

# Assignment 4: Retrieval-Augmented Generation with LangChain

2024 NTHU Natural Language Processing

Hung-Yu Kao

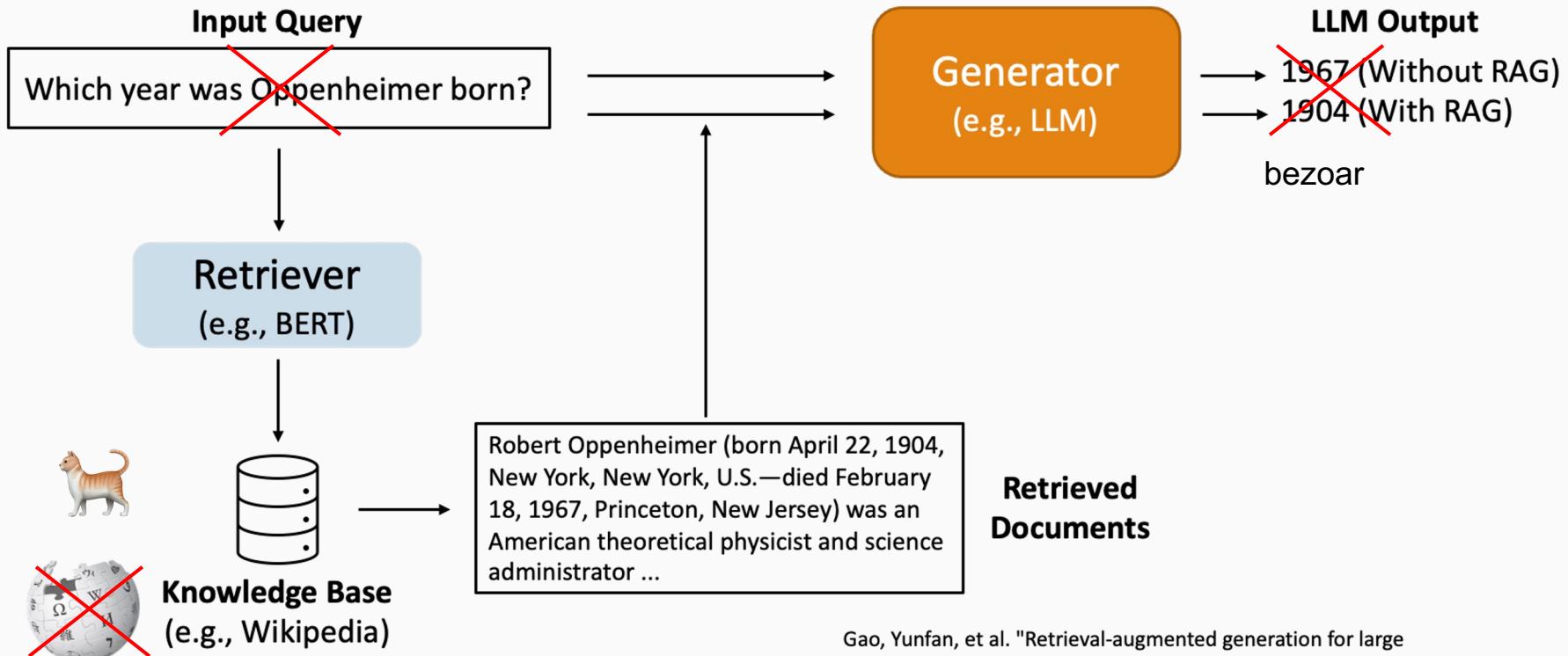
IKM Lab TAs

# Assignment Description

- Task: Retrieval-Augmented Generation (RAG) for **Question Answering**
- Database: Cats' knowledge in texts
- Inputs: **Ten questions** about cats' knowledge
- Answer: Short answers for the ten questions (evaluation: exact matching)
- Restricted models
  - Generator: Llama-3.2-1B (freezed)

