GSC Advanced Research and Reviews
eISSN: 2582-4597 CODEN (USA): GARRC2
Cross Ref DOI: 10.30574/gscarr
Journal homepage: https://gsconlinepress.com/journals/gscarr/

(REVIEW ARTICLE)

# Optimizing Software Engineering Pipelines for Secure Deployment of AI Fraud Detection Systems on Multi-Cloud Infrastructure

Azeez Rabiu [1], Emmanuel Ezeakile [2], Abdulateef Oluwakayode Disu [3], Cynthia Alabi [4] and Moses Oluwasegun Odewale [5,*]

[1] *Department of Computer Science, Faculty of Computing, University of Ibadan, Ibadan, Nigeria.*
[2] *Department of Electrical and Information Engineering, College of Engineering, Covenant University, Ota, Ogun State, Nigeria.*
[3] *Department of Computer Science, School of Computing and Engineering Sciences, Babcock University Ilishan-Remo, Ogun, Nigeria.*
[4] *School of Geography and Natural Sciences, Northumbria University, UK.*
[5] *College of Business, Lamar University, Beaumont, Texas, U.S.A.*

GSC Advanced Research and Reviews, 2026, 26(01), 166-178

Publication history: Received on 05 December 2025; revised on 12 January 2026; accepted on 14 January 2026

Article DOI: https://doi.org/10.30574/gscarr.2026.26.1.0387

## Abstract

The convergence of artificial intelligence, fraud detection, and multi-cloud infrastructure presents unique challenges at the intersection of software engineering, cybersecurity, and distributed systems. This review examines the current state of research on optimizing software engineering pipelines for deploying AI-based fraud detection systems across multi-cloud environments. We synthesize findings from contemporary literature, analyzing architectural patterns, security frameworks, deployment strategies, and performance optimization techniques. The review addresses three critical research questions concerning architectural patterns and pipeline optimization strategies for multi-cloud deployments, security requirements influencing pipeline design, and current limitations with future research directions. Key findings indicate that microservices-based architectures leveraging container orchestration, event-driven processing, and hierarchical feature stores enable effective multi-cloud deployment. However, significant complexities persist in model versioning, data governance, cross-cloud data transfer costs, and security orchestration. We identify critical gaps in standardized pipeline architectures and propose a research agenda focusing on AI-native infrastructure, confidential computing, automated optimization, and sustainable ML practices. Case studies from financial services and e-commerce sectors illustrate practical implementations, while identified challenges and future directions provide a roadmap for advancing this critical domain.

**Keywords:** AI fraud detection; Multi-cloud infrastructure; Microservices architecture; Privacy-preserving machine learning; Real-time transaction scoring

## 1. Introduction

The landscape of financial fraud has transformed dramatically over the past decade, evolving from simple rule-based attacks to sophisticated, adaptive schemes that exploit weaknesses in digital payment systems and online platforms. The global cost of fraud has surged beyond $485 billion annually, creating unprecedented pressure on organizations to deploy advanced detection systems capable of identifying complex patterns in real-time transaction streams[1]. Traditional rule-based fraud detection systems, once the backbone of security operations, have proven increasingly inadequate against adversaries who continuously adapt their tactics to evade detection. Parallel to this evolution, enterprise IT infrastructure has undergone its own transformation, with approximately 87% of enterprises now

* Corresponding author: Cynthia Alabi.

utilizing services from multiple cloud providers [2]. This shift reflects strategic considerations around vendor independence, geographic distribution, regulatory compliance, and cost optimization. However, the marriage of AI-driven fraud detection with multi-cloud infrastructure introduces a complex set of engineering challenges that span machine learning operations, distributed systems, cybersecurity, and software architecture.

The deployment of production-grade AI fraud detection systems demands more than simply running machine learning models in the cloud [3]. These systems must satisfy stringent operational requirements that often conflict with one another. Real-time performance constraints require transaction scoring to occur within 50 to 200 milliseconds to avoid disrupting user experience, while simultaneously maintaining exceptionally high availability typically exceeding 99.99% uptime to avoid financial losses and reputational damage [4]. Regulatory frameworks introduce additional complexity, mandating explainability of automated decisions, protection of personally identifiable information, and maintenance of detailed audit trails. The adaptive nature of fraud patterns necessitates continuous model retraining and deployment, while security requirements demand protection against adversarial attacks designed to evade or manipulate the detection systems themselves. Traditional software engineering pipelines, designed primarily for stateless web applications and microservices, prove fundamentally inadequate for AI systems that introduce unique concerns around data versioning, feature engineering, model training orchestration, experiment tracking, and performance monitoring [5].

This review addresses the current state of research and practice in optimizing software engineering pipelines for secure deployment of AI fraud detection systems across multi-cloud infrastructure. We synthesize findings from recent literature to answer several critical research questions. First, what architectural patterns and pipeline optimization strategies enable effective deployment of AI fraud detection systems across multi-cloud environments? Second, how do the security requirements specific to fraud detection, where systems themselves become targets for adversarial manipulation, influence pipeline design and deployment workflows? Third, what are the current limitations in existing approaches and what future research directions appear most promising? Our scope focuses specifically on production-grade systems rather than experimental prototypes or academic demonstrations, emphasizing practical implementations that must balance multiple competing objectives including prediction accuracy, inference latency, system reliability, security posture, and operational cost.

The structure of this review proceeds systematically through the key dimensions of this problem space. We begin by establishing foundational concepts around AI fraud detection systems, multi-cloud infrastructure characteristics, and the unique requirements of machine learning pipelines. Subsequent sections examine architectural patterns that have emerged as best practices, security frameworks tailored to the unique threat model of fraud detection systems, and optimization strategies across the data, training, and inference pipelines. Case studies from financial services and e-commerce provide concrete illustrations of these concepts in practice, synthesizing lessons learned from deployments processing millions to billions of transactions daily across geographically distributed infrastructure. The review concludes by identifying current limitations and proposing directions for future research that could address remaining gaps in our understanding and capabilities.

