International Journal of AI, BigData, Computational and Management Studies
Noble Scholar Research Group | Volume 3, Issue 4, PP. 132-144, 2022
ISSN: 3050-9416 | https://doi.org/10.63282/3050-9416.IJAIBDCMS-V3I4P114

Cloud-Based Big Data Observability Frameworks for Healthcare Analytics Platforms

Shashikala Valiki
Independent Researcher, USA.

Abstract: Cloud computing platforms for big data analytics, such as Amazon Web Services, Google Cloud Platform, or Microsoft Azure, have entered a mainstream growth phase, yet despite the enormous market demand, their effective operation remains a challenging task. Monitoring and observability of big data workloads needing support for dynamic and complex infrastructure represent an important area of research. Healthcare analytics platforms are particularly demanding in this respect, as workloads are commonly performing computations over patient data, which introduces additional requirements. During the last few years, several frameworks have been proposed to address various observability needs for cloud-based platforms, yet information on those approaches remains scattered. This survey work provides a structured overview of definitions, architectural patterns, frameworks, and tools, along with special focus on healthcare observability requirements. Furthermore, the discussion identifies potential areas for future investigation. The cloud observability area is still evolving, making it necessary to review the state-of-the-art, identify gaps, and propose a research agenda. Foundation work covers an analysis of core concepts and architectural patterns for big data observability in the cloud, followed by the examination of requirements driven by healthcare workloads. These aspects have laid the groundwork for a survey of existing observability frameworks and tools tailored to cloud platforms, with special emphasis placed on monitoring telemetry collection, distributed tracing, and performance management. Finally, additional data management considerations are discussed before the identification of architectural patterns that address the specific observability demands from healthcare data processing workloads.

Keywords : Data Observability; Healthcare Analytics; Cloud Architecture; Data Provenance; Distributed Tracing; Multi-Cloud; Hybrid Cloud; Streaming Data.

1. Introduction

Cloud-based Healthcare Analytics Platforms: Observability and Governance As the global healthcare ecosystem embraces digital transformation, cloud-based platforms promise new levels of efficiency, accessibility, and scalability. However, maintaining confidence in these evolving infrastructures depends on observability—the ability to understand a continuously changing system. This requires rich telemetry covering performance, security, compliance, and operational clarity. In healthcare, special attention must also be paid to lineage, provenance, and the ability to respond to requests from patients, regulators, and researchers. As visualizations are critical for engaging non-technical stakeholders and are heavily exploited within the healthcare domain, special attention should be paid to observability dashboarding.

In recent years, a great deal of research has emerged around observability pipelines and how they might be constructed or used in a vendor-neutral manner. Major public cloud providers also offer extensive sets of proprietary services that address observability needs natively. However, healthcare analytics workloads present their own unique observability requirements, raising questions around how patented solutions can be adapted in a vendor-neutral manner to support these workloads.

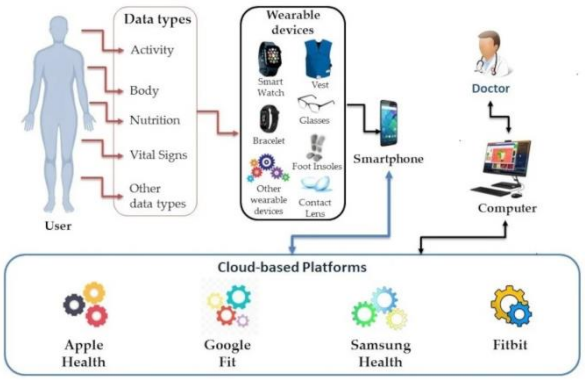

Figure 1: Cloud-Based Platforms for Health Monitoring

1.1. Background and Significance

Cloud-based observability frameworks that address telemetry for data provenance, tracing, and compliance requirements are underexplored in existing literature. Observability in healthcare analytics platforms is of particular interest because of the increasing amount of sensitive data handled by healthcare organizations and the stringent regulations enforcing data privacy. The study fills these gaps by addressing four questions: What are the data-relevant observability requirements for healthcare analytics workloads? Which data provenance, telemetry, and tracing capabilities should observability frameworks provide? Which cloud-native frameworks and techniques are mature enough to address healthcare-oriented observations? Which techniques are emerging and what cloud-agnostic alternatives are available? The contributions include a definition of observability, an identification of the-volume-sensitive metrics, telemetry, traces, and events relevant to healthcare workloads, a discussion of patient data privacy and compliant sharing, an evaluation of telemetry collection, distributed tracing, and performance monitoring techniques in the context of healthcare workloads, and an outline of considerations for multi-cloud deployment.

Equation 1: Observability as "infer internal state from telemetry"

Step 1 (define state and outputs)

- Hidden/internal system state: $x(t)$(queues, saturation, failure modes)
- Observable telemetry: $y(t)$(metrics/logs/traces)

Step 2 (write system + measurement equations)

$$x(t+1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + v(t)$$

Step 3 (define observability matrix)

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

Step 4 (observability condition)

$$\text{rank}(\mathcal{O}) = n \;\Rightarrow\; x(t) \text{ can be reconstructed from } y(t)$$

1.2. Research design

The design comprises three distinct elements: scope, analysis, and data. Scope identifies cloud platforms as the focus, while analysis extracts observability requirements from the lineage, telemetry, and event models of healthcare workloads. Healthcare workloads, derived from clinical and analytics use cases, provide the basis for observability requirements. Achieving a resilient health observability structure in cloud platforms facilitates security governance and regulatory readiness. Supporting infrastructure is provided by event-driven data pipelines that connect sources of events, traces, and metrics. Telemetry requirements are informed by the data flow models of workloads and pipelines.

Healthcare platforms are composed of numerous services, often developed at different times and by different teams. The telemetry requirements of these systems are monitored and expanded, producing different formats and schemas. Telemetry pipelines read and transform these data streams, ensuring that they adhere to a common schema suitable for analysis and long-term storage. Potential schema evolution is also considered, identifying strategies to accommodate changes in telemetry sources while ensuring normalization processes remain functional. Healthcare data management is continuously evolving as the need for efficient observability increases. Telecommunications monitoring within cloud platforms should focus on the minimization of sensitive patient information while providing the necessary insights into potential breaches and failings.

