Continuous Integration and Continuous Deployment, DevOps, and MLOps

Het Ranpura[1], Tirth Patel[2], Prachi Kotadia[3]
Department of Computer Science Illinois Institute of Technology Chicago, IL 60616, USA
[1]E-mail: hranpura@hawk.iit.edu, [2]E-mail: tpatel67@hawk.iit.edu, [3]E-mail: pkotadia@hawk.iit.edu

Abstract—The main aim of industrial machine learning projects is to create ML products and swiftly integrate them into production settings. However, this task is overworked with challenges, particularly in the automation and operationalization of ML products, leading to numerous instances where ML initiatives fall short of their intended outcomes. The paradigm of Machine Learning Operations (MLOps) is tailored to tackle these challenges directly. MLOps encompasses a multifaceted approach, encompassing best practices, conceptual frameworks, and the cultivation of a specialized development culture. However, it's worth noting that MLOps remains a somewhat unclear term, and its implications for researchers and professionals in the field remain somewhat ambiguous. To address this knowledge gap, we as team have undertaken a comprehensive mixed-method research approach. This approach includes an in-depth literature review and a cutting-edge tool assessment. As a result of these thorough investigations, we present a consolidated overview that encapsulates the fundamental principles, key components, and vital roles within the MLOps framework. Additionally, we have shown the associated architectural aspects and workflow intricacies. Additionally, we provided a detailed explanation of MLOps and emphasize the continuous challenges present in this emerging area. Ultimately, the culmination of this work serves as valuable guidance for both ML researchers and practitioners seeking to automate and effectively manage their ML products through the application of a defined set of technologies and methodologies.

Keywords: Machine Learning; DevOps; MLOps; Production Environments; CI/CD; Operations; Workflow Orchestration; Automate; Application.

I. Introduction

Machine Learning (ML) has evolved into a vital tool for harnessing the potential of data, empowering businesses to innovate, operate more efficiently, and promote sustainability [1]. Nevertheless, the realization of productive ML applications in real-world scenarios often does not meet expectations [2]. A significant number of ML projects encounter setbacks, and numerous promising ML proofs of concept never progress to the production stage [3]. From a research perspective, the finding is expected, as the ML community has predominantly concentrated on developing ML models rather than (i) shaping ML products ready for production and (ii) coordinating the intricate processes necessary for the resulting, often complex ML system components and infrastructure, along with the essential roles for automating and managing an ML system in real-world scenarios [1].

To address these challenges, the objective of this study is to explore methods for automating and operationalizing manual ML processes, thus facilitating the transition of more ML proofs of concept into production. In this comprehensive examination of literature, we have studied plunge into the emerging practice of "Machine Learning Operations", or MLOps for brevity. MLOps addresses the precise challenge of creating and sustaining efficient machine learning systems. We adopt a holistic approach to foster a shared understanding of the associated components, principles, roles, and architectures. While Current studies have explained different specific aspects of MLOps, there remains a need for a comprehensive conceptualization, generalization, and clarification of ML system design. Diverse interpretations and definitions of "MLOps" can lead to misunderstandings and miscommunication, potentially resulting in errors within the broader ML system set-up. Consequently, we pose the following central research question is 'What is MLOps?'

To address this query, we embark on a mixed-method research endeavor to (i) identify fundamental principles of MLOps, (ii) delineate core functional components, (iii) outline the essential roles crucial for effective MLOps implementation, and (iv) formulate a generalized architecture for the design of ML systems. These insights collectively culminate in a comprehensive definition of MLOps, which contributes to a shared comprehension of the term and its associated concepts. In practical discussions, providing concise guidelines for professionals and researchers clarifies their respective responsibilities. These insights facilitate the smoother transition of proofs of concept into production, minimizing errors in system design, and ultimately improving the reliability of predictions in real-world scenarios.

This paper is organized as follows: we start by clarifying foundational concepts and reviewing relevant literature. Following that, we outline our methodology, encompassing an extensive literature review, a comprehensive tool assessment, and an information collection from the online interview data [4]. The paper concludes with a concise summary, recognizing limitations, and providing a indication into future prospects.