## 2. Foundational Concepts

Understanding the optimization of software engineering pipelines for AI fraud detection on multi-cloud infrastructure requires establishing clear conceptual foundations across three interconnected domains [6]. The first concerns the architecture and operational requirements of modern fraud detection systems. The second addresses the characteristics, benefits, and challenges of multi-cloud infrastructure. The third examines how software engineering pipelines must evolve to accommodate the unique lifecycle requirements of machine learning systems.

Modern AI fraud detection systems represent a significant departure from their rule-based predecessors [7]. Rather than relying exclusively on manually crafted heuristics, contemporary systems typically employ ensemble approaches that combine multiple machine learning paradigms. Supervised learning algorithms, including gradient boosting machines and deep neural networks, learn patterns from labeled historical fraud cases. Unsupervised methods such as autoencoders and isolation forests identify anomalies that deviate from normal transaction patterns without requiring explicit fraud labels. Graph-based algorithms analyze networks of relationships between entities, uncovering fraud rings and coordinated attacks that might appear innocuous when examined in isolation [8].

These AI systems must satisfy operational requirements that profoundly influence their deployment architecture. The imperative for real-time performance stands paramount, as transaction scoring must occur within tens to hundreds of milliseconds to avoid introducing noticeable delays in user experience. This constraint cascades through every architectural decision, from the design of feature stores that must retrieve user and transaction features with

microsecond latency, to the selection of model serving infrastructure capable of executing inference at scale. Research has consistently demonstrated that fraud detection models experience accuracy degradation over time as fraud patterns evolve, with studies documenting performance declines of 15 to 25 percent within ninety days absent retraining [9]. This phenomenon necessitates systems capable of continuous learning, automated retraining, and seamless model updates without service disruption.

Regulatory frameworks introduce additional requirements that complicate deployment considerably [10]. The General Data Protection Regulation in Europe, the Fair Credit Reporting Act in the United States, and similar legislation globally mandate that automated decisions affecting individuals must be explainable. This requirement influences not only model selection, favoring interpretable algorithms or those amenable to post-hoc explanation methods, but also the architecture of production systems which must capture sufficient context to reconstruct the reasoning behind specific fraud determinations. Financial systems demand exceptional reliability, with availability requirements typically exceeding four nines, meaning less than one hour of downtime annually [11]. Achieving such reliability in the face of software updates, infrastructure failures, and operational changes requires sophisticated deployment strategies and redundancy mechanisms.

Multi-cloud infrastructure has emerged as a dominant architectural paradigm in enterprise IT, yet its adoption introduces substantial complexity alongside its benefits [12]. Organizations deploy workloads across Amazon Web Services, Microsoft Azure, Google Cloud Platform, and increasingly specialized providers to achieve several strategic objectives. Vendor independence represents perhaps the most frequently cited motivation, as organizations seek to avoid lock-in to proprietary services and maintain negotiating leverage with providers. Geographic distribution enables compliance with data residency requirements while simultaneously reducing latency for global operations by placing compute resources closer to end users. Cost optimization becomes possible through strategic workload placement, exploiting differences in pricing models, taking advantage of committed use discounts from multiple providers, and leveraging spot markets for fault-tolerant workloads.

However, multi-cloud architectures demand careful navigation of heterogeneity across multiple dimensions. Each cloud provider offers distinct services with different capabilities, performance characteristics, and operational models. What AWS calls Lambda, Azure terms Functions and Google names Cloud Functions, and despite functional similarity, subtle differences in execution models, scaling behavior, and integration patterns create portability challenges. Network latency between clouds, typically ranging from tens to hundreds of milliseconds depending on geographic distribution, can significantly impact distributed training algorithms and real-time inference when components span provider boundaries [13]. Security policies must be harmonized across platforms with different identity models, networking paradigms, and native security services. Data synchronization and consistency present ongoing challenges, particularly for feature stores that must serve low-latency predictions while maintaining reasonable consistency of feature values computed from data distributed across clouds[14].

Software engineering pipelines for AI systems extend traditional continuous integration and continuous deployment practices to encompass the full machine learning lifecycle [15]. The data pipeline handles ingestion, validation, transformation, and versioning of training data, with fraud detection introducing particular complexities around feature engineering from heterogeneous sources including transaction logs, user behavior analytics, device fingerprinting, and external threat intelligence feeds. Ensuring data quality and consistency across these diverse sources while maintaining the velocity required for timely model updates presents significant engineering challenges.

The training pipeline orchestrates automated model training, hyperparameter optimization, and validation workflows. Contemporary practice emphasizes reproducibility through comprehensive experiment tracking that captures not only model hyperparameters and training metrics but also data versions, code versions, and infrastructure configurations that contributed to any particular model[16]. The stochastic nature of many machine learning algorithms means that training the same model architecture on the same data can produce different results, making reproducibility a non-trivial engineering concern that requires careful attention to random seeds, library versions, and even hardware characteristics that can influence numerical precision.

Deployment pipelines for machine learning models differ substantially from those for traditional applications[17]. Model packaging must include not only the trained parameters but also preprocessing logic, feature engineering code, and inference runtime dependencies. Testing strategies extend beyond functional correctness to include performance validation, bias assessment, and adversarial robustness evaluation[18]. A/B testing infrastructure enables gradual rollout of new models, comparing their performance against current production models on live traffic. Shadow mode deployment, where new models score transactions in parallel with production models but their predictions do not affect

actual decisions, provides a final validation stage that has proven essential for detecting unexpected behaviors that only manifest in production environments with their full complexity and scale.

## 3. Architectural Patterns and Design Principles

### 3.1. Microservices and Container Orchestration

Microservices architecture has emerged as the dominant paradigm for structuring fraud detection systems on multi-cloud platforms [19]. This approach decomposes monolithic fraud detection logic into discrete services, each responsible for a specific capability. Feature extraction services consume raw transaction data and compute derived features. Model inference services execute trained machine learning models against feature vectors. Rule engine services evaluate deterministic rules that complement statistical models. Alert management services orchestrate notifications and case management workflows for detected fraud [20]. This decomposition enables independent scaling of components based on their specific load characteristics, allows different services to utilize different technologies optimized for their particular requirements, and provides fault isolation where failures in one service do not cascade to compromise the entire system.

