Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Serverless AI on Kubernetes: Benefits and Challenges of Using Knative for ML Workloads

Praneel Madabushini
NVIDIA Corporation, USA
Corresponding Author: Praneel Madabushini, **E-mail**: madabushinipraneel@gmail.com

| ABSTRACT

Traditional Kubernetes deployments for artificial intelligence workloads often result in resource underutilization and continuous infrastructure provisioning, leading to significant cost inefficiencies. Serverless computing paradigms address these challenges by enabling dynamic resource allocation and automatic scaling based on demand. Knative emerges as a prominent Kubernetes-native serverless platform that transforms how machine learning models are deployed and executed in containerized environments. The platform provides two core components: Knative Serving for automated deployment and traffic management, and Knative Eventing for creating complex event-driven workflows that enable asynchronous AI workload orchestration. Key advantages include scale-to-zero capabilities that eliminate resource waste during idle periods, seamless integration with existing Kubernetes ecosystems, and support for microservices-based AI applications. However, implementation presents notable challenges including cold start latency that affects real-time inference performance, dependency on specialized GPU optimization plugins, and constraints imposed by stateless architecture requiring external state management solutions. The complexity of debugging multi-component eventing workflows further complicates operational management. These trade-offs between resource efficiency and performance characteristics determine the suitability of Knative for specific machine learning deployment scenarios, particularly influencing decisions around latency-sensitive applications versus cost-optimized batch processing workloads.

| KEYWORDS

Serverless computing, Knative, Machine learning workloads, Kubernetes, Event-driven architecture.

| ARTICLE INFORMATION

ACCEPTED:12 July 2025 **PUBLISHED:** 14 August 2025 **DOI:** 10.32996/jcsts.2025.7.8.110

1. Introduction

1.1 Overview of AI Workload Computational Requirements and Resource Management Challenges

Artificial intelligence workloads present unique computational challenges that demand high-performance infrastructure, dynamic scalability, and sophisticated resource management capabilities. These requirements have positioned Kubernetes as a leading platform for deploying AI applications across diverse domains including image recognition, natural language processing, autonomous driving systems, and fraud detection. The extensive automation features of Kubernetes provide significant advantages for managing AI workloads efficiently, addressing the complex orchestration needs inherent in machine learning operations [1].

1.2 Traditional Kubernetes Deployment Limitations for ML Workloads

Traditional Kubernetes deployment strategies for machine learning workloads exhibit significant limitations that impact resource utilization and operational efficiency. When AI workloads are deployed directly as pods through conventional Kubernetes mechanisms, they operate under static resource allocation models where computational resources are provisioned based on average utilization requirements. This approach results in persistent pod execution regardless of actual demand, leading to substantial resource wastage during periods of low activity and potential overprovisioning to accommodate peak loads. The static nature of traditional deployments fails to capitalize on the intermittent and event-driven characteristics typical of many AI inference scenarios.

1.3 Introduction to Serverless Computing Paradigm and Its Application to AI/ML

The serverless computing paradigm has emerged as a transformative solution within the cloud computing landscape, fundamentally altering how enterprises approach service deployment and infrastructure management. Serverless architecture empowers organizations to deploy applications without the burden of underlying infrastructure provisioning and management, enabling automatic scaling and pay-per-use resource models. When applied to artificial intelligence and machine learning contexts, serverless computing introduces the concept of Serverless AI, where ML models execute exclusively in response to triggering events, with computational resources allocated dynamically based on instantaneous demand [2]. This event-driven execution model optimizes resource utilization by eliminating idle resource consumption and enabling precise scaling aligned with actual workload requirements.

1.4 Research Objectives and Article Scope

This article examines the implementation of serverless AI workloads on Kubernetes through Knative, an open-source serverless platform designed specifically for Kubernetes environments. The scope encompasses the architectural components of Knative, its integration capabilities with existing Kubernetes AI toolkits, and the practical implications of adopting serverless methodologies for machine learning operations. The analysis addresses both the operational benefits achieved through dynamic resource management and the technical challenges associated with serverless AI deployment, providing insights into the trade-offs between resource efficiency and performance characteristics in modern cloud-native AI implementations.

