Google AI

Machine Learning with Google Cloud

Alexander Del Toro Barba, PhD

Machine Learning Specialist, Google Cloud

November 2024

Agenda

Google Cloud Platform
Agent Building Workflow

Workflow

1. Build a Knowledge Base
2. Parsing and Understanding
3. Chunk, Embedding & Search (Retrieval)
4. Rank & Optimization
5. Grounding & Function Calling
6. Guardrails
7. Caching & Quota Optimization
8. Training & Fine-Tuning
9. Evaluation
10. Serving (Inference)
11. Monitoring

Overview

Google AI Platform

Vertex AI

Vertex AI Agent Builder

OOTB and custom Agents | Search | Orchestration | Extensions | Connectors | Document Processors | Retrieval engines | Rankers | Grounding

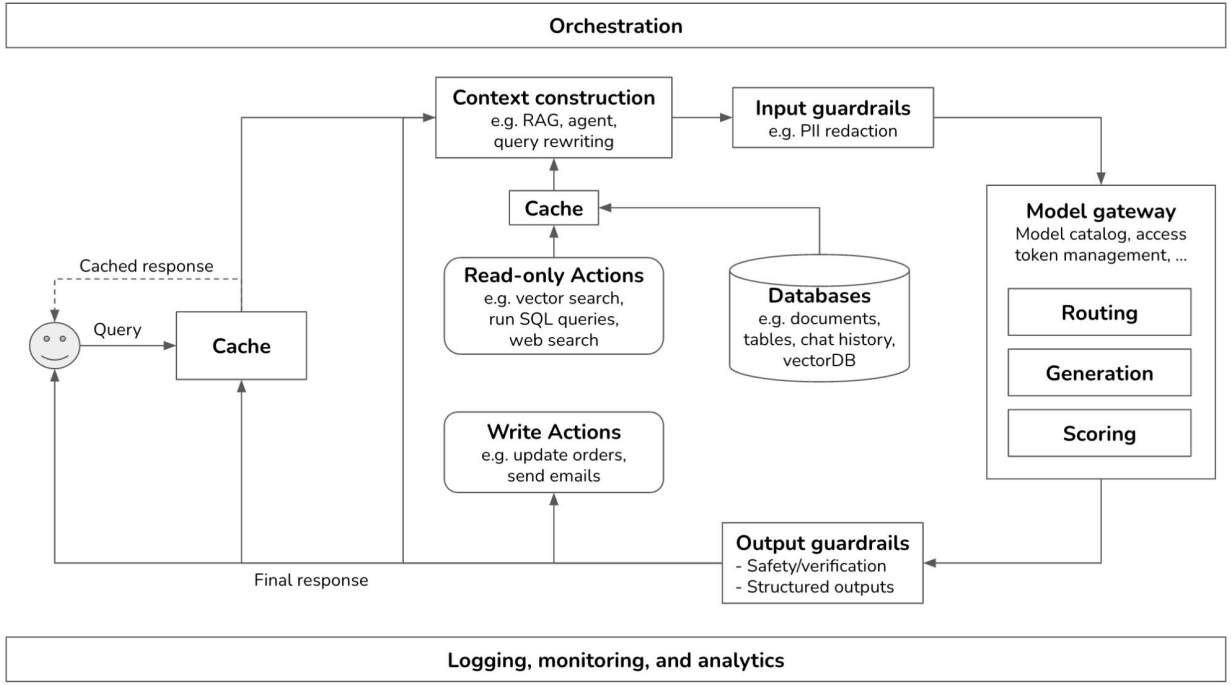
Vertex AI Model Builder

Prompt | Serve | Tune | Distill | Eval | Notebooks I Training | Feature Store | Pipelines | Monitoring