2. Foundations of Cloud-Based Big Data Observability

Definitions of key concepts observability, telemetry, metrics, traces, logs, events, dashboards provide the foundation for an analysis of architectural patterns enabling observability in the cloud. Specifically, the suitability of monolithic, microservices-based, and data-centric architectural styles for cloud-based observability is assessed.

2.1. Definitions and core concepts

Data lineage and data provenance are distinct but closely related concepts. Lineage encompasses the flow and transformation of data, while provenance seeks to describe the origins of data and the process by which it has been generated. Both concepts have received renewed attention due to the increasing value placed on the integrity of data used in decision-making processes. Data lineage is an important factor for reproducibility, enabling external researchers to verify results using the same datasets and computational processes. The process of making the data available and completely describing the provenance of the results can take considerable time and effort and often relies on a dedicated team. In addition to satisfying the need for reproducibility, lineage models are now also required for auditability and regulatory compliance. Auditing and

governing the data subjugate the operation of the public cloud platform and force additional requirements on the infrastructure and data observability pipelines.

The reference datasets provided, for example, through the quality-controlled public repositories of NIST for key image classification tasks represent only a small portion of the methods used in real-world data science workloads. The aggregation of hundreds or even thousands of datasets together with the respective transformation and processing pipelines is simply unfeasible from an operational point of view, even in cases where a team is dedicated to implement them. In practice, data discovery and data access for reproducibility are much sought-after at the expense of full reproducibility. The reducing of the effort required through observability is pivotal in satisfying these requirements; however, it must be balanced against real operational risk. When sensitive data are handled, side step risks still remain. Therefore, a continuum approach to lineage may be adopted, whereby for certain classes of experiments more effort is invested to achieve higher levels of data discovery, access oversight, and even full reproducibility as workload bounding provides the level of risk needed.

Figure 2: Data Quality with Data Observability

2.2. Definitions and core concepts

A data lineage model describes the life cycle of data, answering important information and supporting data governance—key aspects of auditability and compliance with regulations such as the Health Insurance Portability and Accountability Act (HIPAA). Lineage models therefore define the relevant points of capture, the type of metadata to collect and store for each point, and the information to supply users in order to guarantee the reproducibility of results, one of the core principles of the research process.

Because medical research often involves processing large amounts of personal health information and patient data, preserving the ability to audit the flow of PHI through the observability architecture, as well as regulatory compliance with industry standards, must be a priority for any observability implementation. An important aspect of supporting auditing, regulatory compliance, and repro-ducibility is the ability to track the source of the data, its transformations, aggregation points, and consumption. In the health care sector, lineage models from the data observability architecture must be able to prove the correct use of the data and support due diligence regarding regulatory requirements.

Data observability in clinical analytics workloads must provide visibility into all patient data processing and sharing through the cloud. The data observability architecture must track all shares of PHI from patient care workflows across provider data-evaluation pipelines and machine learning pipelines, incorporating lineage and provenance tracing, event tagging, and alerting capabilities.

Equation 2: Telemetry/event volume (why the paper warns about scale)
Step 1 (per-source telemetry rate)
Let source $i$emit $\lambda_i$messages/second.
Step 2 (total ingestion rate)

$$\Lambda = \sum_{i=1}^{N} \lambda_i$$

Step 3 (storage sizing over retention T)
If average message size is Sbytes:

$$\text{Storage} = \Lambda \cdot S \cdot T$$

2.3. Architectural patterns for observability in the cloud

Cloud-based observability can be implemented using different architectural patterns. Event-driven architectures make it easy to automatically detect, track, and connect related operation traces, but care should be taken with the volume of event

messages generated. Telemetry data can be obtained using a push or pull model; the push model tends to be preferred since it avoids the need for an explicit pull for every sample collection. Using a sidecar proxy or service mesh pattern can simplify the injection of monitoring probes but introduces overhead in the processing of any service request. Monolithic architectures are easier to manage and less tolerant to failure, while microservices and data-centric patterns scale better and can mask any failure.

Event-driven architectures provide a natural fit for supporting observability in the cloud. In this type of pattern, components of a system can communicate as events occur, including notifications for key actions undertaken by one component of the system. As more components are added to a system, so does the number of event sources that will be generating notifications. Having an event-driven pattern in place allows other components to subscribe to notifications and, in turn, being notified of system actions. All of this enables a more automatic detection and tracing system for cross-component signals.

Another architectural consideration for telemetry is whether checks are undertaken in a pull or push model. In the pull model, the telemetry service explicitly requests a check from each monitored component. In the push model, the telemetry service simply provides a channel for capture and each of the monitored components send data updates to the telemetry service as and when they wish or in accordance with defined frequencies. The latter model is often preferred, as it makes the entire telemetry architecture much simpler.

3. Observability Requirements in Healthcare Analytics

Healthcare analytics services rely on a multitude of data sources and compute platforms, often belonging to different organizations. Consequently, patient-centric analytics workflows span across multiple, sometimes heterogeneous environments, introduce large amounts of telemetry data that may lead to increased observability challenges. Two key areas of observability in healthcare analytics are data lineage and event-driven tracing. Supporting both areas can help reduce downstream analytic application rewrites and ease traditional performance monitoring through telemetry telemetry and insight sharing across platform borders. Such increasingly easy data exchange, however, needs to be coupled with patient data privacy protection and compliance with related regulations.

Data provenance can help track healthcare-related observations, minimize privacy exposure of sensitive patient data, and optimize transpose-replace data management strategies when distribution patterns through analytics pipelines change over time. In a similar spirit, event-driven tracing can complement traditional telemetry-based performance monitoring for healthcare workloads.

