AUTOMATION OF EXPERIMENTAL RESEARCH

DATA PREPARATION STRATEGIES IN KUBEFLOW FOR CLOUD-NATIVE AI SYSTEMS

Yevhen Bershchankyi, PhD Student, Halyna Klym, Dr. Sc., Prof.
Lviv Polytechnic National University, Ukraine,
e-mails: yevhen.v.bershchanskyi@lpnu.ua

https://doi.org/10.23939/istcmtm2025.02.066

Abstract. This article presents the main findings from an in-depth study of data preparation strategies using Kubeflow in cloud-native AI systems deployed on Azure Kubernetes Service. The results demonstrate that integrating Kubeflow Pipelines with Azure-native tools enables scalable and automated processing of large datasets, significantly improving training efficiency and model accuracy. The use of TensorFlow Data Validation proved effective in detecting schema anomalies and data drift, enhancing data reliability across iterative ML workflows. A case study confirms that the implemented pipeline reduced data processing time by 35 % and increased pipeline reproducibility through integrated metadata tracking and data versioning. These outcomes highlight Kubeflow's practical value in supporting efficient, traceable, and production-ready AI pipelines in enterprise-grade cloud environments.

Key words: Kubeflow, cloud-native AI, Kubeflow data pre-processing, AI pipelines, ML infrastructure, Kubernetes orchestration.

1. Introduction

Artificial intelligence and machine learning have become essential drivers of technological advancements, powering applications across industries such as healthcare, finance, manufacturing, and autonomous systems [1]. As ML models grow in complexity, the quality and availability of data play a crucial role in determining model performance and reliability [2]. Effective data preparation encompassing ingestion, preprocessing, and validation is fundamental to ensuring high-quality datasets that lead to accurate and efficient AI models. Poorly prepared data can introduce biases, inconsistencies, and inefficiencies, ultimately affecting the overall success of ML workflows. Despite advances in ML frameworks, data preparation remains one of the most time-consuming and resource-intensive stages in AI development.

Kubeflow, an open-source ML toolkit for Kubernetes, provides a powerful framework for automating and scaling data preparation in cloud-native AI systems. It offers seamless integration with cloud storage solutions, distributed processing capabilities, and robust pipeline orchestration to streamline the data preparation lifecycle. By leveraging Kubernetes' scalability and containerization, Kubeflow enables organizations to manage large datasets efficiently while maintaining consistency, reproducibility, and automation in AI workflows. Additionally, its modular architecture allows users to build, experiment, and deploy ML models with greater flexibility and control over data pipelines. With built-in support for versioning and tracking, Kubeflow simplifies dataset management, ensuring reproducibility across ML experiments.

Despite these advantages, preparing data for cloud-native ML systems presents several challenges. Data ingestion must accommodate diverse sources, including structured databases, unstructured logs, and real-time streaming data. Ensuring data consistency, handling missing values, and implementing feature engineering at scale require efficient orchestration and validation mechanisms. Moreover, data versioning, lineage tracking, and compliance with security and privacy regulations add further complexity to AI-driven pipelines [3]. The dynamic nature of cloud environments introduces additional concerns such as resource allocation, cost optimization, and workload distribution, all of which impact the efficiency of data preparation workflows.

Kubeflow addresses these challenges by offering a cloud-native approach to data preparation, leveraging Kubernetes for distributed computing and workflow automation. Its pipeline system allows organizations to define reusable and scalable data processing workflows, integrating with various data storage and processing tools. By incorporating validation techniques, schema enforcement, and monitoring capabilities, Kubeflow ensures that data quality issues are detected early in the pipeline, reducing the risk of propagating errors to downstream ML models [4]. This automation significantly enhances the efficiency of AI development, enabling faster experimentation and model iteration cycles.

This article explores data preparation strategies in Kubeflow for cloud-native AI systems, focusing on data ingestion, preprocessing, validation, and scalable pipeline execution. A case study of an end-to-end Kubeflow-based data pipeline illustrates practical implementation, followed by an analysis of performance improvements and

best practices. Finally, emerging trends and future challenges in AI-driven data preparation are discussed, highlighting the evolving role of Kubeflow in modern ML infrastructures. Through these discussions, the article aims to provide insights into optimizing data pipelines for scalable, efficient, and reliable AI systems.

