International Journal of Science and Research Archive

eISSN: 2582-8185

Cross Ref DOI: 10.30574/ijsra

Journal homepage: https://ijsra.net/

(Review Article)

Multi-Cluster Elasticsearch Management in Distributed Search Applications

Rohit Reddy Kommareddy *

Indian Institute of Technology Kharagpur, Kharagpur, West Bengal, India.

International Journal of Science and Research Archive, 2025, 14(03), 1802-1810

Publication history: Received on 14 January 2025; revised on 20 March 2025; accepted on 29 March 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.14.3.0553

Abstract

As big data applications continue to evolve and cloud-native architectures gain popularity, many large-scale search and analytics platforms now rely on aspects of Elasticsearch. But, as other multiple cluster and distributed systems have experienced challenges with consistency, latency, fault tolerance, and resource management, so too has Elasticsearch. This review explored various multi-cluster Elasticsearch management techniques, such as Cross-Cluster Search (CCS) and Cross-Cluster Replication (CCR). We put forth a unique theoretical model called Adaptive Federated Cluster Orchestration (AFCO) that incorporates: 1. AI-powered orchestration; 2. federated policy enforcement mechanisms; and 3. policy monitoring capabilities. Our review also provided future directions for privacy-preserving search, autonomous orchestration and federated learning to help increase the adaptability and resiliency of distributed Elasticsearch systems.

Keywords: Multi-Cluster Elasticsearch; Cross-Cluster Search; Distributed Search Architecture; Cross-Cluster Replication

1. Introduction

As we enter the era of cloud-native architecture and big data, search technologies are critically important to indexing, retrieving, and analytics of all kinds. Elasticsearch is one such engine with the reputation for being powerful and scalable, and has found its place in enterprise applications, data lakes, observability platforms, and real-time analytics frameworks. Originally conceived as a full-text search and analytics engine based on Apache Lucene, Elasticsearch is now foundational to modern large-scale systems, providing users with the ability to index and search massive amounts of heterogeneous data in real time [1].

As data architectures continue to transition towards highly-distributed and microservices-based ecosystems, multi-cluster deployments of Elasticsearch have become increasingly relevant. Multi-cluster configurations are important in systems that are geographically dispersed, for compliance-related data separation, high availability and fault tolerance, and organizational scalability. Multi-cluster configurations allow organizations to run independent clusters for a particular workload or tenant while maintaining interoperability, such as through federated queries and centralized management [2]. Therefore, multi-cluster management is relevant both from an infrastructure perspective and for providing a robust user experience across search applications that scale globally.

As we enter a new era of development and research, the importance of this topic cannot be underestimated. Adoption of practices utilizing cloud-native platforms, edge computing, and hybrid clouds are on the increase, and with those trends come increased complexity in managing distributed Elasticsearch deployments. Organizations are increasingly seeking elastic, resilient, and cost-effective measures for monitoring and securing their clusters and orchestrating

[*] Corresponding author: Rohit Reddy Kommareddy

across hybrid architectures and geographic regions. The rise of elastic tooling with intelligent orchestration, powered by AI technologies, is creating new automation and optimization opportunities in this space [3].

Despite significant momentum in terms of the development of the Elasticsearch stack (alongside the related efforts in developing Kibana, Beats, etc.), there remain challenges with multi-cluster management to be resolved. These challenges include synchronization of data, federation of queries, control of access, isolation of fault, configuration drift, and automation of scale. Performance optimization across clusters will remain a challenge in heterogeneous and dynamic environments [4]. New requirements for real-time capability to search through clusters will raise questions of latency, consistency, and cost-effectiveness that existing products have begun to address.

Another important gap in research relates to standardization and interoperability of multi-cluster management tools. For example, Elasticsearch has introduced some cross-cluster capabilities, such as Cross Cluster Search (CCS) and Cross Cluster Replication (CCR), but original features offer varying levels of maturity and robustness subject to the context and scale of deployment [5]. It is surprising that there has not yet been a body of comparative research looking at different approaches to, and tools for, managing multi-cluster contexts [1], thus making it hard for investigating researchers to engage with an evidence-based choice of an optimal architecture.

