6619 words | 17 minutes

data processing

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

To build our local knowledge base, we need to process local documents stored in various types, read local documents, and convert the contents of local documents into word vectors through the Embedding method described above to build a vector database. In this section, we start with some practical examples to explain how to process local documents.

1. Source document selection

We use some classic open source courses from Datawhale as examples, including:

- "Machine Learning Formula Explanation" PDF version
- "LLM Introductory Tutorial for Developers, Part 1 Prompt Engineering" md version
  We place the knowledge base source data in the ../data_base/knowledge_db directory.

2. Data Reading

1. PDF Document

We can use LangChain's PyMuPDFLoader to read the PDF file of the knowledge base. PyMuPDFLoader is one of the fastest PDF parsers, and the result will contain detailed metadata of the PDF and its pages, and return one document per page.

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

# 创建一个 PyMuPDFLoader Class 实例，输入为待加载的 pdf 文档路径
loader = PyMuPDFLoader("../../data_base/knowledge_db/pumkin_book/pumpkin_
# 调用 PyMuPDFLoader Class 的函数 load 对 pdf 文件进行加载
pdf_pages = loader.load()
```

After the document is loaded it is stored in `pages` a variable:

- `page` The variable type is `List`
- Print `pages` the length to see how many pages the PDF contains

```python
print(f"载入后的变量类型为：{type(pdf_pages)}，", f"该 PDF 一共包含 {len(pdf_p
```

```markup
载入后的变量类型为：<class 'list'>， 该 PDF 一共包含 196 页
```

`page` Each element in is a document, the variable type is `langchain_core.documents.base.Document` , the document variable type contains two attributes

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

```python
pdf_page = pdf_pages[1]
print(f"每一个元素的类型：{type(pdf_page)}.",
    f"该文档的描述性数据：{pdf_page.metadata}",
    f"查看该文档的内容:\n{pdf_page.page_content}",
    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/pumkin_book/pumpkin_
------
查看该文档的内容:
前言
“周志华老师的《机器学习》
（西瓜书）是机器学习领域的经典入门教材之一，周老师为了使尽可能多的读
```

者通过西瓜书对机器学习有所了解， 所以在书中对部分公式的推导细节没有详述，但是这对那些想深
导细节的读者来说可能“不太友好”
，本书旨在对西瓜书里比较难理解的公式加以解析，以及对部分公式补充
具体的推导细节。
”

读到这里，大家可能会疑问为啥前面这段话加了引号，因为这只是我们最初的遐想，后来我们了解到
老师之所以省去这些推导细节的真实原因是，他本尊认为“理工科数学基础扎实点的大二下学生应该
中的推导细节无困难吧，要点在书里都有了，略去的细节应能脑补或做练习”
。所以...... 本南瓜书只能算是我
等数学渣渣在自学的时候记下来的笔记，希望能够帮助大家都成为一名合格的“理工科数学基础扎实
下学生”
。

使用说明

- 南瓜书的所有内容都是以西瓜书的内容为前置知识进行表述的，所以南瓜书的最佳使用方法是以西
为主线，遇到自己推导不出来或者看不懂的公式时再来查阅南瓜书；
- 对于初学机器学习的小白，西瓜书第1 章和第2 章的公式强烈不建议深究，简单过一下即可，等
有点飘的时候再回来啃都来得及；
- 每个公式的解析和推导我们都力(zhi) 争(neng) 以本科数学基础的视角进行讲解，所以超纲的
我们通常都会以附录和参考文献的形式给出，感兴趣的同学可以继续沿着我们给的资料进行深入学习
- 若南瓜书里没有你想要查阅的公式，
或者你发现南瓜书哪个地方有错误，
请毫不犹豫地去我们GitHub 的
Issues（地址：https://github.com/datawhalechina/pumpkin-book/issues）进行反
提交你希望补充的公式编号或者勘误信息，我们通常会在24 小时以内给您回复，超过24 小时未回
话可以微信联系我们（微信号：at-Sm1les）
；

配套视频教程：https://www.bilibili.com/video/BV1Mh411e7VU
在线阅读地址：https://datawhalechina.github.io/pumpkin-book（仅供第1 版）
最新版PDF 获取地址：https://github.com/datawhalechina/pumpkin-book/releases

编委会

主编：Sm1les、archwalker、jbb0523
编委：juxiao、Majingmin、MrBigFan、shanry、Ye980226
封面设计：构思-Sm1les、创作-林王茂盛

致谢

特别感谢awyd234、
feijuan、
Ggmatch、
Heitao5200、
huaqing89、
LongJH、
LilRachel、
LeoLRH、
Nono17、