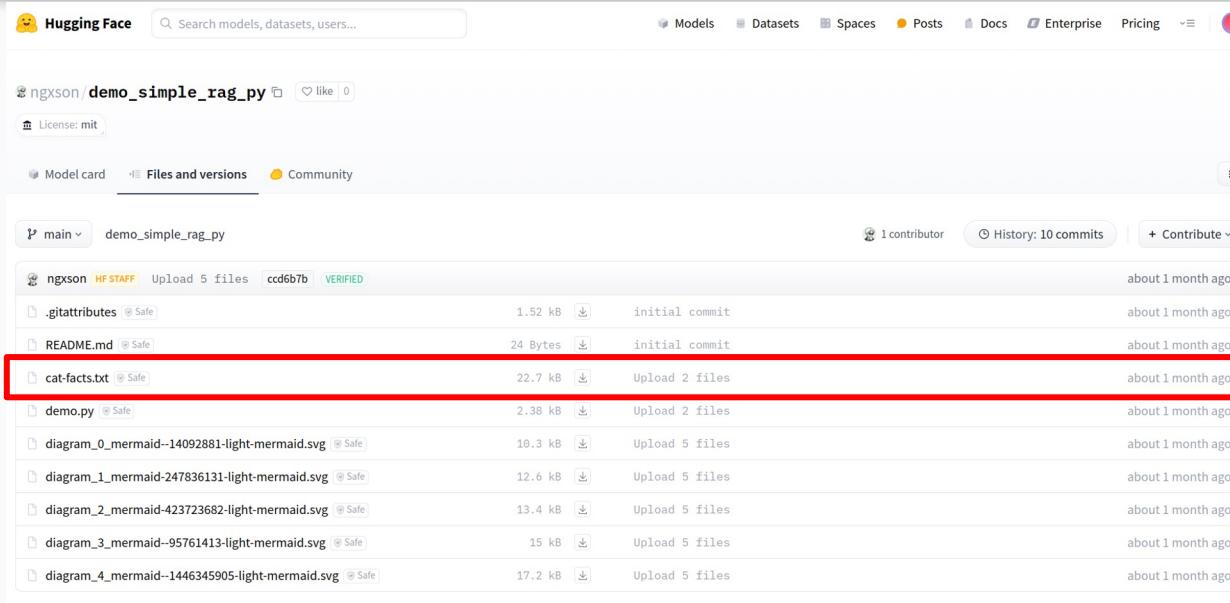
# Retrieval-Augmented Generation (RAG)

What is the technical term for a cat's hairball?



Gao, Yunfan, et al. "Retrieval-augmented generation for large language models: A survey." *arXiv preprint arXiv:2312.10997* (2023).

# Dataset: Cat-facts



The screenshot shows the Hugging Face Model card for the repository `demo_simple_rag_py`. The card includes a search bar, navigation links for Models, Datasets, Spaces, Posts, Docs, Enterprise, and Pricing, and a user profile icon.

The repository details shown are:

- Owner: `ngxson`
- License: MIT
- Model card
- Files and versions (selected)
- Community

The file list table has the following columns: Name, Size, Last Commit, and Actions. A red box highlights the row for `cat-facts.txt`.

| Name                                                        | Size    | Last Commit    | Actions           |
|-------------------------------------------------------------|---------|----------------|-------------------|
| <code>cat-facts.txt</code>                                  | 22.7 kB | Upload 2 files | about 1 month ago |
| <code>demo.py</code>                                        | 2.38 kB | Upload 2 files | about 1 month ago |
| <code>diagram_0_mermaid-14092881-light-mermaid.svg</code>   | 10.3 kB | Upload 5 files | about 1 month ago |
| <code>diagram_1_mermaid-247836131-light-mermaid.svg</code>  | 12.6 kB | Upload 5 files | about 1 month ago |
| <code>diagram_2_mermaid-423723682-light-mermaid.svg</code>  | 13.4 kB | Upload 5 files | about 1 month ago |
| <code>diagram_3_mermaid-95761413-light-mermaid.svg</code>   | 15 kB   | Upload 5 files | about 1 month ago |
| <code>diagram_4_mermaid-1446345905-light-mermaid.svg</code> | 17.2 kB | Upload 5 files | about 1 month ago |

The dataset is presented in `ngxson/demo_simple_rag_py`

# Dataset

## Cat-facts dataset

- Database size: 150 facts
- Each fact is represented as a sentence.
  - E.g., When a cat chases its prey, it keeps its head level. Dogs and humans bob their heads up and down.
  - E.g., The technical term for a cat's hairball is a “bezoar.”
  - ...

