Bridging the MLOps Gap: From AI Research to Production-Ready Systems

Presented by Bharath Reddy Baddam

Campbellsville University

Conf42 MLOps 2025

Today's Agenda

01

The MLOps Challenge

Exploring why many ML models don't make it to production.

02

ML Pipeline Orchestration

Automating ML pipelines using tools like Kubeflow, MLflow, and Airflow.

03

Production Infrastructure

Building scalable, cloud-native infrastructure for ML.

04

Monitoring & Observability

Ensuring model performance and quality in production.

05

CI/CD for ML

Adopting CI/CD for robust ML development and deployment.

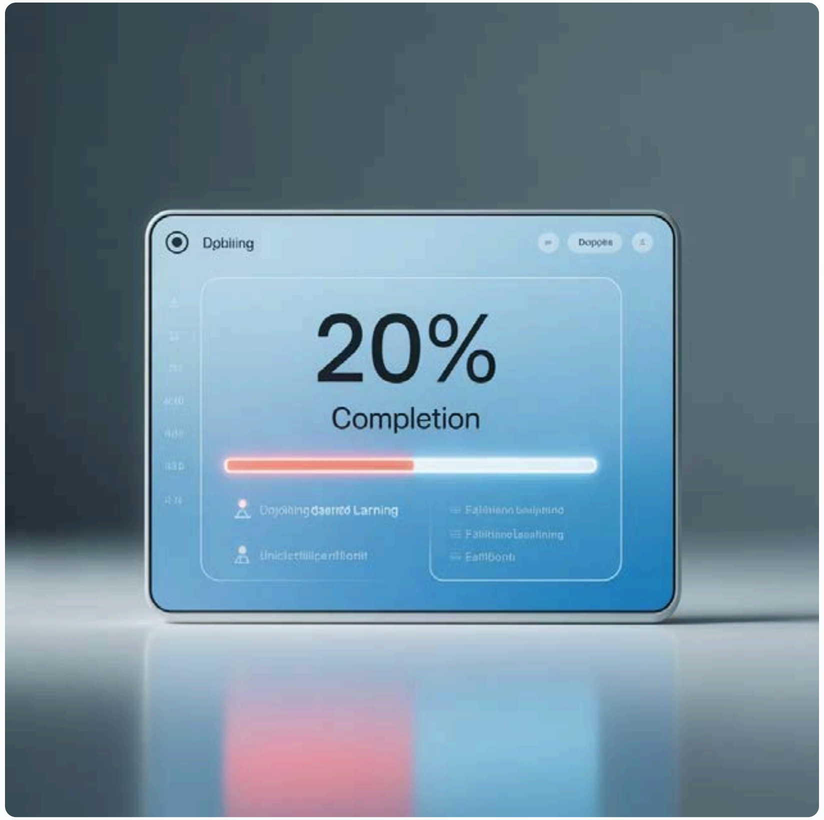
The MLOps Challenge

From Research to Production

Only **20%** of machine learning models successfully transition to production environments.

Key barriers contributing to this challenge include:

- Reproducibility gaps between research and engineering.
- Lack of standardized deployment processes.
- Insufficient monitoring and maintenance frameworks.
- Inadequate testing methodologies for ML systems.
- Poor integration with existing enterprise systems.



The MLOps Evolution

Manual ML Process

Data scientists operate in isolation, relying on notebooks and manual processes. Deployments are often ad-hoc and lack repeatability.

ML Pipeline Automation

Workflow tools and versioning are introduced, automating basic training but with limited production capabilities.

CI/CD for ML

ML workflows are integrated with DevOps practices, enabling automated testing and streamlined deployment pipelines.

Full MLOps

Achieving end-to-end automation from experimentation to production, including comprehensive monitoring, drift detection, and automated retraining.

Achieving MLOps maturity empowers organizations to reduce deployment time by **80%** and maintain over **95%** model accuracy in production.

ML Pipeline Orchestration

Automating the model lifecycle from training to deployment

ML Pipeline Components

Data Preparation

Cleaning, feature engineering, and validation of raw data before model training.

Model Training

Hyperparameter tuning, cross-validation, and performance metric tracking.

Model Evaluation

Rigorous testing against holdout datasets and validation against business KPIs.

