Research Article

Applications of Serverless AI:

Serverless AI is well suited for applications that require on-demand inference, rapid scalability, and cost-efficient execution under highly variable workloads. By removing the need for continuously running infrastructure, the serverless execution model enables organizations to deploy AI-driven functionality across multiple domains while minimizing operational overhead. This section highlights key application areas where Serverless AI provides notable benefits.

A. ***Image and Video Processing***

Applications such as object detection, image classification, and video analytics benefit from Serverless AI by triggering model execution only upon new data uploads. This approach supports real-time surveillance, media tagging, and visual search while reducing idle resource usage [3].

B. ***Chatbots and Conversational Systems***

Natural language processing models for chatbots and virtual assistants are invoked dynamically based on incoming user messages. Serverless AI ensures low-latency responses and can automatically scale during peak interaction periods [8].

C. ***Healthcare and Medical Analytics***

Medical applications, including diagnostic imaging and symptom analysis, leverage Serverless AI for on-demand inference. This reduces infrastructure costs while maintaining responsiveness for sporadic yet critical workloads [6].

D. ***Recommendation and Personalization Systems***

Recommendation engines can dynamically generate personalized suggestions based on user interactions. Event-driven execution allows efficient scaling and reduces the need for continuously active inference servers [9].

E. ***Fraud Detection and Anomaly Analysis***

Serverless AI supports real-time anomaly detection by triggering inference for each transaction or event. This allows systems to automatically scale during periods of high activity and reduce costs during low activity intervals [7].

Overall, Serverless AI is highly effective for applications with bursty workloads, unpredictable demand, and short-lived inference tasks, providing flexibility, efficiency, and cost savings

Challenges and Limitations:

Despite its advantages, Serverless AI inherits several limitations from the underlying serverless computing model that can impact performance, flexibility, and developer experience. These challenges must be carefully considered when deploying AI inference workloads in serverless environments.

A. ***Cold-Start Latency***

Serverless functions often experience a delay when invoked after inactivity. This occurs due to runtime initialization, dependency loading, and resource provisioning, which can impact latency-sensitive applications such as real-time APIs or interactive AI services [2].

B. ***Limited Execution Time***

Cloud providers impose maximum execution times (e.g.15 minutes), making long-running tasks or computationally intensive AI workloads unsuitable without refactoring into smaller sub-tasks [3].

C. ***Hardware Restrictions***

Most serverless platforms offer general-purpose CPU execution and limited or no GPU/TPU access, constraining large scale deep learning inference and training [6].

D. ***Model Size Constraints***

Large models require substantial memory and storage. Loading them into stateless execution

ISSN–2278-5655

AMIERJ Aarhat Multidisciplinary International Education Research Journal

Volume–XV, Special Issues– I(b) Jan – Feb, 2026

OPEN ACCESS Original Research Article

contexts increases cold start latency and initialization overhead [9].

E. ***Debugging Complexity***

Serverless architectures are distributed and ephemeral, making debugging, tracing, and reproducing errors more difficult compared to traditional AI systems [7].

Overall, these limitations necessitate careful consideration when deploying AI workloads in serverless environments, particularly for latency-sensitive, memory-intensive, or long running tasks

Future Research Directions:

Although Serverless AI provides an efficient paradigm for event-driven inference, several open research challenges remain. Addressing these issues can significantly enhance the performance, flexibility, and adoption of Serverless AI systems.

A. ***Accelerator-Aware Serverless Platforms***

Current serverless platforms have limited GPU/TPU support. Future research should explore accelerator-aware architectures, using virtualization and resource pooling to enable efficient, high-performance AI inference for short-lived functions [8], [9].

B. ***Cold-Start Reduction and Model Caching***

Cold-start latency is a key challenge in Serverless AI. Future research should focus on pre-warming, predictive invocation, model caching, and optimized packaging/serialization to reduce startup delays while maintaining stateless execution[3], [1].

C. ***Hybrid Serverless and Edge AI Architectures***

Hybrid serverless-edge AI architectures enable low-latency, bandwidth-efficient inference by combining edge preprocessing with cloud-based serverless execution. Future research can explore workload partitioning, latency-aware orchestration, and energy-efficient execution [10], [11].

D. ***Support for Large and Complex Models***

Deploying large AI models in serverless environments is challenging. Future work can explore model compression, quantization, modular inference, and distributed execution across multiple serverless functions [12], [13].

E. ***Observability and Debugging Frameworks***

Debugging and performance analysis in serverless environments are challenging. Future research should focus on specialized observability frameworks for Serverless AI to monitor latency, resource use, and model behavior across functions [2], [7].

Overall, these research directions indicate that Serverless AI remains an evolving paradigm with substantial potential. Continued advances in cloud infrastructure, AI frameworks, and serverless platforms are expected to mitigate existing limitations and enable broader adoption across real-time and large-scale intelligent applications.

Conclusion:

This paper presented a comprehensive analysis of Serverless AI, examining its architecture, execution workflow, advantages, and limitations. The study demonstrated that Serverless AI offers significant benefits for inference workloads with unpredictable demand patterns. While challenges remain, on-going advancements in cloud infrastructure and AI frameworks position Serverless AI as a promising deployment paradigm for next-generation intelligent systems.

References:

1. *E. Jonas et al., "Cloud programming simplified: A berkeley view on serverless computing," arXiv preprint arXiv:1902.03383, 2019.*
2. *I. Baldini, P. Castro, P. Chang et al., "Serverless computing: Current trends and*

ISSN–2278-5655

AMIERJ Aarhat Multidisciplinary International Education Research Journal

Volume–XV, Special Issues– I(b) Jan – Feb, 2026

OPEN ACCESS Original Research Article

open problems," Research Advances in Cloud Computing,

3. *pp. 1–20, 2017.*
4. *L. Wang et al., "Peeking behind the curtains of serverless platforms,"*
5. *USENIX ATC, 2018.*
6. *C. Pahl, "Containerization and the paas cloud," IEEE Cloud Computing, 2015.*
7. *S. Hendrickson et al., "Serverless computation with openlambda,"*
8. *USENIX HotCloud, 2016.*
9. *R. Buyya et al., Serverless Computing: Principles and Paradigms. Wiley, 2019.*
10. *M. Armbrust et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010.*
11. *M. Shahrad et al., "Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider," USENIX ATC, 2020.*
12. *J. Spillner, "Cloud api management and gateways," IEEE Cloud Com- puting, 2021.*
13. *M. Satyanarayanan, "The emergence of edge computing," Computer, vol. 50, no. 1, pp. 30–39, 2017.*
14. *W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, 2016.*
15. *S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in International Conference on Learning Representations (ICLR), 2016.*
16. *J. Li, Z. Li, Q. Yang, and C. Li, "Model splitting for distributed inference in serverless architectures," IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 2, pp. 291–305, 2020.*

Cite This Article:

Patil A. & Kadam S. (2026). *Serverless AI: Architecture and Execution of AI Models Without Servers.* **In Aarhat Multidisciplinary International Education Research Journal**: Vol. XV (Number I, pp. 48–54)

Doi: https://doi.org/10.5281/zenodo.18637881