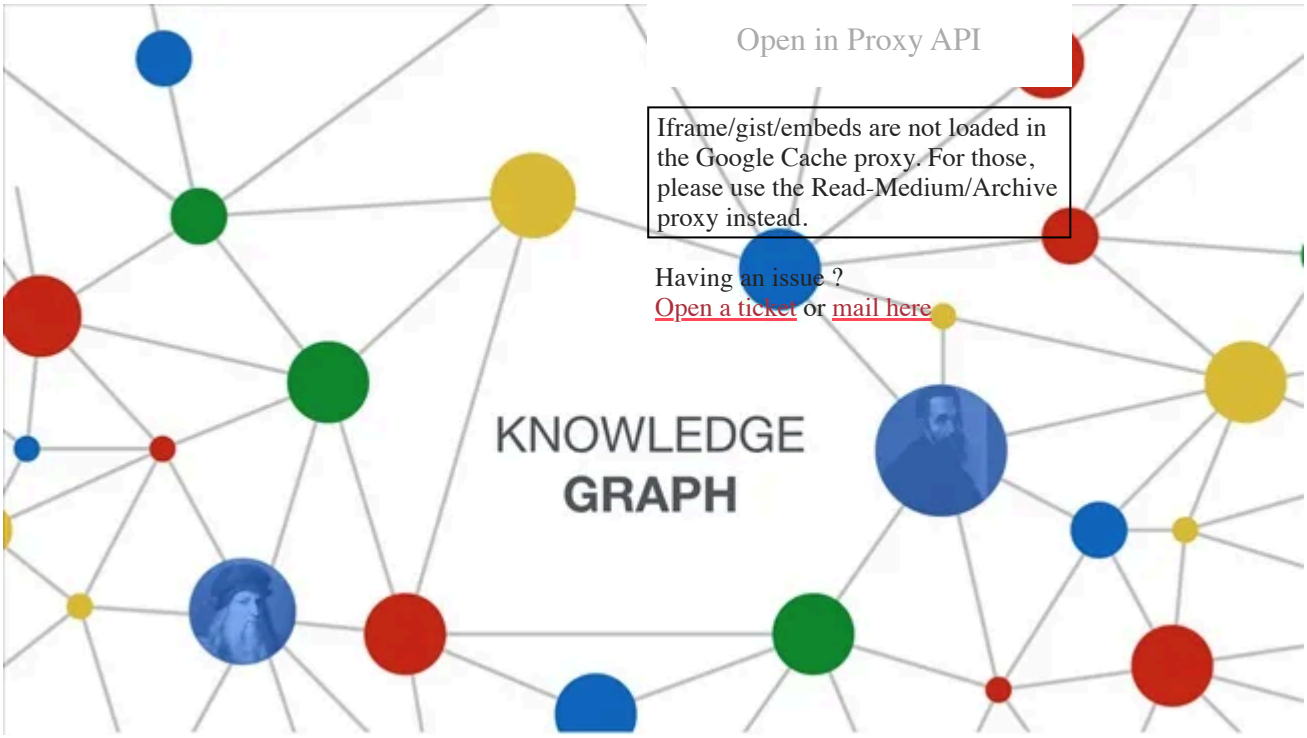
Knowledge Graphs in RAGs (with Llama-Index)

Phil · Follow
3 min read · Jan 29, 2024



Are #KnowledgeGraphs (KGs) better than #VectorDBs in RAGs? Yes and No!

Technically, KGs provide more #precise output than Vector DBs. KGs also support more #diverse and #complex querying than Vector DBs. Moreover, KGs enable better #reasoning and inferencing than Vector DBs

However, the KGs vs Vector DBs comparison is a game of Apples vs Oranges. Knowledge Graphs are best suited for documents with clear #relationships while Vector DBs are more relevant to #similarity-based contexts. So, choosing between

KGs and Vector DBs depends on the specific requirements and objectives of yourRAG project.

Here is my simple demonstration for the stepwise implementation of Knowledge Graph using the #KnowledgeGraphIndex class of LlamaIndex