2. Background and Related Work

2.1 Evolution of AI Workload Deployment Strategies

The deployment of artificial intelligence workloads has undergone significant transformation from traditional monolithic architectures to modern containerized and cloud-native approaches. Early AI deployment strategies relied heavily on dedicated hardware and static resource allocation, which proved inadequate for handling the dynamic nature of machine learning inference and training workloads. The evolution toward distributed computing environments introduced new paradigms that enable more flexible resource management and improved scalability for AI applications [3].

Deployment Era	Architecture Type	Resource Management	Scalability Model	Key Characteristics
Traditional	Monolithic	Static allocation	Manual scaling	Dedicated hardware, fixed resources
Containerized	Microservices	Container-based	Pod-level scaling	Docker containers, orchestration
Kubernetes-Native	Distributed	Dynamic allocation	Horizontal scaling	Declarative configuration, service mesh
Serverless	Event-driven	On-demand	Auto-scaling to zero	Function-as-a-Service, consumption-based

Table 1: Evolution of AI Workload Deployment Strategies [3, 4]

2.2 Kubernetes as a Platform for ML Operations

Kubernetes has established itself as the predominant orchestration platform for machine learning operations, providing essential capabilities for container management, resource scheduling, and service discovery. The platform's declarative configuration model and extensible architecture make it particularly well-suited for managing the complex dependencies and resource requirements inherent in AI workloads. Modern ML operations leverage Kubernetes to orchestrate training pipelines, manage model serving infrastructure, and facilitate continuous integration and deployment workflows for machine learning applications [4].

2.3 Serverless Computing Principles and Cloud Adoption Trends

Serverless computing represents a fundamental shift in cloud service delivery models, emphasizing event-driven execution and automatic resource management without explicit server provisioning. The core principles of serverless architecture include stateless function execution, automatic scaling based on demand, and consumption-based pricing models. Cloud adoption trends demonstrate increasing enterprise preference for serverless solutions due to their ability to reduce operational overhead while maintaining high availability and performance characteristics [3].

2.4 Existing Solutions for Serverless ML Deployment

Contemporary serverless machine learning deployment solutions encompass various platforms and frameworks designed to address the specific requirements of AI workloads. These solutions typically provide managed inference endpoints, automatic model scaling, and integration with popular machine learning frameworks. Existing platforms demonstrate different approaches to handling model lifecycle management, resource optimization, and performance monitoring within serverless execution environments [3].

2.5 Gap Analysis: Traditional vs. Serverless Approaches for AI Workloads

The comparison between traditional and serverless approaches for AI workload deployment reveals significant differences in resource utilization patterns, operational complexity, and cost structures. Traditional deployment methods often result in resource underutilization due to static provisioning strategies, while serverless approaches enable more efficient resource allocation through dynamic scaling mechanisms. However, serverless deployments introduce new challenges related to cold start latency, state management, and debugging complexity that must be carefully considered when selecting deployment strategies for specific AI use cases [4].

3. Knative Architecture and Core Components

3.1 Overview of Knative as a Kubernetes-native Serverless Platform

Knative emerges as a comprehensive Kubernetes-native serverless platform designed to simplify the deployment and management of containerized workloads in cloud-native environments. Built as an open-source toolkit that operates seamlessly within existing Kubernetes clusters, Knative abstracts the complexity of traditional container orchestration while preserving the flexibility and extensibility that characterizes Kubernetes ecosystems. The platform enables organizations to adopt serverless computing principles without migrating away from their established Kubernetes infrastructure, providing a bridge between traditional container deployment models and modern serverless architectures [5].

3.2 Knative Serving: Deployment Automation, Scaling Mechanisms, Networking, and Traffic Management

Knative Serving constitutes the foundational component responsible for automating application deployment, implementing intelligent scaling mechanisms, and managing network connectivity for serverless workloads. The serving component provides sophisticated autoscaling capabilities that dynamically adjust resource allocation based on incoming request patterns, including the ability to scale applications to zero replicas during periods of inactivity. Traffic management features enable advanced deployment strategies such as blue-green deployments and canary releases, while the networking subsystem automatically configures ingress controllers and service meshes to ensure reliable communication between distributed components [5].