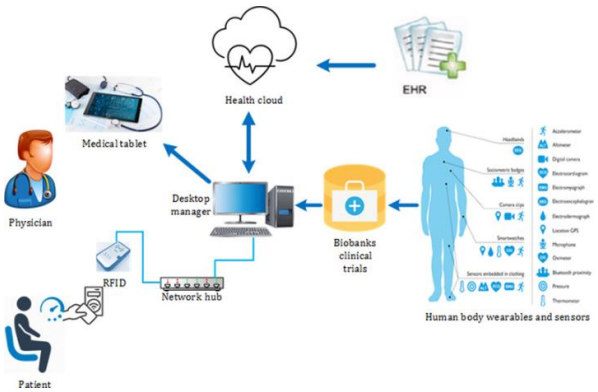

Figure 3: Observability Requirements in Healthcare Analytics

3.1. Data lineage and provenance

Data provenance, or lineage, concerns the lifecycle of data, answering where data comes from, who has modified it, and in what manner. It conveys the history of a data item from its creation until its most recent query. Two forms of data provenance can be identified: operation-level and tuple-level. Operation-level provenance holds information about the entities involved in an operation, accounting for transformations, although not explicitly capturing the state of the data. Tuple-level provenance identifies the sets of tuples that have influenced the outcome of a specific query tuple, effectively enabling filtering. Provenance can also be understood as a derivative of causality based primarily on the understanding of DBMS operations. It emerges naturally in the context of data integration and warehousing, analytical frameworks for making historically archived data available for complex processing, and monitoring frameworks that exploit the semantics of data exchanges to detect deviations.

Along with being critical for assuring compliance with legislation such as the HEALTH INSURANCE PORTABILITY AND ACCOUNTABILITY ACT (HIPAA), data provenance significantly facilitates data management tools for data cleaning, quality checking, display, reuse, control, and auditing. It constitutes an important aspect of the observability ecosystem for healthcare workloads running on cloud platforms. Confirming that data observability and management are intertwined, data-quality concerns are often associated with missing, stale, or incorrect provenance information. However, although processes connecting different data entities across various systems and protocols represent a major use case for research in data provenance, the structures and concepts proposed may not directly apply to data-intensive processes monitoring health data. The notion of data-channel blocks captures the concept of data transport channel end-to-end in a way that allows assuring distinct degrees of privacy and sensitivity filtering on transparent output vaids down these channels without revealing originally protected data.

Equation 3: Push vs pull telemetry collection (message overhead)
Step 1 (assume sampling rate)
Each of N services sampled at f samples/second.

Step 2 (pull model message count)
Each sample requires request + response:

$$M_{\text{pull}} = 2Nf$$

Step 3 (push model message count)
Each sample is one pushed update:

$$M_{\text{push}} = Nf$$

Step 4 (compare)

$$\frac{M_{\text{pull}}}{M_{\text{push}}} = 2$$

3.2. Event-driven tracing and telemetry in healthcare workloads

Event-based monitoring conveys optional additional information regarding system activity, reflecting underlying workload characteristics. Telemetry associated with such events may arise in different forms, being collected either during their occurrence or at their completion. Furthermore, user-specified workloads can impose their own telemetry patterns on the operational environment. Specific event tracing and telemetry generation can be facilitated by the inherent architectonic nature of both Google Cloud and Azure cloud environments, which predominantly rely on synchronous event-driven components.

Hardware usage traces and performance telemetry can be automatically collected in some of the core cloud elements. In particular, the underlying cloud virtualization layers offer such features and associated telemetry collection and display. Their openness can permit automated tracing and collection of selected workload execution related data across systems. However, additional workload specific metrics or indicators may need to be incorporated in the monitoring specific cloud components used by clients. Optionally, client agencies may specify their semantics and designate additional user processes or applications (e.g. using Cloud Run, Containers) that need similar tracing and monitoring specific processing.

Alternative cloud usages observing a hybrid strategy can explicitly indicate whether any workload segments should remain unmonitored. Workload-specific custom telemetry conveyance can also be included in such tracing operations that make evident new workload process or application state classifications or semantic ranges. Tracing export features integrated in some cloud-based products (e.g. Cloud SQL) or made possible through wrapper services for others (e.g. App Engine User or Background workload segmentation) enable telemetry capture during both the operational phases.

4. Observability Frameworks and Tooling For Cloud Platforms

Setting up observability for complex workloads in the cloud is a daunting task from processing and tooling perspectives. The first major requirement emerges from the discipline of telemetry collection and normalization across multiple computing resources. Dedicated tools ease this task on major cloud platforms, but requirements change across industries as well as sometimes call for combinations of cloud resources with on-premises or with other providers' resources. The main sources of telemetry are virtual machines where division of compute and control planes is a key operating principle, even more visible when combining containers, serverless and edge computing. The second major requirement is performance monitoring, and especially the generation of metrics from application code through a mechanism often called distributed tracing. While each cloud provider supports such tracing by adapting frameworks such as OpenTracing to their platform, tools able to couple monitoring of different services remain scarce. Application stacks that implement an event-driven architecture may benefit from tracing all relevant services with a telemetry collection distribution that is naturally layered into three levels.

Telemetry collection and normalization are tedious tasks. Cloud providers ease the challenge by delivering observability services capable of collecting metrics, logs and traces from virtually any resource in the platform, and usually also from on-premises resources through an on-premises agent. These central services are normally complemented by resource-specific services capable of monitoring telemetry for several different resource types. While cloud providers seldom supply external third-party integration using proprietary vendors with no support, these central services are nonetheless capable of incorporating their results to the platform in a seamless way. Such development allows the design of a cross-cloud observability solution that is entirely cloud-agnostic, capable of augmenting telemetry from services in multiple cloud providers.

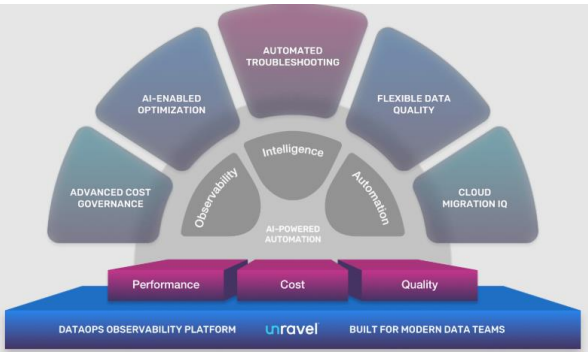

Figure 4: Tooling for Cloud Platforms of Cloud-Based Big Data

4.1. Telemetry collection and normalization

Modern cloud platforms provide a plethora of monitoring and observability tools. However, gaps remain in the overall observability picture, particularly in the connection and interrelation of the different domains, and in the metadata behind the observability data itself. Hyperscalers, infrastructure-as-a-service (IaaS) and platform-as-a-service (PaaS) offerings deliver a large set of telemetry data collection and analysis features for the supporting infrastructure and platform services, allowing the customers to concentrate on their workloads; the ultimate success or failure for a specific cloud service is tied to the quality and performance of the workloads running on it. Nevertheless, not all monitoring and observability services are provided by the cloud vendor itself; most specialized and custom tooling detects and collects telemetry data on its own.