Vertex AI Model Garden

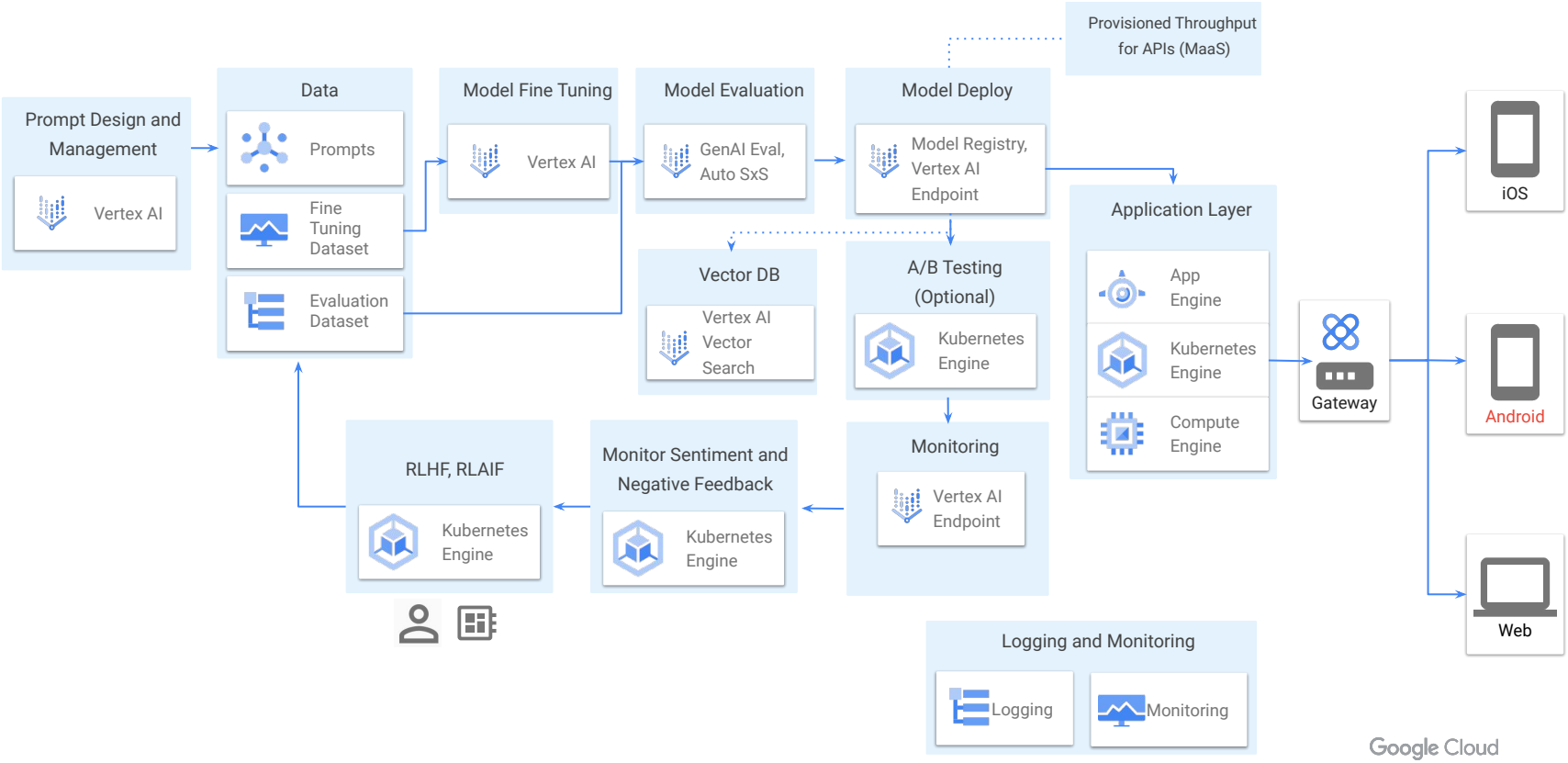
Google | Open | Partner

AI Gateway Platform



https://huyenchip.com/2024/07/25/genai-platform.html#gateway

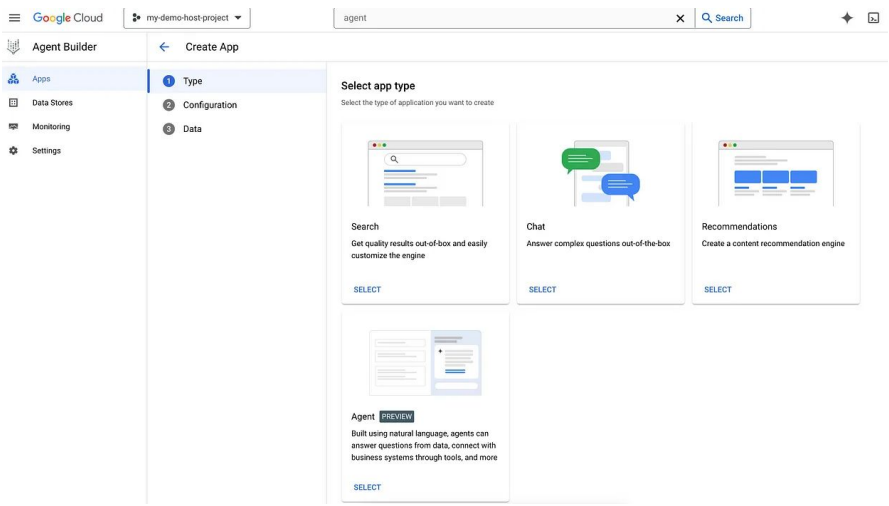
Generative AI with Google



Orchestrators

Orchestrators to build Agents

- Vertex AI Agent Builder as low-code / no-code managed
  - Code: https://dialogflow.cloud.google.com/v2/
- LangChain on Vertex AI (“Reasoning Engine”)
  - Code: Build + deploy LangChain app on Cloud Run
  - Code: Build and Deploy a LangGraph Application
- RAG Engine on Vertex AI (formerly: “Llama Index, Knowledge Engine”)
- Firebase Genkit (with Vertex AI Evaluation plug-in)



Generative AI Workflow

Agent Building Workflow

Preparation

Build a knowledge base
Crawler, File, DB, Connector

How do I get my data, from wherever it is, into the pipeline?

