Google Developer Experts

Vertex AI: Unlocking the Future of Enterprise AI

Rushabh Vasa
Google Developer Experts
Co-founder, Agrahyah Technologies
@rushvasa

Google's Foundation Models on Vertex AI

Across a variety of model sizes to address use cases

GA

Gemini 1.0 Pro

Multimodal reasoning across a wide range of tasks

NEW

Gemini 1.5 Pro

Multimodal reasoning for longer prompts, 1 million context window

Limited Private GA

Gemini 1.0 Ultra

Largest and most capable model for highly complex tasks

NEW

Gemma 2B and 7B

Family of lightweight, state-of-the-art open models

PaLM for Text / Chat

Custom language tasks and multi-turn conversations

Imagen 2.0 for Text to Image

Create and edit images from simple prompts

Chirp for Speech to Text

Build voice enabled applications

Codey for Code Generation

Improve coding and debugging

Embeddings API for Text and Image

Extract semantic information from unstructured data

AI NEW

Claude on Vertex AI

Claude 2, Instant 1.2, and more

MISTRAL AI_ Meta

Open Models on Vertex AI

Mixtral 8x7B, Image Bind, DITO and more

NEW

Hugging Face Models

Few click deployment to Vertex AI

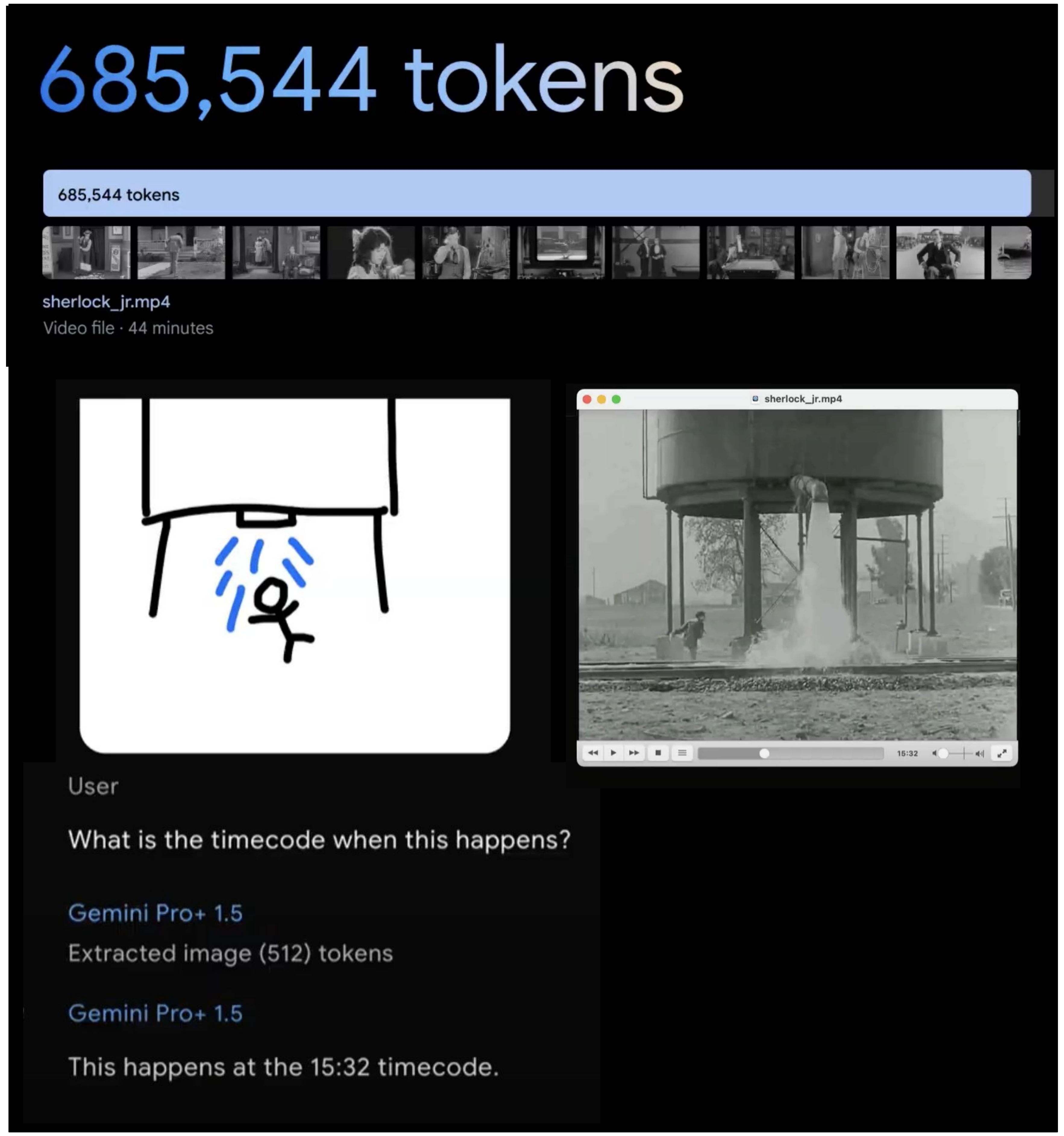
Gemini 1.5 Pro

Private Preview

Mid-size multimodal model with breakthrough long-context understanding

Gemini 1.5 Pro delivers dramatically enhanced performance and represents a step change in our foundation model approach, including:

- A **new Mixture–of-Experts (MoE) architecture** that provides more efficient training and serving, while increasing model performance
- An **expanded context window** (up to 1 million tokens) for complex reasoning across vast amounts of information
- **Better understanding and reasoning across modalities** including text, code, image, audio and video
- **Extensive ethics and safety testing** that builds on novel research on safety risks and leverages red-teaming techniques to test for a range of potential harms



685,544 tokens

User

What is the timecode when this happens?