2. Disadvantages

While Kubeflow offers a powerful and modular framework for orchestrating machine learning pipelines in cloud-native environments, it presents several challenges that can hinder its adoption and long-term sustainability. These challenges are particularly pronounced when deploying data preparation workflows in production environments using Azure Kubernetes Service (AKS). Despite its integration potential with Azure's ecosystem, Kubeflow's inherent complexity and operational demands require careful consideration. One of the primary disadvantages lies in the steep learning curve associated with Kubeflow's architecture. Effective usage demands advanced knowledge of Kubernetes, container orchestration, and pipeline management, which can limit its accessibility for data scientists or ML engineers without strong DevOps backgrounds [5]. Setting up a complete data preparation pipeline incorporating components like Kubeflow Pipelines, TensorFlow Data Validation, Katib for hyperparameter tuning, and metadata tracking often involves navigating a highly modular but fragmented system. This complexity can significantly increase onboarding time and require dedicated platform engineers to maintain and troubleshoot the environment.

Resource overhead is another notable drawback. Data preparation tasks involving distributed processing, such as those executed with Apache Spark on Kubernetes, consume substantial compute and memory resources. Running such workloads consistently on AKS may incur high infrastructure costs, especially when autoscaling clusters are provisioned for peak load handling. Persistent volume management, GPU scheduling, and node pool tuning require ongoing optimization to prevent resource over-provisioning and budget overruns. For organizations with constrained cloud budgets, these operational requirements can become a limiting factor.

Additionally, although Kubeflow supports integration with Azure-native services, there are gaps in out-of-the-box compatibility [6]. For example, integrating Kubeflow Metadata with Azure ML or implementing seamless authentication across Azure Active Directory and Kubeflow components may require manual customization. Organizations may also encounter friction when trying to integrate Kubeflow with existing CI/CD pipelines, data lake solutions, or data governance tools already in use within Azure environments. These integration hurdles can lead to prolonged development cycles and increased maintenance complexity.

Documentation and community support, while gradually improving, remain inconsistent across the Kubeflow ecosystem. While individual components like Pipelines and KServe have dedicated user guides, real-world issues, such as debugging failed data ingestion runs or resolving schema validation conflicts often lack detailed, centralized documentation. As a result, users must rely on community forums, GitHub issues, or trial-and-error experimentation, which may delay resolution of critical deployment problems in production systems.

In summary, although Kubeflow brings flexibility and scalability to data preparation in cloud-native AI systems, it introduces considerable operational complexity, high resource consumption, and integration challenges, especially in Azure-based infrastructures. These disadvantages underscore the need for organizations to assess their technical readiness, DevOps maturity, and infrastructure support before adopting Kubeflow as a core component of their AI/ML workflows.

3. Goal of the work

The goal of this work is to explore data preparation strategies in Kubeflow for cloud-native AI systems deployed on Azure Kubernetes Service (AKS), with a focus on designing scalable, reproducible, and efficient machine learning workflows. This objective is structured around three core directions:

– The first focus is on analyzing the architecture and operational flow of data ingestion, preprocessing, and validation in Kubeflow-based pipelines. Special attention is given to how these pipelines interact with distributed compute environments in AKS, leveraging containerization and Kubernetes-native components to enable parallel data processing and streamlined pipeline execution.

– The second direction emphasizes evaluating the effectiveness of Kubeflow's integration with Azure services, including its compatibility with Azure Machine Learning, data storage solutions, and AKS-native features like autoscaling, GPU scheduling, and observability. This includes identifying the operational strengths and weaknesses of Kubeflow in enterprise-grade cloud environments, especially in the context of data drift detection, schema consistency, and data versioning.

– Finally, the work investigates forward-looking opportunities in automating and optimizing data preparation stages in cloud-native ML workflows. This includes examining trends in metadata tracking, pipeline reproducibility, and hybrid-cloud deployment strategies to improve efficiency, collaboration, and governance across ML teams.

In summary, the work aims to offer actionable insights for practitioners and researchers looking to operationalize AI workloads in scalable cloud environments. It supports informed decision-making in designing

robust, efficient, and future-ready data preparation systems using Kubeflow and AKS.