```
spareribs、sunchaothu、StevenLzq 在最早期的时候对南瓜书所做的贡献。
扫描下方二维码，然后回复关键词“南瓜书”
，即可加入“南瓜书读者交流群”
版权声明
本作品采用知识共享署名-非商业性使用-相同方式共享4.0 国际许可协议进行许可。
```

2. MD Documentation

We can read in a markdown document in almost exactly the same way:

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

loader = UnstructuredMarkdownLoader("../../data_base/knowledge_db/prompt_
md_pages = loader.load()
```

The object read is exactly the same as the PDF document:

```python
print(f"载入后的变量类型为：{type(md_pages)}，",  f"该 Markdown 一共包含 {len(
```

```markup
载入后的变量类型为：<class 'list'>， 该 Markdown 一共包含 1 页
```

```python
md_page = md_pages[0]
print(f"每一个元素的类型：{type(md_page)}.",
    f"该文档的描述性数据：{md_page.metadata}",
    f"查看该文档的内容:\n{md_page.page_content[0:][:200]}",
    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_engineering/
------
查看该文档的内容:
第一章 简介

欢迎来到面向开发者的提示工程部分，本部分内容基于吴恩达老师的《Prompt Engineering for
```

3. Data Cleaning

We expect the data in the knowledge base to be as orderly, high-quality, and concise as possible, so we need to delete low-quality text data that even affects understanding.
It can be seen that the PDF file read above not only adds line breaks to a sentence according to the line breaks of the original text `\n`, but also inserts a line break between the original two symbols `\n`. We can use regular expressions to match and delete them `\n`.

```python
import re
pattern = re.compile(r'[^\u4e00-\u9fff](\n)[^\u4e00-\u9fff]', re.DOTALL)
pdf_page.page_content = re.sub(pattern, lambda match: match.group(0).repl
print(pdf_page.page_content)
```

```markup
前言
“周志华老师的《机器学习》（西瓜书）是机器学习领域的经典入门教材之一，周老师为了使尽可能
者通过西瓜书对机器学习有所了解， 所以在书中对部分公式的推导细节没有详述，但是这对那些想
导细节的读者来说可能“不太友好”，本书旨在对西瓜书里比较难理解的公式加以解析，以及对部分
具体的推导细节。 ”
读到这里，大家可能会疑问为啥前面这段话加了引号，因为这只是我们最初的遐想，后来我们了解到
老师之所以省去这些推导细节的真实原因是，他本尊认为“理工科数学基础扎实点的大二下学生应该
中的推导细节无困难吧，要点在书里都有了，略去的细节应能脑补或做练习”。所以...... 本南瓜
等数学渣渣在自学的时候记下来的笔记，希望能够帮助大家都成为一名合格的“理工科数学基础扎实
下学生”。
使用说明
• 南瓜书的所有内容都是以西瓜书的内容为前置知识进行表述的，所以南瓜书的最佳使用方法是以西
为主线，遇到自己推导不出来或者看不懂的公式时再来查阅南瓜书；• 对于初学机器学习的小白，西
有点飘的时候再回来啃都来得及；• 每个公式的解析和推导我们都力(zhi) 争(neng) 以本科数学
我们通常都会以附录和参考文献的形式给出，感兴趣的同学可以继续沿着我们给的资料进行深入学习
或者你发现南瓜书哪个地方有错误，
```

```
请毫不犹豫地去我们GitHub 的
Issues (地址: https://github.com/datawhalechina/pumpkin-book/issues) 进行反
提交你希望补充的公式编号或者勘误信息，我们通常会在24 小时以内给您回复，超过24 小时未回
话可以微信联系我们 (微信号: at-Sm1les) ;
配套视频教程: https://www.bilibili.com/video/BV1Mh411e7VU
在线阅读地址: https://datawhalechina.github.io/pumpkin-book (仅供第1 版)
最新版PDF 获取地址: https://github.com/datawhalechina/pumpkin-book/releases
编委会
主编: Sm1les、archwalker、jbb0523
编委: juxiao、Majingmin、MrBigFan、shanry、Ye980226
封面设计: 构思-Sm1les、创作-林王茂盛
致谢
特别感谢awyd234、feijuan、Ggmatch、Heitao5200、huaqing89、LongJH、LilRachel、
扫描下方二维码，然后回复关键词“南瓜书”，即可加入“南瓜书读者交流群”
版权声明
本作品采用知识共享署名-非商业性使用-相同方式共享4.0 国际许可协议进行许可。
```