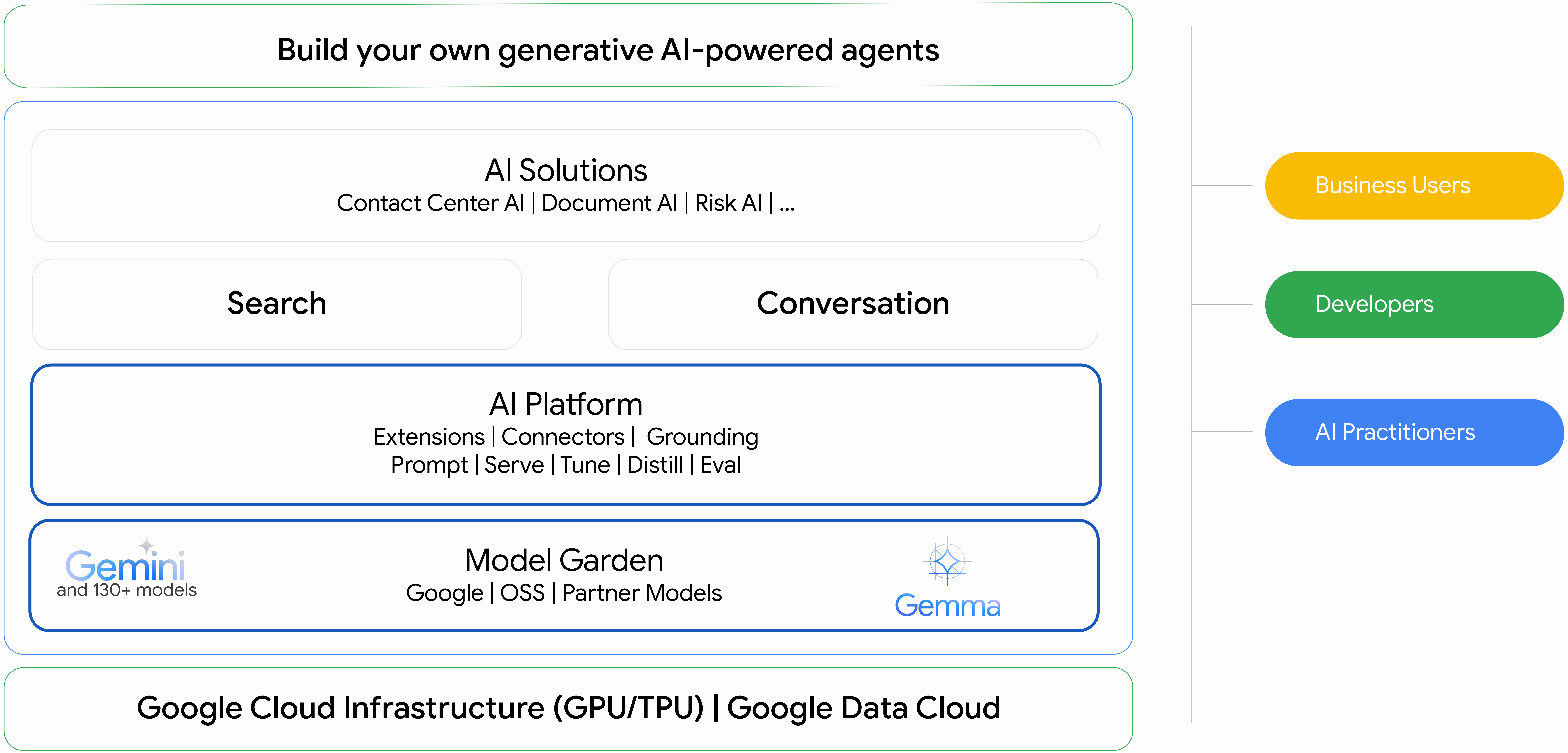
Gemini Pro+ 1.5

Extracted image (512) tokens

Gemini Pro+ 1.5

This happens at the 15:32 timecode.

Vertex AI



Build your own generative AI-powered agents

AI Solutions
Contact Center AI | Document AI | Risk AI | ...

Search

Conversation

AI Platform
Extensions | Connectors | Grounding
Prompt | Serve | Tune | Distill | Eval