# Construction for Questions and Answers

| Index | Question                                                            | Answer Sentence in the database                                                                                                                           | Answer we set                      |
|-------|---------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| 1     | How much of a day do cats spend sleeping on average?                | On average, cats spend <b>2/3</b> of every day sleeping.                                                                                                  | 2/3                                |
| 2     | What is the technical term for a cat's hairball?                    | The technical term for a cat's hairball is a " <b>bezoar</b> ."                                                                                           | bezoar                             |
| 3     | What do scientists believe caused cats to lose their sweet tooth?   | Scientists believe this is due to <b>a mutation in a key taste receptor</b> .                                                                             | a mutation in a key taste receptor |
| 4     | What is the top speed a cat can travel over short distances?        | A cat can travel at a top speed of approximately <b>31 mph</b> (49 km) over a short distance.                                                             | 31 mph or 49 km                    |
| 5     | What is the name of the organ in a cat's mouth that helps it smell? | Besides smelling with their nose, cats can smell with an additional organ called the <b>Jacobson's organ</b> , located in the upper surface of the mouth. | Jacobson's organ                   |

# Construction for Questions and Answers

| Index | Question                                                       | Answer Sentence in the database                                                                                                      | Answer we set         |
|-------|----------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|-----------------------|
| 6     | Which wildcat is considered the ancestor of all domestic cats? | The ancestor of all domestic cats is <b>the African Wild Cat</b> which still exists today.                                           | the African Wild Cat  |
| 7     | What is the group term for cats?                               | A group of cats is called a “ <b>clowder</b> .”                                                                                      | clowder               |
| 8     | How many different sounds can cats make?                       | Cats make about <b>100</b> different sounds.                                                                                         | 100                   |
| 9     | What is the name of the first cat in space?                    | The first cat in space was a French cat named <b>Felicette</b> (a.k.a. “Astrocat”) In 1963, France blasted the cat into outer space. | Felicette or Astrocat |
| 10    | How many toes does a cat have on its back paws?                | Cats have five toes on each front paw, but only <b>four</b> toes on each back paw.                                                   | four                  |

# Code (hints only)

# Connect to Huggingface (1/4)

Step1

The screenshot shows the Huggingface homepage. At the top, there is a navigation bar with links for Datasets, Spaces, Posts, Docs, Enterprise, Pricing, and a user profile icon. Below the navigation bar, there are sections for 'Trending' models and datasets. A red box highlights the 'Profile' section of the user dropdown menu, which shows the name 'ZhiQuanFeng'. A red arrow points from the text 'Step1' down to this highlighted area.

Step2

The screenshot shows the user profile page for 'Zhi-Quan Feng' (ZhiQuanFeng). The profile includes a large circular profile picture, the name 'Zhi-Quan Feng' (ZhiQuanFeng), and a link to their GitHub account ('FengZhiQuan1046'). Below the name, there is a 'Edit profile' button, which is highlighted with a red box and a red arrow pointing from 'Step2'. The page also lists 'AI & ML interests' (NLP) and 'Organizations' (None yet).

## Connect to Huggingface (2/4)

Step4



Zhi-Quan Feng  
ZhiQuanFeng

Profile

Account

Authentication

Organizations

Billing

Access Tokens

SSH and GPG Keys

Webhooks

Papers

Notifications

Local Apps and Hardware

NEW

Gated Repositories

### Access Tokens

#### User Access Tokens

+ Create new token

Access tokens authenticate your identity to the Hugging Face Hub and allow applications to perform actions based on token permissions. ⚠ Do not share your **Access Tokens** with anyone; we regularly check for leaked Access Tokens and remove them immediately.

| Name     | Value     | Last Refreshed Date | Last Used Date     | Permissions | ⋮ |
|----------|-----------|---------------------|--------------------|-------------|---|
| NLP_HW   | hf...pgJF | about 16 hours ago  | about 15 hours ago | FINEGRAINED | ⋮ |
| my_token | hf...kEJe | Sep 24              | 5 days ago         | FINEGRAINED | ⋮ |

Step3



# Connect to Huggingface (3/4)

Create new Access Token

Token type  
 Fine-grained    Read    Write  
⚠️ This cannot be changed after token creation.