Container orchestration platforms, particularly Kubernetes, provide the foundation for managing these microservices across heterogeneous cloud environments [21]. Kubernetes abstracts away provider-specific infrastructure details, offering a consistent API and operational model whether deploying on AWS EKS, Azure AKS, Google GKE, or on-premises infrastructure. The Cloud Native Computing Foundation ecosystem extends Kubernetes with essential capabilities including service mesh implementations that handle service-to-service communication, observability tools that provide visibility into distributed system behavior, and configuration management solutions that enable consistent deployment across environments. Cross-cloud federation approaches, while still maturing, enable managing multiple Kubernetes clusters as a single logical unit, facilitating workload mobility and disaster recovery across provider boundaries[22].

Service mesh architecture provides particularly valuable capabilities for fraud detection systems[23]. By moving concerns around service discovery, load balancing, authentication, authorization, and telemetry collection from application code into infrastructure-level proxies, service meshes reduce application complexity while enabling sophisticated traffic management. Canary deployments, where new model versions initially receive a small percentage of traffic that gradually increases as confidence builds, become straightforward to orchestrate. A/B testing, essential for comparing model performance in production, can route traffic to different model versions based on user cohorts or random sampling. Circuit breaking prevents cascading failures when dependencies become unhealthy, automatically falling back to cached predictions or simpler models when primary inference services experience issues [24].

### 3.2. Event-Driven Architecture and Stream Processing

Event-driven architectures address the real-time processing requirements fundamental to fraud detection. Transaction events flow through distributed streaming platforms, with Apache Kafka and cloud-native alternatives like AWS Kinesis, Azure Event Hubs, and Google Pub/Sub providing the backbone for event distribution [25]. These platforms offer durability, exactly-once or at-least-once processing semantics, and the ability to replay events for reprocessing or debugging. Research into hybrid streaming architectures explores maintaining primary event streams in one cloud while replicating to secondary clouds for disaster recovery and geographic distribution. The tension between consistency models and cross-cloud latency requires careful consideration, with eventual consistency acceptable for many fraud signals while critical transactions may require stronger coordination [26].

Stream processing frameworks including Apache Flink, Spark Structured Streaming, and ksqlDB enable real-time feature computation and model scoring on event streams [27]. These frameworks provide high-level abstractions for windowing, aggregation, and stateful processing that simplify the development of streaming applications. However, deploying them across multiple clouds introduces challenges in state management, particularly ensuring that stateful operations can recover correctly after failures or when workloads migrate between clouds. Recent research on cloud-agnostic state backends enables seamless failover without data loss, but achieving this capability requires careful architecture and testing.

### 3.3. Feature Store and Data Management

Feature stores have crystallized as critical infrastructure for machine learning systems, addressing the challenge of maintaining consistent feature definitions and serving features with minimal latency at inference time[28]. The dual-storage paradigm has emerged as standard practice, with offline storage optimized for historical feature values used during training and online storage optimized for low-latency retrieval during inference. Multi-cloud deployments

require thoughtful data replication strategies that balance consistency requirements against the cost of cross-cloud data transfer. Hierarchical feature store architectures implement local caches at regional edges for sub-10 millisecond retrieval latency, global synchronization layers that propagate feature updates across regions, and conflict resolution mechanisms that handle features computed from data sources distributed across clouds [29].

### 3.4. Model Serving Infrastructure

Model serving infrastructure represents a critical performance bottleneck where architectural decisions directly impact both latency and operational complexity [30]. Research and practice have explored several deployment patterns, each with distinct tradeoffs. Embedded serving approaches compile models into application binaries using frameworks like TensorFlow Lite or ONNX Runtime, minimizing network latency and external dependencies at the cost of complicating model updates which require application redeployment. Model-as-a-service patterns deploy dedicated inference services using specialized serving frameworks such as TensorFlow Serving, TorchServe, or Seldon Core [31]. This separation enables independent scaling of inference infrastructure, simplified A/B testing, and model updates without application changes.

Multi-cloud model serving introduces unique challenges in maintaining consistency of deployed model versions and routing inference requests to appropriate deployments. Organizations must ensure that a specific model version behaves identically whether deployed on AWS, Azure, or GCP, despite differences in instance types, library versions, and numerical precision characteristics of underlying hardware. Service mesh routing rules combined with centralized model registries enable sophisticated traffic management, directing inference requests based on geography, model version, or A/B test cohorts. The model registry serves as the source of truth for which models are approved for production, their version history, performance characteristics, and deployment status across environments [32].

## 4. Security Frameworks and Compliance

### 4.1. Model and Data Security

Security considerations permeate every aspect of deploying AI fraud detection systems, with the ironic reality that systems designed to detect malicious activity themselves become attractive targets for adversaries[33]. Model security addresses threats specific to machine learning systems that have no direct analogues in traditional application security. Adversaries may attempt model extraction attacks, querying the production system strategically to reverse-engineer decision boundaries and recreate functionally similar models. Adversarial examples represent another concern, where small perturbations to input features cause models to misclassify fraudulent transactions as legitimate. Model inversion attacks attempt to reconstruct training data from model parameters or behavior, potentially exposing sensitive transaction patterns or personally identifiable information used during training.

Data protection requirements reflect both regulatory mandates and the inherent sensitivity of financial transaction data[34]. Transaction details, personally identifiable information, and model training datasets require encryption at rest and in transit. Multi-cloud deployments complicate key management substantially, as encryption keys must be available for data access yet protected from unauthorized use. Solutions like HashiCorp Vault or cloud-native key management services provide centralized secrets management, but ensuring proper key rotation, access control, and audit logging across heterogeneous cloud environments demands careful engineering. The principle of data minimization suggests collecting and retaining only information necessary for fraud detection purposes, yet this conflicts with the reality that comprehensive data often improves model accuracy and enables investigation of sophisticated fraud schemes after initial detection.