Parsing / Understanding
DocAI

How process data to extract tables, images etc and to make it easier to find later

Chunking, Embedding and Search (Retrieval)

How should I segment while preserving meaning? Which vector dimensions?

Ranking & Optimization

Boost fresh results, rank based on user behavior, maximize business KPIs e.g. clicks / revenue

Grounding & Function Calling

Prompt Engineering, Tuning, Citations, Agentic iterations, Actions

Runtime

Guardrails for Safety

Rules, Content moderation, Prompt and Model Security

Caching & Quota Optimization

Reduce cost and latency with prompt caching and dynamic shared quota

Training and Fine-tuning

Full finetuning, parameter-efficient tuning

Evaluation

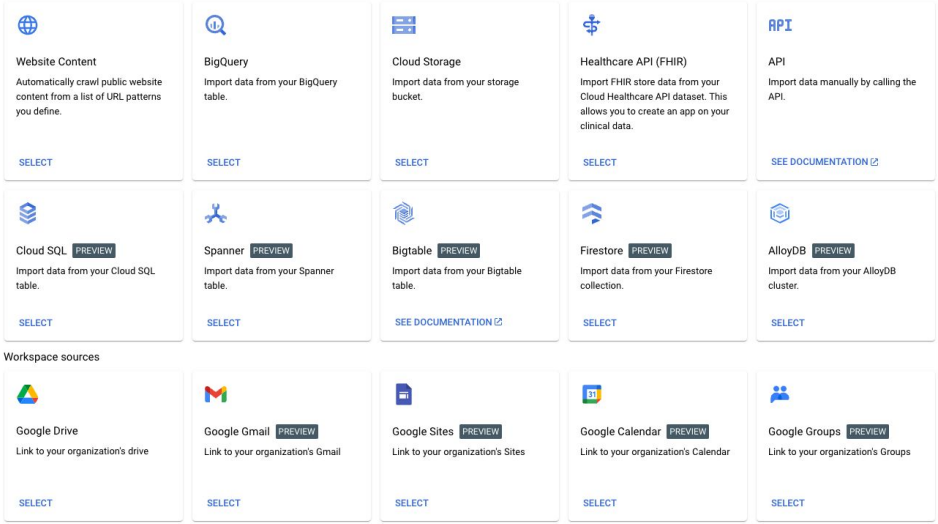
Full finetuning, parameter-efficient tuning