Token name

User permissions (ZhiQuanFeng)

**Repositories**

- Read access to contents of all repos under your personal namespace
- Read access to contents of all public gated repos you can access
- Write access to contents/settings of all repos under your personal namespace

**Inference**

- Make calls to the serverless Inference API
- Make calls to Inference Endpoints
- Manage Inference Endpoints

**Webhooks**

- Access webhooks data
- Create and manage webhooks

**Collections**

- Read access to all collections under your personal namespace
- Write access to all collections under your personal namespace

Step5  
Type a name here  
(Can be arbitrary)

Org permissions  
None if not specified.

Search for orgs

**Repositories**

- Read access to contents of all repos in selected organizations
- Interact with discussions / Open pull requests on repos in selected organizations
- Write access to contents/settings of all repos in selected organizations

**Inference endpoints** ⓘ

- Make calls to inference endpoints
- Manage inference endpoints

**Org settings**

- Read access to organizations settings
- Write access to organizations settings / member management

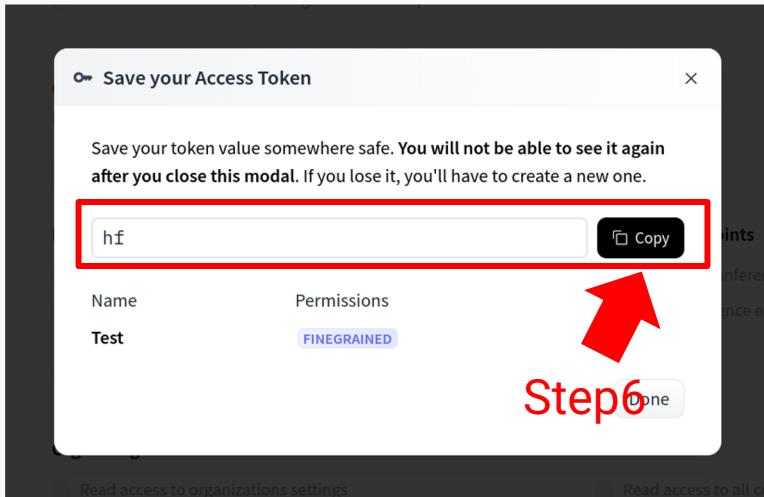
**Collections**

- Read access to all collections in selected organizations
- Write access to all collections in selected organizations

Create token

Step6

## Connect to Huggingface (4/4)



```
1 from huggingface_hub import login
2
3 hf_token =
4 login(token=hf_token, add_to_git_credential=True)

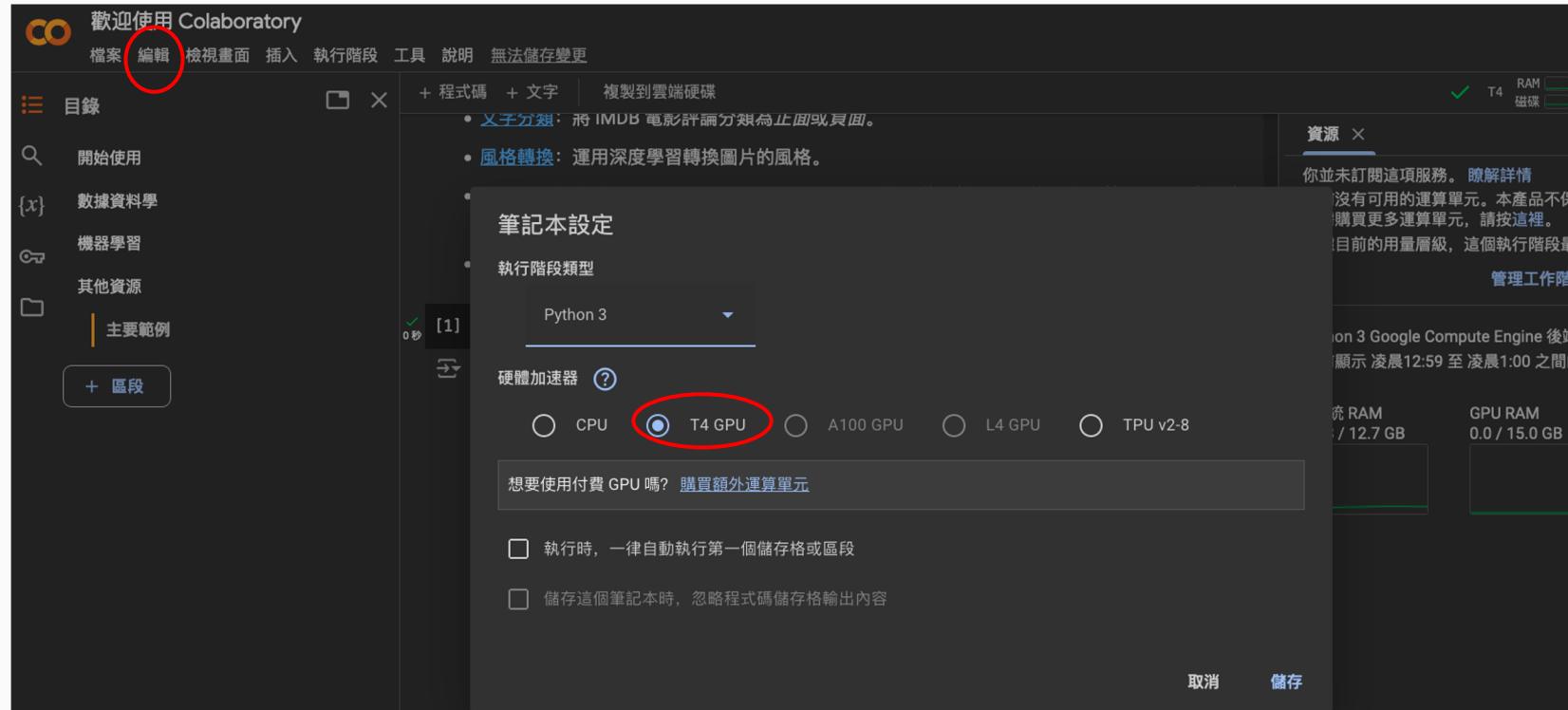
[6] 1 !huggingface-cli whoami
⇒ ZhiQuanFeng
```

