Google Cloud

White paper
May 2021

Practitioners guide to MLOps: A framework for continuous delivery and automation of machine learning.

Authors:
Khalid Salama,
Jarek Kazmierczak,
Donna Schut

Table of Contents

Executive summary

Across industries, DevOps and DataOps have been widely adopted as methodologies to improve quality and reduce the time to market of software engineering and data engineering initiatives. With the rapid growth in machine learning (ML) systems, similar approaches need to be developed in the context of ML engineering, which handle the unique complexities of the practical applications of ML. This is the domain of MLOps. MLOps is a set of standardized processes and technology capabilities for building, deploying, and operationalizing ML systems rapidly and reliably.]

We previously published Google Cloud's AI Adoption Framework to provide guidance for technology leaders who want to build an effective artificial intelligence (AI) capability in order to transform their business. That framework covers AI challenges around people, data, technology, and process, structured in six different themes: *learn, lead, access, secure, scale,* and *automate*.

The current document takes a deeper dive into the themes of *scale* and *automate* to illustrate the requirements for building and operationalizing ML systems. *Scale* concerns the extent to which you use cloud managed ML services that scale with large amounts of data and large numbers of data processing and ML jobs, with reduced operational overhead. *Automate* concerns the extent to which you are able to deploy, execute, and operate technology for data processing and ML pipelines in production efficiently, frequently, and reliably.

We outline an MLOps framework that defines core processes and technical capabilities. Organizations can use this framework to help establish mature MLOps practices for building and operationalizing ML systems. Adopting the framework can help organizations improve collaboration between teams, improve the reliability and scalability of ML systems, and shorten development cycle times. These benefits in turn drive innovation and help gain overall business value from investments in ML.

This document is intended for technology leaders and enterprise architects who want to understand MLOps. It's also for teams who want details about what MLOps looks like in practice. The document assumes that readers are familiar with basic machine learning concepts and with development and deployment practices such as CI/CD.

The document is in two parts. The first part, an overview of the MLOps lifecycle, is for all readers. It introduces MLOps processes and capabilities and why they're important for successful adoption of ML-based systems.

The second part is a deep dive on the MLOps processes and capabilities. This part is for readers who want to understand the concrete details of tasks like running a continuous training pipeline, deploying a model, and monitoring predictive performance of an ML model.

Organizations can use the framework to identify gaps in building an integrated ML platform and to focus on the scale and automate themes from Google's AI Adoption Framework. The decision about whether (or to which degree) to adopt each of these processes and capabilities in your organization depends on your business context. For example, you must determine the business value that the framework creates when compared to the cost of purchasing or building capabilities (for example, the cost in engineering hours).

Overview of MLOps lifecycle and core capabilities

Despite the growing recognition of AI/ML as a crucial pillar of digital transformation, successful deployments and effective operations are a bottleneck for getting value from AI. Only one in two organizations has moved beyond pilots and proofs of concept. Moreover, 72% of a cohort of organizations that began AI pilots before 2019 have not been able to deploy even a single application in production.[1] Algorithmia's survey of the state of enterprise machine learning found that 55% of companies surveyed have not deployed an ML model.[2] To summarize: models don't make it into production, and if they do, they break because they fail to adapt to changes in the environment.

This is due to a variety of issues. Teams engage in a high degree of manual and one-off work. They do not have reusable or reproducible components, and their processes involve difficulties in handoffs between data scientists and IT. Deloitte identified lack of talent and integration issues as factors that can stall or derail AI initiatives.[3] Algorithmia's survey highlighted that challenges in deployment, scaling, and versioning efforts still hinder teams from getting value from their investments in ML. Capgemini Research noted that the top three challenges faced by organizations in achieving deployments at scale are lack of mid- to senior-level talent, lack of change-management processes, and lack of strong governance models for achieving scale.

The common theme in these and other studies is that ML systems cannot be built in an ad hoc manner, isolated from other IT initiatives like DataOps and DevOps. They also cannot be built without adopting and applying sound software engineering practices, while taking into account the factors that make operationalizing ML different from operationalizing other types of software.