Serving & Monitoring
Cloud host

Will my serving API scale to demand? Is it secure?

Build a Knowledge Base

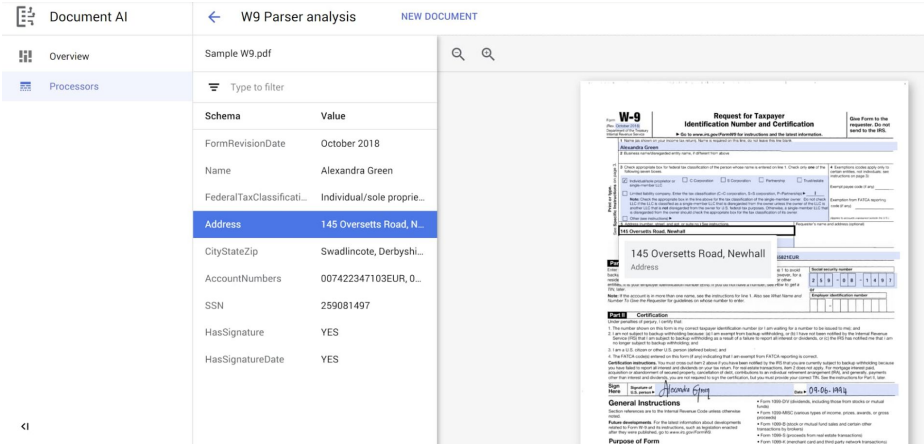
- Create a search data store with Vertex AI Agent Builder
- Connect a third-party data source with Vertex AI Agent Builder



Parsing and Understanding

Working with Tabular Data via Document AI

- Preprocess data that are on PDF: Use Google DocumentAI to extract.
- Data are in Database - Load to BigQuery, then e.g. create SQL command with GenAI
- Connect to relational database and:
  - **Function calling** for including details of customers - will be very large prompts otherwise
  - **Grounding** model on template how model should output the LP proposal (code example, visualization, explain constraints and decision taking). Model should have always the same desired structure in output
  - **Chain of thought** to provide clear explanations how it came to a certain conclusion incl. **Prompt**



Chunk, Embedding & Search (Retrieval)

- Vertex AI Vector Search (out of the box, previously “Matching Engine”)
  - Code: Vector Search quickstart
  - Parse and Chunk documents
- Vector Databases with RAG Engine:
  - Vertex AI Feature Store
  - Weaviate database
  - Pinecone database
  - Vector Search
- AlloyDB Vector Store to Generate Embeddings
  - Code: Getting started with Vector Embeddings with AlloyDB AI
- BigQuery Vector Search

Ranking & Optimization

- Rank and Rerank with RAG on Vertex AI. The following flow outlines how you might use the ranking API to improve the quality of results for chunked documents:
  a. Use Document AI Layout Parser API to split a set of documents into chunks.
  b. Use an embeddings API to create embeddings for each of the chunks.
  c. Load the embeddings into Vector Search or another search solution.
  d. Query your search index and retrieve the most relevant chunks.
  e. Rerank the relevant chunks using the ranking API.
- Prompt Optimization (with announcement). Code: Enhance your prompts with Vertex AI Prompt Optimizer

