5998 words | 15 minutes

Build and use a vector database

The source code for this article is **here** . If you need to reproduce, you can download and run the source code.

1. Pre-order configuration

The focus of this section is to build and use a vector database. Therefore, after reading the data, we will skip the data processing step and go straight to the topic. For steps such as data cleaning, please refer to Section 3.

python

```python
import os
from dotenv import load_dotenv, find_dotenv

# 读取本地/项目的环境变量。
# find_dotenv()寻找并定位.env文件的路径
# load_dotenv()读取该.env文件，并将其中的环境变量加载到当前的运行环境中
# 如果你设置的是全局的环境变量，这行代码则没有任何作用。
_ = load_dotenv(find_dotenv())

# 如果你需要通过代理端口访问，你需要如下配置
# os.environ['HTTPS_PROXY'] = 'http://127.0.0.1:7890'
# os.environ["HTTP_PROXY"] = 'http://127.0.0.1:7890'

# 获取folder_path下所有文件路径，储存在file_paths里
file_paths = []
folder_path = '../../data_base/knowledge_db'
for root, dirs, files in os.walk(folder_path):
    for file in files:
        file_path = os.path.join(root, file)
        file_paths.append(file_path)
print(file_paths[:3])
```

```markup
['../../data_base/knowledge_db/prompt_engineering/6. 文本转换 Transforming.
```

```python
from langchain.document_loaders.pdf import PyMuPDFLoader
from langchain.document_loaders.markdown import UnstructuredMarkdownLoade

# 遍历文件路径并把实例化的loader存放在loaders里
loaders = []

for file_path in file_paths:

    file_type = file_path.split('.')[-1]
    if file_type == 'pdf':
        loaders.append(PyMuPDFLoader(file_path))
    elif file_type == 'md':
        loaders.append(UnstructuredMarkdownLoader(file_path))
```

```python
# 下载文件并存储到text
texts = []

for loader in loaders: texts.extend(loader.load())
```

The variable type after loading is `langchain_core.documents.base.Document` , the document variable type also contains two attributes

- `page_content` Contains the content of this document.
- `meta_data` Descriptive data related to the document.

```python
text = texts[1]
print(f"每一个元素的类型：{type(text)}.",
    f"该文档的描述性数据：{text.metadata}",
    f"查看该文档的内容:\n{text.page_content[0:]}",
    sep="\n------\n")
```

markup

~~~
每一个元素的类型：<class 'langchain_core.documents.base.Document'>.
------
该文档的描述性数据：{'source': '../../data_base/knowledge_db/prompt_engineeri
------
查看该文档的内容:
第四章 文本概括

在繁忙的信息时代，小明是一名热心的开发者，面临着海量的文本信息处理的挑战。他需要通过研究

这个功能对小明来说如同灯塔一样，照亮了他处理信息海洋的道路。LLM 的强大能力在于它可以将

通过编程调用 AP I接口，小明成功实现了这个文本摘要的功能。他感叹道：“这简直就像一道魔法，

一、单一文本概括

以商品评论的总结任务为例：对于电商平台来说，网站上往往存在着海量的商品评论，这些评论反映

接下来我们提供一段在线商品评价作为示例，可能来自于一个在线购物平台，例如亚马逊、淘宝、京

python
prod_review = """
这个熊猫公仔是我给女儿的生日礼物，她很喜欢，去哪都带着。
公仔很软，超级可爱，面部表情也很和善。但是相比于价钱来说，
它有点小，我感觉在别的地方用同样的价钱能买到更大的。
快递比预期提前了一天到货，所以在送给女儿之前，我自己玩了会。
"""

1.1 限制输出文本长度

我们首先尝试将文本的长度限制在30个字以内。