Gemini and 130+ models

Model Garden
Google | OSS | Partner Models

Gemma

Google Cloud Infrastructure (GPU/TPU) | Google Data Cloud

Business Users

Developers

AI Practitioners

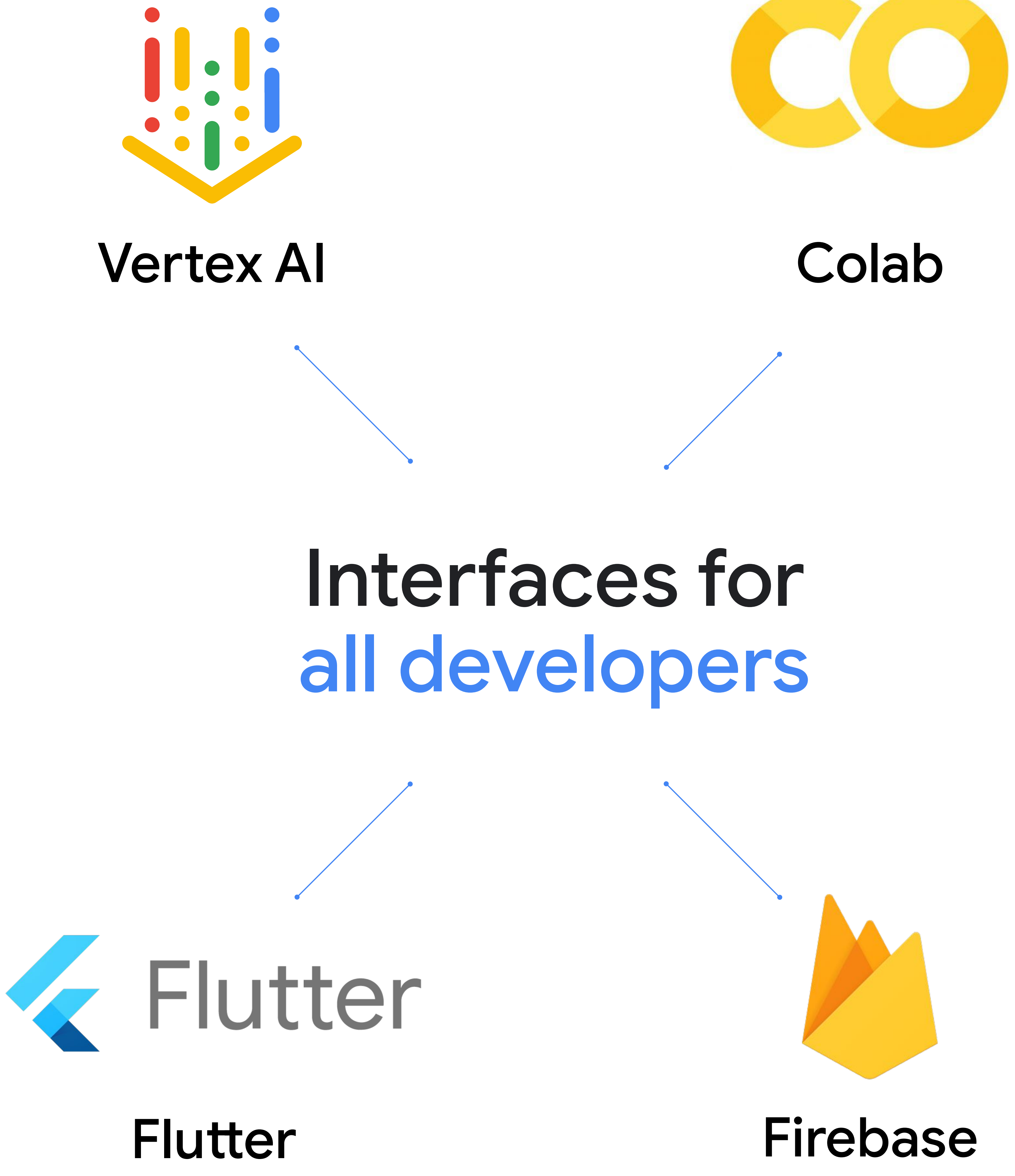
Vertex AI is built for developers

- Extensive **quick start library** with code samples and jumpstarts for **developers of all levels** and ecosystems

- **Free developer labs** and training resources across Vertex products at Cloud Skills Boost

- **Robust integrations** with popular third party developer tools like **Lang Chain, LlamaIndex, Pinecone, and Weaviate.**

- **Packages and extensions** to natively support Google Cloud foundation models in Google app developer frameworks like **Firebase and Flutter**.



Vertex AI

Colab

Interfaces for all developers

Flutter

Firebase

Evolution of AI Use Cases

Predictive AI

Regression & Classification
Forecasting
Sentiment Analysis
Entity Extraction
Object Detection

Generative AI

Text, Image & Code Generation
Text & Code Rewriting & Formatting
Summarization
Extractive Q&A
Image & Video Descriptions

Multimodal Generative AI

Natural Image Understanding
Spatial Reasoning and Logic
Mathematical Reasoning in Visual Contexts
Video Question Answering
Automatic Speech Recognition & Translation

RAG   Function CallingExtensions   Grounding   Punting & Safety

Introduction to RAG

Retrieval Augmented Generation

Typical usage of LLMs

LLMs are phenomenal for knowledge generation and reasoning. They are pre-trained on large amounts of **publicly available data**.

- Text generation
- Summarization
- Q&A

But…. The Grounding Problem (aka Hallucinations)

LLMs can only understand the information

- That they were trained on
- That they are explicitly given in the prompt

They will often assume that the premise of a prompt is true.

LLMs don’t have the capability to ask for more information and potentially need some outside input.

Prompt

What is langchain?

Response

LangChain is a blockchain-based platform that allows users to create, share, and learn languages. It is a decentralized platform that is not controlled by any single entity. This means that users have complete control over their data and content. LangChain is also a secure platform, as all data is encrypted and stored on the blockchain. This makes it very difficult for hackers to access or steal data.

Some naive solutions

(Full) Fine-Tuning

Run fine-tuning to teach the LLM by baking knowledge into the weights of the network.

- Data preparation effort
- Expensive to do
- Online learning/ updating new data
- Still may not work

Make Humans Check

Put a human in-between the output of the LLM and the user.

- Very latent and expensive
- Humans untrustworthy or inattentive

Prompt Engineering

Induce the LLM by adding any known information to the prompt.

- Based on LLMs existing trained knowledge
- Token limit for LLMs
- Error prone: Retrieving the right context
- Trade off: performance, latency, cost

Retrieval Augmented Generation (RAG)

