Kaggle | Google

5-Day Gen AI Intensive Course with Google 2025

Whitepaper Companion Podcast (Notes): Operationalizing Generative AI on Vertex AI using MLOps

Introduction

Generative AI presents unprecedented potential, but transforming this potential into reliable real-world applications requires robust operational frameworks. This document explores how MLOps principles can be adapted and applied to generative AI systems, with a specific focus on Google's Vertex AI platform.

Foundation Models vs. Traditional ML Models

- **Key differences**:
  - Foundation models are multi-purpose rather than task-specific
  - They exhibit emergent properties (capabilities not explicitly trained for)
  - They are extremely sensitive to input prompts
  - They often require adaptation rather than training from scratch

The Generative AI Lifecycle

The lifecycle consists of five key phases:

1. **Discover**
2. **Develop and Experiment**
3. **Evaluate**
4. **Deploy**

5. **Govern**

1. Discovery Phase

- **Challenge**: Navigating the explosion of available models (open-source, proprietary)
- **Key selection factors**:
    - Quality (benchmark scores, task-specific performance)
    - Latency (response time requirements)
    - Cost (infrastructure, hardware, software, usage)
    - Legal and compliance considerations
- **Solution**: Vertex Model Garden provides a curated collection with model cards detailing performance, use cases, and limitations

2. Development and Experimentation Phase

Prompted Model Components

- **Definition**: The combination of a foundation model and a prompt
- **Prompt Engineering**:
    - Prompts are highly sensitive inputs that significantly impact model outputs
    - Prompt templates provide structured ways to combine instructions and examples
    - **Dual nature of prompts**:
        - **Prompt as data**: Few-shot examples, knowledge bases, user queries
        - **Prompt as code**: Instructions, templates, guardrails
    - Requires version control and tracking which prompt versions work best with which model versions

Chaining and Augmentation

- **Purpose**: Address limitations of single prompted models

- **Implementation**: Connecting multiple prompted model components with external APIs and custom logic
- **Common patterns**:
    - **Retrieval Augmented Generation (RAG)**: Augments models with knowledge from external databases to ground responses in facts and reduce hallucinations
    - **Agents**: Use LLMs as decision-makers that can interact with different tools and take actions
- **MLOps implications**:
    - Evaluation must be end-to-end
    - Versioning must encompass the entire chain
    - Input distributions are harder to define upfront due to language's inherent complexity

Tuning and Training

- **Approaches**:
    - **Supervised Fine-Tuning**: Training on labeled datasets for specific tasks
    - **Reinforcement Learning from Human Feedback (RLHF)**: Using human feedback to train a reward model that guides the LLM
- **MLOps considerations**:
    - Track all artifacts (data, parameters, performance metrics)
    - Continuous tuning is often more practical than continuous training due to cost
    - Model quantization can help manage costs

3. Data Practices for Generative AI

- **Unique aspects**:
    - Wider range of data types (prompts, examples, grounding data, feedback)

    - Prototypes can be built with less initial data
    - Models can generate synthetic data for testing and augmentation
- **Challenges**:
    - Managing diverse data ecosystems
    - Unknown training data distributions of foundation models
    - Need for custom evaluation datasets that reflect specific use cases

4. Evaluation

- **Progression**: From manual evaluation in early stages to automated processes as projects mature
- **Challenges**:
    - Outputs are complex and high-dimensional
    - Defining "good" can be subjective
    - Lack of ground truth data
- **Approaches**:
    - Using foundation models as evaluators (Auto-Eval)
    - Testing against adversarial attacks
    - Custom metrics based on use case requirements (factual accuracy, coherence, creativity)

5. Deployment

- **Complexity**: Deploying entire systems rather than single models
- **Best practices**:
    - Version control for all components (prompts, chains, data)
    - CI/CD adapted for generative AI's unique challenges
    - Solutions for managing external data (BigQuery, Vertex Feature Store)

- **Foundation model deployment considerations**:
    - Massive compute and storage requirements
    - Model compression techniques
    - Scalable infrastructure

6. Monitoring and Logging

- **Requirements**: End-to-end tracking across chained components
- **Key concepts**:
    - **Skew detection**: Comparing evaluation data distribution with production data
    - **Drift detection**: Identifying changes in input data over time
    - **Continuous evaluation**: Capturing production outputs for ongoing assessment
    - **Tracing**: Recording event flows to understand component interactions

7. Governance

- **Scope**: Governing entire systems (prompts, chains, data sources)
- **Implementation**: Applying MLOps and DevOps practices with tools like Dataplex, Vertex ML Metadata, and Vertex Experiment

Agent Ops: The Next Frontier

Unique Challenges

- **Autonomy**: Agents make decisions and take actions without direct human intervention
- **External interactions**: Agents interact with various systems and data sources
- **Trust requirements**: Need robust governance, monitoring, and control

Key Concepts

- **Tool orchestration**: Managing the different tools agents use

- **Tool registry**: Centralized catalog for discovering and managing available tools
- **Tool selection strategies**:
    - Generalist approach (access to all tools)
    - Specialist approach (limited set of task-specific tools)
    - Dynamic approach (runtime selection based on relevance)

Evaluation and Optimization

- **Five-stage process**: From tool unit testing to operational metric evaluation
- **Observability and explainability**: Understanding what agents do and why
- **Memory management**:
    - Short-term memory (conversation history)
    - Long-term memory (past interactions)

Deployment Considerations

- Robust CI/CD pipeline
- Automated tool registration
- Continuous monitoring
- Iterative improvement loop

The Changing MLOps Landscape

- **New roles**: Prompt engineers, AI engineers, DevOps engineers
- **Unified platforms**: Vertex AI provides comprehensive functions from data preparation to governance
- **Future challenges**: Rapid innovation will continue to reshape MLOps practices and requirements

Conclusion

Operationalizing generative AI requires adapting existing MLOps principles while developing new approaches specific to foundation models and agents. Platforms like Vertex AI provide the infrastructure and tools needed to build, deploy, and manage these systems effectively, enabling organizations to harness the full potential of generative AI.