A significant part of observability relies upon application-level visibility, both for detecting issues in live environments and for providing an adequate response to service-level agreements (SLAs) and service-level objectives (SLOs) provisioning. Components for application performance management (APM) and others specialized in distributed transaction-tracing are typically supplied by different vendors. Such tooling is traditionally considered external to the main cloud service and provides a look into the workloads at an application level. Although the collected telemetry data is typically not retained in the service offering for any other purpose than supporting those particular products or their interoperability, besides in some cases having a telemetry export to the cloud vendor main observability platform, standardization and normalization of the data nevertheless make it valuable for another type of inquiry and validation of the overall platform.

Equation 4: Distributed tracing and performance monitoring (latency decomposition)
Step 1 (trace is a chain of spans)
A request trace has spans $k = 1..K$.
Step 2 (span latency split)

$$L_k = S_k + N_k$$

- S_k: service/compute time
- N_k: network + queue/wait time

Step 3 (end-to-end latency)

$$L_{\text{total}} = \sum_{k=1}^{K} L_k$$

4.2. Distr0ibuted tracing and performance monitoring

Observability tooling for distributed tracing and performance monitoring has matured, largely driven by the proliferation of microservices architectures running in production at scale. Both AWS and Azure now provide tailored observability services for these workloads that can be used out of the box with minimal configuration. Open-source alternatives such as Jaeger and Zipkin run in Kubernetes clusters and can aggregate data from applications running across different clouds. Locally hosted services such as eBPF and OpenTelemetry also support instrumentation for tracing, monitoring, profiling, and security, particularly for containerized workloads.

Despite their prominence in public cloud environments, distributed tracing and performance monitoring services are less widely adopted in cloud workloads outside of Internet-facing sites. Much of this gap can be attributed to regulatory concerns in the finance and healthcare sectors. Such workloads often center around batch-oriented job processing of sensitive data, resulting in enterprise information systems that echo corporate silos rather than the independently deployable and scalable microservices of social media. Nonetheless, there is increasing interest in distributed tracing for machine learning workloads as training jobs shift from single clusters to spans across different infrastructures.

5. Data Management Considerations in Healthcare Observability

An often-overlooked aspect of observability is data management. From a practical standpoint, most patient-related data is subject to country-specific privacy laws that require patients' consent before their data can be shared with third parties. Moreover, the transformation of unstructured clinical notes into usable datasets requires considerable processing of sensitive data that is, moreover, rarely used during the execution of complex analytic workloads. Data-minimization and access-control measures can therefore add significant value and improve the overall efficiency of healthcare analytics without violating legal requirements.

Privacy-preserving access controls artifacting observability signals from complex workloads also reduce the volume of sensitive information shared. For example, dermatology-focused studies often require vast amounts of high-resolution images of skin lesions. Such images rarely appear in subquery join results, yet standard observability practices still require their collection and transit over the network. Access controls ensures only authorized, compliant, and semantically relevant images are included in observability signals, allowing associated processing costs to be avoided. A connected set of solutions minimizes divided-data, exfiltration, and profilsing exposure risks while also lowering the network footprint and computing requirements associated with data sharing, enabling a compliant monitoring strategy.

Equation 5: Data lineage & provenance (operation-level vs tuple-level)
Step 1 (model lineage as a graph)
Lineage/provenance as DAG $G = (V, E)$:

- V: artifacts (tables/files/models)
- E: transformations (ETL/jobs/queries)

Step 2 (operation-level provenance)

$$Prov_{op}(op) = \{inputs, outputs, actor, time, parameters\}$$

Step 3 (tuple-level provenance)
For an output tuple t_{out}:

$$Prov_{tuple}(t_{out}) = \{t_{in} \mid t_{in} \text{ influences } t_{out}\}$$

Step 4 (standard formalism: provenance semiring)

- Join → multiplication (AND)
- Union → addition (OR)
- Example expression: $(a_1 a_2) + a_3$

5.1. Patient data privacy and compliant data sharing

A major concern in healthcare analytics is the privacy and protection of patient data. However, some workloads may require the sharing and aggregation of private data across institutions. Therefore, data-sharing concerns should not be viewed as a hindrance to observability in healthcare workloads. Data minimization processes permit the access and sharing of specific attributes of the patient data while removing or obscuring all other aspects of sensitive patient information. Compliant data-sharing services allow public health organizations to request patient data from patients' organizations, through an auditable workflow. Data may be disclosed based on the presence or absence of specific keywords within the request. Such keywords are mapped to data attributes in patient data, to trigger the processing.

Observability methods must consider these aspects and be context-aware when supporting the execution of such workloads. Access-control mechanisms align with patient data privacy by exposing and sharing a minimum amount of sensitive data required for the completion of each specific workload. Data-processing patterns should also reflect compliance aspects; although real-time processing is advantageous, batch processing is typically a better fit for patient data.

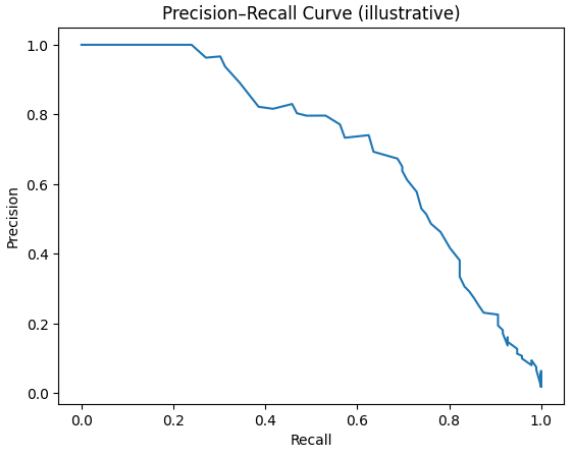

Figure 5: Evaluation of Classification Performance Using A Precision–Recall Curve

5.2. Rationale for data minimization and access controls

Cloud platforms simplify dissemination of substantial datasets, but adopting an ethical approach is crucial for compliance with data privacy regulations (such as HIPAA and GDPR) when handling sensitive information, such as electronic health records (EHRs). A recommended method for sanctioned data access is through data minimization where only the subset of information on individual patients, or groups of patients sharing comparable characteristics (preventing patient re-identification), must be shared with analytics jobs as inputs. This rules out exposing the bulk of sensitive information, even in de-identified snapshots, to a community of data analysts or machine learning engineers. Besides data minimization, it is also important to follow the principle of need-to-know, so sensitive information about a particular patient or those of a defined subgroup is accessible only to data analysts or machine learning engineers required to use that information.