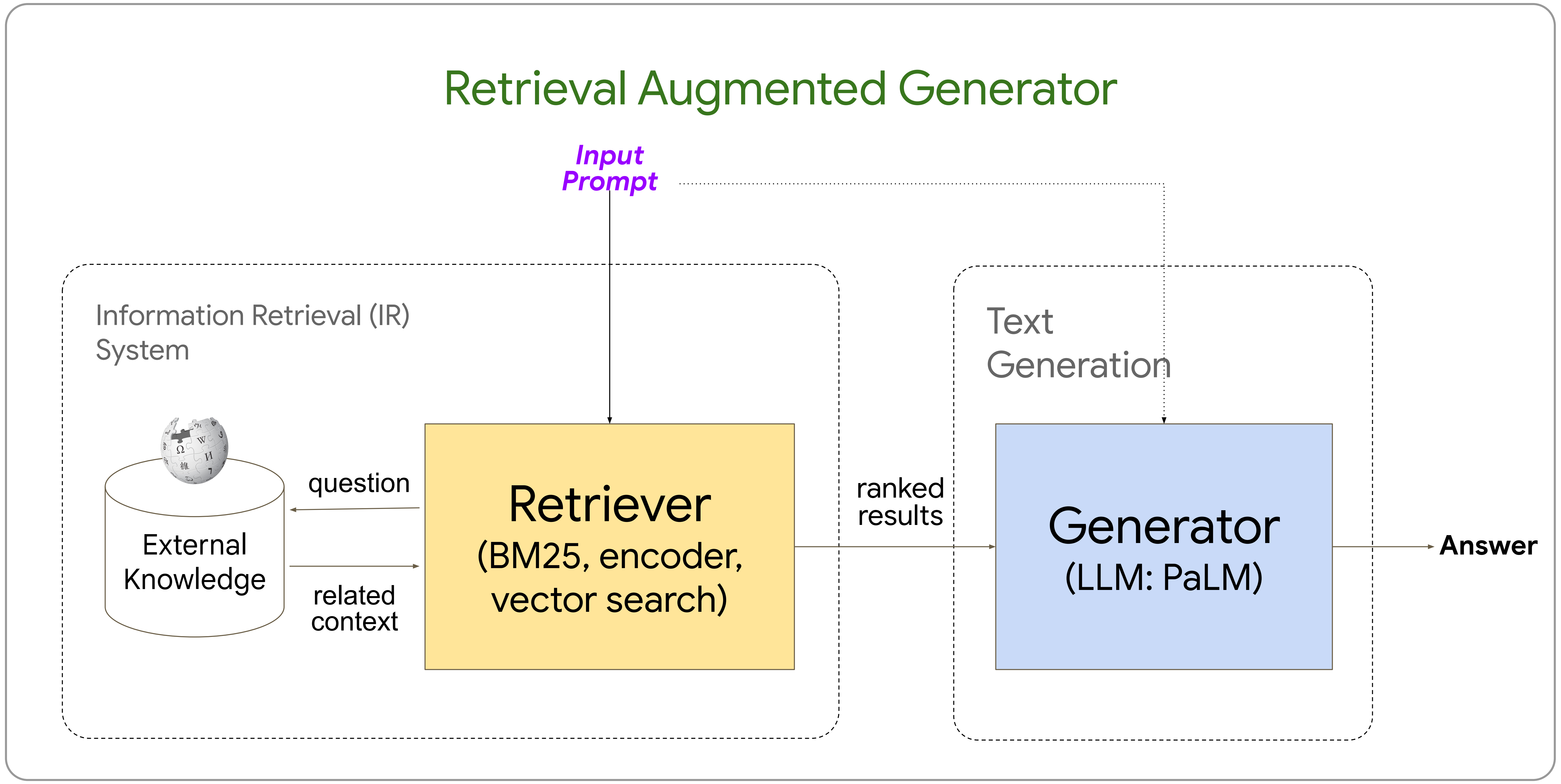
‘Grounding’ on user data

The Problem:

- LLMs do not know your business’s proprietary or domain specific data
- LLMs do not have real-time information
- LLMs find it challenging to provide accurate citations from their parametric knowledge

The Solution:

Feed the LLM *relevant* context in real-time, by using an information retrieval system



Google Cloud

Generative AI applications with Vertex AI PaLM 2 Models and LangChain

Modified Prompt

You are an intelligent assistant helping the users with their questions on {{company | research papers | …}}. Strictly Use ONLY the following pieces of context to answer the question at the end. Think step-by-step and then answer.

Do not try to make up an answer:
 - If the answer to the question cannot be determined from the context alone, say "I cannot determine the answer to that."
 - If the context is empty, just say "I do not know the answer to that."

CONTEXT:
{{retrieved_information}}

QUESTION:
{{question}}

Helpful Answer:

Common use cases / applications

Question & Answering

Semantic search and/or summarization over unstructured documents or structured data sources.

Can involve breaking down complex question, combining heterogeneous data sources or multiple documents.

Chatbots

Instead of a single question and answer, a chatbot can handle multiple back-and-forth queries and answers, getting clarification or answering follow-up questions.

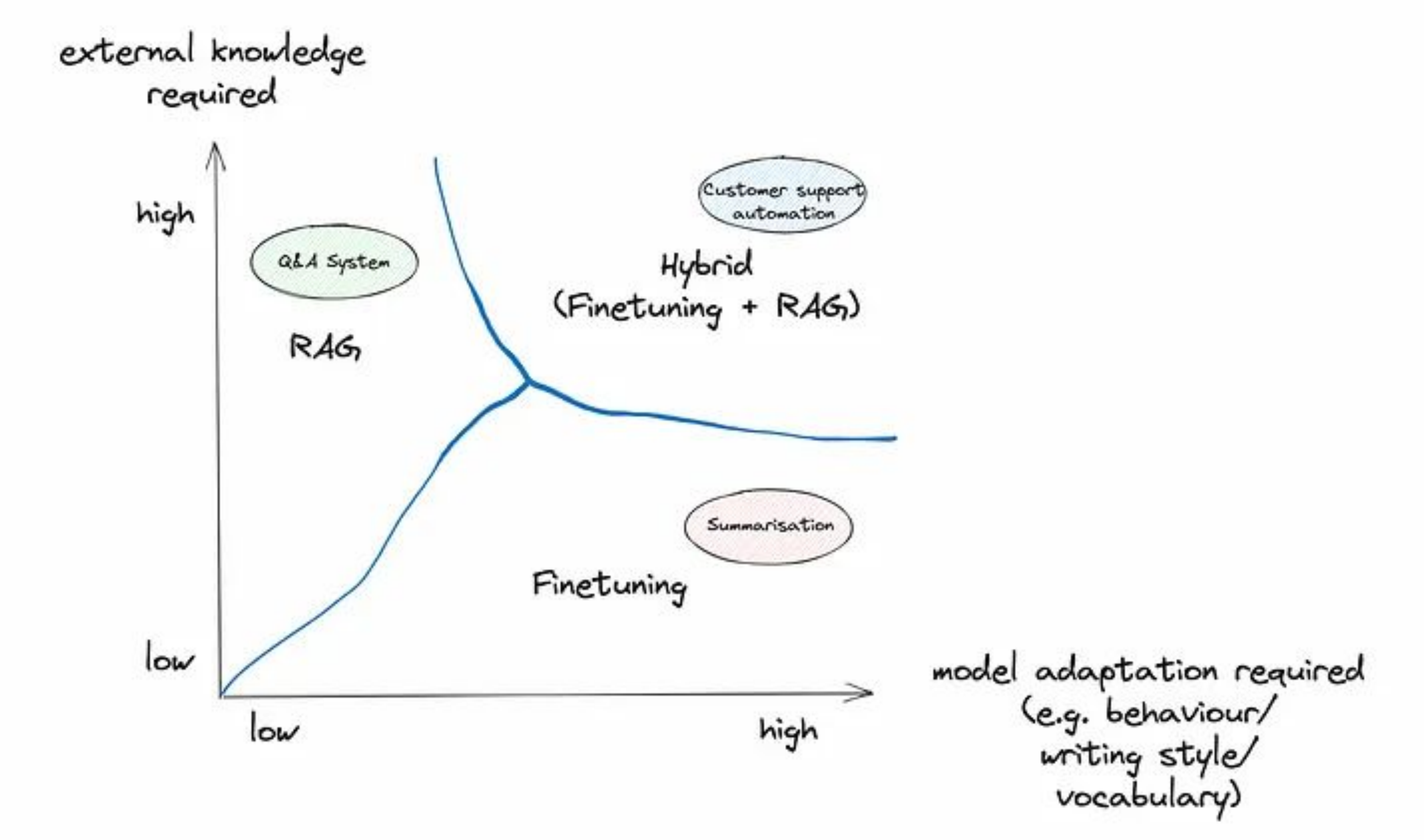
Agents

An “agent” is an automated reasoning and decision engine that takes in a user input/query and make internal decisions for executing that query to return results.