```python
from tool import get_completion

prompt = f"""
您的任务是从电子商务网站上生成一个产品评论的简短摘要。

请对三个反引号之间的评论文本进行概括，最多30个字。

评论: {prod_review}
"""
~~~

```
response = get_completion(prompt)
print(response)
```

我们可以看到语言模型给了我们一个符合要求的结果。

注意：在上一节中我们提到了语言模型在计算和判断文本长度时依赖于分词器，而分词器在字符统计

1.2 设置关键角度侧重

在某些情况下，我们会针对不同的业务场景对文本的侧重会有所不同。例如，在商品评论文本中，物

我们可以通过增强输入提示（Prompt），来强调我们对某一特定视角的重视。

1.2.1 侧重于快递服务

```python
prompt = f"""
您的任务是从电子商务网站上生成一个产品评论的简短摘要。

请对三个反引号之间的评论文本进行概括，最多30个字，并且侧重在快递服务上。

评论: {prod_review}
"""

response = get_completion(prompt)
print(response)
```

通过输出结果，我们可以看到，文本以“快递提前到货”开头，体现了对于快递效率的侧重。

1.2.2 侧重于价格与质量

```python
prompt = f"""
您的任务是从电子商务网站上生成一个产品评论的简短摘要。

请对三个反引号之间的评论文本进行概括，最多30个词汇，并且侧重在产品价格和质量上。

评论: {prod_review}
"""

response = get_completion(prompt)
```

```
print(response)
```

通过输出的结果，我们可以看到，文本以“可爱的熊猫公仔，质量好但有点小，价格稍高”开头，体现

1.3 关键信息提取

在1.2节中，虽然我们通过添加关键角度侧重的 Prompt ，确实让文本摘要更侧重于某一特定方面，

下面让我们来一起来对文本进行提取信息吧！

```python
prompt = f"""
您的任务是从电子商务网站上的产品评论中提取相关信息。

请从以下三个反引号之间的评论文本中提取产品运输相关的信息，最多30个词汇。

评论: {prod_review}
"""

response = get_completion(prompt)
print(response)
```

二、同时概括多条文本

在实际的工作流中，我们往往要处理大量的评论文本，下面的示例将多条用户评价集合在一个列表中

```python
review_1 = prod_review

一盏落地灯的评论

review_2 = """
我需要一盏漂亮的卧室灯，这款灯不仅具备额外的储物功能，价格也并不算太高。
收货速度非常快，仅用了两天的时间就送到了。
不过，在运输过程中，灯的拉线出了问题，幸好，公司很乐意寄送了一根全新的灯线。
新的灯线也很快就送到手了，只用了几天的时间。
装配非常容易。然而，之后我发现有一个零件丢失了，于是我联系了客服，他们迅速地给我寄来了缺
对我来说，这是一家非常关心客户和产品的优秀公司。
"""

一把电动牙刷的评论
```

~~~~
review_3 = """
我的牙科卫生员推荐了电动牙刷，所以我就买了这款。
到目前为止，电池续航表现相当不错。
初次充电后，我在第一周一直将充电器插着，为的是对电池进行条件养护。
过去的3周里，我每天早晚都使用它刷牙，但电池依然维持着原来的充电状态。
不过，牙刷头太小了。我见过比这个牙刷头还大的婴儿牙刷。
我希望牙刷头更大一些，带有不同长度的刷毛，
这样可以更好地清洁牙齿间的空隙，但这款牙刷做不到。
总的来说，如果你能以50美元左右的价格购买到这款牙刷，那是一个不错的交易。
制造商的替换刷头相当昂贵，但你可以购买价格更为合理的通用刷头。
这款牙刷让我感觉就像每天都去了一次牙医，我的牙齿感觉非常干净!
"""

