名词向量处理

笔记本:	我的第一个笔记本		
创建时间:	2024/11/19 12:05	**更新时间:**	2024/11/19 12:09
作者:	153klixx022		
URL:	file:///D:/科大讯飞实习/上海市交通系统交易问答框架v3.0/项目详细说明/名词向量存...		

Milvus数据库部分的存储和检索

search_Milvus.py

用途

该脚本通过 Milvus 向量数据库进行语义搜索，主要用于从一个预定义的向量集合中检索最相关的文档。

主要功能

1. **初始化嵌入模型**
   使用 OllamaEmbeddings 的嵌入模型 bge-m3:latest，将文本转换为向量表示。
2. **连接 Milvus 向量数据库**
3. **检索功能**

代码如下

```
from langchain_ollama import OllamaEmbeddings
from langchain_chroma import Chroma
from langchain_milvus import Milvus


# Initialize embeddings and vector store
embeddings = OllamaEmbeddings(model="bge-m3:latest")


URI = "tcp://47.102.103.246:19530"
vector_store = Milvus(
    embeddings,
    connection_args={"uri": URI},
    collection_name="Noun_Collection_bge_v3",
)


retriever = vector_store.as_retriever(search_type="mmr", search_kwargs={"k": 1})
```

```
#docs = retriever.invoke("阜阳站信什么锁室内设备招标公告这个项目完整名称是什么，我不太记得")
#docs = retriever.invoke("上海城建审图咨询有限公司能否提供相关网址")
#docs = retriever.invoke("新建杭州乘务员工程施工总价承包招标公告")
docs = retriever.invoke("阜阳站信什么锁室内设备招标公告这个项目完整名称是什么，我不太记得")
# Print the retrieved documents
infer_info =[]
for doc in docs:
    #print(doc.page_content)
    #infer_info.append(doc.page_content)
    infer_info.append(doc.metadata["content"])
print(infer_info)
```

---

vectorstore_company_name_Milvus.py

用途

用于加载、处理与企业相关的中标信息并将其存储到 Milvus 数据库中。

主要功能

1. **加载 JSON 文件**
   - 使用 JSONLoader 加载名为 中标总表.json 的 JSON 文件。
   - 通过 JQ schema .prizes[] 定位 JSON 中的具体内容。
2. **数据处理**
   - 解码 Unicode 并提取相关字段：
3. **存储向量**
   - 初始化 Milvus 数据库：
   - 生成 UUID 并将文档添加到数据库：

代码如下

```
from langchain_ollama import ChatOllama, OllamaEmbeddings
from langchain_community.document_loaders import JSONLoader
import json
from langchain_chroma import Chroma
from uuid import uuid4
from langchain_milvus import Milvus
from uuid import uuid4
from langchain_core.documents import Document

# Load the documents
loader = JSONLoader(file_path="中标总表.json", jq_schema=".prizes[]",
text_content=False)
documents = loader.load()

# Process the documents to decode Unicode sequences and extract text
for doc in documents:
    content_dict = json.loads(doc.page_content)
    name = content_dict.get('中标人名称', '')
    prizes = content_dict.get('价格（万）', '')
    info = content_dict.get('主要标的信息', '')
    web_url = content_dict.get('页面网址', '')
    num = content_dict.get('项目编号', '')
    pro_name = content_dict.get('项目名称', '')
```

```
    date = content_dict.get('日期', '')
    pro_class = content_dict.get('公路项目', '')
    comp_url = content_dict.get('爱企查网址', '')
    com_ui = content_dict.get('公司官网', '')
    # Update the page_content with the actual Chinese text
    doc.page_content = name
    doc.metadata["content"] = '中标人名称：' + str(name) + ' ' +'价格（万）：'+
str(prizes) + ' ' +'主要标的信息：'+ str(info) + ' ' +'页面网址：' +
str(web_url)  + ' ' +         '项目编号：' + str(num) + ' ' + '项目名称：'+pro_name
+ ' ' +'日期：'+ str(date) + ' ' + '公路项目：'+ str(pro_class) + ' ' + '爱企查网
址'+str(comp_url)  + ' ' +        '公司官网：'+str(com_ui)

# Initialize embeddings and vector store
embeddings = OllamaEmbeddings(model="bge-m3:latest")

URI = "tcp://47.102.103.246:19530"

vector_store = Milvus(
    embedding_function=embeddings,
    connection_args={"uri": URI},
    collection_name="Noun_Collection_bge_v3",
)


# vector_store_saved = Milvus.from_documents(
#     [Document(page_content="foo!")],
#     embeddings,
#     collection_name="langchain_example",
#     connection_args={"uri": URI},
# )


uuids = [str(uuid4()) for _ in range(len(documents))]
vector_store.add_documents(documents=documents, ids=uuids)
```

---

vectorstore_project_name_Milvus.py

用途

该脚本加载与项目相关的信息，并将其存储到 Milvus 数据库中。

主要功能

1. **加载 JSON 文件**
   - 使用 JSONLoader 加载名为 local_file.json 的 JSON 文件。
   - 通过 JQ schema .prizes[] 定位 JSON 中的具体内容。
2. **数据处理**
   - 提取字段：
3. **存储向量**
   - 初始化 Milvus 数据库，地址与 vectorstore_company_name_Milvus.py 相同。
   - 将生成的文档添加到数据库。

代码如下

```
from langchain_ollama import ChatOllama, OllamaEmbeddings
```

```
from langchain_community.document_loaders import JSONLoader
import json
from langchain_chroma import Chroma
from uuid import uuid4
from langchain_milvus import Milvus
from uuid import uuid4
from langchain_core.documents import Document


# Load the documents
loader = JSONLoader(file_path="local_file.json", jq_schema=".prizes[]",
text_content=False)
documents = loader.load()

# Process the documents to decode Unicode sequences and extract text
for doc in documents:
    content_dict = json.loads(doc.page_content)
    name = content_dict.get('项目名称', '')
    num = content_dict.get('项目编号', '')
    time = content_dict.get('时间', '')
    class1 = content_dict.get('交易分类', '')
    class2 = content_dict.get('项目类型', '')
    class3 = content_dict.get('公告类型', '')
    # Update the page_content with the actual Chinese text
    doc.page_content = name
    doc.metadata["content"] ='项目名称:'+ name + ' ' +'项目编号:'+ num + ' ' + '时
间:' + time + ' ' +'交易分类:'+ class1  + ' ' + '项目类型:' + class2 + ' ' + '公告
类型:' +       class3

# Initialize embeddings and vector store
embeddings = OllamaEmbeddings(model="bge-m3:latest")

URI = "tcp://47.102.103.246:19530"

vector_store = Milvus(
    embedding_function=embeddings,
    connection_args={"uri": URI},
    collection_name="Noun_Collection_bge_v3",
)
# vector_store_saved = Milvus.from_documents(
#     [Document(page_content="foo!")],
#     embeddings,
#     collection_name="langchain_example",
#     connection_args={"uri": URI},
# )


uuids = [str(uuid4()) for _ in range(len(documents))]
vector_store.add_documents(documents=documents, ids=uuids)
```

Chromadb本地部分的存储和检索

类似的方法去存储和检索