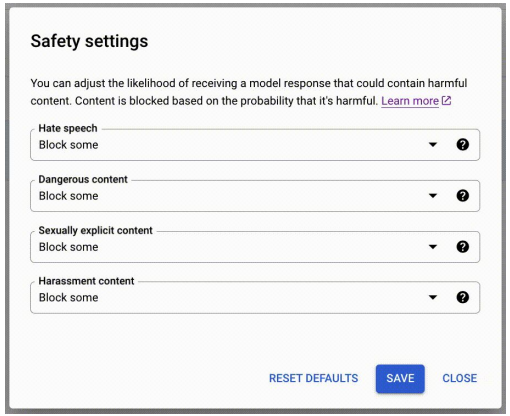
Grounding & Function Calling

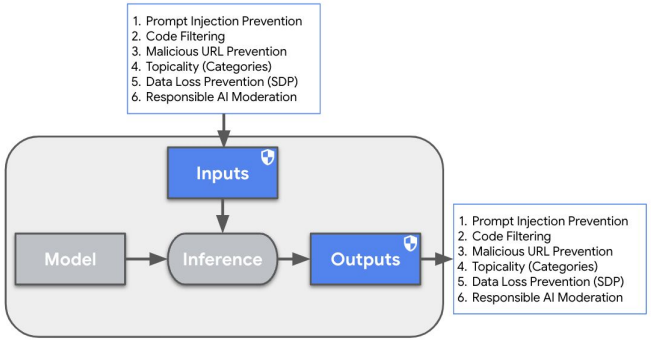
- Ground with public data via Google Search - ground a model with publicly available web data
  - Code: Ground responses for Gemini models
  - Dynamic Retrieval for improved price performance
- Ground with private data via Vertex AI Search as a data store
- Check Grounding API for RAG
- Third party content: Ground in content from authoritative sources

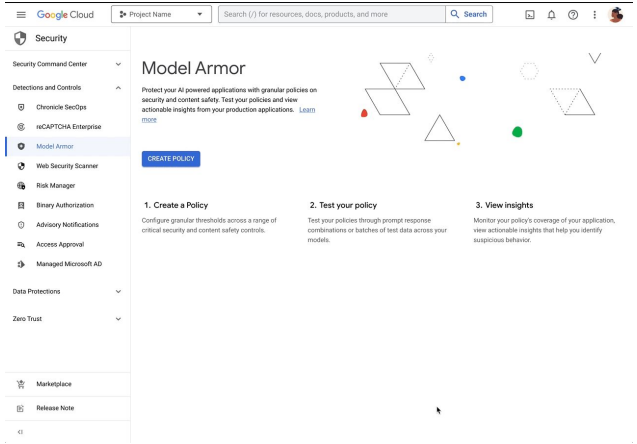
MOODY'S THOMSON REUTERS MSCI

Guardrails

- **Content Moderation**
  - Safety Attributes in Google's 1P models
  - Content Moderation API (open source + 3P models)
- **Model Armor:** more categories and topicality. prompt injections, jailbreaks, toxic content, and sensitive data leakage (Security Command Center is used to monitor what Model Armor has detected and blocked.)