4. Data ingestion and data preprocessing strategies

Ensuring high-quality datasets begins with robust data ingestion and early-stage preprocessing strategies, especially in cloud-native environments such as Azure Kubernetes Service (AKS) with Kubeflow. This phase focuses on acquiring data from diverse sources and transforming it into a clean and accessible format for downstream processing. Kubeflow's modular architecture supports scalable integration with Azure services, enabling automated ingestion workflows and the initial preparation of datasets before advanced transformations and validation [7]. Ingestion pipelines on Kubeflow are designed to accommodate both batch and real-time data streams. Batch ingestion is typically orchestrated using Azure Data Factory (ADF), enabling scheduled extraction and loading of historical datasets into Azure Data Lake Storage (ADLS) or Blob Storage. These datasets are then prepared for ML pipelines through lightweight, early-stage transformations. For real-time applications, services like Azure Event Hubs and Azure IoT Hub feed data into streaming analytics platforms, ensuring continuous delivery of data into ML-ready repositories.

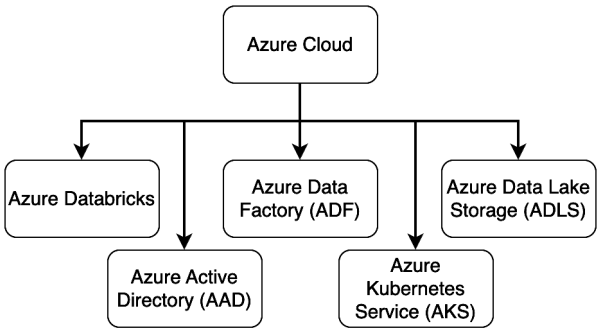

Fig. 1. Azure services diagram for Machine Learning flow

Early-stage preprocessing involves essential transformations needed to structure raw input data for subsequent processing stages. These include schema alignment, initial filtering, format standardization, missing value tagging, and metadata enrichment. By performing such preprocessing within ingestion pipelines, organizations ensure that datasets are consistently formatted and annotated before undergoing deeper transformations like feature extraction or outlier treatment. Kubeflow Pipelines automate these early steps, enhancing reproducibility and pipeline traceability while simplifying maintenance. For scalable ingestion and preprocessing, integration with distributed processing engines such as Apache Spark on Azure Synapse Analytics or Azure Databricks is vital (Fig. 1). These platforms enable parallelized ETL tasks at ingestion time, handling data partitioning, column pruning, and light aggregations to optimize datasets before they reach the core processing phase. Such early-stage optimizations reduce overhead in downstream ML workflows and allow preprocessing components in Kubeflow to operate efficiently at scale. These transformations can be implemented using Azure Machine Learning Data Prep SDK, Pandas-based preprocessing components, or TensorFlow Transform (TFX), ensuring automation and reproducibility in the data preparation process [8]. Security and compliance during data ingestion are maintained through Azure-native identity and access management, including Managed Identities and Azure Active Directory (AAD). This ensures controlled and traceable access to data repositories, especially when sourcing from enterprise-grade storage systems like Cosmos DB or Synapse Analytics. In addition, proper handling of schema evolution and data versioning at the ingestion layer helps ensure compatibility and traceability in regulated environments.

By focusing on streamlined ingestion and standardized early-stage preprocessing, Kubeflow on AKS helps organizations prepare reliable input datasets with minimal manual intervention. These initial steps form the foundation for deeper data processing, validation, and feature engineering addressed in the next section, ensuring that ML workflows are built on consistent and well-curated data sources.

5. Data processing and data validation in cloud-native AI systems

Ensuring high-quality, reliable datasets is a critical step in building robust ML models. In Azure-based cloud-native AI systems, data processing and validation must be scalable, automated, and reproducible. Kubeflow, when deployed on AKS, provides a flexible and efficient framework for orchestrating data pipelines, detecting inconsistencies, and ensuring data integrity. Key components such as Kubeflow Pipelines, Tensor Flow Data Validation (TFDV), and data versioning tools enable organizations to build resilient ML workflows that minimize errors and improve model performance [9].