Involves breaking down complex question, choosing external tools, planning tasks and caching completed tasks.

When do you fine–tune vs RAG?

	Fine-Tuning	RAG
External knowledge required?	❌	✅
Model adaptation required?	✅	❌
Minimize hallucinations?	❌	✅
Is training data available?	✅	❌
How dynamic is the data?	❌	✅
Interpretability required?	❌	✅


external knowledge required
high
low
Q&A System
RAG
Customer support automation
Hybrid (Finetuning + RAG)
Summarisation
Finetuning
low
high
model adaptation required (e.g. behaviour/ writing style/ vocabulary)

[RAG vs Finetuning — Which Is the Best Tool to Boost Your LLM Application?]

Benefits of RAG-based workflows / applications

- Factuality & Grounding: Provides context, and accuracy grounded in evidence to generative AI, beyond what the LLM can provide.
- Better context: Can contain data that's more contextual than the data in a generalized LLM.
- Fresher data: Access to information which could be more recent than the data used to train the LLM.
- Quicker updates to data: Data in the RAG can be continually updated without incurring significant costs.
- Cheaper: Relatively cheaper than fine-tuning and quicker to implement
- Governance: Control LLM response based on who is accessing, by implementing access control and entitlements.

Challenges with RAG

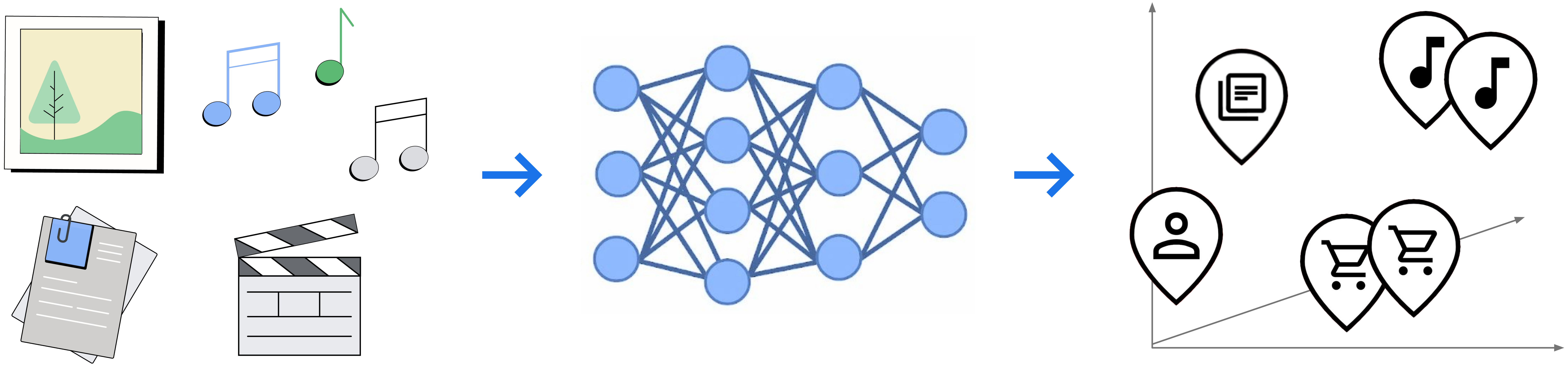
Quality related

- **Multiple failure modes** due to multiple hops
- **Requires tooling to measure quality** of the workflow and components
- **Bad retrieval ⟶ Bad results**
  - *Low precision:* not all chunks in retrieved set are relevant
  - *Low recall:* Not all relevant chunks are retrieved.
  - *Outdated* information or *redundant* data
- **Bad response generation:** hallucination, irrelevance and toxicity/bias

Non-Quality related

- **Operational overhead:** maintaining pipelines, handle inaccuracies, correcting and updating sources, governance
- **Data redundancy:** copies of data in different formats: embeddings, original content
- **Incurs additional costs:** storage, pipelines, LLM but relatively lower than retraining LLM
- **Increased latency:** Added latency with multiple hops deteriorating user experience
- **Additional tooling for observability** to observe, debug, and evaluate pipeline or each component
- **Requires a team** of data/software engineers and ML engineers

Embeddings



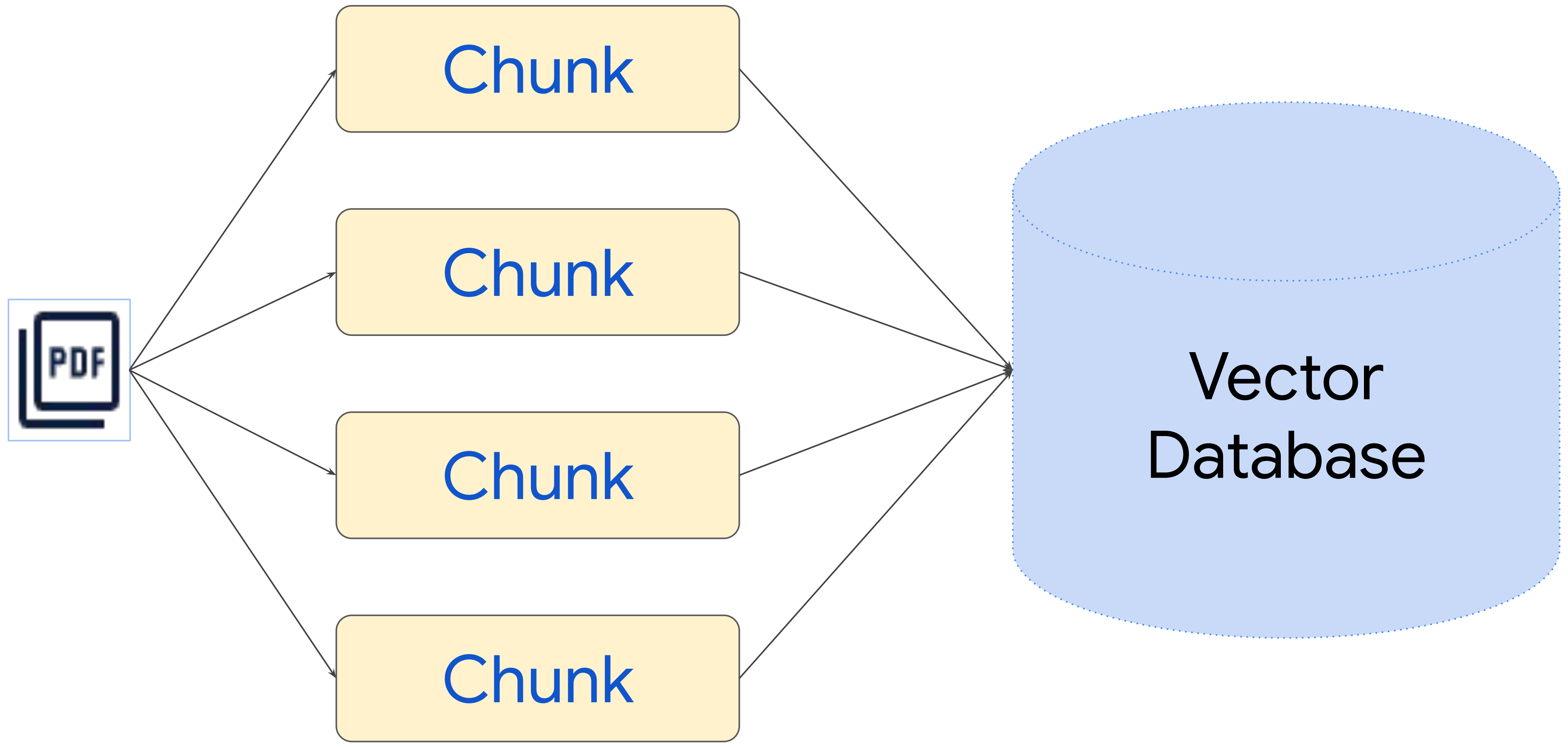
Data (10^4~10^6 dims) **DL models** **Embs (10^2~10^4 dims)**