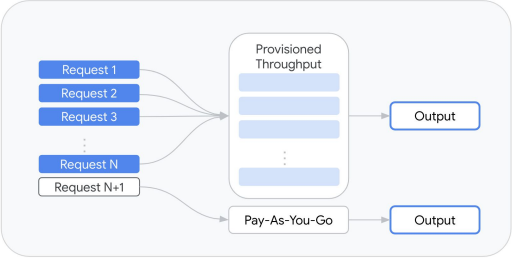






Caching & Quota Optimization

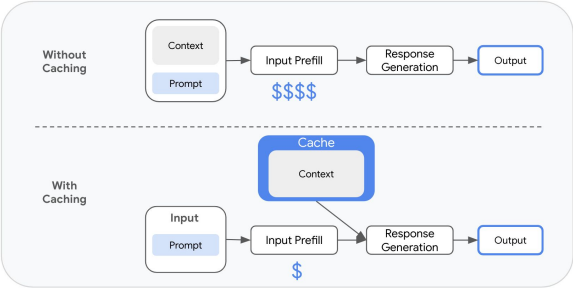
- Context Caching (prompt caching)
- Dynamic Shared Quota (DSQ): distributes on-demand capacity among all queries being processed by Google Cloud services. Eliminates the need for quota increase requests (QIRs).
- Provisioned Throughput: fixed-cost monthly subscription or weekly service that reserves throughput for supported generative AI models on Vertex AI



75% **Lower input price** with context caching*

Take advantage of millions-of-tokens context windows, available across both 1.5 Pro (June 27th) and 1.5 Flash (July 2nd)

**with >=32K context window*



Training & Fine-Tuning

- Model Garden on Google Cloud
  - Explore Models and Pricing for training
- Model Tuning - Parameter Efficient Tuning (PEFT)
- **PEFT for Google Models**
  - Gemini Parameter-efficient tuning (PEFT, Adapter Tuning), e.g LoRa
  - Code: Tune Gemini models by using supervised fine-tuning
  - Code: Supervised Fine-tuning
- **PEFT for Third Party Models**
  - Code: PEFT for LLama3 also see Video: NO-CODE Llama 3 Fine Tuning (Train AND Deploy)
  - Code: PEFT for Mistral
- **Upcoming: Managed PEFT for open-weight models in model garden**
- **Full Finetune**: Not planned yet as managed service, only very few customers would do that
- Reinforcement Finetuing

Evaluation

Evaluation: Gen AI evaluation service overview

- Code: Quickstart: Gen AI evaluation service workflow
  - RapidEval: lets developers evaluate model performance in seconds based on a small data set
  - Special: Run AutoSxS pipeline to perform pairwise model-based evaluation - assess the performance of two different models
  - Special: Computation-based evaluation pipeline - compare output against ground truth
- **Upcoming**: Multimodal Evaluation

Serving (Inference)

- Deploy Models
  - GPU on Cloud Run: fully managed, with no extra drivers or libraries needed
  - Open weight models: How to deploy Llama 3.2-1B-Instruct model with Google Cloud Run
  - Model Garden as Manager Service: Deploy and inference Gemma using Model Garden (example)
- Serving with vLLM:
  - Serve an LLM using GPUs on GKE with vLLM
  - Serve an LLM using TPUs on GKE with vLLM
- Open Source Managed Platforms for training and serving:
  - Ray for LLMs: Ray on GKE and Ray on Vertex AI (with guide to create a cluster)
  - PyTorch & Saxml: Serve models on multi-host TPUs with pre-built Saxml containers and PyTorch
- Register and Versionize Models
  - Vertex AI Model Registry
  - Import models to Vertex AI

Monitoring

Model Monitoring 2.0

- Centralize model monitoring configuration and reporting on the model (version), not the serving infrastructure
- Enable monitoring of models being served outside of Vertex AI (e.g. GKE, Cloud Run, and even multi/hybrid cloud environments)

	v1	v2
Configured per	Endpoint & Batch Prediction job	Model version
Serving Infrastructure	Vertex AI Endpoints & Batch Prediction jobs	**Any serving infrastructure** including GCE, GKE, and Cloud Run (just need access to serving data)
Supported Objectives	• Input feature drift • Feature attribution drift	• Input feature drift • Output prediction drift • Feature attribution drift • More coming soon...
Schema Required?	No	Yes (BigQuery, CSV, or JSONL)
GUI	Embedded in Endpoints & Batch Prediction GUIs	New, model-level GUI

Thank you