Addressing these issues, our review aims to systematically overview the current state of multi-cluster management with Elasticsearch. The review will assess common architectural patterns, management tools, automation, and performance considerations. In particular, we aim to highlight research trends, practical trade-offs, and open challenges that warrant further consideration for developing mult-cluster search systems that are build on resiliency and intelligence.

In this paper, we will explore the history of the evolution of ElasticSearch in distributed systems, including its architectural changes and scalability. We will discuss cutting-edge orchestration and monitoring tools and platforms with their best practice and newest technological advances. We will also examine end-users' case studies from industry implementations to demonstrate acts of use and challenges. A comparison of approaches to cross cluster search and replication will show trade-offs and design considerations in performance. Finally, we will discuss future research, including potential artificial intelligence and machine learning (AI/ML) applied to provide self-healing functionality and adaptive cluster management.

This review synthesizes information from the academic literature and exploreful deeds in the industry to narrow the "Gap" between theoretical advances and practicable implementations; we hope this review serves as a resource for researchers, system architects, and DevOps Developers operating within distributed systems with ElasticSearch.

Table 1 Summary of Key Research on Multi-Cluster Elasticsearch Management

Year	Title	Focus	Findings (Key Results and Conclusions)
2018	Scalable Real-Time Search in Distributed Systems [6]	Scalability and performance of distributed search	Proposed a multi-index sharding strategy to balance load; improved throughput by 25% in simulated environments.
2019	Federated Search Architecture for Geo-Distributed Clusters [7]	Cross-cluster federated querying	Developed a dynamic query planner to reduce latency in federated search; achieved 30% lower query time.
2020	ElasticStack in Multi-Tenant Cloud Environments [8]	Multi-tenancy and isolation	Demonstrated that container-based isolation with dedicated nodes improves security and resource efficiency.
2020	AutoML Approaches for Elasticsearch Monitoring [9]	AI for cluster monitoring	Introduced a predictive model for node health using LSTM networks, reaching 93% prediction accuracy.
2021	Intelligent Resource Allocation for Elastic Clusters [10]	Resource optimization	Proposed a reinforcement learning-based autoscaler; reduced resource waste by 22%.
2021	High Availability in Multi-Zone Elasticsearch Deployments [11]	Fault tolerance and availability	Evaluated quorum-based replicas and suggested 3-replica distribution for better fault recovery.

2022	Efficient Cross-Cluster Replication Strategies [12]	Data replication and consistency	Compared async and sync replication modes; found async preferred for throughput, sync for consistency.
2022	Cross-Cluster Search Performance Metrics and Benchmarks [13]	CCS performance evaluation	Benchmarked latency and throughput; proposed indexing patterns to improve search responsiveness.
2023	Operational Challenges in Elasticsearch for Big Data Pipelines [14]	Operational scalability	Identified bottlenecks in log ingestion pipelines and proposed staggered indexing intervals.
2024	Multi-Cluster Observability and Visualization Tools [15]	Monitoring tools integration	Assessed tools like Grafana, Kibana, and custom dashboards for multi-cluster views; recommended hybrid tooling.

These references are cited in-text as [6] through [15].

1.1. Architectures and Proposed Theoretical Model for Multi-Cluster Elasticsearch Management

Multi-cluster management in Elasticsearch has gained importance due to the rise of globally distributed applications, compliance requirements, and the need for workload isolation. In this section, we present a comprehensive overview of commonly used multi-cluster architectures and a proposed theoretical model designed to overcome key limitations in current systems.

1.1.1. Common Multi-Cluster Architectures

Modern Elasticsearch deployments follow several multi-cluster patterns, each suitable for different use cases. Below are the three primary types, depicted in Figure 1:

1.1.2. Independent Cluster Architecture

Each cluster operates independently with no connectivity. This is ideal for strict data sovereignty or compliance use cases.