Further analyzing the data, we found that there are still a lot of • spaces in the data, so our simple and practical replace method can be used.

python

```python
pdf_page.page_content = pdf_page.page_content.replace('•', '')
pdf_page.page_content = pdf_page.page_content.replace(' ', '')
print(pdf_page.page_content)
```

markup

```
前言
“周志华老师的《机器学习》 (西瓜书) 是机器学习领域的经典入门教材之一，周老师为了使尽可能
者通过西瓜书对机器学习有所了解,所以在书中对部分公式的推导细节没有详述，但是这对那些想深
导细节的读者来说可能“不太友好”，本书旨在对西瓜书里比较难理解的公式加以解析，以及对部分
具体的推导细节。”
读到这里，大家可能会疑问为啥前面这段话加了引号，因为这只是我们最初的遐想，后来我们了解到
老师之所以省去这些推导细节的真实原因是，他本尊认为“理工科数学基础扎实点的大二下学生应该
中的推导细节无困难吧，要点在书里都有了，略去的细节应能脑补或做练习”。所以......本南瓜
等数学渣渣在自学的时候记下来的笔记，希望能够帮助大家都成为一名合格的“理工科数学基础扎实
下学生”。
使用说明
南瓜书的所有内容都是以西瓜书的内容为前置知识进行表述的，所以南瓜书的最佳使用方法是以西瓜
为主线，遇到自己推导不出来或者看不懂的公式时再来查阅南瓜书；对于初学机器学习的小白，西瓜
有点飘的时候再回来啃都来得及；每个公式的解析和推导我们都力(zhi)争(neng)以本科数学基础
```

```
我们通常都会以附录和参考文献的形式给出，感兴趣的同学可以继续沿着我们给的资料进行深入学习
或者你发现南瓜书哪个地方有错误，
请毫不犹豫地去我们GitHub的
Issues（地址：https://github.com/datawhalechina/pumpkin-book/issues）进行反
提交你希望补充的公式编号或者勘误信息，我们通常会在24小时以内给您回复，超过24小时未回复
话可以微信联系我们（微信号：at-Sm1les）；
配套视频教程：https://www.bilibili.com/video/BV1Mh411e7VU
在线阅读地址：https://datawhalechina.github.io/pumpkin-book（仅供第1版）
最新版PDF获取地址：https://github.com/datawhalechina/pumpkin-book/releases
编委会
主编：Sm1les、archwalker、jbb0523
编委：juxiao、Majingmin、MrBigFan、shanry、Ye980226
封面设计：构思-Sm1les、创作-林王茂盛
致谢
特别感谢awyd234、feijuan、Ggmatch、Heitao5200、huaqing89、LongJH、LilRachel、
扫描下方二维码，然后回复关键词“南瓜书”，即可加入“南瓜书读者交流群”
版权声明
本作品采用知识共享署名-非商业性使用-相同方式共享4.0国际许可协议进行许可。
```