Organizations need an automated and streamlined ML process. This process does not just help the organization successfully deploy ML models in production. It also helps manage risk when organizations scale the number of ML applications to more use cases in changing environments, and it helps ensure that the applications are still in line with business goals. McKinsey's Global Survey on AI found that having standard frameworks and development

[1] The AI-powered enterprise, CapGemini Research Institute, 2020.
[2] 2020 state of enterprise machine learning, Algorithmia, 2020.
[3] Artificial intelligence for the real world, Deloitte, 2017.
[4] The state of AI in 2020, McKinsey, 2020.

processes in place is one of the differentiating factors of high-performing ML teams.[4]

This is where ML engineering can be essential. ML engineering is at the center of building ML-enabled systems, which concerns the development and operationalizing of production-grade ML systems. ML engineering provides a superset of the discipline of software engineering that handles the unique complexities of the practical applications of ML.[5] These complexities include the following:

- Preparing and maintaining high-quality data for training ML models.
- Tracking models in production to detect performance degradation.
- Performing ongoing experimentation of new data sources, ML algorithms, and hyperparameters, and then tracking these experiments.
- Maintaining the veracity of models by continuously retraining them on fresh data.
- Avoiding training-serving skews that are due to inconsistencies in data and in runtime dependencies between training environments and serving environments.
- Handling concerns about model fairness and adversarial attacks.

MLOps is a methodology for ML engineering that unifies ML system development (the ML element) with ML system operations (the Ops element). It advocates formalizing and (when beneficial) automating critical steps of ML system construction. MLOps provides a set of standardized processes and technology capabilities for building, deploying, and operationalizing ML systems rapidly and reliably.

MLOps supports ML development and deployment in the way that DevOps and DataOps support application engineering and data engineering (analytics). The difference is that when you deploy a web service, you care about resilience, queries per second, load balancing, and so on. When you deploy an ML model, you also need to worry about changes in the data, changes in the model, users trying to game the system, and so on. This is what MLOps is about.

MLOps practices can result in the following benefits over systems that do not follow MLOps practices:

- Shorter development cycles, and as a result, shorter time to market.
- Better collaboration between teams.
- Increased reliability, performance, scalability, and security of ML systems.
- Streamlined operational and governance processes.
- Increased return on investment of ML projects.

In this section, you learn about the MLOps lifecycle and workflow, and about the individual capabilities that are re-

[5] Towards ML Engineering, Google, 2020.

quired for a robust MLOps implementation.

Building an ML-enabled system

Building an ML-enabled system is a multifaceted undertaking that combines data engineering, ML engineering, and application engineering tasks, as shown in figure 1.