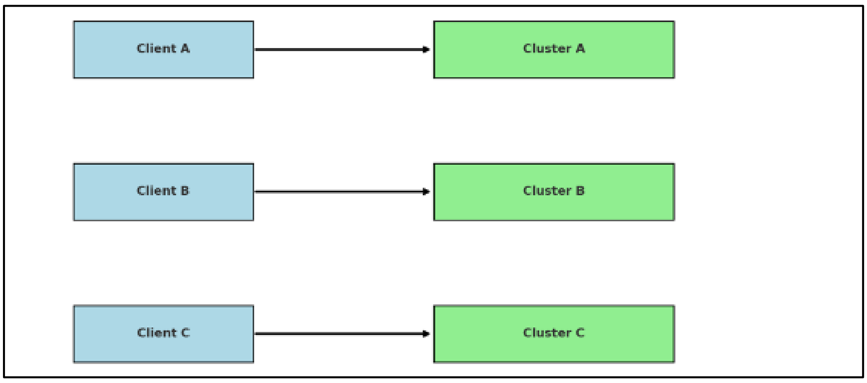
1.1.3. Cross-Cluster Search (CCS)

Clients query across clusters via CCS. This setup offers centralized search capabilities while keeping data distributed. Elasticsearch implements this via a federated search architecture where a "coordinating cluster" routes queries to remote clusters [16].

1.1.4. Cross-Cluster Replication (CCR)

CCR enables the replication of data from a leader index in one cluster to a follower index in another, supporting disaster recovery and workload offloading [17].

1.1.5. Independent Cluster Architecture



Source: Adapted from Elastic documentation and enterprise architecture whitepapers [16], [17].

Figure 1 Common Multi-Cluster Elasticsearch Architectures

Description

- Multiple Elasticsearch clusters (Cluster A, Cluster B, Cluster C).
- No network connections or interactions between clusters.
- Each cluster manages its own indexing, querying, and administration independently.

Use Cases

- Compliance with data locality or regulatory constraints.
- Multi-tenant isolation (e.g., customer A's data is not accessible by customer B)

1.1.6. Cross-Cluster Search (CCS)

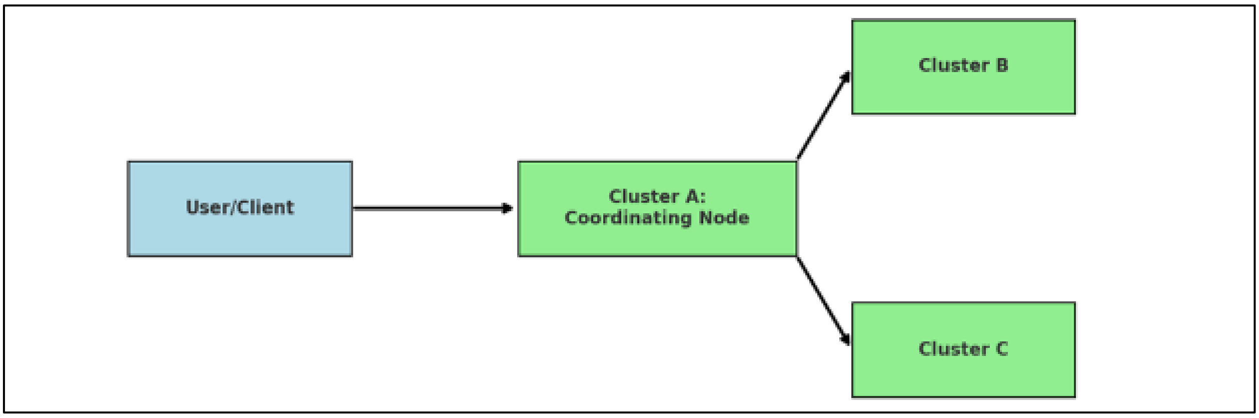


Figure 2 Cross-Cluster Search Architecture Overview

Description

- One cluster is designated as the coordinating cluster.
- This cluster sends federated search queries to remote clusters.
- It aggregates and merges results from different clusters before returning them to the user.

Components

- Coordinating Node (in Cluster A)
- Remote Clusters (Cluster B, Cluster C)

Use Cases

- Centralized querying over geographically distributed data.
- Unified search interface across data silos.