一台搅拌机的评论

review_4 = """
在11月份期间，这个17件套装还在季节性促销中，售价约为49美元，打了五折左右。
可是由于某种原因（我们可以称之为价格上涨），到了12月的第二周，所有的价格都上涨了，
同样的套装价格涨到了70-89美元不等。而11件套装的价格也从之前的29美元上涨了约10美元。
看起来还算不错，但是如果你仔细看底座，刀片锁定的部分看起来没有前几年版本的那么漂亮。
然而，我打算非常小心地使用它
（例如，我会先在搅拌机中研磨豆类、冰块、大米等坚硬的食物，然后再将它们研磨成所需的粒度，
接着切换到打蛋器刀片以获得更细的面粉，如果我需要制作更细腻/少果肉的食物）。
在制作冰沙时，我会将要使用的水果和蔬菜切成细小块并冷冻
（如果使用菠菜，我会先轻微煮熟菠菜，然后冷冻，直到使用时准备食用。
如果要制作冰糕，我会使用一个小到中号的食物加工器），这样你就可以避免添加过多的冰块。
大约一年后，电机开始发出奇怪的声音。我打电话给客户服务，但保修期已经过期了，
所以我只好购买了另一台。值得注意的是，这类产品的整体质量在过去几年里有所下降
，所以他们在一定程度上依靠品牌认知和消费者忠诚来维持销售。在大约两天内，我收到了新的搅拌
"""

reviews = [review_1, review_2, review_3, review_4]

```


```python
for i in range(len(reviews)):
    prompt = f"""
    你的任务是从电子商务网站上的产品评论中提取相关信息。

```
~~~~

三、英文版

1.1 单一文本概括

```
python
prod_review = """
Got this panda plush toy for my daughter's birthday, \
who loves it and takes it everywhere. It's soft and \
super cute, and its face has a friendly look. It's \
a bit small for what I paid though. I think there \
might be other options that are bigger for the \
same price. It arrived a day earlier than expected, \
so I got to play with it myself before I gave it \
to her.
"""
```

```python
prompt = f"""
Your task is to generate a short summary of a product \
review from an ecommerce site.

Summarize the review below, delimited by triple
backticks, in at most 30 words.

Review: {prod_review}
"""

response = get_completion(prompt)
print(response)
```

1.2 设置关键角度侧重

1.2.1 侧重于快递服务

```python
prompt = f"""
Your task is to generate a short summary of a product \
review from an ecommerce site to give feedback to the \
Shipping deparmtment.

Summarize the review below, delimited by triple
backticks, in at most 30 words, and focusing on any aspects \
that mention shipping and delivery of the product.

Review: {prod_review}
"""

response = get_completion(prompt)
print(response)
```

1.2.2 侧重于价格和质量

```python
prompt = f"""
Your task is to generate a short summary of a product \
review from an ecommerce site to give feedback to the \
pricing deparmtment, responsible for determining the \
price of the product.

Summarize the review below, delimited by triple
backticks, in at most 30 words, and focusing on any aspects \
that are relevant to the price and perceived value.

Review: {prod_review}
"""

response = get_completion(prompt)
print(response)
```

1.3 关键信息提取

```python
prompt = f"""
Your task is to extract relevant information from \
a product review from an ecommerce site to give \
feedback to the Shipping department.

From the review below, delimited by triple quotes \
extract the information relevant to shipping and \
delivery. Limit to 30 words.

Review: {prod_review}
"""
```

~~~
response = get_completion(prompt)
print(response)
```