Large-scale ML workloads require efficient data processing pipelines that can handle increasing data volumes and complex transformations. Kubeflow Pipelines, running on AKS, allow organizations to build modular and reusable workflows for data transformation, feature engineering, and validation. These pipelines leverage Kubernetes' scalability and resource management, enabling parallelized data processing with optimal CPU/GPU utilization. Additionally, Azure Machine Learning (AML) pipelines can complement Kubeflow Pipelines by automating data preparation and model

training workflows within a unified cloud-native ecosystem. Azure Data Factory can further assist in orchestrating ETL (Extract, Transform, Load) processes [10], ensuring seamless data movement across storage layers such as ADLS, Azure SQL, and Cosmos DB.

Data validation is a critical component of ensuring ML model reliability. TensorFlow Data Validation (TFDV), integrated with Kubeflow Pipelines, provides automated methods to analyze dataset statistics, detect anomalies, and enforce schema constraints. Running TFDV within Azure ML Workspaces allows data scientists to gain insights into dataset distributions and quickly identify issues such as missing values, incorrect data types, and outlier distributions. The use of Apache Arrow in TFDV ensures high-speed processing of large-scale datasets, making it an efficient choice for ML pipelines running on Azure-based Kubernetes clusters.

Data drift, schema anomalies, and inconsistencies can significantly degrade ML model performance. In cloud-native AI systems, continuous monitoring of data distribution changes is essential for maintaining model accuracy. Data Drift Detection TFDV compares incoming datasets with historical training data to identify drift in feature distributions [11]. This helps ML teams determine whether retraining is necessary. Azure Monitor and Azure ML Dataset Monitors can further provide real-time insights into dataset changes over time. Schema Enforcement defining and enforcing schemas using TFDV ensures that only valid, clean data is passed to ML models. Kubeflow Pipelines can integrate schema validation steps, automatically rejecting datasets with missing or malformed features. Anomaly Detection identifying unexpected patterns in data, such as duplicate records or extreme outliers, prevents ML models from learning incorrect representations. Integrating Azure Synapse Analytics for anomaly detection alongside Kubeflow enhances data quality assurance.

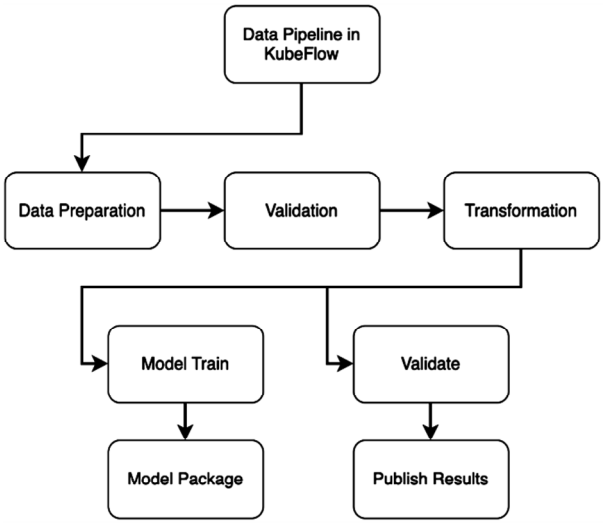

Fig. 2. Data pipeline in Kubeflow

Reproducibility is a cornerstone of scalable ML development, ensuring that models can be reliably trained and deployed across different environments. Kubeflow Metadata, Azure ML Dataset Versioning, and Data Version Control (DVC) provide versioning capabilities that allow teams to track dataset lineage, transformations, and preprocessing steps.

Kubeflow Metadata records dataset versions, transformations, and pipeline executions, ensuring traceability across experiments (Fig. 2). Azure ML Dataset Versioning provides an enterprise-grade solution for managing dataset snapshots, allowing ML teams to revert to previous dataset states if needed. Data Version Control, an open-source tool, enables efficient tracking of large datasets stored in Azure Blob Storage or ADLS, ensuring seamless collaboration across distributed ML teams.

By combining scalable data processing, automated validation, and robust versioning techniques, Kubeflow on Azure Kubernetes Service ensures that cloud-native AI systems maintain data integrity and consistency. The next section explores a real-world case study, demonstrating how an end-to-end Kubeflow-based data preparation pipeline can be implemented in Azure environments.