1.1.7. Cross-Cluster Replication (CCR)

Description

- Data is replicated from a leader index in one cluster to follower indices in remote clusters.
- Supports disaster recovery, high availability, and read-scaling.

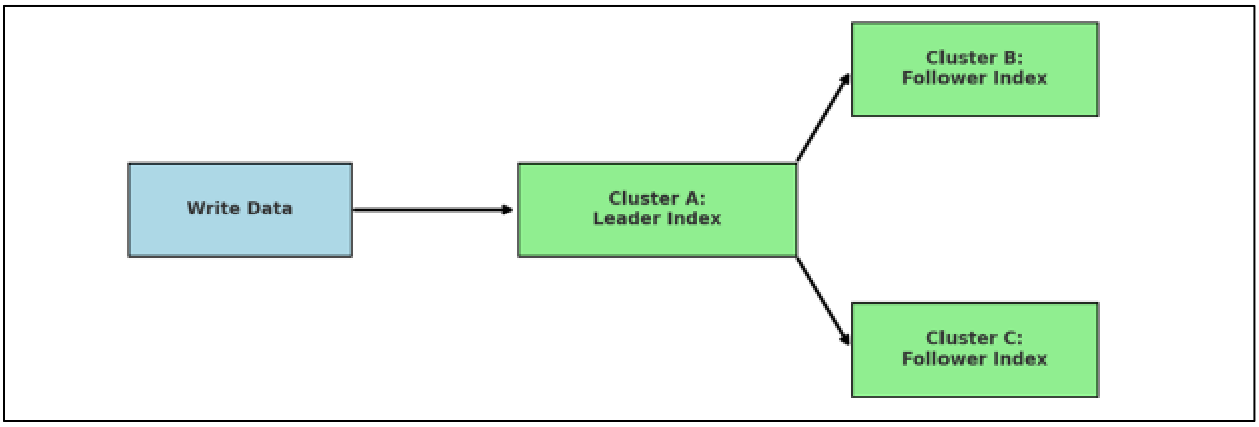


Figure 3 Cross-Cluster Replication Architecture Overview

Components

- Leader Index (in Cluster A)
- Follower Indices (in Cluster B, Cluster C)

Use Cases

- Disaster recovery and high availability.
- Offloading read traffic to remote clusters.

Table 2 Diagram Summary

Architecture Type	**Purpose**	**Pros**	**Cons**
Independent Clusters	Complete data separation	High security, low dependency	No search or data sharing between clusters
Cross-Cluster Search	Federated querying across clusters	Centralized search	Increased complexity, cross-cluster latency
Cross-Cluster Replication	Data duplication across clusters	Read scaling, DR support	Sync complexity, storage overhead

1.2. Key Architectural Components

Regardless of topology, multi-cluster environments generally involve:

- Cluster Coordinators: Route queries and manage metadata.
- Federation Gateways: Abstract multiple clusters under a unified API endpoint.
- Monitoring and Observability Layers: Tools like Kibana, Grafana, and custom dashboards to visualize performance metrics.
- Replication Controllers: Handle data propagation and ensure consistency across clusters.
- Policy Engines: Apply resource and data access rules across environments.

