Polish Your RAG app

with Gemini Pro (Vision) API

Discord: **Google for Developers**.

jimmy.liao, fb: **jimmyliao.tw**

*** Includes personal research only.**

Agenda

- RAG - Retrieval Augmented Generation
- Multi-Modal LMM Example
- Lab and Discussion

Workshop Repo

https://github.com/jimmyliao/genai-gdg.git

gemini_multi_modal.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

Share

TPU RAM Disk Colab AI

GenAI Workshop for Google Developer Groups Taipei - Jan. 25

Open in Colab

Using Google's Gemini model for image understanding and build Retrieval Augmented Generation

Description

In this notebook, we show how to use Google's Gemini Vision models for image understanding.

Lab01, Show several Gemini functions:

- `complete`, `chat`, `stream complete`, `stream chat`

Lab02, `Gemini` + `Pydantic` to parse structured information from Google Maps screenshots.

- Define the desired Pydantic class with attribution fields
- Let `gemini-pro-vision` model understand each image and output structured results

Lab03, Build a simple `Retrieval Augmented Generation` Flow for a small Google Maps restaurant dataset.

- Build vector index based on the structured outputs from Step 2
- Using the `gemini-pro` model to synthesize the results and recommends restaurants based on user query.

Note: `google-generativeai` is only available for certain countries and regions.

- (Credit: LlamaIndex multi_modal example)

Prerequisite part 1: Get Gemini API from Google AI Studio

1 cell hidden

RAG terminology

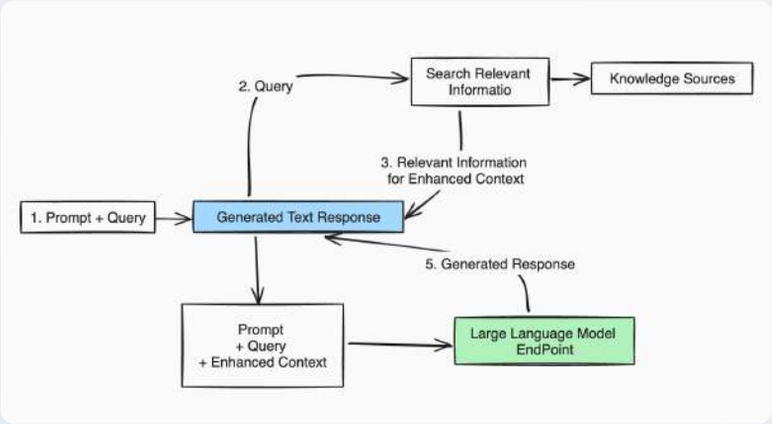
- **R**etrieval **A**ugmented **G**eneration
  - Before generate a response
  - it references an authoritative knowledge base

- Before Retrieval → **I**nformation **R**etrieval → **Index**ing

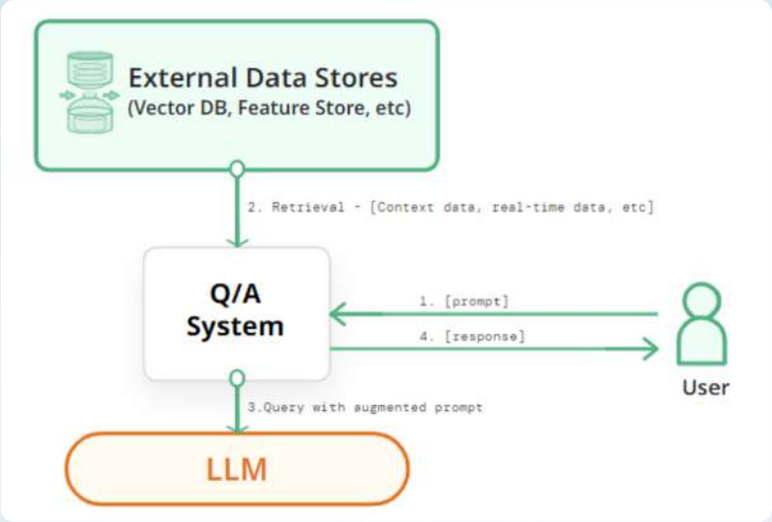
- ***RAG= f(LLM, IR)***
- Indexing → **Data Ingestion**