1. **Import Modules**

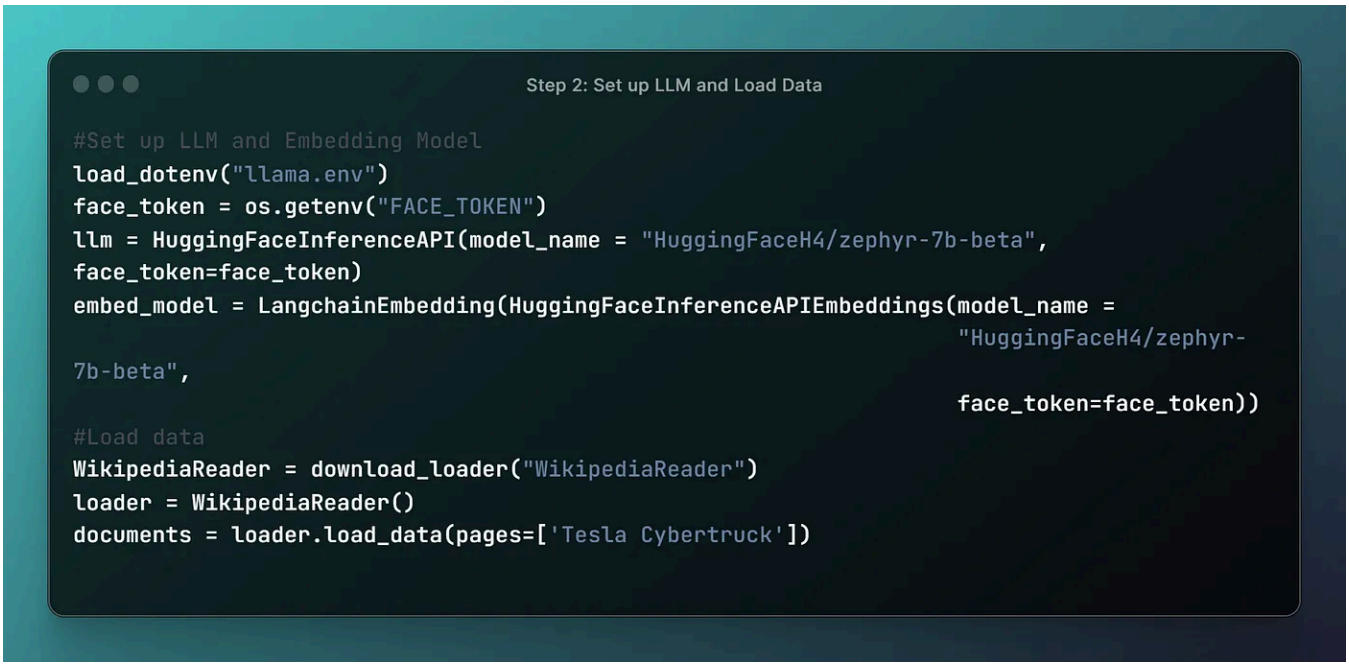
Step 1: Import Modules and set up Logging

```
#Setup logging diagnostics
import logging
import sys
logging.basicConfig(stream=sys.stdout, level=logging.INFO)
logging.getLogger().addHandler(logging.StreamHandler(stream=sys.stdout))

#Import all necessary modules
from llama_index import (ServiceContext, download_loader, KnowledgeGraphIndex)
from llama_index.graph_stores import SimpleGraphStore
from llama_index.storage.storage_context import StorageContext
from llama_index.llms import HuggingFaceInferenceAPI
from llama_index.embeddings import LangchainEmbedding
from langchain.embeddings import HuggingFaceInferenceAPIEmbeddings
from pyvis.network import Network
from Ipython.display import display, HTML
from dotenv import load_dotenv
import Ipython
import os
```

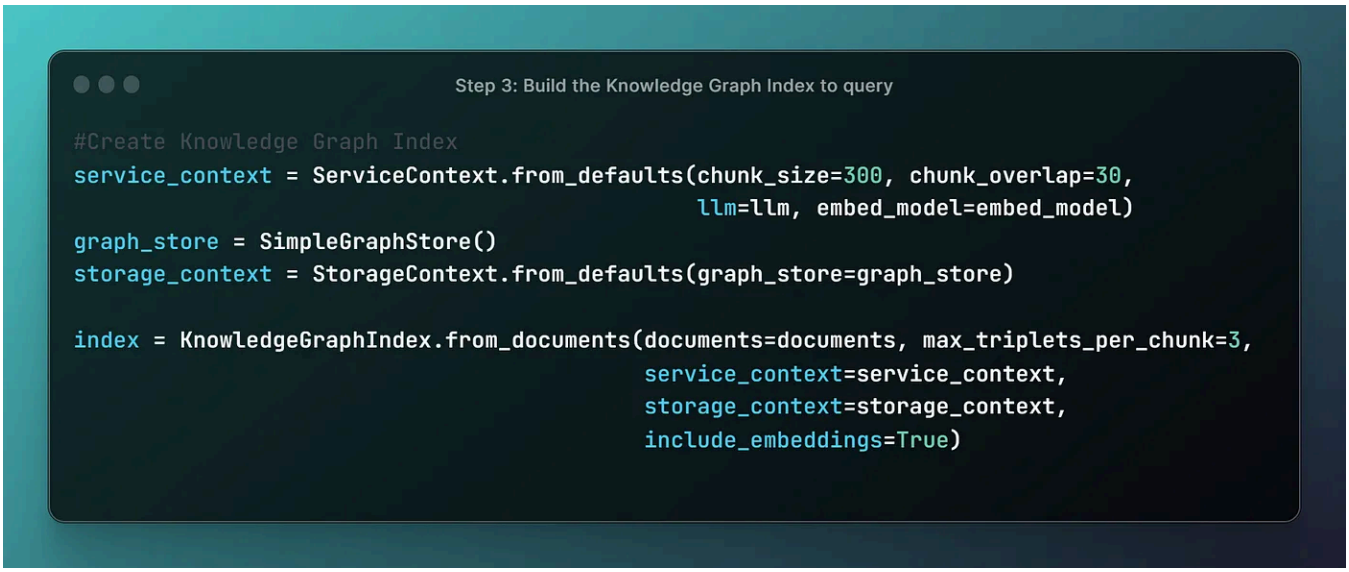
The **KnowledgeGraphIndex** class streamlines RAG knowledge graph construction by automating entity extraction & relationship recognition from text, eliminating complex manual parsing. It also offers flexibility in customization, allowing you to tailor the graph structure and reasoning rules to your specific needs.