"An embedding is a relatively low-dimensional vector into which you can translate high-dimensional vectors. Ideally, an embedding captures some of the semantics of the input by placing semantically similar inputs close together in the embedding space."

Meet AI's multitool: Vector embeddings

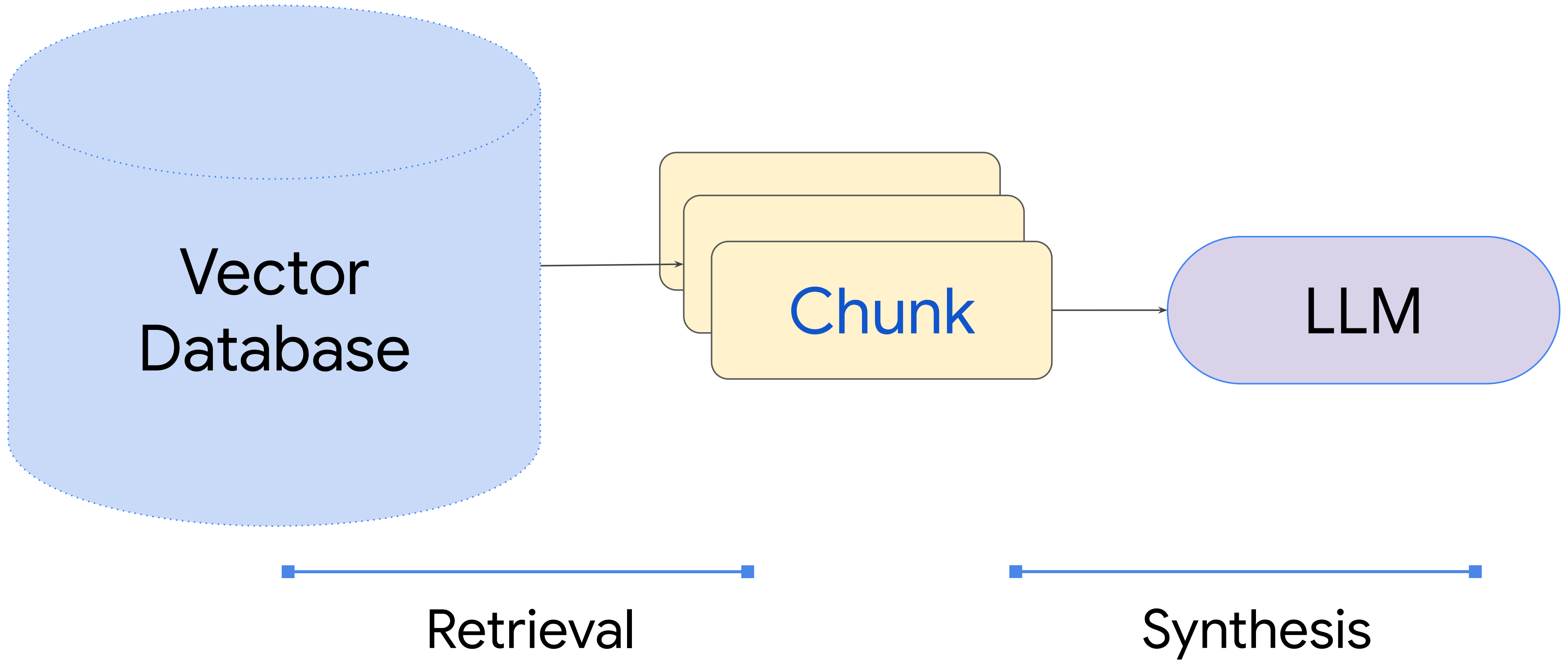
From: Google Machine Learning Crash Course

RAG workflow for building a QA System



Data Ingestion / Parsing

- Split up document(s) into even chunks.
- Each chunk is a piece of raw text.
- Generate embedding for each chunk
- Store each chunk into a vector database