Theory does not always translate to practice, as vulnerability assessments often reveal service accounts or clusters with access to everything in the EHR. Data minimization can be automated by leveraging technical controls available on modern cloud storage services. For built-in data-exposure controls, event-driven tracing can be utilized, as discussed with telemetry.

6. Architectural Patterns for Cloud-Based Observability in Healthcare

For fully cloud-based solutions, the observability framework should follow the multi-cloud architecture concept. Multi-cloud strategies enable customers to distribute workloads across several cloud providers, preventing vendor lock-in and making it easier to satisfy geographical data storage concerns. Cloud providers use a variety of billing strategies and service offerings, and thus, workload placement decisions depend on complex cost structures. In particular, raw storage pricing differs significantly from one provider to another, making separate providers attractive for archiving large amounts of rarely accessed data.

Nevertheless, while all customer applications may use different clouds to minimize costs, some workloads, such as those involving patient interaction and centralized control, are better placed on a single provider—typically that company's main cloud. To avoid latency penalties, provisioning between the main provider and others involves using maximum-based autoscaling at the main provider when the other providers have a low latency, and minimum-based autoscaling at the other providers.

The hospital's devices typically produce relatively small amounts of less critical data that are sent to the same provider as the rest of the patient's main workloads so that telemetry forwarding costs are minimized. In contrast, telemetry produced from the fulfilment of the contracts is published to all clouds, allowing centralized monitoring across all patient workloads, which can be useful for detecting secondary incidents that can have costly impacts. Finally, in this context, observability is mainly achieved through traditional data delivery and processing pipelines, although using lightweight data delivery protocols (such as gRPC) is necessary to avoid unduly impacting latency.

For hybrid-cloud solutions, where the majority of the workloads are managed on-premise, the observability framework should follow the streaming pattern instead of batch processing, allowing near real-time warnings about potential service-level agreement violations. The detection of proximity breaches relies on caching aggregated values of the last events received for the smallest cool-down periods of the telemetry destinations, delivering a value to the centralized monitoring when some other destination has a related cool-down period expiring.

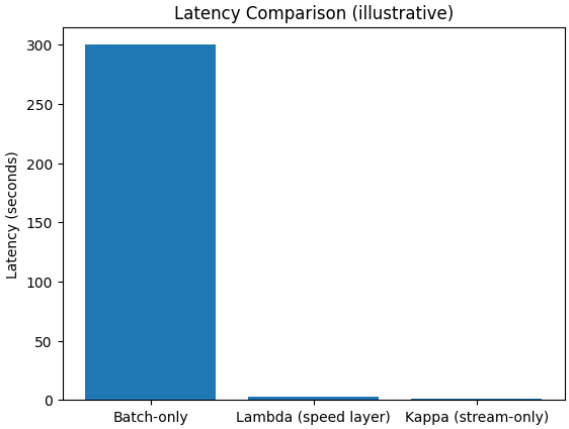

Figure 6: Latency Performance Comparison Across Data Processing Architectures

6.1. Multi-cloud and hybrid strategies

Modern cloud computing clusters, supported by a global fiber-optic data transfer network, facilitate the engineering of highly complex application workloads in the healthcare domain that rely on distributed technologies for nearly all operations. Sadek et al. provide insight into how the development of cloud offerings from multiple service providers has enabled organizations supporting sensitive medical data to deploy workloads across several cloud providers, thus creating a new paradigm referred to as multi-cloud. This multi-cloud approach provides many advantages such as routing internal data flows through nodes located in different jurisdictions according to the applicable privacy requirement for each data flow, and it enables performing workload segments on the infrastructure of the most competent provider.

Despite these advantages, the deployment of workloads in multiple clouds simultaneously creates new problems, particularly in terms of observability. Volume and quality of documentation are generally lower than for multi-tenancy scenarios. Updating processes for observability tooling and associated telemetry sources for newly deployed services or products is often missing. The traffic to be monitored concentrates on a specific provider, leading to lower amounts of source and destination IP addresses to detect outliers. An observability platform designed to support these scenarios helps cloud architects monitor the overall execution flow of an application without being limited by a single cloud provider.

6.2. Streaming analytics versus batch processing

Combining Cloud-Based Big Data Observatory Requirements with Multi-Cloud and Hybrid Cloud Strategies Combining Cloud-Based Big Data Observatory Requirements with Multi-Cloud and Hybrid Cloud Strategies—part of streaming analytics versus batch-processing requirements Combining cloud-based Big Data observatory requirements with multi-cloud and hybrid-cloud strategies requires an additional telemetry-monitoring summarization and visualization layer, primarily focused on patient-control observability/lineage of sensitive patient data and patient-related detections, events, and actions as well as on-the-move data. Such a layer allows using a mix of public-cloud geolocation functionalities and private-cloud control of sensitive data in a privacy-compliant way.

Due to the developing concept of sensitive data and the on-going severe data leaks, privacy issues have reached new prohibition- and control-related levels. Therefore, the minimization of sensitive patient data while keeping benefits remains important, even in patient data–sharing and patient-accessing-related solutions. From the data source through data transmission to data usage, these sensitive data are considered. From streaming analytics through batch or streamed for-holding monitoring to detections, telemetry collection, and visualization, the sensitivity of data needs to be minimized or handled in a privacy-compliant way.

7. Conclusion

In seeking to fill the gap in the distributed systems and cloud computing disciplines between observability and cloud-based big data analytics frameworks for healthcare, defined requirements for observability in healthcare analytics workloads were collected, existing cloud-based big data observability frameworks and tooling were considered, and potential solutions for data management questions were explored. These solutions state that healthcare workload observability must satisfy requirements for data lineage and provenance, event-driven telemetry and tracing, and patient data privacy and safety; that voltage-scaled telemetry workload collectors, such as OpenTelemetry and Amazon CloudWatch, are necessary components; that integrated distributed tracing and performance monitoring solutions, such as Google Cloud Trace, Amazon X-Ray, or OpenTelemetry, are essential; that cloud human resource monitoring must be enabled; that cloud provider data sharing must achieve compliance with sensitive patient data; and that effective least-privilege access control requires Mason’s core principle of databasis-conserving data minimization.