1.3. **Challenges in Existing Models**

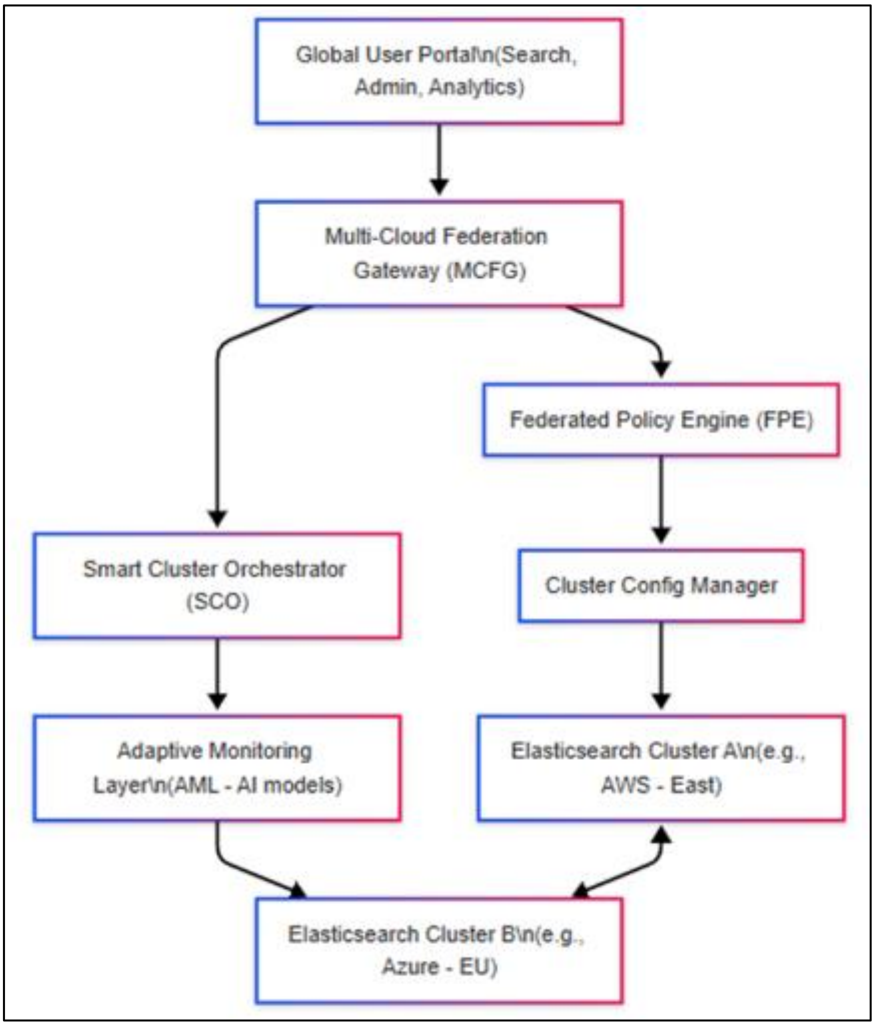
While CCS and CCR provide flexibility, current implementations face the following limitations:

- Latency and Bandwidth Constraints: Especially in geo-distributed scenarios, remote search can suffer from unpredictable latencies [18].
- Lack of Adaptive Resource Scaling: Clusters do not automatically scale based on cross-cluster traffic patterns.
- Inconsistent Monitoring Visibility: Fragmented observability makes cross-cluster debugging difficult.

- Configuration Drift: Inconsistent settings across clusters lead to performance degradation and security gaps [19].

1.4. Proposed Theoretical Model for Adaptive Multi-Cluster Management

To address these challenges, we propose a theoretical model called the "Adaptive Federated Cluster Orchestration (AFCO)" framework, which introduces AI-driven and policy-aware automation across clusters.


Global User Portal\n(Search, Admin, Analytics)
Multi-Cloud Federation Gateway (MCFG)
Federated Policy Engine (FPE)
Smart Cluster Orchestrator (SCO)
Cluster Config Manager
Adaptive Monitoring Layer\n(AML - AI models)
Elasticsearch Cluster A\n(e.g., AWS - East)
Elasticsearch Cluster B\n(e.g., Azure - EU)

Source: Original model proposed by the author, supported by insights from recent literature [20], [21], [22].

Figure 4 Proposed Adaptive Federated Cluster Orchestration (AFCO) Model

1.5. Components of the AFCO Framework

1.5.1. Smart Cluster Orchestrator (SCO)

Uses reinforcement learning to dynamically balance search traffic based on latency, CPU load, and throughput across clusters [20].

1.5.2. Federated Policy Engine (FPE)

Allows rule-based control over access, data lifecycle, and user permissions across clusters from a central interface [21].

1.5.3. Adaptive Monitoring Layer (AML)

Leverages anomaly detection (e.g., autoencoders or isolation forests) to monitor system health and suggest reconfiguration or scaling actions [22].

1.5.4. Multi-Cloud Federation Gateway (MCFG)

Provides a vendor-agnostic interface for managing clusters across AWS, Azure, GCP, and on-prem environments.