II. Foundations of DevOps in Software Engineering

(1) The evolution of software engineering methodologies has witnessed the introduction and adoption of diverse models, such as the sequential Waterfall model and the dynamic Agile manifesto. Despite their distinctions, these methodologies share a common objective: optimizing the creation and delivery of software products for production environments [5].

(2) In the late 2000s, the transformative movement known as 'DevOps' emerged to address and mitigate

challenges in software development. DevOps goes beyond a broad methodology, embodying a holistic paradigm that tackles both social dynamics and technical complexities within software development teams and processes [5].

(3) DevOps aims to align the efforts of development (Dev) and operations (Ops) teams, fostering collaboration, effective communication, and shared knowledge. This integration is achieved through automation practices, including Continuous Integration (CI), Continuous Delivery (CD), and Continuous Deployment. These practices enable rapid, consistent, and reliable software delivery cycles, supported by continuous testing and quality assurance. DevOps also underscores the significance of ongoing monitoring, logging, and feedback loops for system reliability and performance [5].

(4) The advent of DevOps has given rise to a multitude of tools that augment various aspects of the software development lifecycle. These tools can be categorized into areas such as Collaboration and Knowledge Sharing (e.g., Slack, Trello, GitLab wiki), Source Code Management (e.g., GitHub, GitLab), Build Process (e.g., Maven), Continuous Integration (e.g., Jenkins, GitLab CI), Deployment Automation (e.g., Kubernetes, Docker), and Monitoring and Logging (e.g., Prometheus, Logstash). Their collective contribution enhances the overall efficiency and effectiveness of the development process [5].

(5) DevOps has broadened the role of software developers to include operational responsibilities, enhancing software quality and efficiency. The principles of DevOps are now applied to automate Machine Learning (ML) processes, known as MLOps, leading to more efficient and scalable ML workflows. This paradigm shift emphasizes collaboration, automation, and continuous improvement, influencing both software development and ML operations [5].

III. METHODOLOGYS

To gain insights from academic knowledge and industry expertise, we use a comprehensive mixed-method approach, as depicted in Figure 1. This involves sequential steps to ensure a thorough understanding of the subject.

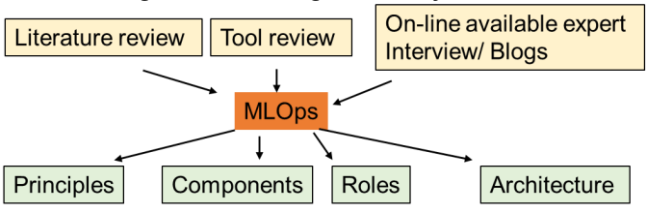

Figure 1. Overview of the methodology.

A. **Literature review:** We begin by conducting a systematic literature review using of scholarly articles to explore and gather insights from existing academic research and publications.

The approach involved an exploratory search phase, leading to the development of a targeted search query: (("DevOps" OR "CICD" OR "Continuous Integration" OR "Continuous Delivery" OR "Continuous Deployment") AND "Machine Learning") OR "MLOps" OR "CD4ML." We searched across databases like Google Scholar and Science Direct, and the library. Our study found various articles and we have thoroughly investigated 9 articles. This initial phase establishes the foundation for understanding the current state of knowledge within the field.

B. **Evaluation of tooling support:** Furthermore, we conducted a thorough assessment of relevant tools and technological support in the MLOps domain. This evaluation enhances our understanding of the crucial technical components and solutions integral to MLOps.

C. **Collecting experts' interview from on-line/Blogs:** Gather insights from online expert interviews spanning diverse domains and review relevant blogs. This information will complement and support the findings in the literature review.

Building on insights from three phases, we conceptualize 'MLOps.' The next chapter, 'Results,' elaborates on findings from the literature review and other information.