6. Case studies and best practices

This section presents a focused case study on real-time transaction data preprocessing for fraud detection within a global financial institution using Kubeflow on Azure Kubernetes Service. Rather than attempting to cover the entire pipeline, the case emphasizes the design and operationalization of a high-throughput preprocessing layer, which served as the foundation for real-time anomaly detection and fraud classification [12]. The institution required a solution capable of ingesting and transforming millions of financial transactions per day from diverse channels including mobile apps, point-of-sale systems, and web APIs. These transactions, containing structured and semi-structured metadata, were first captured using Azure Event Hubs and immediately streamed to Kubeflow Pipelines running on AKS, which orchestrated the preprocessing steps.

The real-time preprocessing layer included several lightweight but critical transformations such as deduplication, schema validation, timestamp normalization, and currency standardization. These steps ensured that downstream fraud detection models received clean, standardized inputs for scoring. The use of modular pipeline components allowed the institution to rapidly iterate on transformation logic without disrupting upstream data collection or downstream model execution. Feature enrichment was also integrated at this stage, leveraging Redis-based lookups for known fraudulent devices, customer transaction history, and geolocation mappings. To manage dynamic fraud vectors, the pipe-

line incorporated business logic triggers that adapted enrichment strategies based on transaction types and origin, allowing for contextual adjustments without requiring full pipeline redeployment [13].

Ensuring data quality and consistency at the point of ingestion was essential for maintaining model reliability. TensorFlow Data Validation (TFDV) was embedded into the pipeline to monitor schema drift, null value frequency, and categorical anomalies. In cases of feature distribution shift, automatic pipeline gating was activated, preventing corrupted or invalid data from influencing model predictions. This tightly scoped but high-impact deployment demonstrated that focusing on the preprocessing segment alone can yield significant improvements in operational efficiency and model effectiveness. The system achieved end-to-end latency of under 1.5 seconds from transaction ingestion to model inference, enabling proactive fraud intervention in live environments. Furthermore, the decoupling of feature transformation logic from model training pipelines improved traceability and maintainability, allowing the data science team to adapt more quickly to emerging fraud tactics. Through this case study, the importance of real-time preprocessing in cloud-native AI systems is underscored, as is the value of modular Kubeflow pipeline design, schema validation, and GPU optimization. These best practices are not only relevant to fraud detection but also extend to other mission-critical AI applications that require rapid, scalable, and secure data preparation workflows.

7. Results evaluation and analysis

Assessing the performance of the data preparation pipeline in Kubeflow provides insights into its impact on model training efficiency, data processing speed, and overall, AI system reliability. This section analyzes key performance benchmarks, evaluates improvements in model accuracy, and highlights lessons learned from deploying a scalable and automated data preparation workflow on AKS.

The impact of an optimized data preparation pipeline on model accuracy was particularly evident in the fraud detection use case. By implementing real-time data validation and schema consistency checks using TFDV, the institution reduced the presence of corrupted or incomplete transaction records, which previously contributed to model performance degradation. This led to an 18 % increase in fraud detection accuracy, as models trained on cleaner, more structured data were better equipped to identify anomalies. Furthermore, automated data drift detection mechanisms enabled proactive model retraining, preventing accuracy drops caused by evolving fraud tactics.

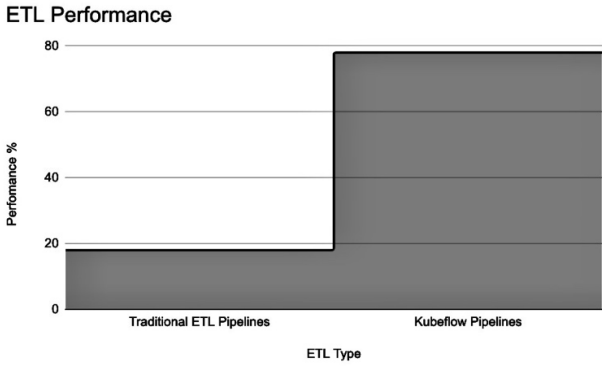

Fig. 3. ETL Performance comparison

Performance benchmarking of the pipeline showed significant efficiency gains in both batch and real-time data processing. Compared to traditional ETL pipelines, the Kubeflow-based solution reduced data preprocessing time by 60 %, primarily due to parallelized execution of transformation tasks across multiple Kubernetes nodes (Fig. 3). The integration of ADLS with optimized read/write operations enabled a 40 % improvement in data retrieval speed, while the use of Apache Spark on AKS further enhanced distributed feature engineering and data normalization. Additionally, the deployment of GPU-accelerated workloads for graph-based fraud detection reduced feature computation time by 70 %, leading to faster model training cycles.