2. Set up LLM, embed_model, and Load Data



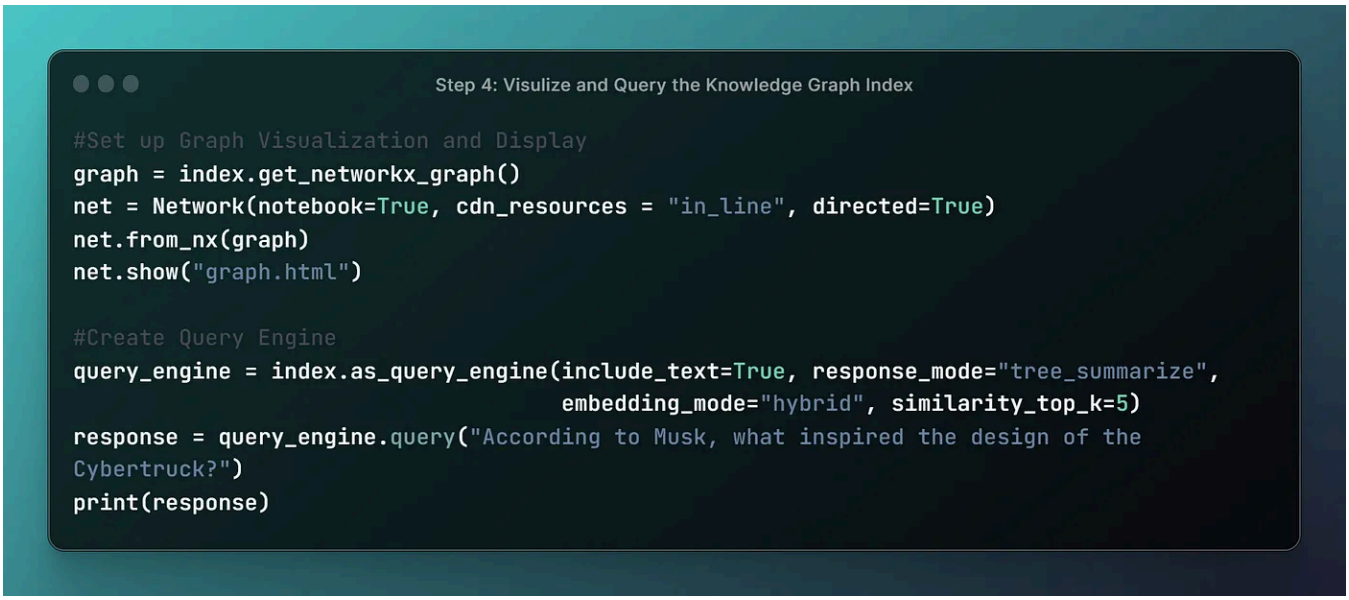
The `HuggingFaceInferenceAPI` class simplifies the consumption of Large Language Models (LLMs) for RAGs by providing a straightforward interface to pretrained models, facilitating efficient retrieval and generation of responses.

3. Construct the Knowledge Graph Index



The `SimpleGraphStore` class in `llama_index` handles storage and retrieval of entities and relationships automatically, offers lightweight, in-memory storage solution for agile development, and integrates seamlessly with `KnowledgeGraphIndex` for a streamlined workflow.

4. Visualize and Query the Index



The `pyvis` module turns your RAG's knowledge graph into an interactive web visualization, letting you explore connections and patterns at a glance. But that's not all! `LlamaIndex`'s `KnowledgeGraphQueryEngine` class is your key to advanced questioning. Craft nuanced queries, navigate entities through different modes, and uncover hidden insights your RAG holds — all with flexibility and precision.

Conclusion

Overall, **Knowledge Graphs** capture rich relationships among entities, enabling more precise, diverse, and complex querying and reasoning in RAGs than vector databases.

Retrieval Augmented | Rags | Ai Engineer | Python | Llamaindex

Follow

Written by Phil

6 Followers