IV. RESULTS

We systematically employed our defined methodology, resulting in a structured synthesis of key principles, components, roles, and an architectural framework. This synthesis leads to a refined understanding and a comprehensive definition of MLOps. A principle in the context of MLOps is comparable to a foundational truth or a guiding standard, akin to 'best practices' in professional settings. These principles serve as blueprints for implementing strategies and processes in the MLOps domain.

A. Identified nine core principles crucial for the successful execution of MLOp [4, 6]:

P1: CI/CD automation principle: This pivotal principle in MLOps focuses on the streamlined automation of Continuous Integration and Continuous Delivery (CI/CD). It involves automating processes such as building, testing, and deploying code, providing rapid feedback to enhance developer efficiency.

P2: Workflow orchestration principle: This principle emphasizes strategic coordination and management of tasks within an ML workflow. Using Directed Acyclic Graphs (DAGs) for task sequences, it ensures a coherent and efficient workflow by considering dependencies and relationships between tasks.

P3: Reproducibility principle: Reproducibility in MLOps involves duplicating ML experiments consistently to achieve the same results. Critical for validating model reliability and accuracy, this principle ensures the dependability of ML models.

P4: Versioning principle: Fundamental in MLOps, effective version control of data, models, and code supports reproducibility and traceability. It is crucial for compliance,

auditing, and understanding the evolution of models and datasets.

P5: Collaboration principle: Highlighting the importance of interdisciplinary collaboration among different teams, this principle promotes a culture of communication and teamwork beyond technical boundaries.

P6: Continuous ML training evaluation principle: This principle underscores the ongoing importance of model training and evaluation. It involves regular retraining with new data, supported by monitoring components and feedback mechanisms to maintain model quality.

P7: ML metadata tracking/logging principle: Essential for comprehensive record-keeping, this principle involves tracking and logging metadata for every aspect of the ML workflow, including training jobs, model parameters, and lineage information.

P8: Continuous monitoring principle: Regular monitoring of data quality, model performance, code changes, and infrastructure resources is crucial. This principle ensures early detection of issues or changes affecting ML product quality.

P9: Feedback loops principle: Critical in MLOps, this principle involves implementing multiple feedback loops to integrate insights from different stages of the ML lifecycle. It enables continuous improvement, such as adjusting model training based on feedback from performance monitoring.

These principles shape the construction of an MLOps architecture, influencing component design and role delineation. The resulting architecture is comprehensive, facilitating ML operations from data preparation to model deployment and monitoring. It fosters a collaborative environment with integral elements such as continuous integration, monitoring, and feedback in the workflow. This architecture represents the essence of MLOps, representing a structured approach to ML project management. It prioritizes scalability, adaptability, and efficiency, ensuring the development, deployment, and maintenance of ML models align with high standards of quality and performance. Emphasizing a methodical and integrated approach, it reflects the dynamic and complex nature of managing machine learning operations.

B. Technical Components in MLOps

To establish a robust MLOps system, integrating various technical components is essential, each contributing uniquely to the overall functionality and efficiency of machine learning operations. Aligned with identified principles as above, these components ensure a cohesive MLOps environment [7].

CI/CD Component (Principles: P1, P6, P9): The CI/CD component is fundamental, enabling continuous integration, delivery, and deployment. It automates building, testing, and deploying ML models, providing immediate feedback on pipeline stages. Tools like Jenkins and GitHub Actions exemplify this component.

Source code repository (Principles: P4, P5): Serving as the foundation for collaborative coding, it ensures efficient version control and code management. Platforms like Bitbucket, GitHub, and Gitea support code storage and collaboration.

Workflow orchestration component (Principles: P2, P3, P6): Responsible for strategic task management within an ML workflow, it orchestrates tasks using Directed Acyclic Graphs. Tools like Apache Airflow and AWS Sage Maker Pipelines manage complex ML workflows.

Feature store system (Principles: P3, P4): The feature store system acts as a centralized repository for ML features, facilitating efficient management and reuse. Examples include Google Feast and Amazon AWS Feature Store.

Model training infrastructure (Principle: P6): This provides computational resources for model training, prioritizing scalability. Kubernetes and Red Hat OpenShift support scalable and distributed computing.