Step7

Your access token here

If you succeed, your Hugging Face ID will be displayed here.

# How to use GPU on Colab?



Type in a code block: !nvidia-smi

# TOD01: Set up the environment of Ollama (1/3)

## Introduction to Ollama

- Ollama is a platform designed for running and managing large language models (LLMs) directly **on local devices**, providing a balance between performance, privacy, and control.
- There are also other tools support users to manage LLM on local devices and accelerate it like *vllm*, *Llamofile*, *GPT4ALL*...etc.

### ▼ Launch colabxterm

```
[ ] 1 # TOD01-1: You should install colab-xterm and launch it.  
2 # Write your commands here.
```

```
[ ] 1 # TOD01-2: You should install Ollama.  
2 # You may need root privileges if you use a local machine instead of Colab.
```

Install the colab-xterm package and use “load\_ext” to use it.

# TOD01: Set up the environment of Ollama (2/3)

```
[ ] 1 %xterm
```

```
[ ] 1 # TOD01-3: Pull Llama3.2:1b via Ollama and start the Ollama service in the xterm  
2 # Write your commands in the xterm
```

Use “%xterm” and the followint command to the command line:

- curl -fsSL https://ollama.com/install.sh | sh
- Download Llama3.2-1b using pull command in Ollama
- ollama serve

# A simple test of Ollama

## ▼ Ollama testing

You can test your Ollama status with the following cells.

```
[ ] 1 # Setting up the model that this tutorial will use  
2 MODEL = "llama3.2:1b" # https://ollama.com/library/llama3.2:3b  
3 EMBED_MODEL = "jinaai/jina-embeddings-v2-base-en"
```

```
[ ] 1 # Initialize an instance of the Ollama model  
2 llm = Ollama(model=MODEL)  
3 # Invoke the model to generate responses  
4 response = llm.invoke("What is the capital of Taiwan?")  
5 print(response)
```

- Here we define the retrieval model and document reader model.
- We can perform a simple test for checking Ollama's status.
- The .invoke API is to input sequence to the model and obtain the outputs.

## Define the retriever model

```
[ ] 1 # Create an embedding model
2 model_kwargs = {'trust_remote_code': True}
3 encode_kwargs = {'normalize_embeddings': False}
4 embeddings_model = HuggingFaceEmbeddings(
5     model_name=EMBED_MODEL,
6     model_kwargs=model_kwargs,
7     encode_kwargs=encode_kwargs
8 )
```

Build a simple document retriever using LangChain (HuggingFaceEmbeddings).