2.1 同时概括多条文本

```python
review_1 = prod_review

review for a standing lamp

review_2 = """
Needed a nice lamp for my bedroom, and this one \
had additional storage and not too high of a price \
point. Got it fast - arrived in 2 days. The string \
to the lamp broke during the transit and the company \
happily sent over a new one. Came within a few days \
as well. It was easy to put together. Then I had a \
missing part, so I contacted their support and they \
very quickly got me the missing piece! Seems to me \
to be a great company that cares about their customers \
and products.
"""

review for an electric toothbrush

review_3 = """
My dental hygienist recommended an electric toothbrush, \
which is why I got this. The battery life seems to be \
pretty impressive so far. After initial charging and \
leaving the charger plugged in for the first week to \
condition the battery, I've unplugged the charger and \
been using it for twice daily brushing for the last \
3 weeks all on the same charge. But the toothbrush head \
is too small. I’ve seen baby toothbrushes bigger than \
this one. I wish the head was bigger with different \
length bristles to get between teeth better because \
this one doesn’t.  Overall if you can get this one \
around the $50 mark, it's a good deal. The manufactuer's \
replacements heads are pretty expensive, but you can \
get generic ones that're more reasonably priced. This \
toothbrush makes me feel like I've been to the dentist \
~~~

~~~~
every day. My teeth feel sparkly clean!
"""

review for a blender

review_4 = """
So, they still had the 17 piece system on seasonal \
sale for around $49 in the month of November, about \
half off, but for some reason (call it price gouging) \
around the second week of December the prices all went \
up to about anywhere from between $70-$89 for the same \
system. And the 11 piece system went up around $10 or \
so in price also from the earlier sale price of $29. \
So it looks okay, but if you look at the base, the part \
where the blade locks into place doesn’t look as good \
as in previous editions from a few years ago, but I \
plan to be very gentle with it (example, I crush \
very hard items like beans, ice, rice, etc. in the \
blender first then pulverize them in the serving size \
I want in the blender then switch to the whipping \
blade for a finer flour, and use the cross cutting blade \
first when making smoothies, then use the flat blade \
if I need them finer/less pulpy). Special tip when making \
smoothies, finely cut and freeze the fruits and \
vegetables (if using spinach-lightly stew soften the \
spinach then freeze until ready for use-and if making \
sorbet, use a small to medium sized food processor) \
that you plan to use that way you can avoid adding so \
much ice if at all-when making your smoothie. \
After about a year, the motor was making a funny noise. \
I called customer service but the warranty expired \
already, so I had to buy another one. FYI: The overall \
quality has gone done in these types of products, so \
they are kind of counting on brand recognition and \
consumer loyalty to maintain sales. Got it in about \
two days.
"""

reviews = [review_1, review_2, review_3, review_4]
```

```python
for i in range(len(reviews)):
~~~~

~~~
    prompt = f"""
    Your task is to generate a short summary of a product \
    review from an ecommerce site.

```
~~~

```python
from langchain.text_splitter import RecursiveCharacterTextSplitter

# 切分文档
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=500, chunk_overlap=50)

split_docs = text_splitter.split_documents(texts)
```

2. Build Chroma vector library

Langchain integrates with over 30 different vector repositories. We chose Chroma because it is lightweight and data is stored in memory, which makes it very easy to launch and start using.

LangChain can directly use the embeddings of OpenAI and Baidu Qianfan. At the same time, we can also customize the embedding APIs that they do not support. For example, we can encapsulate a zhupuai_embedding based on the interface provided by LangChain to connect Zhipu's Embedding API to LangChain. In the explanation of LangChain custom embedding encapsulation in the appendix of this chapter , we take Zhipu Embedding API as an example to introduce how to encapsulate other Embedding APIs into LangChain. Interested readers are welcome to read.

Note: If you use Zhipu API, you can refer to the explanation content to implement the encapsulation code, or you can directly use the encapsulated code zhipuai_embedding.py , which can also be downloaded to the same directory as this notebook, and then you can directly import our encapsulated function. In the following code Cell, we use Zhipu's Embedding by default, and present the other two Embedding usage codes as comments. If you are using Baidu API or OpenAI API, you can use the code in the following Cell according to the situation.

```python
# 使用 OpenAI Embedding
# from langchain.embeddings.openai import OpenAIEmbeddings
# 使用百度千帆 Embedding
# from langchain.embeddings.baidu_qianfan_endpoint import QianfanEmbeddin
# 使用我们自己封装的智谱 Embedding，需要将封装代码下载到本地使用
from zhipuai_embedding import ZhipuAIEmbeddings

# 定义 Embeddings
# embedding = OpenAIEmbeddings()
embedding = ZhipuAIEmbeddings()
# embedding = QianfanEmbeddingsEndpoint()

# 定义持久化路径
persist_directory = '../../data_base/vector_db/chroma'
```

```python
!rm -rf '../../data_base/vector_db/chroma'  # 删除旧的数据库文件（如果文件夹中有
```

```python
from langchain.vectorstores.chroma import Chroma

vectordb = Chroma.from_documents(
    documents=split_docs[:20], # 为了速度，只选择前 20 个切分的 doc 进行生成；使
    embedding=embedding,
    persist_directory=persist_directory  # 允许我们将persist_directory目录保
)
```

After this, we make sure to persist the vector database by running vectordb.persist so that we can use it in future lessons.

Let's save it for later!