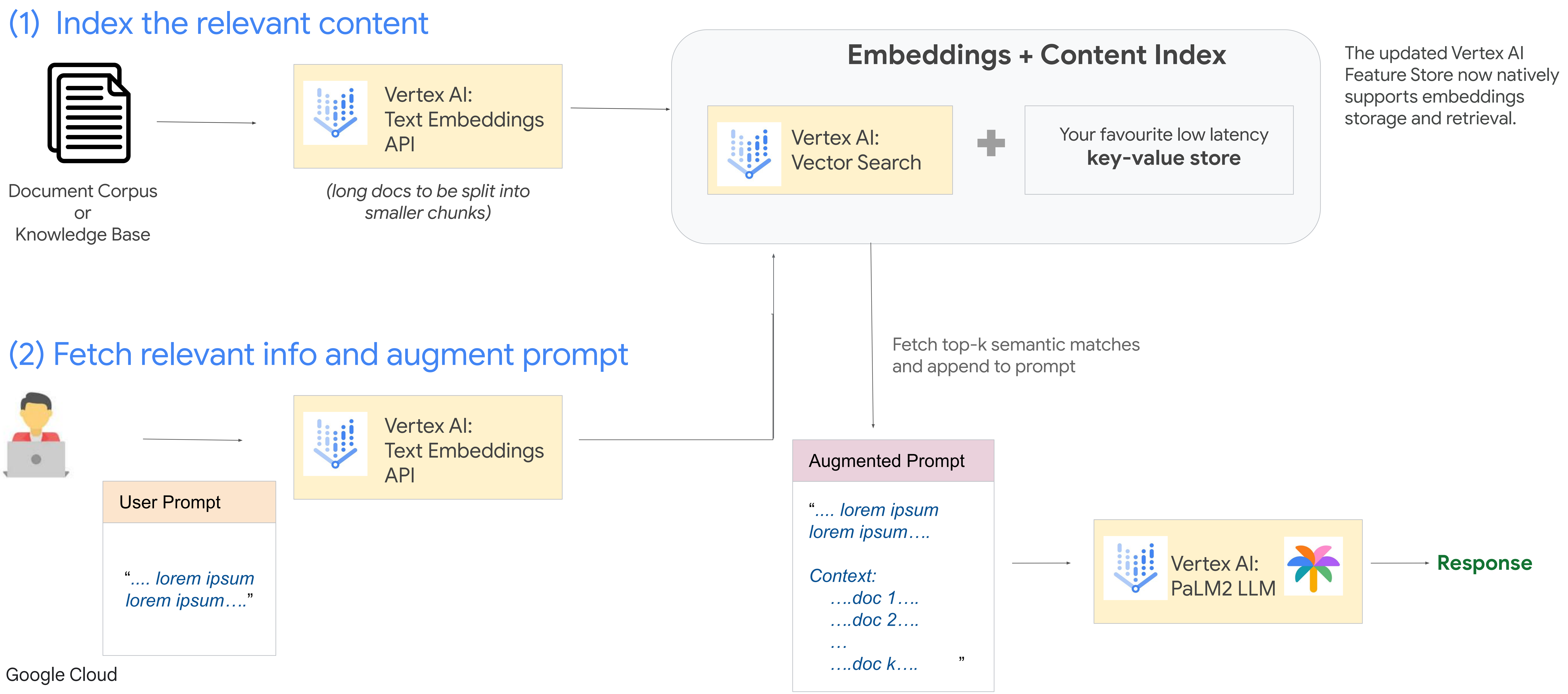


Querying

- Generate embedding for query
- Find top-k most similar chunks from vector database
- Plug into LLM response synthesis

Retrieval Augmented Generation:

Example architecture powered by Vertex AI Text Embeddings and Vector Search



(1) Index the relevant content

Document Corpus
or
Knowledge Base

Vertex AI:
Text Embeddings
API

(long docs to be split into smaller chunks)

Embeddings + Content Index

Vertex AI:
Vector Search

Your favourite low latency
key-value store

The updated Vertex AI Feature Store now natively supports embeddings storage and retrieval.

(2) Fetch relevant info and augment prompt

User Prompt

"…. lorem ipsum lorem ipsum…."

Vertex AI:
Text Embeddings
API

Fetch top-k semantic matches and append to prompt

Augmented Prompt

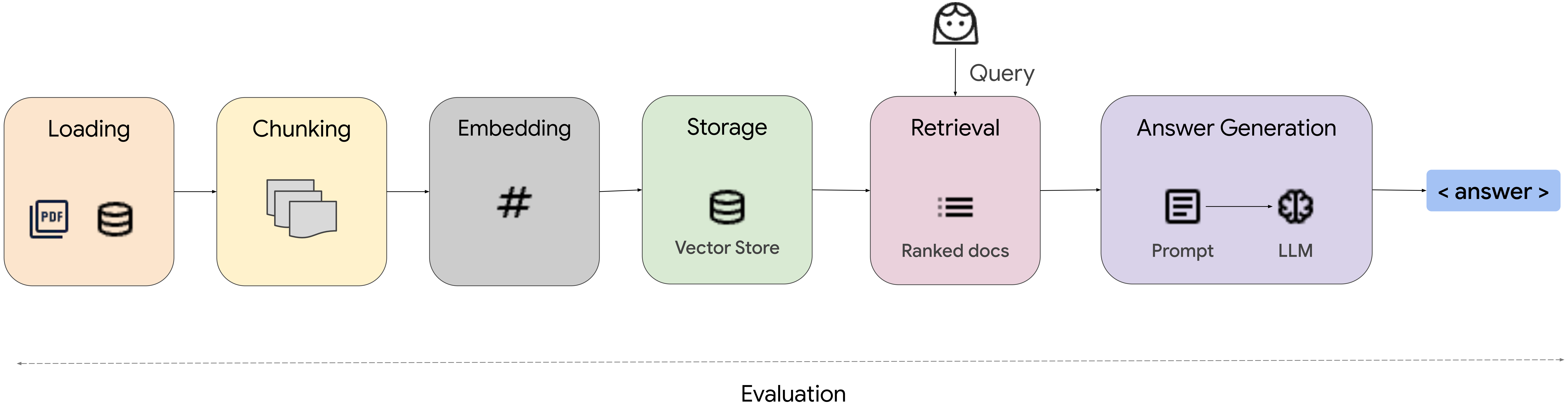
"…. lorem ipsum
lorem ipsum….

Context:
….doc 1….
….doc 2….
…
….doc k…. "

Vertex AI:
PaLM2 LLM

Response

Improving performance
Better retrieval == better results



Potential approaches

Build versus buy?

Build vs Buy: Vertex AI Search vs DIY RAG

DIY RAG

Parsing

Chunking

Embedding

Indexing/Storage

Search

Summarization & Conversation

e.g. Vertex AI Vector Search

Painful BUT Fully customizable!

How to chunk to preserve consistent knowledge?

How to pick/tune the right embedding?

Which database to pick? Self-managed

What similarity metric should I use?