# TOD02: Load the cat-facts dataset and prepare the retrieval database

```
[ ] 1 !wget https://huggingface.co/ngxson/demo\_simple\_rag\_py/resolve/main/cat-facts.txt
[ ] 1 # TOD02-1: Load the cat-facts dataset (as `refs`, which is a list of strings for all the cat facts)
[ ] 2 # Write your code here
[ ] 1 from langchain_core.documents import Document
[ ] 2 docs = [Document(page_content=doc, metadata={"id": i}) for i, doc in enumerate(refs)]
```

Load the data to be retrieved and use the  
`langchain.docstore.document.Document` API to construct a database.

# TOD02: Load the cat-facts dataset and prepare the retrieval database

```
[ ] 1 # TOD02-2: Prepare the retrieval database
2 # You should create a Chroma vector store.
3 # search_type can be "similarity" (default), "mmr", or "similarity_score_threshold"
4 vector_store = Chroma.from_documents(
5     # Write your code here
6 )
7 retriever = vector_store.as_retriever(
8     # Write your code here
9 )
```

## TODO3: Set up the `system\_prompt` and configure the prompt

### ▼ Prompt setting

```
[ ] 1 # TODO3: Set up the `system_prompt` and configure the prompt.  
2 system_prompt = # Write your code here  
3 prompt = ChatPromptTemplate.from_messages(  
4     [  
5         ("system", system_prompt),  
6         ("human", "{input}"),  
7     ]  
8 )
```

- For the vectorspace, the common algorithm would be used like Faiss, Chroma...(<https://python.langchain.com/docs/integrations/vectorstores/>) to deal with the extreme huge database.

Write a system prompt to guide the document reader model in performing the desired task.

# TOD04: Build and run the RAG system

```
[ ] 1 # TOD04: Build and run the RAG system
2 # TOD04-1: Load the QA chain
3 # You should create a chain for passing a list of Documents to a model.
4 question_answer_chain = # Write your code here
5                         stuff_documents_chain
6 # TOD04-2: Create retrieval chain
7 # You should create retrieval chain that retrieves documents and then passes them on.
8 chain = # Write your code here
9          retrieval_chain
```

# The Question and Answer Pairs

```
[1] 1 # Question (queries) and answer pairs
2 # Please do not modify this cell.
3 queries = [
4     "How much of a day do cats spend sleeping on average?",
5     "What is the technical term for a cat's hairball?",
6     "What do scientists believe caused cats to lose their sweet tooth?",
7     "What is the top speed a cat can travel over short distances?",
8     "What is the name of the organ in a cat's mouth that helps it smell?",
9     "Which wildcat is considered the ancestor of all domestic cats?",
10    "What is the group term for cats?",
11    "How many different sounds can cats make?",
12    "What is the name of the first cat in space?",
13    "How many toes does a cat have on its back paws?"
14 ]
15 answers = [
16     "2/3",
17     "Bezoar",
18     "a mutation in a key taste receptor",
19     ["31 mph", "49 km"], ←
20     "Jacobson's organ",
21     "the African Wild Cat",
22     "clowder",
23     "100",
24     ["Felicette", "Astrocat"], ←
25     "four",
26 ]
```

The queries (the ten questions) and the correct answers are provided in the notebook file.

## TOD04-3 and TODO5: Improve to let the LLM correctly answer the ten questions.

```
[ ] 1 counts = 0
2 for i, query in enumerate(queries):
3     # TOD04-3: Run the RAG system
4     response = # Write your code here
5     print(f"Query: {query}\nResponse: {response['answer']}\n")
6     # The following lines perform evaluations.
7     # if the answer shows up in your response, the response is considered correct.
8     if type(answers[i]) == list:
9         for answer in answers[i]:
10            if answer.lower() in response['answer'].lower():
11                counts += 1
12                break
13     else:
14         if answers[i].lower() in response['answer'].lower():
15             counts += 1
16
17 # TOD05: Improve to let the LLM correctly answer the ten questions.
18 print(f"Correct numbers: {counts}")
```

Input the query to the RAG system and get the response

Evaluate the response

# Submission

# Scoring

Coding work : **60%**

- (10%) TODO1: Set up the environment of Ollama
- (10%) TODO2: Load the cat-facts dataset and prepare the retrieval database
- (15%) TODO3: Set up the `system\_prompt` and configure the prompt.
- (15%) TODO4: Build and run the RAG system
- (10%, 1% per question) TODO5: Improve to let the LLM correctly answer the ten questions.