Pipeline security recognizes that continuous integration and continuous deployment infrastructure represents a high-value target for supply chain attacks[35]. Compromising the CI/CD pipeline enables adversaries to inject malicious code into production systems, deploy backdoored models that intentionally misclassify fraudulent transactions, or exfiltrate sensitive data processed during training or inference. Software Bill of Materials tracking provides visibility into all dependencies incorporated into deployments, enabling rapid response when vulnerabilities are discovered in upstream libraries. Artifact signing using cryptographic signatures ensures that only authorized code and models reach production environments, while admission controllers in Kubernetes enforce policies requiring cryptographic verification before allowing containers to run.

### 4.2. Compliance and Regulatory Requirements

Compliance with regulatory frameworks introduces specific technical requirements that influence architectural decisions[36]. Payment Card Industry Data Security Standard imposes detailed controls around storage, transmission,

and processing of cardholder data. General Data Protection Regulation requires that automated decisions significantly affecting individuals must be explainable and that individuals have rights to data access, correction, and deletion. Service Organization Control Type 2 audits assess whether organizations maintain effective internal controls over data security, availability, and confidentiality. Multi-cloud deployments must harmonize compliance across jurisdictions with varying requirements, documenting data flows across cloud boundaries and ensuring appropriate controls at each stage.

### 4.3. Zero Trust Architecture and Privacy-Preserving Techniques

Zero trust architecture principles align naturally with the requirements of multi-cloud fraud detection systems[37]. The fundamental tenet that trust should never be assumed but always explicitly verified applies particularly well in environments where workloads span administrative domains with different security postures. Identity-based access control ensures that every service authenticates explicitly using cryptographic credentials rather than relying on network location. Micro-segmentation applies zero trust principles at the network level, restricting communication between services to explicitly authorized paths, denying all traffic by default and requiring explicit policies for permitted flows. Continuous verification monitors runtime behavior for anomalous activity that might indicate compromise, with tools like Falco detecting unexpected system calls or network connections in containerized applications.

Privacy-preserving machine learning techniques address the tension between utilizing sensitive data for model training and protecting individual privacy[38]. Differential privacy adds carefully calibrated statistical noise to training data or model gradients, providing mathematical guarantees that individual records cannot be identified from trained model behavior. Federated learning enables training models across distributed data sources without centralizing sensitive data, proving particularly valuable for consortium models where competing financial institutions collaborate on fraud detection while maintaining commercial confidentiality. Homomorphic encryption enables computation on encrypted data, allowing fraud detection inference on encrypted transactions without revealing transaction details to the inference service, though computational overhead currently limits practical applications to simpler models [39].

## 5. Pipeline Optimization Strategies

### 5.1. Model Training and Retraining Optimization

The continuous evolution of fraud patterns necessitates frequent model retraining, yet full retraining from scratch on complete historical datasets becomes computationally expensive and time-consuming as data volumes grow[40]. Incremental learning approaches update models with new data without discarding accumulated knowledge, maintaining institutional learning about historical fraud patterns while adapting to emerging tactics. Online learning algorithms adapted for fraud detection, including Follow The Regularized Leader and stochastic gradient descent variants, enable real-time adaptation as new transactions arrive. However, these approaches require careful tuning to balance plasticity, the ability to learn from new patterns, against stability, the preservation of previously learned knowledge.

Trigger-based retraining replaces fixed retraining schedules with automated workflows initiated by performance degradation signals [41]. Rather than retraining daily or weekly regardless of need, drift detection algorithms continuously assess whether model performance remains acceptable or data distributions have shifted significantly from training conditions. Kolmogorov-Smirnov tests compare feature distributions between training data and recent production data, while Population Stability Index metrics quantify overall distribution shift. When degradation exceeds configured thresholds, retraining workflows launch automatically, training new models and evaluating them in shadow mode before promotion to production. Model ensemble management adds another optimization dimension, maintaining diverse model portfolios that provide robustness against evolving fraud tactics, with dynamic ensembles adjusting weights based on recent performance[42].

### 5.2. Data Pipeline and Infrastructure Optimization

Data pipeline optimization addresses the reality that data preparation typically consumes the majority of pipeline execution time[43]. Incremental feature engineering computes features only for changed data rather than reprocessing complete datasets with each pipeline execution. Change data capture patterns identify modified records in source systems, propagating only deltas through feature engineering workflows. Feature materialization pre-computes and caches frequently used features, trading storage cost for reduced inference latency. Distributed data processing frameworks leverage cluster computing for parallel processing of feature engineering and model training workloads, with Apache Spark, Dask, and Ray providing abstractions that enable near-linear scaling to hundreds of nodes for data-parallel tasks.

Infrastructure optimization for multi-cloud deployments balances multiple competing objectives [44]. Cost optimization through strategic workload placement exploits differences in cloud provider pricing models and resource characteristics. Training workloads, typically batch-oriented and fault-tolerant, benefit from spot or preemptible instances that offer substantial discounts in exchange for potential interruption. Inference workloads require stable, low-latency instances despite higher costs given their impact on user-facing performance. Latency optimization requires careful consideration of geographic distribution and network topology, deploying inference services in multiple regions worldwide to minimize network latency for global operations. Resource rightsizing addresses the common problem of over-provisioned infrastructure consuming unnecessary costs or under-provisioned resources causing performance degradation [45].

### 5.3. Testing and Validation Frameworks

Comprehensive testing frameworks ensure quality and reliability despite rapid iteration. Model validation extends beyond traditional accuracy metrics to evaluate false positive rates at operational thresholds, fairness across demographic groups, and robustness against adversarial perturbations [46]. Financial impact analysis estimates expected fraud losses and operational costs associated with model decisions, enabling business-oriented assessment of model quality. Integration testing validates end-to-end pipeline functionality across cloud boundaries, with chaos engineering practices deliberately introducing failures such as network partitions or service unavailability to verify resilience mechanisms. Shadow mode deployment runs new models in parallel with production systems, logging predictions without affecting actual decisions, enabling validation against live traffic and detecting unexpected behaviors that test environments cannot replicate due to their limited ability to capture production complexity and scale.