```python
vectordb.persist()
```

```python
print(f"向量库中存储的数量：{vectordb._collection.count()}")
```

```markup
向量库中存储的数量：20
```

3. Vector Search

3.1 Similarity Retrieval

Chroma's similarity search uses the cosine distance, which is:

$$similari\ ty = cos(A,B) = \frac{A \cdot B}{\| A \| \| B \|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2}\sqrt{\sum_1^n b_i^2}}$$

ina_i,b_iThey are vectors$A$,BThe amount.

This function can be used when you need the database to return results strictly sorted by cosine similarity `similarity_search` .

```python
question="什么是大语言模型"
```

```python
sim_docs = vectordb.similarity_search(question,k=3)
print(f"检索到的内容数：{len(sim_docs)}")
```

```markup
检索到的内容数：3
```

```python
for i, sim_doc in enumerate(sim_docs):
    print(f"检索到的第{i}个内容：\n{sim_doc.page_content[:200]}", end="\n---
```

```markup
检索到的第0个内容:
第六章 文本转换

大语言模型具有强大的文本转换能力，可以实现多语言翻译、拼写纠正、语法调整、格式转换等不同

在本章中，我们将介绍如何通过编程调用API接口，使用语言模型实现文本转换功能。通过代码示例，

掌握调用大语言模型接口进行文本转换的技能，是开发各种语言类应用的重要一步。文
--------------
检索到的第1个内容:
以英译汉为例，传统统计机器翻译多倾向直接替换英文词汇，语序保持英语结构，容易出现中文词汇

大语言模型翻译的这些优势使其生成的中文文本更加地道、流畅，兼具准确的意义表达。利用大语言
--------------
检索到的第2个内容:
通过这个例子，我们可以看到大语言模型可以流畅地处理多个转换要求，实现中文翻译、拼写纠正、

利用大语言模型强大的组合转换能力，我们可以避免多次调用模型来进行不同转换，极大地简化了工

六、英文版

1.1 翻译为西班牙语

python
prompt = f"""
Translate the fo
--------------
```

3.2 MMR retrieval

If we only consider the relevance of the retrieved content, the content will be too monotonous and important information may be lost.

Maximum marginal relevance (`MMR, Maximum marginal relevance`) can help us increase the richness of content while maintaining relevance.

The core idea is to select a document with low relevance but rich information after selecting a highly relevant document. This can increase the diversity of content while maintaining relevance and avoid too single results.

python

```python
mmr_docs = vectordb.max_marginal_relevance_search(question,k=3)
```

python

```python
for i, sim_doc in enumerate(mmr_docs):
    print(f"MMR 检索到的第{i}个内容: \n{sim_doc.page_content[:200]}", end="\
```

markup

```
MMR 检索到的第0个内容:
第六章 文本转换

大语言模型具有强大的文本转换能力，可以实现多语言翻译、拼写纠正、语法调整、格式转换等不同

在本章中，我们将介绍如何通过编程调用API接口，使用语言模型实现文本转换功能。通过代码示例，

掌握调用大语言模型接口进行文本转换的技能，是开发各种语言类应用的重要一步。文
--------------
MMR 检索到的第1个内容:
"This phrase is to cherck chatGPT for spelling abilitty"  # spelling
]
--------------
MMR 检索到的第2个内容:
room.

room. Yes, adults also like pandas

too.

too. She takes it everywhere with her, and it's super soft and

cute.  One

cute. However, one of the ears is a bit lower than the other, and I don't
--------------
```

The source code for this article is **here** . If you need to reproduce, you can download and run the source code.

‹ Previous chapter
3. Data processing

Next Chapter ›
Appendix. LangChain Custom Embedding Explanation