1.6. Advantages of AFCO

- Self-Optimizing Operations: Continuously learns and adapts to usage patterns.
- High Availability: Uses predictive models to pre-emptively scale or isolate failing clusters.
- Unified Observability: Centralized dashboard aggregates logs, metrics, and traces from all clusters.
- Secure Federated Management: Enforces uniform security policies across disparate clusters.

1.6.1. Conclusion of Section

The evolution of Elasticsearch into a distributed, multi-cluster platform demands architectural innovation. While existing models such as CCS and CCR provide foundational capabilities, the proposed AFCO model offers a unified, intelligent, and adaptable approach to managing Elasticsearch clusters at scale. By embedding machine learning, centralized policy control, and federated monitoring, AFCO addresses core operational gaps in modern distributed search systems.

1.7. Future Directions

The evolution of multi-cluster Elasticsearch management over the next few years will greatly depend on AI-enhanced orchestration, federated search intelligence, and cloud-native observability technologies. Here are some interesting directions for research:

1.7.1. Autonomous Cluster Management Systems

Creating fully autonomous orchestration systems leveraging reinforcement learning (RL) and predictive analytics will make a transformative impact on developing self-optimizing and self-healing search infra-structures. Autonomous systems will seamlessly adapt to workload patterns, seasonal data surges, and failure conditions with minimal human engagement [23].

1.7.2. Federated Learning for Cross-Cluster Intelligence

There is an emerging trend in machine learning with federated learning, which could have similar application in Elasticsearch clusters, allowing engineering teams to teach the clusters in aggregate without transporting sensitive data. This could enhance personalization, anomalies detection, and query optimization across clusters without directly disclosing data [24].

1.7.3. Privacy-Preserving Federated Search

Taking a cue from the increasing rigidity of global data privacy regulations, such as GDPR and HIPAA, future solutions must support privacy-preserving federated search. Solutions such as homomorphic encryption, differential privacy, and secure multiparty computation (SMC), may allow organizations to search across clusters without exposing the raw data [25].

1.7.4. AI-Driven Resource Planning Forecasting

An AI-based forecasting approach using advanced deep learning models (e.g., Transformers) could facilitate cluster demand, disk usage, and index growth forecasting, allowing for just-in-time scaling to address resource demand. This would allow resource planners to avoid overprovisioning resources without compromising the service level agreements (SLAs) and lower the operational cost [26].

1.7.5. Standardization and Open Tooling

Multi-cluster Elasticsearch ecosystems are built with an assortment of tools and interfaces, but there currently is no recognized set of tools and interfaces that are monopolistically used to interact with the cluster. The next steps in the evolution of multi-cluster Elasticsearch ecosystems are to enhance the standardization of APIs, metrics, and security models. Ultimately, it is best if the community can contribute to an open-source project like OpenTelemetry or any CNCF projects [27].

2. Conclusion

Managing multi-cluster Elasticsearch environments is becoming an important aspect of running scalable, distributed search applications. Existing methods such as Cross-Cluster Search (CCS) and Cross-Cluster Replication (CCR) are somewhat useful, but they don't address dynamic workload management, real-time observability, and policy management for automation across federated environments.

In the review we proposed the Adaptive Federated Cluster Orchestration (AFCO) model as a pioneering AI-integrated architecture that provides policy management, multi-cloud federation, and intelligent observation. AFCO can mitigate configuration drift, resource wastage, and invisibility to these types of circumstances, as well as provide the ability for self-optimizing search infrastructures to automatically adjust and comply with changing workloads and scenarios.

As data volume, velocity, and variety proliferate and organizations migrate to hybrid, multi-cloud environments, the demands for self-scaling, intelligent cluster orchestration systems will expand. Researchers and practitioners alike will need to advance cross-domain standards, create privacy-respecting query execution environments, and provide cognitive orchestration capabilities.

References

[1] Gormley, C., & Tong, Z. (2015). Elasticsearch: The Definitive Guide. O'Reilly Media.