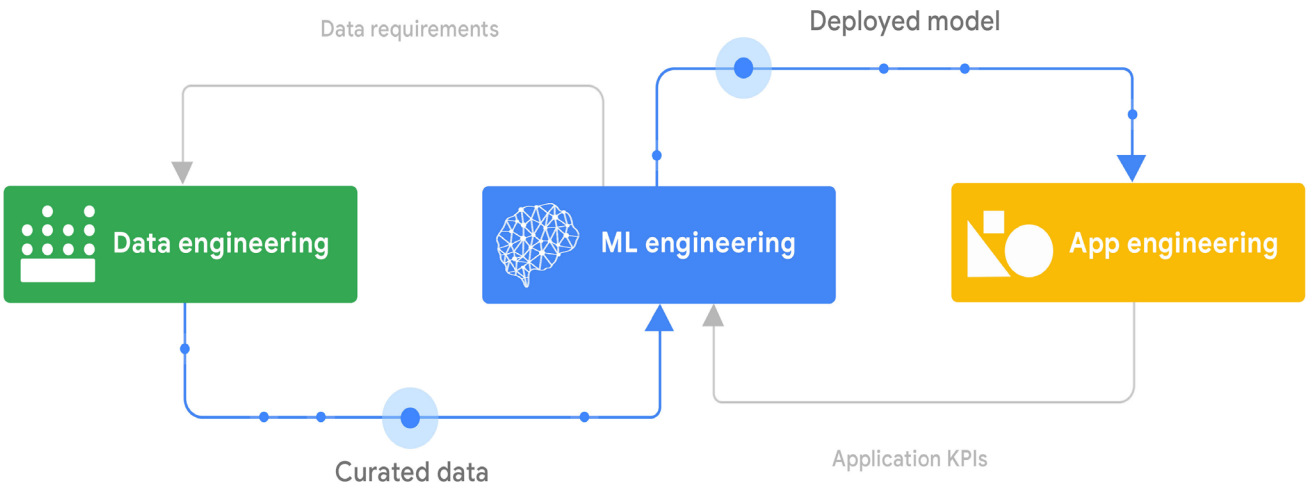

Figure 1. The relationship of data engineering, ML engineering, and app engineering

Data engineering involves ingesting, integrating, curating, and refining data to facilitate a broad spectrum of operational tasks, data analytics tasks, and ML tasks. Data engineering can be crucial to the success of the analytics and ML initiatives. If an organization does not have robust data engineering processes and technologies, it might not be set up for success with downstream business intelligence, advanced analytics, or ML projects.

ML models are built and deployed in production using curated data that is usually created by the data engineering team. The models do not operate in silos; they are components of, and support, a large range of application systems, such as business intelligence systems, line of business applications, process control systems, and embedded systems. Integrating an ML model into an application is a critical task that involves making sure first that the deployed model is used effectively by the applications, and then monitoring model performance. In addition to this, you should also collect and monitor relevant business KPIs (for example, click-through rate, revenue uplift, and user experience). This information helps you understand the impact of the ML model on the business and adapt accordingly.

The MLOps lifecycle

The MLOps lifecycle encompasses seven integrated and iterative processes, as shown in figure 2.

Figure 2. The MLOps lifecycle

The processes can consist of the following:

- **ML development** concerns experimenting and developing a robust and reproducible model training procedure (training pipeline code), which consists of multiple tasks from data preparation and transformation to model training and evaluation.
- **Training operationalization** concerns automating the process of packaging, testing, and deploying repeatable and reliable training pipelines.
- **Continuous training** concerns repeatedly executing the training pipeline in response to new data or to code changes, or on a schedule, potentially with new training settings.
- **Model deployment** concerns packaging, testing, and deploying a model to a serving environment for online experimentation and production serving.

- **Prediction serving** is about serving the model that is deployed in production for inference.
- **Continuous monitoring** is about monitoring the effectiveness and efficiency of a deployed model.
- **Data and model management** is a central, cross-cutting function for governing ML artifacts to support auditability, traceability, and compliance. Data and model management can also promote shareability, reusability, and discoverability of ML assets.

MLOps: An end-to-end workflow

Figure 3 shows a simplified but canonical flow for how the MLOps processes interact with each other, focusing on high-level flow of control and on key inputs and outputs.