Report: **40%**

- (10%) Please describe the **details of your implementation for the RAG system** (please tell us 1. What's in your RAG system? 2. Which retrieval model you use? 3. What's your prompt? 4. What's new in your code in comparison with the code from our lab course?) in this assignment and **list your best score for the ten questions**.
- (10%) Please provide analysis for the RAG performance using different prompts.
- (10%) Please compare the RAG performance with different retrieval models and the performance without using RAG (note that Llama 3.2 should not be fine-tuned in this assignment).
- (10%) Anything that can strengthen your report.

# Delivery policies: File formats

- Coding work: Python file (.py)
  - Download your script via Colab.
- Package list: requirements.txt
  - E.g., numpy==1.26.3
- Report: Microsoft Word (.docx)
- **No other formats are allowed.**
- Zip the files above before uploading you assignment.



# Delivery policies: Filenames

- Do not forget the correct formats!

|              | <b>Filename rule</b>                    | <b>Filename example</b>             |
|--------------|-----------------------------------------|-------------------------------------|
| Coding work  | NLP_HW4_ <b>school</b> _student_ID.py   | NLP_HW4_ <b>NTHU</b> _12345678.py   |
| Report       | NLP_HW4_ <b>school</b> _student_ID.docx | NLP_HW4_ <b>NTHU</b> _12345678.docx |
| Package list | requirements.txt                        |                                     |
| Zipped file  | NLP_HW4_ <b>school</b> _student_ID.zip  | NLP_HW4_ <b>NTHU</b> _12345678.zip  |

# Delivery policies: Things You should include

- In your report:

|                     | Example            |                                               |
|---------------------|--------------------|-----------------------------------------------|
| Environment types   | If Colab or Kaggle | If local                                      |
| Running environment | Colab              | System: Ubuntu 22.04,<br>CPU: Ryzen 7-7800X3D |
| Python version      | Colab              | Python 3.10.1                                 |

# Delivery policies: Rules of coding

- If you use ChatGPT or Generative AI, please specify your usage **both** in:
  - **Code comments**
  - **Reports**
- **No plagiarism.** You should not copy and paste from your classmates. **Submit duplicate code or report will get 0 point !**
- Please provide links if you take the code from the Internet as reference.
- The following behaviors **will cause loss in the score of the assignment:** (1) **Usage with Generative AI without specifications** (2) **Internet sources without specifications** (3) **Plagiarism.**

# Uploading the zipped file

- Please upload your file to NTU COOL.
- Typically, you will have at most three weeks to finish this assignment.
- If you have any question, please e-mail to [nthuikmlab@gmail.com](mailto:nthuikmlab@gmail.com)
- For sending an email, please include **[NLP AS4]** in your email title.
  - Example email title: [NLP AS4] Question about the scoring

# Punishments

|                   |                                                                                    |                                                                                        |                                                               |                                                                                        |                                                |
|-------------------|------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|---------------------------------------------------------------|----------------------------------------------------------------------------------------|------------------------------------------------|
| <b>Rules</b>      | Name your code as<br><b>NLP_HW4_school_student_ID.py</b><br>(only .py is allowed!) | Name your code as<br><b>NLP_HW4_school_student_ID.docx</b><br>(only .docx is allowed!) | Include<br><b>requirements.txt</b> in the folder              | Zip your files into<br><b>NLP_HW4_school_student_ID.zip</b><br>(only .zip is allowed!) | Include the running environment in your report |
| <b>Punishment</b> | -5                                                                                 | -5                                                                                     | -5                                                            | -5                                                                                     | -5                                             |
| <b>Rules</b>      | Include the Python version in your report                                          | Include the GPU card(s) you use in your report                                         | Usage with GAI or Internet sources with proper specifications | Do not copy and paste the code from your classmates                                    |                                                |
| <b>Punishment</b> | -5                                                                                 | -5                                                                                     | -100                                                          | -100 (for both)                                                                        |                                                |