

AI 大模型开发工程师 LangChain全面剖析之Retrieval

讲师：李希沅

目录

- 1 Retrieval模块的设计理念和意义
- 2 Data Connection流程
- 3 LangChain API总结

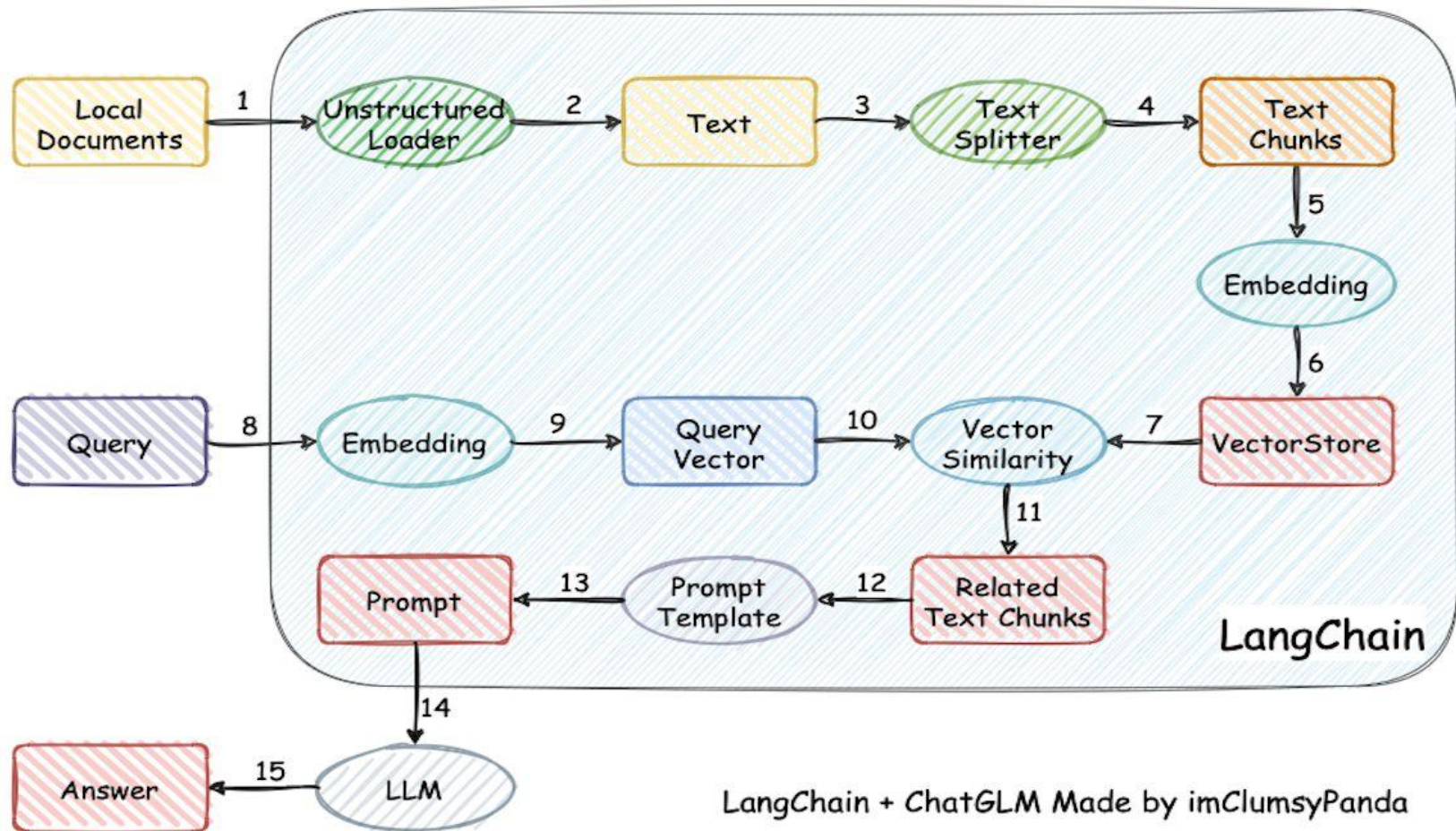
1 Retrieval模块的设计理念和意义

01、LangChain之RAG架构支持

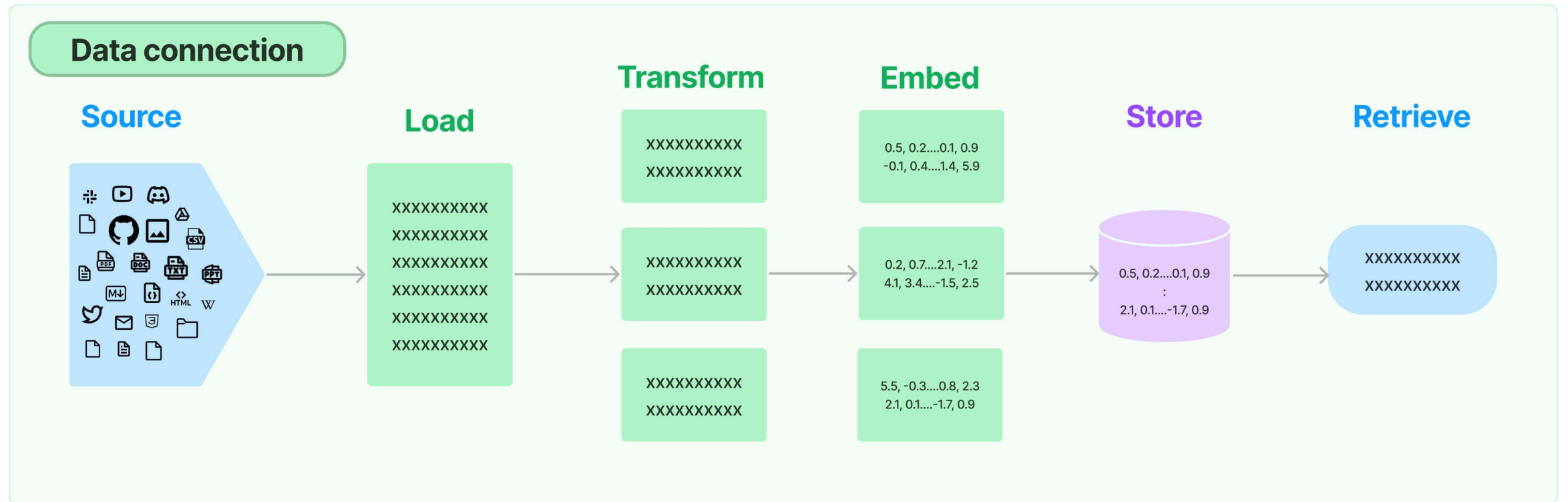
RAG: Retrieval-Augmented Generation (检索增强生成)

大模型不够聪明

Token有限制



02、Data connection



② Data Connection流程

01、Loader

Retrieval

Document loaders

Custom Document Loader

CSV

File Directory

HTML

JSON

Markdown

Microsoft Office

PDF

- Chat models
- LLMs
- Embedding models
- Document loaders
 - acream
 - AirbyteLoader
 - Airbyte CDK (Deprecated)
 - Airbyte Gong (Deprecated)
 - Airbyte Hubspot (Deprecated)
 - Airbyte JSON (Deprecated)
 - Airbyte Salesforce (Deprecated)
 - Airbyte Shopify (Deprecated)
 - Airbyte Stripe (Deprecated)
 - Airbyte Typeform (Deprecated)
 - Airbyte Zendesk Support (Deprecated)
 - Airtable
 - Alibaba Cloud MaxCompute
 - Amazon Textract
 - Apify Dataset
 - ArcGIS
 - Arxiv
 - AssemblyAI Audio Transcripts
 - AstraDB
 - Async Chromium
 - AsyncHtml
 - Athena
 - AWS S3 Directory
 - AWS S3 File
 - AZLyrics
 - Azure AI Data
 - Azure Blob Storage Container

PDF

Portable Document Format (PDF), standardized as ISO 32000, is a file format developed by Adobe in 1992 to present documents, including text formatting and images, in a manner independent of application software, hardware, and operating systems.

This covers how to load PDF documents into the Document format that we use downstream.