Model registry (Principles: P3, P4): The model registry serves as a central hub for storing trained ML models and metadata. Solutions like MLflow and AWS SageMaker Model Registry offer advanced capabilities.

ML metadata stores (Principles: P4, P7): Crucial for tracking metadata across ML workflows, these stores record detailed information about training jobs. Platforms like Kubeflow Pipelines and AWS SageMaker Pipelines provide comprehensive tracking.

Model serving component (Principle: P1): Tailored for deploying ML models, it often uses REST APIs. Kubernetes and Docker containerize models, while services like Microsoft Azure ML REST API facilitate model serving.

Monitoring component (Principles: P8, P9): Essential for continuous performance monitoring, it tracks aspects like model accuracy and infrastructure health. Monitoring solutions like Prometheus with Grafana ensure ongoing evaluation of ML systems.

C. Essential Roles in MLOps

Successful MLOps implementation relies on orchestrated specialized roles, each contributing critical expertise to ML projects. Exploring these roles and their interdependencies [8, 9]:

R1: Business stakeholder (Aligned roles: Product owner, Project manager)

(1) Key in setting strategic direction for ML initiatives.
(2) Articulates business objectives and expected outcomes.
(3) Communicates ML product value proposition and ROI.
(4) Ensures alignment between business goals and ML capabilities.

R2: Solution architect (Similar role: IT Architect)

(1) Designs overarching system architecture for ML projects.
(2) Selects technologies and frameworks through comprehensive evaluation.
(3) Makes critical decisions on component integration.
(4) Ensures architecture supports all roles in the ML lifecycle.

R3: Data scientist (Aligned roles: ML specialist, ML developer)

(1) Translates business problems into ML problems.
(2) Designs and develops ML models, selects algorithms, and hyperparameters.
(3) Applies data analysis and ML techniques for actionable insights.

R4: Data engineer (Similar role: DataOps engineer)

(1) Constructs and maintains data pipelines for efficient data flow.
(2) Ensures efficient data ingestion, processing, and storage.
(3) Supports data scientists with high-quality data for training and evaluation.

R5: Software engineer

(1) Transforms ML models into well-engineered software products.
(2) Applies software engineering principles for robust, scalable applications.
(3) Writes clean, efficient code adhering to coding guidelines.

R6: DevOps engineer

(1) Links development and operational aspects in MLOps.
(2) Automates and optimizes the CI/CD pipeline for rapid and reliable deployment.
(3) Manages orchestration of ML workflows and continuous monitoring.

R7: ML engineer/MLOps engineer

(1) Possesses cross-disciplinary knowledge in data science, engineering, and operations.
(2) Manages ML infrastructure, automates workflows, deploys models, and monitors both models and infrastructure.
(3) Ensures smooth functioning of the entire ML lifecycle.

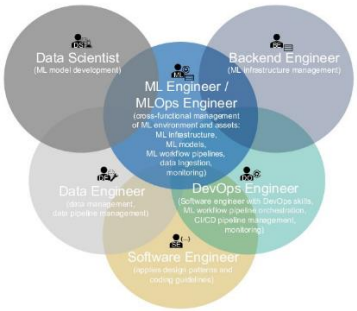

Figure 2. Roles and their intersections contributing to the MLOps paradigm [4].

D. Architectural Workflow

The architectural framework and workflow in Machine Learning Operations (MLOps) play a pivotal role in shaping the future of automated machine learning systems. Figure 4 visualizes this comprehensive design, strategically amalgamating technological components, roles, and operational methodologies, designed to be technology neutral. This neutrality empowers ML researchers and practitioners to customize their technological stack based on unique project demands.

The figure details the roles within the MLOps paradigm, illustrating the complexity and interdependencies of these roles for successful implementation. The architecture outlines an end-to-end MLOps process, from project conceptualization to final model deployment. This process comprises several key phases:

MLOps project initiation: Strategic decisions are made, including defining project goals, aligning with business strategies, and outlining the technical roadmap. Collaboration between business stakeholders, solution architects, data scientists, and data engineers is critical.