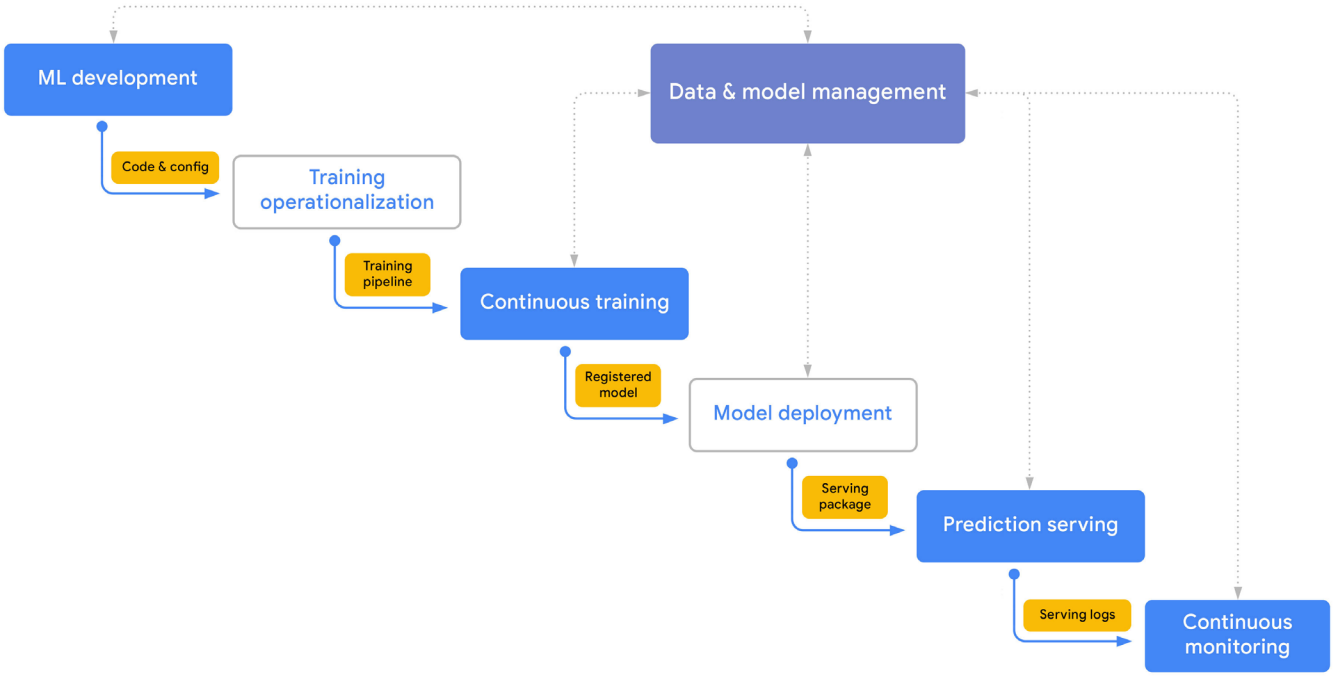

Figure 3. The MLOps process

This is not a waterfall workflow that has to sequentially pass through all the processes. The processes can be skipped, or the flow can repeat a given phase or a subsequence of the processes. The diagram shows the following flow:

1. The core activity during this ML development phase is experimentation. As data scientists and ML researchers prototype model architectures and training routines, they create labeled datasets, and they use features and other reusable ML artifacts that are governed through the data and model management process. The

primary output of this process is a formalized training procedure, which includes data preprocessing, model architecture, and model training settings.

2. If the ML system requires continuous training (repeated retraining of the model), the training procedure is operationalized as a training pipeline. This requires a CI/CD routine to build, test, and deploy the pipeline to the target execution environment.
3. The continuous training pipeline is executed repeatedly based on retraining triggers, and it produces a model as output. The model is retrained as new data becomes available, or if model performance decay is detected. Other training artifacts and metadata that are produced by a training pipeline are also tracked. If the pipeline produces a successful model candidate, that candidate is then tracked by the model management process as a registered model.
4. The registered model is annotated, reviewed, and approved for release and is then deployed to a production environment. This process might be relatively opaque if you are using a no-code solution, or it can involve building a custom CI/CD pipeline for progressive delivery.
5. The deployed model serves predictions using the deployment pattern that you have specified: online, batch, or streaming predictions. In addition to serving predictions, the serving runtime can generate model explanations and capture serving logs to be used by the continuous monitoring process.
6. The continuous monitoring process monitors the model for predictive effectiveness and service. The primary concern of effectiveness performance monitoring is detecting model decay—for example, data and concept drift. The model deployment can also be monitored for efficiency metrics like latency, throughput, hardware resource utilization, and execution errors.

MLOps capabilities

To effectively implement the key MLOps processes outlined in the previous section, organizations need to establish a set of core technical capabilities. These capabilities can be provided by a single integrated ML platform. Alternatively, they can be created by combining vendor tools that each are best suited to particular tasks, developed as custom services, or created as a combination of these approaches.

In most cases, the processes are deployed in stages rather than all at once in a single deployment. An organization's plan for adopting these processes and capabilities should align with business priorities and with the organization's technical and skills maturity. For example, many organizations start by focusing on the processes for ML development, model deployment, and prediction serving. For these organizations, continuous training and continuous monitoring might not be necessary if they are piloting a relatively small number of ML systems.

Figure 4 shows the core set of technical capabilities that are generally required for MLOps. They are abstracted as functional components that can have many-to-many mappings to specific products and technologies.