The md file read above has a line break between each section, which we can also remove using the replace method.

python

```python
md_page.page_content = md_page.page_content.replace('\n\n', '\n')
print(md_page.page_content)
```

markup

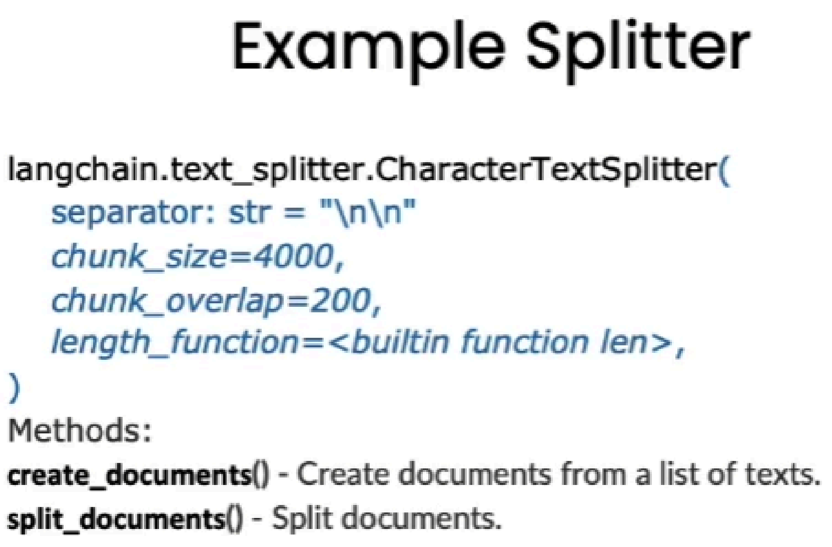
```
第一章 简介
欢迎来到面向开发者的提示工程部分，本部分内容基于吴恩达老师的《Prompt Engineering fo
网络上有许多关于提示词（Prompt， 本教程中将保留该术语）设计的材料，例如《30 prompts
在本模块，我们将与读者分享提升大语言模型应用效果的各种技巧和最佳实践。书中内容涵盖广泛，
随着 LLM 的发展，其大致可以分为两种类型，后续称为基础 LLM 和指令微调（Instruction T
与基础语言模型不同，指令微调 LLM 通过专门的训练，可以更好地理解并遵循指令。举个例子，当
因此，本课程将重点介绍针对指令微调 LLM 的最佳实践，我们也建议您将其用于大多数使用场景。
如果你将 LLM 视为一名新毕业的大学生，要求他完成这个任务，你甚至可以提前指定他们应该阅读
```

4. Document Segmentation

Since the length of a single document often exceeds the context supported by the model, the retrieved knowledge is too long to be processed by the model. Therefore, in the process of building a vector knowledge base, we often need to segment the documents, split a single document into several chunks according to length or fixed rules, and then convert each chunk into a word vector and store it in the vector database.

When searching, we will use chunk as the unit of search, that is, each time we retrieve k chunks of knowledge that the model can refer to to answer user questions. This k can be set freely.

The text segmenters in Langchain are segmented based on `chunk_size` (block size) and `chunk_overlap` (overlap size between blocks).