Component	Primary Function	Key Features	Integration Capabilities	Use Cases
Knative Serving	Application deployment	Auto-scaling, traffic management, blue-green deployment	Istio, Contour, Kourier	Model serving, API endpoints
Knative Eventing	Event orchestration	Event routing, filtering, transformation	Kafka, RabbitMQ, Cloud Events	Workflow automation, data pipelines
Build Component	CI/CD integration	Source-to-container builds	Tekton, Cloud Build	Model deployment pipelines

Table 2: Knative Architecture Components Comparison [5, 6]

3.3 Knative Eventing: Event-driven Workflows and Asynchronous AI Workload Orchestration

Knative Eventing delivers a robust framework for creating complex event-driven workflows that enable sophisticated orchestration of AI workloads through asynchronous communication patterns. The eventing system supports the development of loosely coupled microservices architectures where individual AI models can be triggered independently based on specific events, creating cascading workflows where the output of one model serves as input events for subsequent processing stages.

This architectural approach facilitates the construction of complex AI pipelines that can adapt dynamically to varying workload patterns while maintaining high availability and fault tolerance [6].

3.4 Integration Capabilities with Kubernetes AI Toolkits and Message Queues

Knative demonstrates extensive integration capabilities with established Kubernetes AI toolkits, particularly Kubeflow, enabling seamless incorporation into existing machine learning operations workflows. The platform supports native integration with popular message queuing systems including Apache Kafka, facilitating reliable event delivery and supporting high-throughput data processing scenarios common in AI applications. These integration capabilities allow organizations to leverage their existing investments in Kubernetes-based AI infrastructure while gaining the benefits of serverless execution models [5].

3.5 Comparison with Other Serverless Platforms for ML Workloads

When compared to alternative serverless platforms designed for machine learning workloads, Knative distinguishes itself through its deep integration with Kubernetes ecosystems and its emphasis on open-source extensibility. Unlike proprietary cloud-specific serverless offerings, Knative provides vendor-neutral deployment capabilities that support multi-cloud and hybrid cloud strategies. The platform's architectural design prioritizes compatibility with existing Kubernetes tooling and workflows, making it particularly attractive for organizations seeking to adopt serverless principles without abandoning their current container orchestration investments [6].

4. Benefits of Knative for ML Workloads

4.1 Dynamic Resource Allocation and Cost Optimization Through Scale-to-Zero Capabilities

Knative's scale-to-zero functionality represents a fundamental advancement in resource utilization efficiency for machine learning workloads, enabling automatic termination of idle resources and instantaneous scaling based on demand patterns. This capability eliminates the persistent resource consumption characteristic of traditional Kubernetes deployments, where pods remain active regardless of actual utilization levels. The dynamic scaling mechanisms continuously monitor incoming requests and automatically provision or deallocate computational resources, resulting in significant cost reductions for organizations deploying AI workloads with variable traffic patterns [7].

Aspect	Traditional Kubernetes	Knative Serverless	Impact on ML Workloads
Resource Utilization	Persistent pod execution	Scale-to-zero capability	Eliminates idle resource waste
Infrastructure Management	Manual configuration	Automated provisioning	Reduced operational overhead
Scaling Model	Manual/HPA scaling	Event-driven auto-scaling	Responsive to inference demand
Traffic Management	LoadBalancer/Ingress	Built-in traffic splitting	A/B testing for model versions
Deployment Strategy	Rolling updates	Blue-green/Canary	Zero-downtime model updates

Table 3: Benefits of Knative vs Traditional Kubernetes for ML Workloads [7, 8]

4.2 Automated Infrastructure Management and Reduced Operational Overhead

The platform abstracts complex infrastructure management tasks through intelligent automation, significantly reducing the operational burden associated with maintaining AI workload deployments. Knative automatically handles load balancing, service discovery, health monitoring, and failure recovery without requiring explicit configuration or manual intervention from operations teams. This automation extends to networking configuration, SSL certificate management, and ingress controller setup, enabling development teams to focus on model development and optimization rather than infrastructure concerns [8].

4.3 Event-Driven Architecture Enabling Complex AI Workflow Orchestration

Knative's event-driven architecture facilitates the construction of sophisticated AI workflow orchestrations through loosely coupled microservices that communicate via asynchronous event streams. This architectural approach enables the development of complex processing pipelines where individual AI models operate independently while contributing to larger analytical workflows. The event-driven model supports advanced patterns such as parallel processing, conditional branching, and dynamic workflow adaptation based on intermediate results, providing flexibility for implementing complex AI use cases [8].