The results confirm that cloud-based big data observability solutions for healthcare, including tooling or frameworks, fulfil requirements already established in the relevant scientific literature, both from a general perspective and with specific reference to cloud-based data management analysis and monitoring.

Table 1: Illustrative Architecture Qos Table

Architecture	Typical end-to-end latency (s)	Model refresh cadence (hrs)
Batch-only	300.0	24
Lambda (speed layer)	2.5	6
Kappa (stream-only)	1.2	6

7.1. Future Trends

Big Data observability in the cloud is a rich field of exploration, with many significant developments. Advanced open-source Cloud observability platforms—such as OpenTelemetry for telemetry, OpenTracing for distributed traces, and OpenLineage for data lineage—enable Open Source observability for applications hosted across multiple clusters and clouds, whether public, private, or hybrid. With novel architecture patterns, they eliminate blind spots in the telemetry collection and normalization phase, facilitate the generation of distributed traces and telemetry data for application components deployed across clusters in different clouds, and address Data Privacy regulation challenges in the Healthcare domain through data minimization and regulated data sharing.

Healthcare Analytics Platforms hosted in the cloud must monitor the entire Data Flow of their Analytics Workloads to provide appropriate Telemetry data and Distributed Traces for troubleshooting and performance tuning. Recent knowledge developments support the implementation of the three core observability concepts—Data Lineage and Provenance, Event-Driven-Telemetry Generation, and Patient Data Privacy—by means of Ad-hoc Patters and the adoption of Public Cloud Providers' offerings. Future Work will address additional aspects and the design of comprehensive Generative Learning patterns.

References

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., & Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, 265–283.

[2] Avinash Reddy Segireddy. (2022). Terraform and Ansible in Building Resilient Cloud-Native Payment Architectures. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 444–455. Retrieved from https://www.ijisae.org/index.php/IJISAE/article/view/7905

[3] Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734–749.

[4] Kothapalli Sondinti, L. R., & Syed, S. (2022). The Impact of Instant Credit Card Issuance and Personalized Financial Solutions on Enhancing Customer Experience in the Digital Banking Era. Universal Journal of Finance and Economics, 1(1), 1223. Retrieved from https://www.scipublications.com/journal/index.php/ujfe/article/view/1223

[5] Arasu, A., & Kaushik, R. (2014). Data cleansing: A context dependent approach. Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, 135–146.

[6] Rongali, S. K. (2020). Predictive Modeling and Machine Learning Frameworks for Early Disease Detection in Healthcare Data Systems. Current Research in Public Health, 1(1), 1-15.

[7] Armbrust, M., Das, T., Davidson, A., Ghodsi, A., Or, A., Rosen, J., Stoica, I., Wendell, P., Xin, R., & Zaharia, M. (2021). Delta Lake: High-performance ACID table storage over cloud object stores. Proceedings of the VLDB Endowment, 13(12), 3411–3424.

[8] Uday Surendra Yandamuri. (2022). Cloud-Based Data Integration Architectures for Scalable Enterprise Analytics. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 472–483. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/8005

[9] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50–58.

[10] Chava, K., Chakilam, C., & Recharla, M. (2021). Machine Learning Models for Early Disease Detection: A Big Data Approach to Personalized Healthcare. International Journal of Engineering and Computer Science, 10(12), 25709–25730. https://doi.org/10.18535/ijecs.v10i12.4678

[11] Babcock, J., Chaudhuri, S., & Das, G. (2004). Dynamic sample selection for approximate query processing. Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, 539–550.

[12] Sriram, H. K. (2022). Advancements in Credit Score Analytics using Deep Learning and Predictive Modeling Techniques. Available at SSRN 5255128.