Keeping context windows, Prompt Engineering, Tuning

Vertex AI Search

Built-in Summarization & Conversation

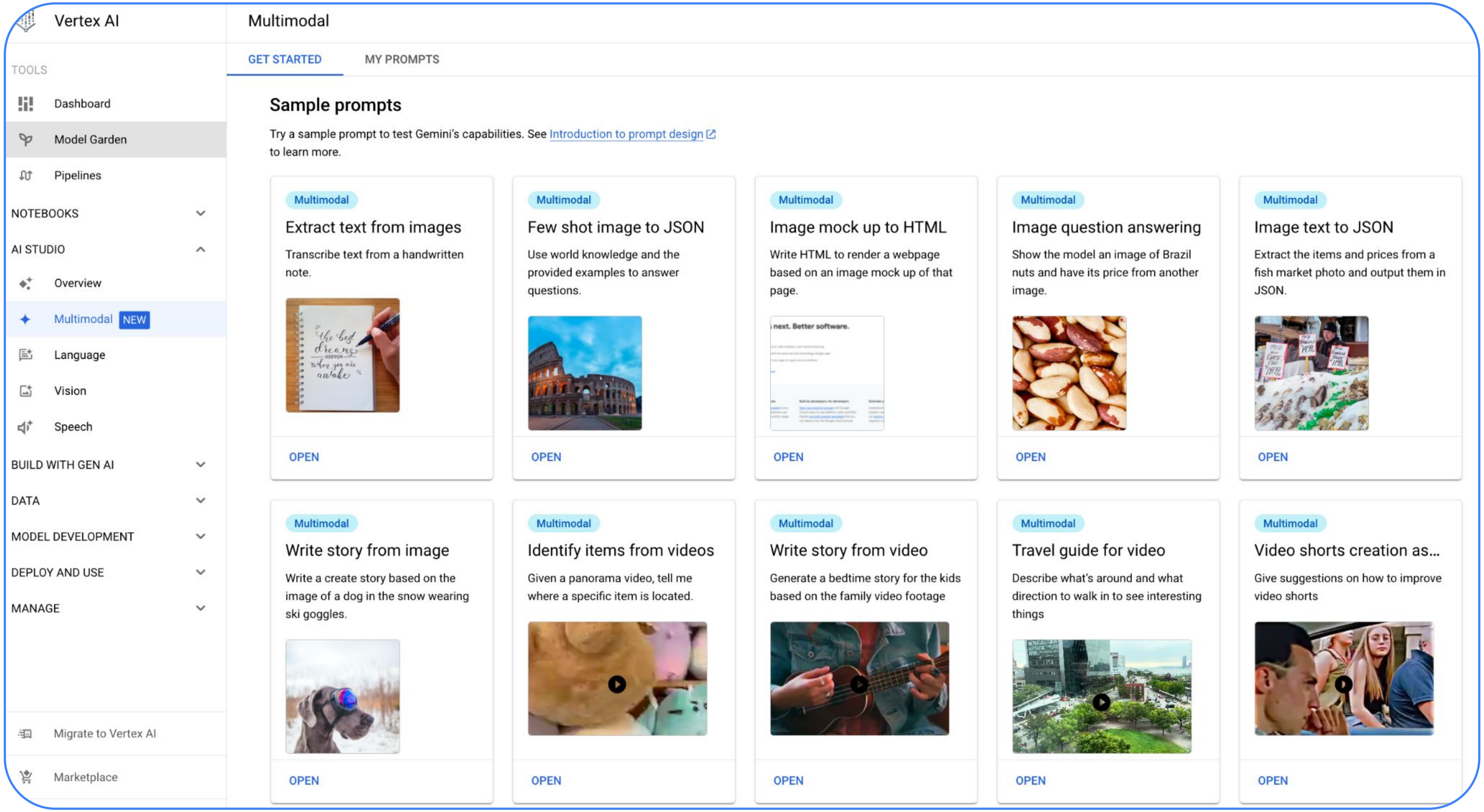
OR

DIY Summarization & Conversation

Full-Fledged Search Engine OOTB
Parsing, chunking, embedding, indexing/storage, semantic+token-based search, summarization and conversation
Better query understanding, user events

Tuning:
Bring you own doc parsers (roadmap)
Bring your own embeddings
Bring your own ranking model (roadmap)
Tune your own quality Search Adapters (roadmap)

Try Gemini-powered Multimodal experiences on Vertex AI



Vertex AI

Multimodal

GET STARTED

MY PROMPTS

TOOLS

- Dashboard
- Model Garden
- Pipelines

NOTEBOOKS

AI STUDIO

- Overview
- Multimodal NEW
- Language
- Vision
- Speech

BUILD WITH GEN AI

DATA

MODEL DEVELOPMENT

DEPLOY AND USE

MANAGE

Migrate to Vertex AI

Marketplace

Sample prompts

Try a sample prompt to test Gemini's capabilities. See Introduction to prompt design to learn more.

Multimodal
Extract text from images
Transcribe text from a handwritten note.
OPEN

Multimodal
Few shot image to JSON
Use world knowledge and the provided examples to answer questions.
OPEN

Multimodal
Image mock up to HTML
Write HTML to render a webpage based on an image mock up of that page.
OPEN

Multimodal
Image question answering
Show the model an image of Brazil nuts and have its price from another image.
OPEN

Multimodal
Image text to JSON
Extract the items and prices from a fish market photo and output them in JSON.
OPEN

Multimodal
Write story from image
Write a create story based on the image of a dog in the snow wearing ski goggles.
OPEN

Multimodal
Identify items from videos
Given a panorama video, tell me where a specific item is located.
OPEN

Multimodal
Write story from video
Generate a bedtime story for the kids based on the family video footage
OPEN

Multimodal
Travel guide for video
Describe what's around and what direction to walk in to see interesting things
OPEN

Multimodal
Video shorts creation as...
Give suggestions on how to improve video shorts
OPEN

Feedback

I want to know more / want to try this out

- Gemini: Unlocking insights in scientific literature: https://youtu.be/sPiOP_CB54A
- GitHub repo: goo.gle/gen-ai-github
  - `gemini > use-cases > retrieval-augmented-generation`
- Qwiklabs on Google Cloud Skills Boost https://www.cloudskillsboost.google/:
  - Integrate Search in Applications using Vertex AI Search
  - Multimodality with Gemini

Thank you

Google Cloud