## 6. Case Studies and Industry Applications

### 6.1. Financial Services Implementations

Examining real-world implementations provides valuable insights into how organizations have translated architectural principles and optimization strategies into production systems processing billions of transactions. The financial services sector, facing the highest fraud exposure and strictest regulatory requirements, has pioneered many approaches that subsequently diffused to other industries. A multinational bank operating across six continents implemented a hybrid architecture spanning AWS and Azure to achieve geographic distribution while maintaining regulatory compliance with data residency requirements. Their system processes approximately 500 million transactions daily with real-time scoring requirements under 100 milliseconds [47]. The architecture employs active-active deployment patterns with continuous replication of models and feature stores across cloud providers, enabling automatic failover without service disruption.

The technical implementation reveals several sophisticated design choices that address multi-cloud complexity. Feature engineering pipelines run independently in each cloud, processing regional transaction streams locally to minimize cross-cloud data transfer [48]. A global feature store synchronization layer propagates derived features that benefit from worldwide context, such as merchant reputation scores and fraud pattern intelligence, with eventual consistency acceptable given these features' relatively slow rate of change. Model training occurs primarily in AWS due to more favorable pricing for GPU-intensive workloads, with trained models distributed to Azure for redundancy and regional inference [49]. The deployment pipeline implements progressive rollout, initially deploying new models to low-traffic regions for validation before global deployment.

A major payment processor adopted a different architectural approach driven by stringent data sovereignty requirements in multiple jurisdictions. Their fraud detection system operates across AWS, Azure, and Google Cloud Platform, with complete data isolation between regions to satisfy regulatory mandates. Rather than synchronizing a global model, they implemented federated learning to build region-specific models that benefit from global fraud intelligence without data movement. Each region trains local models on transactions from that jurisdiction, sharing only model updates rather than raw data with a central aggregation service that combines insights into a global model template [50]. The technical challenges they encountered illuminate common pitfalls in multi-cloud ML deployments, including model version drift where slight differences in library versions or hardware characteristics across clouds caused models to produce inconsistent predictions for identical inputs.

### 6.2. E-commerce and Digital Platforms

E-commerce platforms face distinct fraud challenges including account takeover, payment fraud, promotion abuse, and marketplace fraud where sellers engage in fraudulent activities [51]. A global marketplace operating in over 100

countries deployed fraud detection across Google Cloud Platform and AWS to balance cost optimization with reliability requirements. Their architecture routes inference requests based on geographic proximity to minimize latency for users worldwide. The feature store maintains approximately 50 million user profiles synchronized across clouds with sub-second consistency achieved through conflict-free replicated data types that allow concurrent updates while ensuring eventual convergence.

Dynamic model ensembles adapt their architecture to address seasonal patterns and emerging fraud tactics[52]. During high-traffic periods like major shopping holidays, the system adjusts ensemble weights to favor high-precision models that minimize false positives even at the cost of some missed fraud, recognizing that customer friction during peak sales periods carries exceptional business impact. During normal periods, the ensemble shifts toward balanced precision-recall tradeoffs. This adaptive approach required development of custom monitoring infrastructure that correlates model performance with business metrics, enabling automated decision-making about ensemble configuration.

Their deployment strategy emphasizes rapid experimentation and learning through continuous validation mechanisms[53]. Shadow mode deployment runs continuously, with multiple candidate models scoring all transactions but only the production model affecting actual decisions. This creates a live testbed for evaluating new approaches under realistic conditions. When candidate models demonstrate superior performance in shadow mode, they enter gradual rollout through canary deployment, initially handling a small percentage of traffic with continuous monitoring for unexpected behaviors. This approach has enabled deployment of over 200 model updates annually while maintaining system stability.

### 6.3. Cross-Industry Patterns and Insights

Analysis across these case studies and others reveals common patterns that characterize successful implementations. Organizations that successfully deployed fraud detection on multi-cloud infrastructure typically adopted phased migration strategies rather than attempting wholesale transitions[54]. Initial phases focused on non-critical workloads such as model training and batch scoring, allowing teams to develop expertise with multi-cloud operations before migrating real-time inference serving. This gradual approach reduced risk while building organizational capability and provided opportunities to refine processes before deploying mission-critical components.

Abstraction layers that hide cloud-specific details from application logic emerged as a critical success factor[55]. Organizations that heavily invested in portability layers reported faster iteration and lower costs for managing multi-cloud complexity compared to those that built directly against native cloud services. These abstraction layers encompass infrastructure provisioning through tools like Terraform, application deployment through Kubernetes, and data access through cloud-agnostic APIs. While abstraction introduces some overhead, the ability to migrate workloads between clouds for cost optimization or to respond to capacity constraints justifies this investment.

Observability received consistent emphasis as a prerequisite for successful multi-cloud operations rather than an afterthought[56]. Teams that implemented comprehensive monitoring, distributed tracing, and anomaly detection before migrating to multi-cloud reported significantly smoother transitions than those that attempted to retrofit observability after encountering operational challenges. Security integration from the earliest stages of pipeline design proved more effective and less costly than retrofitting security controls onto existing systems, with organizations treating security as an integral architectural concern implementing more robust defenses with less operational friction.

### 6.4. Lessons Learned and Operational Considerations

Several lessons emerge from production experience that warrant attention from organizations contemplating similar deployments[57]. The hidden complexity of multi-cloud operations exceeds initial estimates substantially, with organizations reporting that operational overhead grows super-linearly with the number of cloud providers. This suggests that multi-cloud strategies should be adopted deliberately for specific strategic benefits rather than simply pursuing diversification for its own sake. The coordination required across different platform paradigms, security models, and operational procedures creates ongoing management overhead that must be weighed against the resilience and flexibility benefits.

Cost visibility and optimization require sustained attention and specialized tooling beyond what native cloud providers offer. Native cloud cost management tools provide visibility within individual clouds but lack the cross-cloud perspective necessary for holistic optimization[58]. Organizations that developed or adopted tools providing unified cost visibility reported better control of multi-cloud spending and more informed decisions about workload placement. The most sophisticated implementations incorporate cost considerations into workload scheduling decisions,

automatically migrating batch workloads to the most cost-effective cloud based on current spot pricing and committed capacity utilization.