[13] Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. Proceedings of the 2007 SIAM International Conference on Data Mining, 443–448.
[14] Muthusamy, S., Kannan, S., Lee, M., Sanjairaj, V., Lu, W. F., Fuh, J. Y., ... & Cao, T. (2021). Cover Image, Volume 118, Number 8, August 2021. Biotechnology and Bioengineering, 118(8), i-i.
[15] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32.
[16] Aitha, A. R. (2022). Cloud Native ETL Pipelines for Real Time Claims Processing in Large Scale Insurers. Available at SSRN 5532601.
[17] Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. Mobile Networks and Applications, 19(2), 171–209.
[18] Dwaraka Nath Kummari. (2022). Fiscal Policy Simulation Using AI And Big Data: Improving Government Financial Planning. Kurdish Studies, 10(2), 934–945. https://doi.org/10.53555/ks.v10i2.3855
[19] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794.
[20] Gadi, A. L. The Role of Digital Twins in Automotive R&D for Rapid Prototyping and System Integration.
[21] Das, T., Zhu, A., Li, S., Narayanamurthy, S., & Bhat, P. (2013). Distributed and fault-tolerant streaming computation in Spark. Proceedings of the ACM Symposium on Cloud Computing, 1–12.
[22] Siva Hemanth Kolla. (2022). Knowledge Retrieval Systems for Enterprise Service Environments. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 495–506. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/8037
[23] Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified data processing on large clusters. Communications of the ACM, 51(1), 107–113.
[24] Paleti, S. (2022). Financial Innovation through AI and Data Engineering: Rethinking Risk and Compliance in the Banking Industry. Available at SSRN 5250726.
[25] DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., & Vogels, W. (2007). Dynamo: Amazon's highly available key-value store. Proceedings of the 21st ACM Symposium on Operating Systems Principles, 205–220.
[26] Sriram, H. K., ADUSUPALLI, B., & Malempati, M. (2021). Revolutionizing Risk Assessment and Financial Ecosystems with Smart Automation, Secure Digital Solutions, and Advanced Analytical Frameworks.
[27] Dwork, C. (2008). Differential privacy: A survey of results. Proceedings of the 5th International Conference on Theory and Applications of Models of Computation, 1–19.
[28] Varri, D. B. S. (2021). Cloud-Native Security Architecture for Hybrid Healthcare Infrastructure. Available at SSRN 5785982.
[29] Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering, 19(1), 1–16.
[30] Dwaraka Nath Kummari,. (2022). Machine Learning Approaches to Real-Time Quality Control in Automotive Assembly Lines. Mathematical Statistician and Engineering Applications, 71(4), 16801–16820. Retrieved from https://philstat.org/index.php/MSEA/article/view/2972
[31] Fader, P. S., Hardie, B. G. S., & Lee, K. L. (2005). "Counting your customers" the easy way: An alternative to the Pareto/NBD model. Marketing Science, 24(2), 275–284.
[32] Inala, R. (2022). Engineering Data Products for Investment Analytics: The Role of Product Master Data and Scalable Big Data Solutions. International Journal of Scientific Research and Modern Technology, 155-171.
[33] Davuluri, P. N. (2020). Improving Data Quality and Lineage in Regulated Financial Data Platforms. Finance and Economics, 1(1), 1-14.
[34] Meda, R. Enabling Sustainable Manufacturing Through AI-Optimized Supply Chains.
[35] Ghemawat, S., Gobioff, H., & Leung, S. T. (2003). The Google file system. Proceedings of the 19th ACM Symposium on Operating Systems Principles, 29–43.
[36] Varri, D. B. S. (2022). A Framework for Cloud-Integrated Database Hardening in Hybrid AWS-Azure Environments: Security Posture Automation Through Wiz-Driven Insights. International Journal of Scientific Research and Modern Technology, 1(12), 216-226.
[37] Yandamuri, U. S. (2021). A Comparative Study of Traditional Reporting Systems versus Real-Time Analytics Dashboards in Enterprise Operations. Universal Journal of Business and Management, 1(1), 1–13. Retrieved from https://www.scipublications.com/journal/index.php/ujbm/article/view/1357
[38] Gottimukkala, V. R. R. (2022). Licensing Innovation in the Financial Messaging Ecosystem: Business Models and Global Compliance Impact. International Journal of Scientific Research and Modern Technology, 1(12), 177-186.
[39] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). Springer.
[40] Vadisetty, R., Polamarasetti, A., Guntupalli, R., Raghunath, V., Jyothi, V. K., & Kudithipudi, K. (2022). AI-Driven Cybersecurity: Enhancing Cloud Security with Machine Learning and AI Agents. Sateesh kumar and Raghunath, Vedaprada and Jyothi, Vinaya Kumar and Kudithipudi, Karthik, AI-Driven Cybersecurity: Enhancing Cloud Security with Machine Learning and AI Agents (February 07, 2022).

[41] Hellerstein, J. M., Haas, P. J., & Wang, H. J. (1997). Online aggregation. Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, 171–182.
[42] Garapati, R. S. (2022). Web-Centric Cloud Framework for Real-Time Monitoring and Risk Prediction in Clinical Trials Using Machine Learning. Current Research in Public Health, 2, 1346.
[43] Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. Proceedings of the 2008 IEEE International Conference on Data Mining, 263–272.
[44] Amistapuram, K. (2022). Fraud Detection and Risk Modeling in Insurance: Early Adoption of Machine Learning in Claims Processing. Available at SSRN 5741982.
[45] Davuluri, P. S. L. N. (2021). Event-Driven Compliance Systems: Modernizing Financial Crime Detection Without Machine Intelligence. Journal of International Crisis and Risk Communication Research , 339–354. https://doi.org/10.63278/jicrcr.vi.3636
[46] Meda, R. (2022). Integrating Edge AI in Smart Factories: A Case Study from the Paint Manufacturing Industry. International Journal of Science and Research (IJSR), 1473-1489.
[47] Jagadish, H. V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J. M., Ramakrishnan, R., & Shahabi, C. (2014). Big data and its technical challenges. Communications of the ACM, 57(7), 86–94.
[48] Segireddy, A. R. (2020). Cloud Migration Strategies for High-Volume Financial Messaging Systems.
[49] Khatri, V., & Brown, C. V. (2010). Designing data governance. Communications of the ACM, 53(1), 148–152.
[50] Amistapuram, K. (2021). Digital Transformation in Insurance: Migrating Enterprise Policy Systems to .NET Core. Universal Journal of Computer Sciences and Communications, 1(1), 1–17.
[51] Kleppmann, M. (2017). Designing data-intensive applications. O'Reilly Media.
[52] Nagabhyru, K. C. (2022). Bridging Traditional ETL Pipelines with AI Enhanced Data Workflows: Foundations of Intelligent Automation in Data Engineering. Available at SSRN 5505199.
[53] Lahiri, M., & Venkatasubramanian, S. (2013). Robust record linkage. Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, 101–112.
[54] Avinash Reddy Aitha. (2022). Deep Neural Networks for Property Risk Prediction Leveraging Aerial and Satellite Imaging. International Journal of Communication Networks and Information Security (IJCNIS), 14(3), 1308–1318. Retrieved from https://www.ijcnis.org/index.php/ijcnis/article/view/8609
[55] Leskovec, J., Rajaraman, A., & Ullman, J. D. (2014). Mining of massive datasets (2nd ed.). Cambridge University Press.
[56] Rongali, S. K. (2022). AI-Driven Automation in Healthcare Claims and EHR Processing Using MuleSoft and Machine Learning Pipelines. Available at SSRN 5763022.
[57] Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing, 7(1), 76–80.
[58] Meda, R. (2021). Digital Infrastructure for Predictive Inventory Management in Retail Using Machine Learning. International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI, 10.
[59] Lin, J., Kolcz, A., & Szymanski, B. K. (2012). Large-scale machine learning at Twitter. Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, 793–804.
[60] Sheelam, G. K. Power-Efficient Semiconductors for AI at the Edge: Enabling Scalable Intelligence in Wireless Systems. International Journal of Innovative Research in Electrical, Elec-tronics, Instrumentation and Control Engineering (IJIREEICE), DOI, 10.
[61] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Byers, A. H. (2011). Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute.
[62] Vadisetty, R., Polamarasetti, A., Guntupalli, R., Rongali, S. K., Raghunath, V., Jyothi, V. K., & Kudithipudi, K. (2021). Legal and Ethical Considerations for Hosting GenAI on the Cloud. International Journal of AI, BigData, Computational and Management Studies, 2(2), 28-34.
[63] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. Proceedings of the International Conference on Learning Representations, 1–12.
[64] Ramesh Inala. (2022). Cross-Domain MDM Integration Using AI-Driven Data Governance: A Case Study In Financial Technology Architecture. Migration Letters, 19(2), 280–304. Retrieved from https://migrationletters.com/index.php/ml/article/view/11982
[65] Montoya, D. Y., Neto, A. M., & da Silva, A. S. (2016). A survey of entity resolution in big data. Journal of Big Data, 3(1), 1–22.
[66] Aitha, A. R. (2021). Optimizing Data Warehousing for Large Scale Policy Management Using Advanced ETL Frameworks.
[67] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: Cluster computing with working sets. Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing, 1–7.
[68] Varri, D. B. S. (2022). AI-Driven Risk Assessment and Compliance Automation in Multi-Cloud Environments. Available at SSRN 5774924.
[69] Zaharia, M., Das, T., Li, H., Shenker, S., & Stoica, I. (2012). Discretized streams: Fault-tolerant streaming computation at scale. Proceedings of the 24th ACM Symposium on Operating Systems Principles, 423–438.