Feature engineering pipeline development: Data engineers establish rules for data transformation, collaborating with data scientists to develop sophisticated feature engineering strategies, transforming raw data into a format suitable for model training.

Experimentation and model refinement: Data scientists and ML specialists iteratively test, train, and validate models. Feedback informs adjustments to model parameters and feature engineering strategies.

Automated ML workflow and model deployment: Managed by DevOps and ML engineers, this phase involves automating tasks, setting up infrastructure, and deploying models efficiently, often utilizing technologies like Kubernetes for container orchestration.

Continuous monitoring and adaptive feedback loop: Post-deployment, continuous monitoring tracks metrics to ensure optimal model performance. An adaptive feedback loop facilitates quick issue resolution, transferring insights back to model development.

Iterative retraining and model evolution: Automated iterative retraining adapts models to new data or changes, ensuring relevance and accuracy over time.

Shown in Figure 3 is an exhaustive process that spans the complete path from the inception of MLOps products to model serving. This includes (A) the stages of MLOps product initiation; (B) the feature engineering pipeline, which involves ingesting data into the feature store; (C) the experimentation phase; and (D) the automated ML workflow pipeline culminating in model serving.

(A) MLOps project initiation: The business stakeholder (R1) evaluates the business landscape, identifying potential ML solutions. The solution architect (R2) designs the ML system architecture, making technology decisions after rigorous evaluation. The data scientist (R3) translates business

goals into specific ML problems, collaborating with the data engineer (R4) to understand data requirements. Together, they identify raw data sources, assess data distribution, quality, and conduct validation, ensuring labeled data availability for supervised ML due to prior labeling processes.

(B1) Requirements for feature engineering pipeline: The data engineer (R4) establishes data transformation and cleaning rules for preprocessing. Collaboratively, the data scientist (R3) and data engineer (R4) define feature engineering rules, subject to iterative adjustments based on feedback from experimental model engineering or monitoring component observations.

(B2) Feature engineering pipeline: Using initial requirements, the data engineer (R4) and software engineer (R5) construct a prototype, incorporating iterative updates from feedback. The data engineer (R4) defines essential code for CI/CD (C1) and orchestrates task orchestration within the feature engineering pipeline (C3). The pipeline connects to raw data sources, initiates data extraction, preprocessing, and computes new features based on existing ones. A data ingestion job loads data into the feature store system (C4).

(C) Experimentation: In this stage, the data scientist (R3) leads tasks, connecting to the feature store system (C4) or raw data for analysis. Necessary data adjustments are promptly communicated to the data engineering zone, maintaining a vital feedback loop.

(D) Automated ML workflow pipeline: Managed by the DevOps engineer (R6) and ML engineer (R7), this stage oversees the model training infrastructure. The workflow orchestration component (C3) orchestrates tasks, retrieving artifacts from the artifact store and collecting metadata. Automated tasks include extraction of versioned features, data preparation, model training, evaluation, and export. The trained model is pushed to the model registry (C6), recorded in the ML metadata store (C7).

Once a well-performing model transitions to production, the CI/CD component (C1) triggers the continuous deployment pipeline. The model serving component (C8), configured by the software engineer (R5) and managed by the ML engineer (R7), makes predictions. The monitoring component (C9) observes real-time performance, forwarding feedback via the feedback loop to support continuous training, retraining, and improvement. The feedback loop connects to upstream points, including the experimental stage and data engineering zone, enabling adjustments based on concept drift detection for continuous training.