Skill development presents an ongoing challenge, with demand for personnel who understand machine learning, cloud infrastructure, security, and operations exceeding supply[59]. Successful organizations invested heavily in training existing staff rather than relying exclusively on external hiring. Building cross-functional teams where data scientists, cloud engineers, and security specialists collaborate closely proved more effective than siloed organization structures. Vendor relationship management gained strategic importance in multi-cloud contexts, with organizations maintaining good relationships with multiple cloud providers reporting better access to technical expertise, advance notice of platform changes, and flexibility in contract negotiations.

## 7. Current Challenges and Future Directions

### 7.1. Technical and Operational Challenges

Despite significant progress in deploying AI fraud detection systems on multi-cloud infrastructure, several technical challenges remain only partially addressed. Cross-cloud data transfer costs continue to present a major obstacle, with organizations reporting that data movement between clouds can consume 15 to 25 percent of total infrastructure spending[60]. Network latency between clouds exhibits high variability that complicates performance guarantees for distributed operations, with cross-cloud latency varying from tens to hundreds of milliseconds depending on routing, peering relationships, and congestion. Model version management across heterogeneous cloud environments presents subtle but impactful challenges, as differences in underlying infrastructure can produce variations in model behavior even when deploying identical container images. Tool fragmentation in the MLOps ecosystem creates integration challenges as organizations attempt to build coherent pipelines from disparate components.

The organizational challenges of operating fraud detection systems on multi-cloud infrastructure often exceed purely technical concerns. The skill gap presents a persistent obstacle, with demand for personnel possessing the necessary combination of expertise far exceeding supply. Cost visibility and attribution in multi-cloud environments proves substantially more complex than in single-cloud deployments, as native cloud billing systems lack the cross-cloud perspective necessary for holistic cost management. Governance complexity grows with each additional cloud provider, as data governance policies, model governance procedures, and security controls must be harmonized across platforms with different native capabilities and paradigms. Change management becomes more challenging when modifications potentially impact systems spanning multiple clouds, with many organizations struggling to maintain deployment velocity in multi-cloud environments[61].

Security challenges specific to multi-cloud fraud detection systems remain areas of active concern. The expanded attack surface inherent in multi-cloud deployments creates additional vulnerabilities that adversaries may exploit, with each cloud provider introducing unique security models and potential misconfigurations. Adversarial attacks targeting machine learning models create adversarial environments more intense than typically considered in ML security research, particularly in the multi-cloud context where attackers might exploit differences in model behavior across clouds[62]. The lack of standardization across the MLOps ecosystem hinders portability and increases development costs, with each tool and platform defining its own abstractions that require translation layers and custom integration code.

### 7.2. Future Research Directions

The future of fraud detection deployment likely involves cloud platforms specifically designed for AI workloads rather than generic compute infrastructure adapted for machine learning. AI-native infrastructure would provide primitives optimized for common ML operations, with seamless access to diverse accelerators including GPUs, TPUs, and emerging architectures. Cognitive infrastructure that learns from workload patterns and automatically optimizes resource allocation, data placement, and network routing offers another promising direction, observing operational behavior and continuously adjusting to optimize objectives including latency, cost, and reliability. Cloud-agnostic ML operations platforms that provide true portability across providers represent a critical need, dramatically reducing the engineering cost of multi-cloud deployments[63].

Confidential computing using trusted execution environments offers promising capabilities for protecting both data and models during processing, with technologies like Intel SGX, AMD SEV, and ARM TrustZone creating hardware-isolated enclaves where sensitive computation occurs protected from the host operating system and cloud provider. Privacy-preserving federated learning that combines multiple techniques could enable unprecedented collaboration on fraud

detection while maintaining competitive confidentiality and regulatory compliance[64]. Real-time adaptive fraud detection systems that continuously learn from streaming data without batch retraining represent an important research direction, with online learning at scale presenting challenges around maintaining model stability while incorporating new patterns and preventing adversarial manipulation of learning processes.

Energy-efficient model design addresses growing concerns about the environmental impact of large-scale machine learning systems, with research on neural architecture search optimized for energy efficiency alongside accuracy identifying model designs that maintain detection performance while reducing computational requirements. Carbon-aware scheduling represents a relatively straightforward optimization that could reduce environmental impact substantially by routing training workloads to data centers powered by renewable energy or scheduling computationally intensive operations during periods of low carbon intensity[65]. Responsible AI practices ensuring fairness, transparency, and accountability in fraud detection require continued research and development, with fraud detection systems making decisions with significant impacts on individuals' financial access requiring inherently interpretable models that maintain competitive accuracy and fairness constraints that prevent discriminatory outcomes.

## 8. Conclusion

The deployment of AI fraud detection systems on multi-cloud infrastructure represents a complex engineering challenge at the intersection of machine learning, distributed systems, and cybersecurity. This review has synthesized current research across architectural patterns, security frameworks, and optimization strategies that characterize successful implementations. Contemporary systems achieve remarkable capabilities through microservices-based designs leveraging container orchestration, event-driven architectures processing high-velocity transaction streams, and sophisticated security frameworks implementing defense-in-depth strategies. Despite substantial progress, significant challenges persist including cross-cloud data transfer costs, model version management complexity, organizational hurdles around skill development and governance, and research gaps in standardization and automated optimization that create unnecessary integration overhead.

Future research directions toward AI-native infrastructure, advanced security mechanisms including confidential computing and post-quantum cryptography, adaptive autonomous systems using online learning and reinforcement learning, and sustainable ML practices offer promising paths forward. The field stands at an inflection point where theoretical advances in machine learning and practical innovations in cloud platforms converge to enable fraud detection capabilities that were impossible just a few years ago. Success requires breaking down traditional disciplinary boundaries, bringing together expertise in data science, software engineering, cloud architecture, security, and fraud domain knowledge. As digital commerce continues expanding globally and fraud tactics evolve in sophistication, the synthesis presented in this review provides a foundation for understanding current capabilities and limitations, while the identified research directions point toward opportunities for meaningful advancement that will likely generalize to other AI applications facing similar challenges around real-time performance, security, and operational complexity at scale.

## Compliance with ethical standards

### *Disclosure of conflict of interest*

No conflict of interest to be disclosed.

## References

[1] Bakare FA, Ikumapayi OJ, England UK. Securing Government Revenue: A Cloud-Based AI Model for Predictive Detection of Tax-Related Financial Crimes.