[70] Segireddy, A. R. (2021). Containerization and Microservices in Payment Systems: A Study of Kubernetes and Docker in Financial Applications. Universal Journal of Business and Management, 1(1), 1–17.
[71] Zhai, C., & Massung, S. (2016). Text data management and analysis: A practical introduction to information retrieval and text mining. ACM & Morgan Claypool.
[72] Kolla, S. H. (2021). Rule-Based Automation for IT Service Management Workflows. Online Journal of Engineering Sciences, 1(1), 1–14. Retrieved from https://www.scipublications.com/journal/index.php/ojes/article/view/1360
[73] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135–146.
[74] Keerthi Amistapuram , "Energy-Efficient System Design for High-Volume Insurance Applications in Cloud-Native Environments," International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering (IJIREEICE), DOI 10.17148/IJIREEICE.2020.81209
[75] Goutham Kumar Sheelam. (2022). Reconfigurable Semiconductor Architectures For AI-Enhanced Wireless Communication Networks. Kurdish Studies, 10(2), 1027–1040. https://doi.org/10.53555/ks.v10i2.3867
[76] Batarseh, F. A., & Yang, R. (2019). Federal data science: Transforming government and society. Academic Press.
[77] Gottimukkala, V. R. R. (2021). Digital Signal Processing Challenges in Financial Messaging Systems: Case Studies in High-Volume SWIFT Flows.
[78] Bhasin, H., & Bhatia, P. (2020). Clickstream data mining for web analytics and customer behavior modeling: A review. ACM Computing Surveys, 53(6), 1–34.
[79] Rongali, S. K. (2021). Cloud-Native API-Led Integration Using MuleSoft and .NET for Scalable Healthcare Interoperability. Available at SSRN 5814563.
[80] Davuluri, P. N. (2020). Event-Driven Architectures for Real-Time Regulatory Monitoring in Global Banking.
[81] Abedjan, Z., Golab, L., & Naumann, F. (2016). Profiling relational data: A survey. The VLDB Journal, 24(4), 557–581.
[82] Yandamuri, U. S. (2022). Big Data Pipelines for Cross-Domain Decision Support: A Cloud-Centric Approach. International Journal of Scientific Research and Modern Technology, 1(12), 227–237. https://doi.org/10.38124/ijsrmt.v1i12.1111
[83] Dwaraka Nath Kummari. (2022). AI-Driven Audit Frameworks For Enhancing Compliance In Modern Manufacturing Systems. Migration Letters, 19(S8), 2150–2177. Retrieved from https://migrationletters.com/index.php/ml/article/view/11912
[84] Davuluri, P. N. Event-Driven Compliance Systems: Modernizing Financial Crime Detection Without Machine Intelligence.
[85] Baesens, B., Van Vlasselaer, V., & Verbeke, W. (2021). Fraud analytics using descriptive, predictive, and social network techniques: A guide to data science for fraud detection (2nd ed.). Wiley.
[86] Vadisetty, R., Polamarasetti, A., Guntupalli, R., Raghunath, V., Jyothi, V. K., & Kudithipudi, K. (2021). Privacy-Preserving Gen AI in Multi-Tenant Cloud Environments. Sateesh kumar and Raghunath, Vedaprada and Jyothi, Vinaya Kumar and Kudithipudi, Karthik, Privacy-Preserving Gen AI in Multi-Tenant Cloud Environments (January 20, 2021).
[87] Barocas, S., Hardt, M., & Narayanan, A. (2019). Fairness and machine learning. fairmlbook.org (Book manuscript).
[88] Garapati, R. S. (2022). AI-Augmented Virtual Health Assistant: A Web-Based Solution for Personalized Medication Management and Patient Engagement. Available at SSRN 5639650.
[89] Gottimukkala, V. R. R. (2020). Energy-Efficient Design Patterns for Large-Scale Banking Applications Deployed on AWS Cloud. power, 9(12).
[90] Ahmad, M. A., Eckert, C., & Teredesai, A. (2018). Interpretable machine learning in healthcare**.** Proceedings of the ACM Conference on Health, Informatics, and Data Science, 1–10.
[91] Inala, R. Advancing Group Insurance Solutions Through Ai-Enhanced Technology Architectures And Big Data Insights.
[92] Aljabre, A. (2019). Cloud computing security in healthcare**.** Journal of King Saud University – Computer and Information Sciences, 31(1), 10–18.
[93] Kolla, S. K. (2021). Architectural Frameworks for Large-Scale Electronic Health Record Data Platforms. Current Research in Public Health, 1(1), 1–19. Retrieved from https://www.scipublications.com/journal/index.php/crph/article/view/1372
[94] Davuluri, P. N. (2020). Improving Data Quality and Lineage in Regulated Financial Data Platforms. Finance and Economics, 1(1), 1-14.
[95] Bani-Salameh, N., & Olariu, C. (2019). Efficient healthcare big data processing using cloud computing**.** Journal of Parallel and Distributed Computing, 132, 47–55.
[96] Goutham Kumar Sheelam, "Semiconductor Innovation for Edge AI: Enabling Ultra-Low Latency in Next-Gen Wireless Networks," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJARCCE.2022.111258
[97] Dey, N., Ashour, A. S., & Balas, V. E. (Eds.). (2018). Smart medical data sensing and IoT systems design in healthcare**.** Springer.