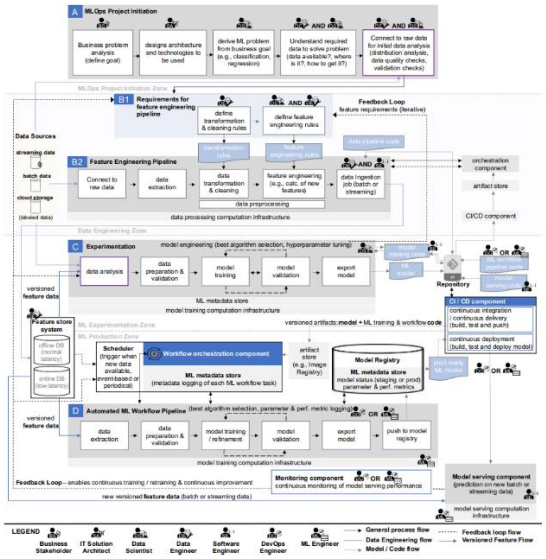

Figure. 3. End-to-end MLOps architecture and workflow with functional (Ref. Machine Learning Operations (MLOps)) [9].

CONCLUSION

The research extensively delved into the MLOps paradigm, aiming to bridge the gap between theoretical advancements in machine learning and their practical implementation in production environments. Following conclusion can be drawn from the literature study:

(1) Through a applied methodology involving systematic literature review, tool assessment, and on-line interview/blogs, the study provided valuable insights into the challenges and nuances of automating and operationalizing ML products.

(2) A key finding highlighted the paramount importance of continuous integration and deployment (CI/CD) in ML projects, addressing the dynamic nature of ML models and the necessity for frequent updates to adapt to evolving data.

(3) The paper emphasized that MLOps transcends technology, emphasizing its dependence on cultural and practical shifts within teams. The successful implementation of MLOps requires a collaborative mindset, breaking down silos between data scientists, engineers, and operational staff.

(4) Reproducibility, versioning, and collaboration emerged as crucial principles in MLOps, with the study emphasizing their significance in maintaining the integrity and traceability of ML models. These principles play a pivotal role in compliance, auditing, and ensuring continuous improvement in ML projects.

References

[1] T. Ahmad, R. Madonski, D. Zhang, C. Huang, A. Mujeeb, Data-driven probabilistic machine learning in sustainable smart energy/smart energy systems: Key developments, challenges, and future research opportunities in the context of smart grid paradigm, Renewable and Sustainable Energy Reviews 160 (2022) 112128.

[2] L.E. Lwakatare, A. Raj, I. Crnkovic, J. Bosch, H.H. Olsson, Large-scale machine learning systems in real-world industrial settings: A review of challenges and solutions, Information and Software Technology 127 (2020) 106368.

[3] V. Sebastian, A.M. Bilal, B. Gergo, J.K. Franz, G. Rayid, J. Pall, C. Sarah, J. Adrian, S.L.M. Katherine, M. Puja, G. David, B. Mark, B. Richard, G.M.M. Karel, S.C. Gary, P.A.I. John, H. Chris, H. Harry, Machine learning and artificial intelligence research for patient benefit: 20 critical questions on transparency, replicability, ethics, and effectiveness, BMJ 368 (2020) l6927.

[4] D. Kreuzberger, N. Kühl, S. Hirschl, Machine Learning Operations (MLOps): Overview, Definition, and Architecture, IEEE Access 11 (2022) 31866-31879.

[5] M. Muñoz, O. Díaz, DevOps: Foundations and Its Utilization in Data Center, in: J. Marx Gómez, M. Mora, M.S. Raisinghani, W. Nebel, R.V. O'Connor (Eds.), Engineering and Management of Data Centers: An IT Service Management Approach, Springer International Publishing, Cham, 2017, pp. 205-225.

[6] I. Karamitsos, S. Albarhami, C. Apostolopoulos, Applying DevOps Practices of Continuous Automation for Machine Learning, Information, 2020.

[7] M. Ali, Machine Learning, Pipelines, Deployment and MLOps Tutorial, (2022).

[8] D. Kreuzberger, N. Kühl, S. Hirschl, Machine Learning Operations (MLOps): Overview, Definition, and Architecture, IEEE Access 11 (2023) 31866-31879.

[9] M.M. John, H.H. Olsson, J. Bosch, Towards MLOps: A Framework and Maturity Model, 2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2021, pp. 1-8.