[2] Govindaraju R, Akbar R, Suryadi K. IT Infrastructure Transformation and its Impact on IT Capabilities in the Cloud Computing Context. International Journal on Electrical Engineering & Informatics. 2018 Jun 1;10(2).

[3] Sethupathy UK. Risk-Aware AI Models for Financial Fraud Detection: Scalable Inference from Big Transactional Data.

[4] Mutahhar A, Khanzada TJ, Shahid MF. Enhanced Scalability and Security in Blockchain-Based Transportation Systems for Mass Gatherings. Information. 2025 Jul 28;16(8):641.

[5] Albada OM, Salman N. Software Architecture in Practice: Challenges and Opportunities in the Age of Digital Transformation: Literature Review.

[6] Jagarlamudi J. *A Qualitative Study of Multi-Cloud Applications Security for Multi-Cloud Developers* (Doctoral dissertation, Colorado Technical University).

[7] Aziz LA, Andriansyah Y. The role artificial intelligence in modern banking: an exploration of AI-driven approaches for enhanced fraud prevention, risk management, and regulatory compliance. Reviews of Contemporary Business Analytics. 2023 Aug;6(1):110-32.

[8] Immaneni J. Strengthening Fraud Detection with Swarm Intelligence and Graph Analytics. International Journal of Digital Innovation. 2022 Nov 14;3(1).

[9] Al Lawati HM, Zainal A, Al-Rimy BA, Al-Azawi M, Kassim MN, Almalki SA, Alghamdi TA. An Integrated Preprocessing and Drift Detection Approach With Adaptive Windowing For Fraud Detection In Payment Systems (February 2025). IEEE Access. 2025 May 13.

[10] Romasheva N, Ilinova A. CCS projects: How regulatory framework influences their deployment. Resources. 2019 Dec 9;8(4):181.

[11] Bauer E. Design for reliability: information and computer-based systems. John Wiley & Sons; 2011 Feb 11.

[12] Hope OS. Multi-cloud strategy for enterprise applications: Cost, performance, and resilience considerations.

[13] Barros S. Solving AI Foundational Model Latency with Telco Infrastructure. arXiv preprint arXiv:2504.03708. 2025 Mar 27.

[14] Mansouri Y, Toosi AN, Buyya R. Data storage management in cloud environments: Taxonomy, survey, and future directions. ACM Computing Surveys (CSUR). 2017 Dec 11;50(6):1-51.

[15] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.

[16] Melchor F, Rodriguez-Echeverria R, Conejero JM, Prieto AE, Gutiérrez JD. A model-driven approach for systematic reproducibility and replicability of data science projects. InInternational Conference on Advanced Information Systems Engineering 2022 Jun 3 (pp. 147-163). Cham: Springer International Publishing.

[17] Paleyes A, Urma RG, Lawrence ND. Challenges in deploying machine learning: a survey of case studies. ACM computing surveys. 2022 Dec 7;55(6):1-29.

[18] Wang J, Ai J, Lu M, Su H, Yu D, Zhang Y, Zhu J, Liu J. A survey of neural network robustness assessment in image recognition. arXiv preprint arXiv:2404.08285. 2024 Apr 12.

[19] Merseedi KJ, Zeebaree SR. The cloud architectures for distributed multi-cloud computing: a review of hybrid and federated cloud environment. The Indonesian Journal of Computer Science. 2024 Apr 1;13(2).

[20] Chandrasekaran T. Project Management for Anti-Fraud Platforms in Financial Institutions: Integrating AI, Real-Time Alerts, and Compliance Automation. International Journal of Advanced Research in Computer Science & Technology (IJARCST). 2023 Jun 15;6(3):8258-66.

[21] Al Qassem LM, Stouraitis T, Damiani E, Elfadel IM. Containerized microservices: A survey of resource management frameworks. IEEE Transactions on Network and Service Management. 2024 Apr 15;21(4):3775-96.

[22] Thummarakoti S. Advanced Container Orchestration Strategies for Multi-Cloud Environments: Enhancing Performance, Scalability, and Resilience. Scalability, and Resilience (February 28, 2025). 2025 Feb 28.

[23] Emma L. Real-Time Fraud Analytics in Financial Systems Using AI-Powered Cloud Micro services Architecture.

[24] Tellnes J. *Dependencies: No software is an island* (Master's thesis, The University of Bergen).

[25] Kumar R. Event-Driven Architectures for Real-Time Data Processing: A Deep Dive into System Design and Optimization. evolution.;7:8.

[26] Deevi SR, Patil SP, Pillai P, Nakamura Y. Cognitive Security in the Cloud Era. Cari Journals USA LLC; 2025 Nov 28.

[27] Rozony FZ. A Comprehensive Review Of Real-Time Analytics Techniques And Applications In Streaming Big Data. Available at SSRN 5256050. 2024 Nov 4.

[28] Vuppalapati SM, Vemasani P, Modi S. THE CORNERSTONE OF SCALABLE ML: EXPLORING THE CRITICAL ROLE OF FEATURE STORES IN LARGE-SCALE VIDEO APPLICATIONS.

[29] Antwi NW. Designing Scalable Cybersecurity Architectures for Edge-Cloud Continuums Using Secure SDN and Distributed Identity Authentication Models.

[30] Shahrad M, Balkind J, Wentzlaff D. Architectural implications of function-as-a-service computing. InProceedings of the 52nd annual IEEE/ACM international symposium on microarchitecture 2019 Oct 12 (pp. 1063-1075).

[31] Yumo L. An open-source and portable mLOps pipeline for continuous training and continuous deployment.

[32] David O, Ascough II JC, Lloyd W, Green TR, Rojas KW, Leavesley GH, Ahuja LR. A software engineering perspective on environmental modeling framework design: The Object Modeling System. Environmental Modelling & Software. 2013 Jan 1;39:201-13.

[33] Ijiga OM, Idoko IP, Ebiega GI, Olajide FI, Olatunde TI, Ukaegbu C. Harnessing adversarial machine learning for advanced threat detection: AI-driven strategies in cybersecurity risk assessment and fraud prevention. J. Sci. Technol. 2024;11:001-24.

