Gauntlet-AI /
class-3-rag-fusion-similarity-search

Code Issues Pull requests Actions Projects Security Insights

1 star 59 forks 0 watching Branches Activity Custom properties
Tags

Public repository

1 Branch 0 Tags Go to file t Go to file Add file Code

ashtilawat23 added gitignore 4378e5f · 16 hours ago

docs	first commit	2 days ago
.env.sample	updated file name	16 hours ago
.gitignore	added gitignore	16 hours ago
Dockerfile	first commit	2 days ago
RAG_fusion_101.ipynb	first commit	2 days ago
README.md	Adjusted envs	18 hours ago
docker-compose.yml	first commit	2 days ago
main.py	first commit	2 days ago
requirements.txt	first commit	2 days ago
run.sh	first commit	2 days ago
similarity_search.ipynb	first commit	2 days ago
upload.py	first commit	2 days ago

README

Class 3: RAG (Retrieval-Augmented Generation)

Introduction

Master a technique that integrates relevant documents into the generation process, enhancing the quality and contextuality of LLM responses. This repo demonstrates a simple RAG example and more advanced implementations.

Class Materials

- Link to Class Slides

Prerequisites

- Docker (for containerized usage - recommended)
- Python 3.11 (for local setup)
- Pinecone account (free tier is sufficient)

Setup

1. Create a Pinecone Index:

Before running the application, you need to set up a Pinecone index:

1. Log in to your Pinecone account at https://app.pinecone.io/
2. Navigate to the "Indexes" section and click "Create Index"
3. Fill in the following details:
   - Name: Choose a name for your index (e.g., "rag-project-index")
   - Dimensions: Set to `3072`
   - Metric: Choose `cosine`
4. For Environment and Region:
   - Cloud Provider: Select `aws`
   - Region: Choose `us-east-1` (Note: These last two options are required for a free instance)
5. Click "Create Index" to confirm
6. If you wish to run `RAG_fusion_101.ipynb` you must create a separate Pinecone index
7. For this second index, follow the above procedure but set the `Dimensions` to `1536`, and give the index a name of your choice
8. Make sure and enter this second index into the `.env` file

Remember the names you gave to your indices, as you'll need them for the `.env` file.

2. Set up environment variables:

- Copy the sample environment file:

```
cp .env.sample .env
```

- Edit the `.env` file and add your API keys and other required variables:

```
OPENAI_API_KEY=your-secret-key
LANGCHAIN_API_KEY=your-secret-key
LANGCHAIN_TRACING_V2=true
LANGCHAIN_PROJECT=gauntlet_class_3
PINECONE_API_KEY=your-secret-key
PINECONE_INDEX=your-3072-dimension-index-name
PINECONE_INDEX_TWO=your-1536-dimension-index-name
```

Make sure to use the relevant index names you created for `PINECONE_INDEX` and `PINECONE_INDEX_TWO`, respectfully.

Quick Start with Docker

1. Run the upload script:

```
docker compose run --rm upload_service
```

2. Run the main RAG example:

```
docker compose run --rm rag_app
```

3. Start Jupyter for notebook examples:

```
docker compose up jupyter
```

4. Run a specific script (any new `.py` file you may add):

```
docker compose run --rm rag_app python <script_name.py>
```

Note: The Docker setup will automatically use the environment variables from your `.env` file. You don't need to export them to your system environment when using Docker.

Running Different Scripts

You can use the provided `run.sh` script for easier execution. Make sure to make the script executable with `chmod +x run.sh` before using:

```
./run.sh upload
./run.sh main
./run.sh jupyter
./run.sh <your_new_py_file>
```

Local Setup (Alternative to Docker)

If you prefer to run the examples locally:

1. Ensure you have Python 3.10+ installed.
2. Clone the repository:

```
git clone [repository-url]
cd [repository-name]
```

3. Set up the environment:

```
python3 -m venv .venv
source .venv/bin/activate  # On Windows use `.venv\Scripts\activate`
pip install -r requirements.txt
```

4. Configure environment variables as described in the Setup section.
5. Export the environment variables (python-dotenv should handle this automatically in the):

```
export $(cat .env | xargs)
```

6. Run an example:

```
python3 upload.py
```

Troubleshooting

- Ensure you're using Python 3.11.0 or later for local setup.
- For Docker issues, check your Docker installation and version.
- If you encounter package issues, try updating pip: `pip install --upgrade pip`
- Make sure all required environment variables are set in your `.env` file.
- If you're having issues with environment variables in Docker, ensure your `.env` file is in the same directory as your `docker-compose.yml` file.
- If you encounter issues with Pinecone, double-check that your index is created correctly and that you're using the correct API key and index name(s) in your `.env` file.

Need Help?

Releases

No releases published

Packages

No packages published

Languages


Jupyter Notebook 80.4%
Python 16.8%
Shell 1.6%
Dockerfile 1.2%