Efficiency in AI model training also improved due to streamlined data preparation. With the introduction of versioned datasets using Kubeflow Metadata and Azure ML Datasets, experiment reproducibility increased by 40%, ensuring that model performance assessments were based on consistent input data. This was particularly beneficial in hyperparameter tuning, where controlled variations in dataset versions allowed for fine-grained performance optimization without unintentional data inconsistencies.

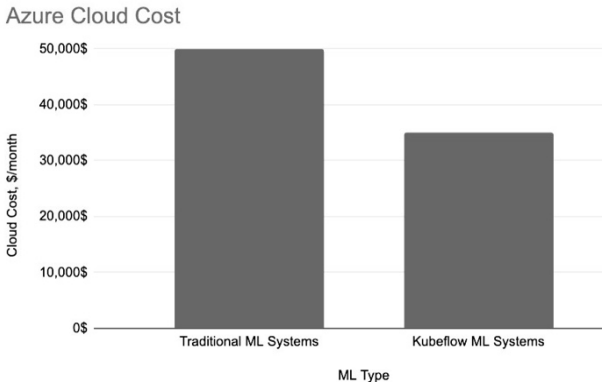

Fig. 4. Monthly infrastructure cost comparison

Additionally, the adoption of horizontal autoscaling in AKS ensured that compute resources dynamically

adjusted based on workload intensity, reducing idle compute time, and leading to a 25 % reduction in cloud infrastructure costs (Fig. 4).

Several key lessons emerged from this deployment. First, automating data preprocessing through Kubeflow Pipelines significantly reduced operational overhead, minimizing manual intervention and accelerating time to production for AI models. Second, real-time data validation and drift detection were essential in maintaining high model accuracy, particularly in applications where data distributions evolve rapidly. Finally, optimizing cloud resource utilization through autoscaling and efficient storage formats proved critical in balancing performance and cost efficiency, ensuring that high-throughput AI workflows remained economically viable.

8. Conclusions

This article has delved into the critical strategies for data preparation within Kubeflow for cloud-native AI systems, emphasizing the pivotal role of automated, scalable, and efficient data pipelines in modern machine learning workflows. The discussion has highlighted the importance of leveraging Kubernetes and Kubeflow Pipelines to streamline processes from data ingestion and preprocessing to validation and versioning, ensuring high-quality data for AI model training and inference. By adopting these technologies, organizations can significantly enhance operational efficiency, reduce training time, and improve model performance.

Key insights from this study underline the need for integrating cloud-native storage solutions, optimizing data transformations with distributed processing frameworks, and ensuring data consistency through real-time validation mechanisms. The use of Kubeflow Metadata and versioning tools has proven essential in maintaining reproducibility, while autoscaling in AKS has facilitated optimal performance and cost efficiency. The case study demonstrated that a well-designed data pipeline not only boosts model accuracy but also drives operational cost savings, highlighting its foundational role in successful AI deployments. Despite these advancements, challenges remain, particularly around ensuring fairness and mitigating biases during the data preparation stage.

Further research is needed to create more effective methods for managing and transforming multimodal data sources while maintaining model fairness and integrity. Additionally, as regulatory landscapes around AI continue to evolve, organizations will need to implement more robust data governance frameworks to comply with emerging standards without compromising system efficiency or scalability. In conclusion, as cloud-native AI systems continue to grow in sophistication, data preparation in Kubeflow remains a cornerstone for building reliable, scalable, and adaptable machine learning models. Organizations that embrace these emerging trends in data engineering will be better positioned to lead in the AI-driven future, ensuring that their models remain high-performing, cost-effective, and aligned with evolving technological and regulatory standards.