- The following will be explained by **Data Ingestion** → **Retrieval** → Retrieval Augmented **Generation**

User how to use



Flow



Retrieval Concept

- Ref: https://www.hopsworks.ai/dictionary/retrieval-augmented-generation-llm

Data Ingestion - Preparation

- Sometimes you will also hear Load / Prepare the data
- Keyword Chat With Your Data/Excel, etc.
- We use images as data source for this workshop

Download example images for Gemini to understand

```
from pathlib import Path

input_image_path = Path("google_restaurants")
if not input_image_path.exists():
    Path.mkdir(input_image_path)
```

```
!wget "https://docs.google.com/uc?export=download&id=1Pg04p6ss0FlBgz00noHAOAJ1EYXiosKg" -O ./google_restaurants/m
!wget "https://docs.google.com/uc?export=download&id=1dYZy17bD6pSsEyACXx9fRMNx93ok-kTJ" -O ./google_restaurants/c
!wget "https://docs.google.com/uc?export=download&id=1ShPnYVc1iL_TA1t7ErCFEAHT74-qvMrn" -O ./google_restaurants/s
!wget "https://docs.google.com/uc?export=download&id=1WjISWnatHjwL4z5VD_9o09ORWhRJuYqm" -O ./google_restaurants/t

!wget "https://docs.google.com/uc?export=download&id=14wsesUbq_pOj1xvhqqJKogwcPXvdUUhC" -O ./google_restaurants/t
!wget "https://docs.google.com/uc?export=download&id=1now7y3RH9FuTvefe_b66Y4RVLBkZ0mDv" -O ./google_restaurants/t
!wget "https://docs.google.com/uc?export=download&id=1lyUbzkt-JYO7NR5BVfxfZiZkOnny8pu4" -O ./google_restaurants/t
```

```
--2023-12-29 00:53:13--  https://docs.google.com/uc?export=download&id=1Pg04p6ss0FlBgz00noHAOAJ1EYXiosKg
```

Data ingestion - load data

- If you're using LlamaIndex or LangChain, can use built-in `PyMuPDFReader` to read PDF
  - also support multiple files

```
from llama_hub.file.pymu_pdf.base import PyMuPDFReader

loader = PyMuPDFReader()
documents = loader.load(file_path="./data/llama2.pdf")
```

- Pros/Cos of using built-in Document Reader as RAG app?
  - memory
  - search / matching

Data ingestion - Index creation

- Similarity: Embeddings => Vector Database
- NodeParser → Nodes → Create Vector Store

Index Creation by using LlamaIndex `VectorStoreIndex`

```
# use Azure OpenAI as llm
from llama_index.llms import AzureOpenAI

llm = AzureOpenAI(
    deployment=AZURE_OPENAI_DEPLOYMENT,
    api_key=AZURE_OPENAI_KEY,
    azure_endpoint=f"https://{AZURE_OPENAI_ENDPOINT}",
    api_version=AZURE_OPENAI_API_VERSION
)

node_parser = SentenceSplitter(chunk_size=1024)
service_context = ServiceContext.from_defaults(llm=llm)

nodes = node_parser.get_nodes_from_documents(documents)

## Setup Index from Nodes VectorStoreIndex
## index time varies depending on the number of nodes / documents
## For this simple PDF, takes around 10~15 seconds
index = VectorStoreIndex(nodes, service_context=service_context)

## Setup Query Engine from Index
query_engine = index.as_query_engine()
```

Retrieval

```
## Send first query to verify the RAG is working

query = "What is the purpose of this paper?"
print(query_engine.query(query))
```

Practical Example

Showcasing MultiModal Models

What is LMM (Multi-Modal Models)

A Multi-Modal model is a model can process and understand information from different sources, such as text, images, and sounds.