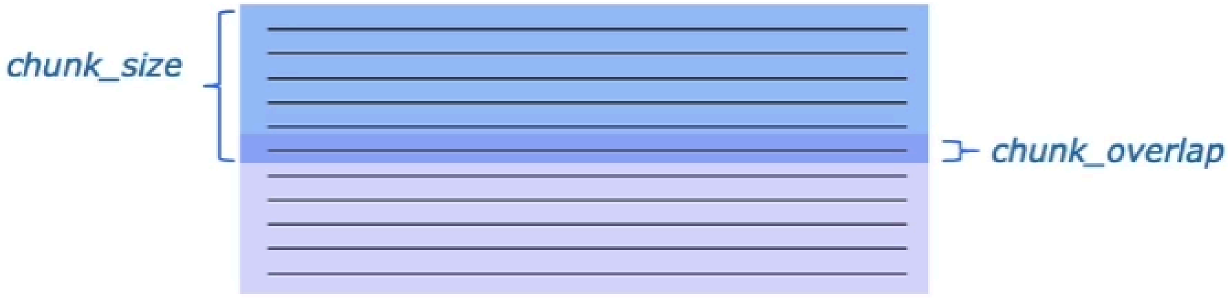
Example Splitter

```
langchain.text_splitter.CharacterTextSplitter(
    separator: str = "\n\n"
    chunk_size=4000,
    chunk_overlap=200,
    length_function=<builtin function len>,
)
```

Methods:

create_documents() - Create documents from a list of texts.

split_documents() - Split documents.



- chunk_size refers to the number of characters or tokens (such as words, sentences, etc.) contained in each chunk
- chunk_overlap refers to the number of characters shared between two chunks, which is used to maintain the coherence of the context and avoid losing context information during segmentation

Langchain provides multiple ways to segment documents, which differ in how to determine the boundaries between blocks, which characters/tokens a block consists of, and how to measure the block size.

- RecursiveCharacterTextSplitter(): Splits text by a string, recursively trying to split the text by different separators.
- CharacterTextSplitter(): Splits text by characters.
- MarkdownHeaderTextSplitter(): Splits a markdown file based on the specified header.
- TokenTextSplitter(): Splits text by token.
- SentenceTransformersTokenTextSplitter(): Split text by token
- Language(): for CPP, Python, Ruby, Markdown, etc.
- NLTKTextSplitter(): Splits text into sentences using NLTK (Natural Language Toolkit).
- SpacyTextSplitter(): Split text into sentences using Spacy.

```python
'''
* RecursiveCharacterTextSplitter 递归字符文本分割
RecursiveCharacterTextSplitter 将按不同的字符递归地分割(按照这个优先级["\n\n", '
    这样就能尽量把所有和语义相关的内容尽可能长时间地保留在同一位置
RecursiveCharacterTextSplitter需要关注的是4个参数:

* separators - 分隔符字符串数组
* chunk_size - 每个文档的字符数量限制
* chunk_overlap - 两份文档重叠区域的长度
* length_function - 长度计算函数
'''
#导入文本分割器
from langchain.text_splitter import RecursiveCharacterTextSplitter
```

```python
# 知识库中单段文本长度
CHUNK_SIZE = 500

# 知识库中相邻文本重合长度
OVERLAP_SIZE = 50
```

```python
# 使用递归字符文本分割器
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=CHUNK_SIZE,
    chunk_overlap=OVERLAP_SIZE
)
text_splitter.split_text(pdf_page.page_content[0:1000])
```

markup

```
['前言\n“周志华老师的《机器学习》（西瓜书）是机器学习领域的经典入门教材之一，周老师为了
 '有点飘的时候再回来啃都来得及；每个公式的解析和推导我们都力(zhi)争(neng)以本科数学基
 '编委会\n主编：Sm1les、archwalk']
```

python

```
split_docs = text_splitter.split_documents(pdf_pages)
print(f"切分后的文件数量：{len(split_docs)}")
```

markup

```
切分后的文件数量：720
```

python

```
print(f"切分后的字符数（可以用来大致评估 token 数）：{sum([len(doc.page_content)
```

markup

```
切分后的字符数（可以用来大致评估 token 数）：308931
```

Note: How to segment documents is actually the most important step in data processing, which often determines the lower limit of the retrieval system. However, how to choose the segmentation method is often highly business-related - for different businesses and different source data, it is often necessary to set a personalized document segmentation method. Therefore, in this chapter, we simply segment documents based on chunk_size. For readers who are interested in further exploration, please read our project examples in Part 3 to refer to how existing projects perform document segmentation.

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

‹ Previous chapter
2. Use the Embedding API

Next Chapter ›
4. Build and use vector database