4.4 Seamless Integration with Existing Kubernetes Ecosystems

The platform's native Kubernetes integration ensures compatibility with established container orchestration workflows, enabling organizations to leverage existing investments in Kubernetes infrastructure and tooling. Knative operates transparently within standard Kubernetes clusters, supporting existing monitoring solutions, logging frameworks, and security policies without requiring architectural modifications. This integration approach minimizes migration complexity while providing immediate access to serverless capabilities for organizations already committed to Kubernetes-based infrastructure strategies [7].

4.5 Support for Microservices-Based AI Applications and Modular Deployment Strategies

Knative facilitates the decomposition of monolithic AI applications into discrete microservices, enabling independent scaling, deployment, and maintenance of individual model components. This modular approach supports advanced deployment strategies including canary releases, A/B testing, and gradual rollouts for AI model updates. The platform's traffic splitting capabilities enable sophisticated experimentation workflows where multiple model versions can operate simultaneously, allowing for performance comparison and gradual migration strategies that minimize risk during model deployment cycles [8].

5. Challenges and Limitations

5.1 Cold Start Latency Issues and Performance Implications for Real-Time AI Applications

Cold start latency represents a significant challenge for serverless AI deployments, particularly when applications require immediate response times for real-time inference scenarios. The initialization process for AI workloads involves container startup, model loading, and dependency resolution, which can introduce substantial delays before the first request can be processed. This latency becomes particularly problematic for applications requiring consistent sub-second response times, such as autonomous vehicle decision systems or real-time fraud detection services, where delays can compromise system effectiveness and user experience [8].

5.2 GPU Optimization Requirements and Dependency on Specialized Plugins

Knative's default configuration lacks native optimization for GPU-accelerated workloads, requiring integration with specialized plugins and inference servers to achieve optimal performance for computationally intensive AI models. The dependency on external components such as the NVIDIA Triton inference server introduces additional complexity in deployment pipelines and requires specialized knowledge for proper configuration and maintenance. GPU resource scheduling and allocation mechanisms must be carefully managed to prevent resource conflicts and ensure efficient utilization across multiple concurrent AI workloads [9].

5.3 Stateless Architecture Constraints and External State Management Solutions

The inherently stateless nature of serverless architectures creates challenges for AI workloads that require persistent state management, session continuity, or intermediate result storage during multi-step inference processes. Applications must rely on external state management solutions such as Redis for caching, MinIO for object storage, or database systems for persistent data, introducing additional infrastructure dependencies and potential performance bottlenecks. These external dependencies can complicate deployment architectures and introduce new failure modes that must be carefully managed [8].

Challenge Category	Specific Issues	Performance Impact	Mitigation Strategies	External Dependencies
Cold Start Latency	Container initialization, model loading	High response time	Pre-warming, model caching	Container registries
GPU Optimization	Resource scheduling, driver compatibility	Suboptimal performance	NVIDIA Triton, GPU operators	Specialized plugins
State Management	Session persistence, intermediate results	Data consistency issues	Redis, MinIO, databases	External storage systems
Debugging Complexity	Distributed tracing, event correlation	Increased troubleshooting time	Observability tools, logging	Monitoring platforms
Resource Scheduling	Memory allocation, CPU intensive tasks	Resource contention	Resource quotas, node affinity	Cluster autoscaling

Table 4: Challenges and Mitigation Strategies for Knative ML Deployments [9, 10]

5.4 Debugging Complexity in Multi-Component Eventing Workflows

The distributed nature of event-driven AI workflows significantly complicates debugging and troubleshooting processes, particularly when issues arise from interactions between multiple loosely coupled components. Tracing request flows across asynchronous event chains becomes challenging, especially when failures occur in intermediate processing stages or when event delivery mechanisms experience delays or failures. The lack of centralized logging and monitoring for complex eventing workflows can impede rapid problem resolution and system maintenance [9].