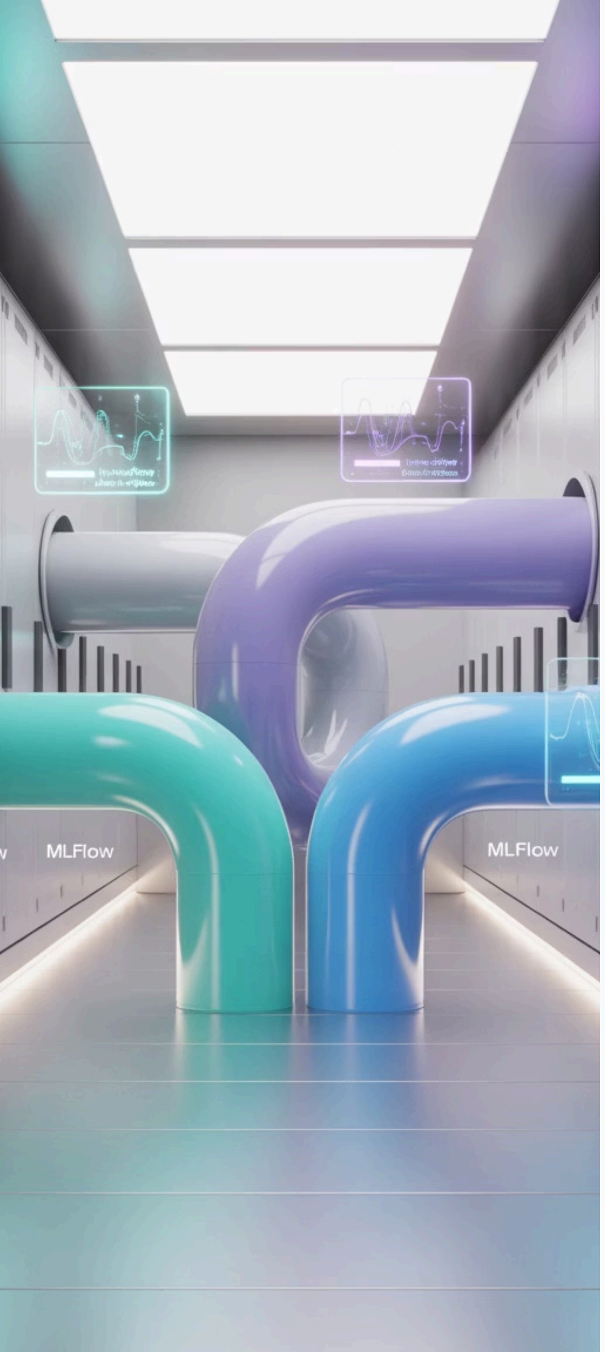
Deployment

Packaging models for inference and seamlessly deploying to target environments.

Monitoring

Continuous observation of model performance, data drift, and system health.

A fully automated pipeline reduces human intervention points and creates reproducible, auditable workflows, accelerating the journey from research to production.



Orchestration Tools Comparison

Kubeflow

- Kubernetes-native ML platform
- Integrated notebook environments
- Comprehensive pipeline component library
- Robust model serving capabilities

Best for: Organizations deeply integrated with Kubernetes

MLflow

- Centralized experiment tracking
- Versioned model registry
- Streamlined model deployment
- Language-agnostic design for broad compatibility

Best for: Teams prioritizing experimentation and version control

Apache Airflow

- Versatile workflow orchestration engine
- Extensive operator ecosystem
- Advanced dependency management
- Powerful scheduling for recurring tasks

Best for: Orchestrating complex, data-intensive ML pipelines

Production Infrastructure

Designing cloud-native ML systems that scale

Infrastructure as Code for ML Environments

- **Infrastructure Definition**

  Defining cloud resources using tools like Terraform, CloudFormation, or Pulumi.

- **Containerization**

  Packaging models and dependencies in versioned environments using Docker.

- **Deployment Automation**

  Automating the provisioning of consistent environments across development, testing, and production.

Benefits

- Reproducible environments eliminate "it works on my machine" issues.
- Enabling version control for infrastructure, aligning with model versioning.
- Facilitating easy rollback capabilities for both models and infrastructure.
- Cost optimisation through right-sizing and auto-scaling.

Real-time Inference Architectures

Design Considerations

High-throughput prediction systems require specialised architectures:

Horizontal Scaling

Kubernetes-based deployments with auto-scaling based on request volume

Load Balancing

Request distribution across multiple model servers with health checks

Caching Layers

Redis or in-memory caching for frequently requested predictions

Hardware Acceleration

GPU/TPU optimisation for deep learning models

Target latency: 50-200ms for real-time applications

Monitoring & Observability

Preventing model drift and maintaining production quality

Model Monitoring Systems

Data Drift Detection

Statistical monitoring of input distributions compared to training data

- KL divergence
- Population Stability Index
- Feature correlation changes

Model Performance Tracking

Continuous evaluation of prediction quality

- Accuracy, F1, AUC metrics
- Confusion matrix changes
- Business KPI impact

Operational Metrics

System health and performance indicators

- Latency percentiles
- Throughput measurements
- Resource utilisation

Automated Retraining and Deployment

Trigger-based Retraining

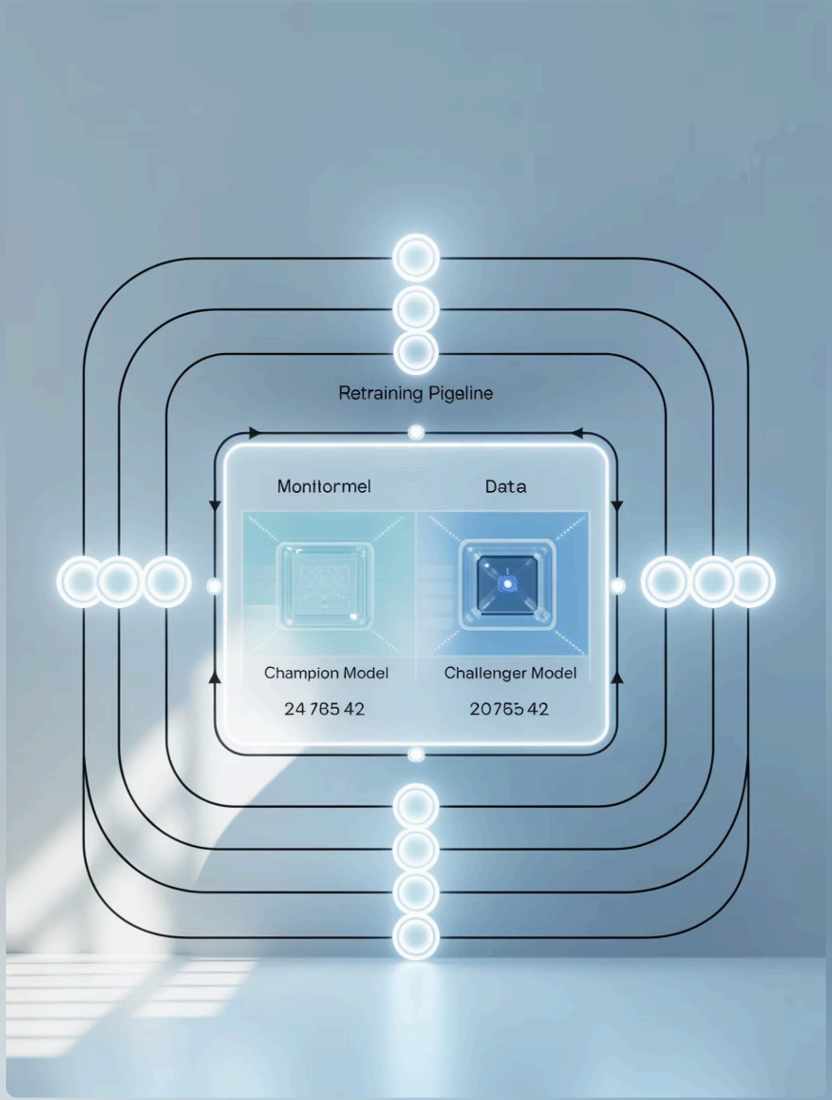
Automate model refreshing based on monitoring signals:

- **Performance-based:** When accuracy drops below threshold
- **Time-based:** Regular intervals (weekly/monthly)
- **Data-based:** When drift exceeds acceptable levels
- **Volume-based:** After processing N new examples

Champion-Challenger Deployment

Safely validate new models before full deployment:

- Deploy new model alongside existing one
- Shadow deployment (no live traffic)
- Canary deployment (small % of traffic)
- A/B testing for business impact
- Automated rollback if performance degrades



CI/CD for Machine Learning

Implementing continuous integration patterns for ML workflows

ML-Specific Testing Framework

1

Data Validation Tests

Verify data quality, schema compliance, and distribution characteristics using tools like TensorFlow Data Validation or Great Expectations

2

Model Performance Tests

Ensure model metrics meet minimum thresholds on validation data and test for regressions against previous versions

3

Integration Tests

Verify end-to-end pipeline functionality, including data preprocessing, training, and deployment steps

4

Load & Performance Tests

Assess prediction latency, throughput capacity, and resource consumption under simulated load conditions

5

Security & Compliance Tests

Check for data leakage, PII exposure, and compliance with regulatory requirements like GDPR or HIPAA

A comprehensive testing framework reduces deployment risk and maintains quality across the entire MLOps lifecycle, ensuring models are both technically sound and business-ready.

Thank You!