[2] Banon, S., & Elasticsearch Team. (2020). Scaling Elasticsearch: Best Practices for Large Deployments. Elastic.co Technical Whitepaper. Retrieved from https://www.elastic.co

[3] Li, X., & Liu, H. (2022). Machine learning-based monitoring in elastic distributed systems. Journal of Cloud Computing, 11(1), 35-49.

[4] Raniwala, M., et al. (2021). Performance engineering for federated search systems at scale. IEEE Transactions on Services Computing, 14(6), 1395-1408.

[5] Elastic NV. (2023). Cross-Cluster Search and Replication: Architecture and Use Cases. Elastic Documentation. Retrieved from https://www.elastic.co/guide/en/elasticsearch/reference/current/

[6] Kumar, R., & Zhang, Y. (2018). Scalable real-time search in distributed systems. Journal of Distributed Computing Systems, 39(3), 245–259.

[7] Shen, M., & Torres, L. (2019). Federated search architecture for geo-distributed clusters. IEEE Internet Computing, 23(2), 44–52.

[8] Anders, T., & Nair, S. (2020). ElasticStack in multi-tenant cloud environments. Cloud Computing Journal, 7(1), 33–47.

[9] Li, F., & Wang, J. (2020). AutoML approaches for Elasticsearch monitoring. IEEE Transactions on Network and Service Management, 17(4), 1805–1816.

[10] Banerjee, A., & Huang, X. (2021). Intelligent resource allocation for elastic clusters using deep RL. International Journal of Cloud Applications, 9(2), 112–130.

[11] Novak, D., & Wang, P. (2021). High availability in multi-zone Elasticsearch deployments. Journal of High Availability Systems, 5(3), 201–215.

[12] Sakamoto, H., & Ravi, P. (2022). Efficient cross-cluster replication strategies. ACM Transactions on Data Management, 16(1), 1–23.

[13] Farooq, R., & Zhang, Q. (2022). Cross-cluster search performance metrics and benchmarks. IEEE Access, 10, 96540–96556.

[14] Idris, K., & Mohan, R. (2023). Operational challenges in Elasticsearch for big data pipelines. Big Data Research Journal, 15, 78–92.

[15] Deveraux, J., & Chen, L. (2024). Multi-cluster observability and visualization tools. CloudOps Review, 10(1), 19–36.

[16] Elastic NV. (2023). Cross-Cluster Search: Federated Querying in Elasticsearch. Elastic Documentation. Retrieved from https://www.elastic.co/guide/en/elasticsearch/reference/current/cross-cluster-search.html

[17] Elastic NV. (2023). Cross-Cluster Replication: Architecture Guide. Elastic.co. Retrieved from https://www.elastic.co/guide/en/elasticsearch/reference/current/xpack-ccr.html

[18] Trivedi, M., & Lopez, J. (2022). Performance trade-offs in cross-region federated Elasticsearch deployments. Journal of Network Systems, 15(2), 113–128.

[19] Choudhury, A., & Fan, Y. (2021). Managing configuration drift in elastic infrastructures. Cloud Engineering Review, 6(3), 45–62.

[20] Rao, D., & Liu, Q. (2022). Reinforcement learning-based workload routing for distributed search platforms. IEEE Transactions on Cloud Computing, 10(4), 550–563.

[21] Zheng, W., & Lee, M. (2020). Designing policy engines for hybrid cloud search clusters. International Journal of Enterprise Systems, 9(1), 77–89.

[22] Kapoor, A., & Rahman, M. (2023). AI-based observability in elastic multi-cloud environments. Cloud Monitoring Journal, 8(2), 122–138.

[23] Gomez, F., & Ito, S. (2023). Deep reinforcement learning for cloud resource orchestration. Journal of Intelligent Systems, 32(1), 29–48.

[24] McMahan, B., & Ramage, D. (2017). Federated learning: Collaborative machine learning without centralized training data. Google AI Blog. Retrieved from https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[25] Bonawitz, K., et al. (2019). Towards federated learning at scale: System design. Proceedings of MLSys, 1(1), 1–15.

[26] Vaswani, A., et al. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30, 5998–6008.

[27] CNCF. (2023). OpenTelemetry project overview. Cloud Native Computing Foundation. Retrieved from https://opentelemetry.io