[34] Oyewole AT, Oguejiofor BB, Eneh NE, Akpuokwe CU, Bakare SS. Data privacy laws and their impact on financial technology companies: a review. Computer Science & IT Research Journal. 2024 Mar 18;5(3):628-50.

[35] Örnell P. Security Assessment of Continuous Deployment Pipelines.

[36] Volker L. Deciding about design quality: Value judgements and decision making in the selection of architects by public clients under European tendering regulations. Sidestone Press; 2010.

[37] Adanigbo OS, Adekunle BI, Ogbuefi E, Odofin OT, Agboola OA, Kisina D. Implementing Zero Trust Security in Multi-Cloud Microservices Platforms: A Review and Architectural Framework. ecosystems.;13:14.

[38] Shokri R, Shmatikov V. Privacy-preserving deep learning. InProceedings of the 22nd ACM SIGSAC conference on computer and communications security 2015 Oct 12 (pp. 1310-1321).

[39] Kolawole AF, Rahmon SO, Akinnagbe O. Designing secure data pipelines for medical billing fraud detection using homomorphic encryption and federated learning. Int. J. Sci. Res. Arch. 2024;10:1210-22.

[40] Ikemefuna CD, Okusi O, Iwuh AC, Yusuf S. Adaptive fraud detection systems: Using ML to identify and respond to evolving financial threats. International Research Journal of Modernization in Engineering. 2024;6:2077-92.

[41] Behere SS. AI-Enabled Third-Party Risk Management: Advancing Governance In Digital Ecosystems. Journal of International Crisis and Risk Communication Research. 2025;8(S12):10.

[42] Shaikh TA, Rasool T, Verma P, Mir WA. A fundamental overview of ensemble deep learning models and applications: systematic literature and state of the art. Annals of Operations Research. 2024 Dec 24:1-77.

[43] High Point NC. Optimizing Data Management Pipelines With Artificial Intelligence Challenges And Opportunities. Journal of Computational Analysis and Applications. 2024;33(8).

[44] Vankayalapati RK. Cost optimization in hybrid cloud. The Synergy Between Public and Private Clouds in Hybrid Infrastructure Models: Real-World Case Studies and Best Practices. 2025 Jan 10:93.

[45] Owen A, Samson E. Validating AI/ML-Based Systems: Frameworks for Testing Model Accuracy, Fairness, Robustness, and Generalization in QA.

[46] Sengupta M. Service-Oriented Financial Architecture: A Framework for Modern Financial System Design. Journal Of Engineering And Computer Sciences. 2025 Sep 21;4(9):445-53.

[47] Bolormaa S. Scalable Computational Frameworks for Big Data Processing in Multi-Cloud Environments. American International Journal of Computer Science and Technology. 2024 Nov 6;6(6):13-24.

[48] del Rey S, Martínez-Fernández S, Cruz L, Franch X. Do DL models and training environments have an impact on energy consumption?. In2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) 2023 Sep 6 (pp. 150-158). IEEE.

[49] Hasan MM. Federated Learning Models for Privacy-Preserving AI In Enterprise Decision Systems. International Journal of Business and Economics Insights. 2025 Sep 15;5(3):238-69.

[50] Mutemi A, Bacao F. E-commerce fraud detection based on machine learning techniques: Systematic literature review. Big Data Mining and Analytics. 2024 Apr 22;7(2):419-44.

[51] Okunola OA, Adebayo OS, Favour-Bethy TA, Otasowie O. Dynamic Resilience in Credit Card Fraud Detection: The Adaptive Accuracy Weighted Ensemble Approach.

[52] Chitraju Gopal Varma S. From Experimentation to Production: Streamlining the ML Model Lifecycle for Faster Insights. From Experimentation to Production: Streamlining the ML Model Lifecycle for Faster Insights (November 23, 2024). 2024 Nov 23.

[53] Somanathan S. Data Science in Multi-Cloud Governance: Insights for Security, Scalability, and Risk Mitigation.

[54] Ranabahu AH. Abstraction driven application and data portability in cloud computing.

[55] Marie-Magdelaine N. *Observability and resources managements in cloud-native environnements* (Doctoral dissertation, Université de Bordeaux).

[56] Davenport T, Malone K. Deployment as a critical business data science discipline. Harvard Data Science Review. 2021 Feb 10;3(1).

[57] Kaul D. Optimizing resource allocation in multi-cloud environments with artificial intelligence: Balancing cost, performance, and security. JICET. 2019;4:1-25.

[58] Sundaramurthy SK, Ravichandran N, Inaganti AC, Muppalaneni R. The future of enterprise automation: Integrating AI in cybersecurity, cloud operations, and workforce analytics. Artificial Intelligence and Machine Learning Review. 2022 Apr 6;3(2):1-5.

[59] Oloruntoba O. Architecting Resilient Multi-Cloud Database Systems: Distributed Ledger Technology, Fault Tolerance, and Cross-Platform Synchronization. International Journal of Research Publication and Reviews. 2025;6(2):2358-76.

[60] Raj P, Raman A. Multi-cloud management: Technologies, tools, and techniques. InSoftware-defined cloud centers: Operational and management technologies and tools 2018 May 5 (pp. 219-240). Cham: Springer International Publishing.

[61] Yedalla J. The dark side of AI: How cybercriminals are leveraging machine learning for attacks in Multi-Cloud hosted applications. International Journal for Multidisciplinary Research. 2025;7(2).

[62] Arul K. Data Engineering Challenges in Multi-cloud Environments: Strategies for Efficient Big Data Integration and Analytics. International Journal of Scientific Research and Management (IJSRM). 2023;10(6).

[63] Umakor MF, Iheanyi IK, Ofurum UD, Ibecheozor UH, Adeyefa EA. Federated learning for privacy-preserving fraud detection in digital banking: balancing algorithmic performance, privacy, and regulatory compliance. Iconic Res Eng J. 2025 Jul;9(1):215-31.

[64] Lin WT, Chen G, Li H. Carbon-aware load balance control of data centers with renewable generations. IEEE Transactions on Cloud Computing. 2022 Feb 11;11(2):1111-21.