In our lab showcase, the use case:

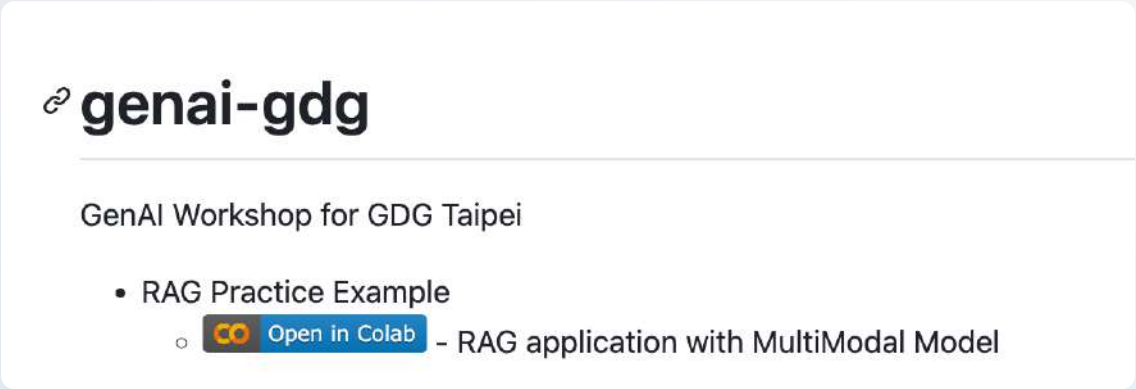
- Please recommend American restaurants, near XinYi District, Taipei City
  - Datasource is based on screenshots of some map crop
  - Understand/Analytics Images with `Gemini Vision Model`
  - Build Vector Store with `Gemini Embedding` Model and `Qdrant`
  - Index Restaurant as nodes in Vector Store
  - Query with the index

Prerequisite of workshop

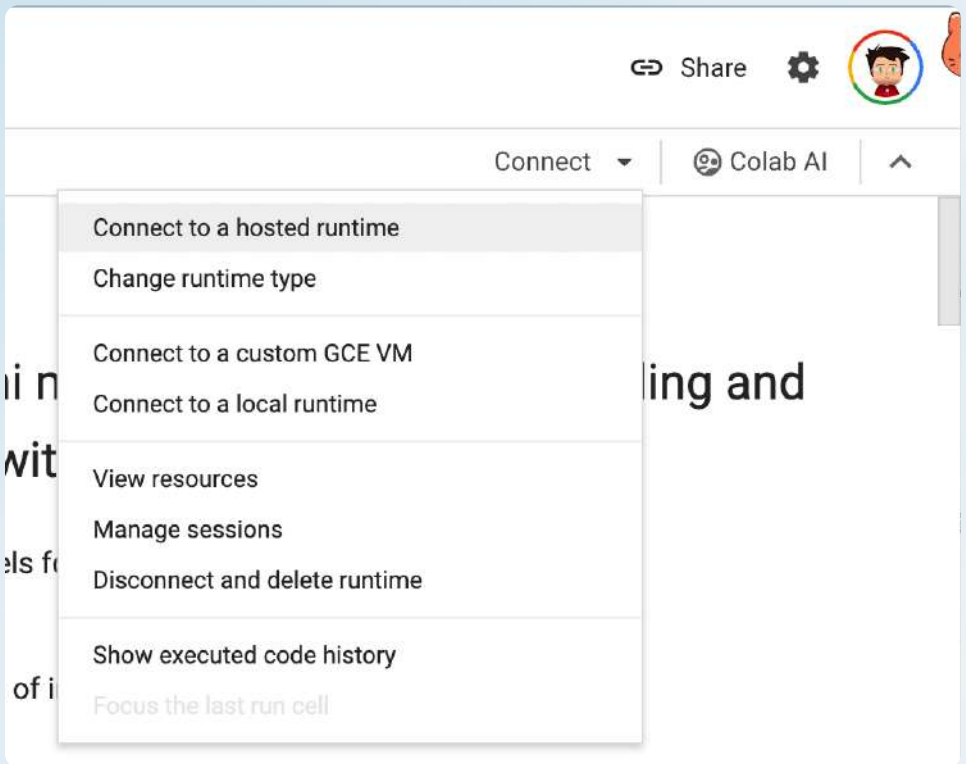
- ☐ Your Google Account sign-in
- ☐ Open in Colab

Open in Colab

https://colab.research.google.com/github/jimmyliao/genai-gdg/blob/main/gemini-lmm.ipynb



- ☐ Click Connect to startup a Runtime → **TPU (Recommeded)**



- ☐ Run first cell to install python libraries

Ready for the LMM RAG journey!

Let's look the Notebook with Colab!