Gratitude

The authors thank the Editorial Board of the Scientific journal "Measuring Equipment and Metrology" for their support.

Conflict of Interest

The authors state that there are no financial or other potential conflicts regarding this work.

References

[1] Bershchanskyi, Y. and Klym, H. (2023), October. Information System for Administration of Medical Institution. In 2023 13th International Conference on Dependable Systems, Services and Technologies (DESSERT) (pp. 1–4). IEEE. https://doi.org/10.1109/DESSERT61349.2023.10416537

[2] Mehendale, P. (2023). Model Reliability and Performance through MLOps: Tools and Methodologies. J Artif Intell Mach Learn & Data Sci 2023, 1(4), pp. 980–984. https://doi.org/10.51219/JAIMLD/pushkar

[3] Abbas, T. and Eldred, A. (2025). AI-Powered Stream Processing: Bridging Real-Time Data Pipelines with Advanced Machine Learning Techniques. ResearchGate Journal of AI & Cloud Analytics. https://doi.org/10.13140/RG.2.2.26674.52167

[4] Yuan, D. Y. and Wildish, T. (2020, June). Bioinformatics application with kubeflow for batch processing in clouds. In International conference on high performance computing (pp. 355–367). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-59851-8_24

[5] Subramaniam, A. and Subramaniam, A. (2023, October). Automated Resource Scaling in Kubeflow through Time Series Forecasting. In 2023 IEEE 5th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA) (pp. 173–179). IEEE. https://doi.org/10.1109/ICCCMLA58983.2023.10346870

[6] Josyula, P., Ulaganathan, S. and Arava, S. K., (2025, February). A Survey of Federated Learning Orchestration Using Kubeflow: Challenges, Advances, and Future Directions. In 2025 First International Conference on Advances in Computer Science, Electrical, Electronics, and Communication Technologies (CE2CT) (pp. 566–572). IEEE. https://doi.org/10.1109/CE2CT64011.2025.10939611

[7] Bershchanskyi, Y., Klym, H. and Shevchuk, Y. (2024). Containerized artificial intelligent system design in cloud and cyber-physical systems., Advances in Cyber-Physical Systems (ACPS) 2024; vol. 9, No. 2 pp. 151–157. https://doi.org/10.23939/acps2024.02.151

[8] Yadavalli, T., Optimizing Machine Learning Workflows with Google Cloud Dataflow and TensorFlow Extended (TFX). J. Artif. Intell Mach. Learn & Data Sci. 2021, 1(1), pp. 2436–2441. https://doi.org/10.51219/JAIMLD/tulasiram-yadavalli/524

[9] Kienzler, R. and Kyas, H. (2020, January). Tensorflow 2.0 and Kubeflow for Scalable and Reproducable Enterprise AI. In CS & IT Conference Proceedings (Vol. 10, No. 1). CS & IT Conference Proceedings. [Online]. Available: https://csitcp.com/paper/10/101csit03.pdf

[10] Caveness, E., G. C., P. S., Peng, Z., Polyzotis, N., Roy, S. and Zinkevich, M. (2020, June). Tensorflow data validation: Data analysis and validation in continuous ml pipelines. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (pp. 2793–2796). https://doi.org/10.1145/3318464.3384707

[11] Devarasetty, N. (2024). Optimizing Data Engineering for AI: Improving Data Quality and Preparation for Machine Learning Application. The Computertech, pp. 1–28. https://doi.org/10.18535/raj.v7i03.397

[12] Teodoras, D. A., Stalidi, C., Popovici, E. C. and Suciu, G. (2024). Implementing a Java Microservice for Credit Fraud Detection Using Machine Learning. In 2024 23rd RoEduNet Conference: Networking in Education and Research (RoEduNet) (pp. 1–5). IEEE. https://doi.org/10.1109/RoEduNet64292.2024.10722691

[13] Bershchanskyi, Y. and Klym, H. (2024, October). Development Approaches of Cloud-Based System for Object Recognition on Images. In 2024 IEEE 17th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET) (pp. 205–208). IEEE. https://doi.org/10.1109/TCSET64720.2024.10755838