5.5 Resource Scheduling Limitations for Computationally Intensive ML Models

Knative's resource scheduling mechanisms may prove inadequate for extremely computationally intensive machine learning models that require specialized hardware configurations, extended processing times, or large memory allocations. The platform's emphasis on rapid scaling and stateless execution can conflict with the requirements of long-running training jobs or models that need warm-up periods to achieve optimal performance. Resource contention and scheduling inefficiencies can emerge when multiple resource-intensive AI workloads compete for limited computational resources within the same cluster [8].

6. Conclusion

Knative emerges as a transformative platform for deploying machine learning workloads in Kubernetes environments, offering significant advantages through dynamic resource allocation, automated infrastructure management, and event-driven orchestration capabilities. The platform's scale-to-zero functionality addresses the persistent resource waste characteristic of traditional Kubernetes deployments, while its native integration with existing Kubernetes ecosystems enables organizations to adopt serverless principles without abandoning established infrastructure investments. The event-driven architecture facilitates sophisticated AI workflow orchestrations, supporting complex processing pipelines and microservices-based applications that can adapt dynamically to varying workload patterns. However, implementation challenges including cold start latency, GPU optimization requirements, and debugging complexity in distributed eventing workflows must be carefully evaluated against specific application requirements. The stateless architecture constraints and resource scheduling considerations for computationally intensive models further influence deployment decisions. Organizations considering Knative adoption should evaluate these trade-offs between resource efficiency gains and performance characteristics based on their specific AI workload profiles, latency requirements, and operational constraints. The platform demonstrates particular suitability for cost-sensitive applications with variable traffic patterns, while latency-critical real-time inference scenarios may require additional optimization strategies or hybrid deployment approaches to achieve optimal performance outcomes.

Funding: This research received no external funding
Conflicts of interest: The authors declare no conflict of interest

References

[1] Amine B, et al., (2022). Serverless on Machine Learning: A Systematic Mapping Study, IEEE Access, Volume 10, 12 September 2022. https://ieeexplore.ieee.org/document/9888122

[2] An Z et al., (2023). RTGPU: Real-Time GPU Scheduling of Hard Deadline Parallel Tasks with Fine-Grain Utilization, IEEE Transactions on Parallel and Distributed Systems, 6 Feb 2023. https://arxiv.org/pdf/2101.10463

[3] Davide L et al., (2025). Serverless Microservice Architecture for Cloud-Edge Intelligence in Sensor Networks, *IEEE Sensors Journal,* Vol. 25, No. 5, 1 March 2025. https://sisinflab.poliba.it/publications/2024/LIGBFPLSRD24/Loconte_et_al_IEEE_SensorsJ2024_editorial.pdf

[4] Dayong F & Dongzhi H, (2020). Knative Autoscaler Optimize Based on Double Exponential Smoothing, 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC), 16 July 2020. https://ieeexplore.ieee.org/document/9141858/similar#similar

[5] Faisal M & Ross B, (2022). Machine Learning on Kubernetes: A Practical Handbook, Packt Publishing eBook (Indexed by IEEE), 2022. https://ieeexplore.ieee.org/book/10162846

[6] Muhammed G, et al., (2024). MASTER: Machine Learning-Based Cold Start Latency Prediction Framework in Serverless Edge Computing Environments for Industry 4.0, *IEEE Journal of Selected Areas in Sensors (JSAS),* 2024. https://xplorestaging.ieee.org/ielx7/10057477/10185156/10517641.pdf?arnumber=10517641&isnumber=10185156

[7] Ravi P & Tirimula R B, (2025). An Overview of AI Workload Optimization Techniques, Artificial Intelligence-Enhanced Software and Systems Engineering (AISSE), Volume 7, Springer, 24 May 2025. https://link.springer.com/chapter/10.1007/978-3-031-88188-6_12

[8] Xiangxian & Yuanyi, (2025). An Automatic Scaling Solution for LLM Inference Services Based on Knative," Alibaba Cloud Technical Community (IEEE-indexed), 14 May 2025. https://www.alibabacloud.com/blog/an-automatic-scaling-solution-for-llm-inference-services-based-on-knative_602223

[9] Yuanyi, (2021). Knative: Serverless Practices in AI Event-Driven Scenarios, Alibaba Cloud Technical Community (IEEE-indexed), 19 July 2021. https://www.alibabacloud.com/blog/knative-serverless-practices-in-ai-event-driven-scenarios_597929