Using PyPDF

Load PDF using `pypdf` into array of documents, where each document contains the page content and metadata with `page` number.

```
pip install pypdf
```

```
from langchain_community.document_loaders import PyPDFLoader

loader = PyPDFLoader("example_data/layout-parser-paper.pdf")
pages = loader.load_and_split()
```

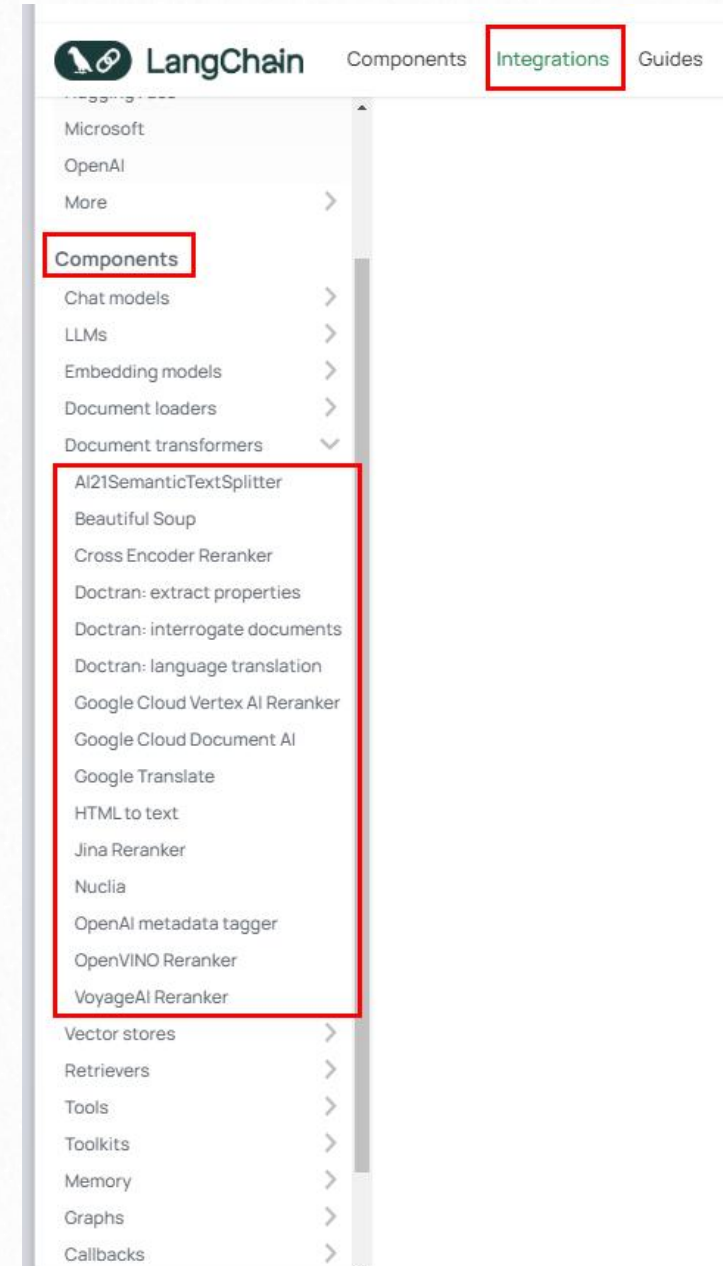
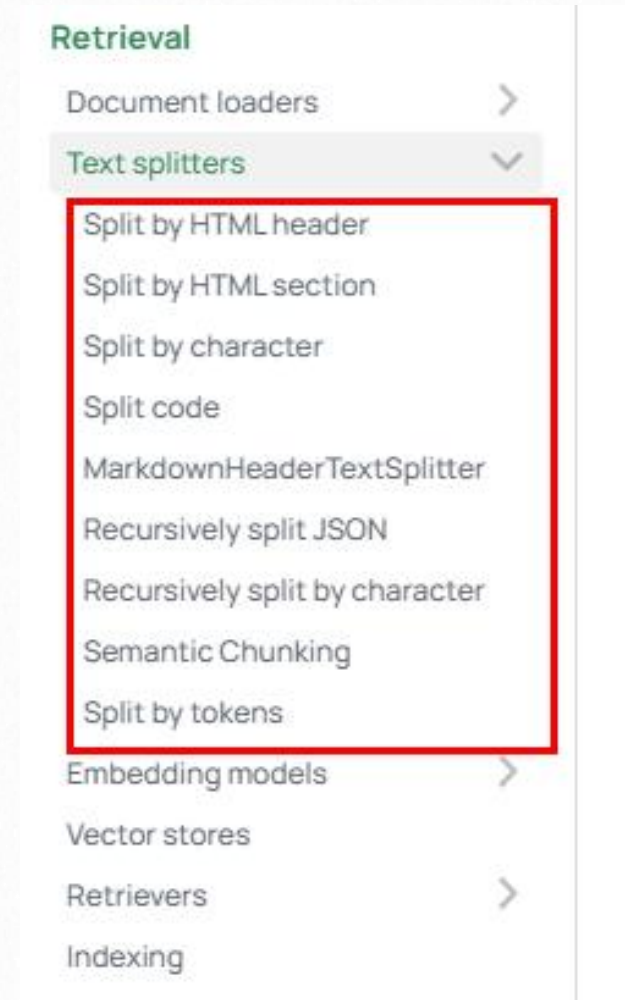
API Reference:

- PyPDFLoader

```
pages[0]
```

```
Document(page_content='LayoutParser : A Uni\x0ced Toolkit for Deep\nLearning Based Document Image Analysis\nZ
```

02、 transformers-Text Splitters



03、transformers-可视化切割效果

Text Splitter Playground

Split a text into chunks using a Text Splitter. Parameters include:

- chunk_size: Max size of the resulting chunks (in either characters or tokens, as selected)
- chunk_overlap: Overlap between the resulting chunks (in either characters or tokens, as selected)
- length_function: How to measure lengths of chunks, examples are included for either characters or tokens
- The type of the text splitter, this largely controls the separators used to split on

Chunk Size

Chunk Overlap

Length Function

Select a Text Splitter

1000

200

Characters

RecursiveCharacter

from langchain.text_splitter import RecursiveCharacterTextSplitter

length_function = len

The default list of split characters is [\n\n, \n, " ", ""]

Tries to split on them in order until the chunks are small enough

Keep paragraphs, sentences, words together as long as possible

splitter = RecursiveCharacterTextSplitter(

separators=["\n\n", "\n", " ", ""],

chunk_size=1000,

chunk_overlap=200,

length_function=length_function,

)

text = "foo bar"

splits = splitter.split_text(text)

Paste your text here:

Split Text

点击这个进行切分，切分结果会展现在当前页面下半部分

切分的文本示例

切割的最大长度

测量长度的方法

重叠的字符数量

具体实现的切分方法

切分方法对应的核心代码

测试的文本数据

Chunk Size

Chunk Overlap

Length Function

Select a Text Splitter

100

20

Characters

RecursiveCharacter

from langchain.text_splitter import RecursiveCharacterTextSplitter

length_function = len

The default list of split characters is [\n\n, \n, " ", ""]

Tries to split on them in order until the chunks are small enough

Keep paragraphs, sentences, words together as long as possible

splitter = RecursiveCharacterTextSplitter(

separators=["\n\n", "\n", " ", ""],

chunk_size=100,

chunk_overlap=20,

length_function=length_function,

)

text = "foo bar"

splits = splitter.split_text(text)

Paste your text here:

盼望着，盼望着，东风来了，春天的脚步近了。

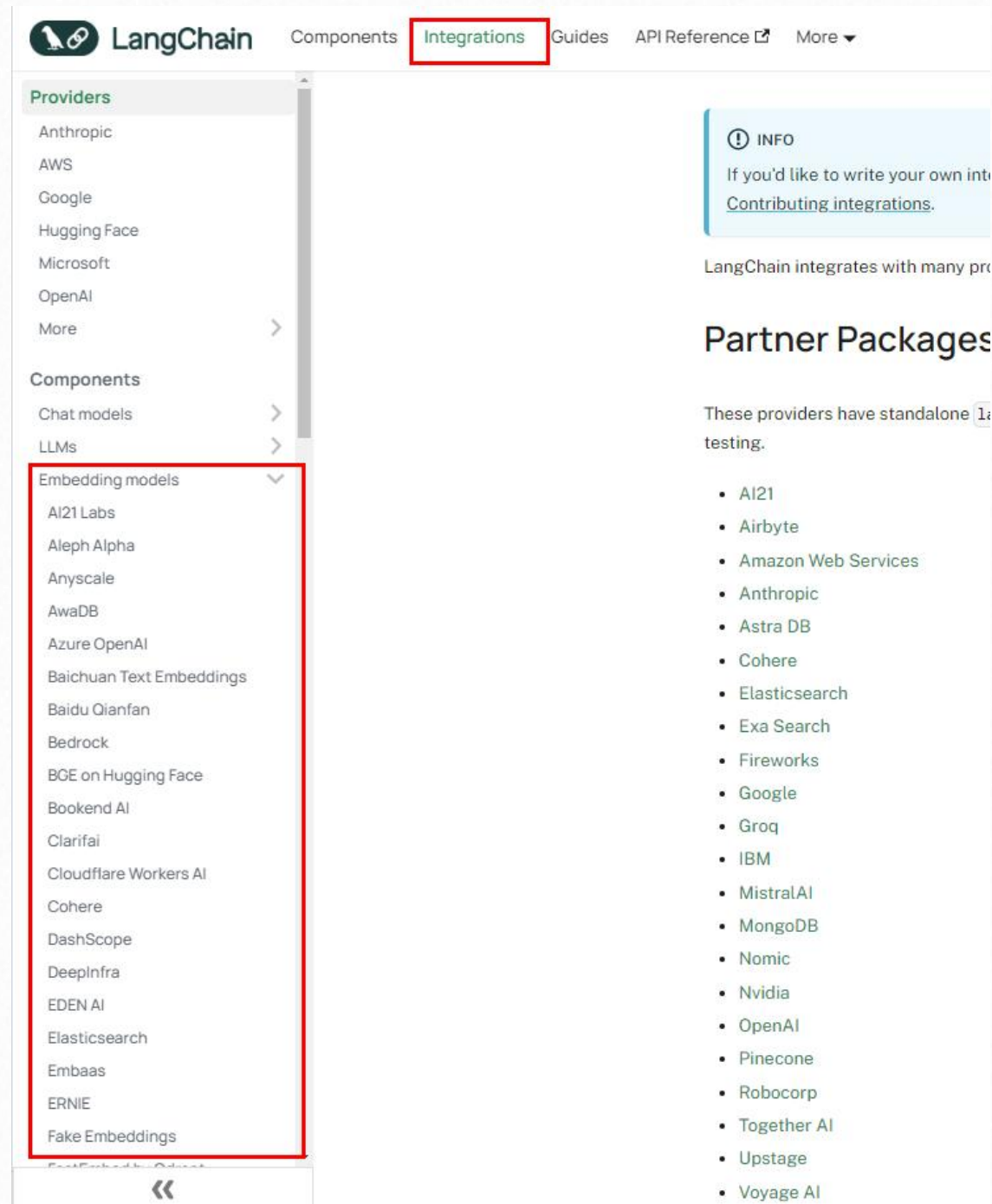
一切都象刚睡醒的样子，欣欣然张开了眼。山朗润起来了，水涨起来了，太阳的脸红起来了！

小草偷偷地从土里钻出来，嫩嫩的，绿绿的。园子里，田野里，瞧去，一大片一大片满是的。

坐着，躺着，打两个滚，踢几脚球，赛几趟跑，捉几回迷藏。风轻悄悄的，草软绵绵的。”

Split Text

04、 Embedding Models



The screenshot shows the LangChain website's 'Integrations' page. The 'Integrations' tab is highlighted in the top navigation bar. On the left, a sidebar menu lists 'Providers' and 'Components'. Under 'Components', the 'Embedding models' category is expanded and highlighted with a red box, showing a list of 25 providers. On the right, there is an 'INFO' box with a link to 'Contributing integrations', a paragraph stating 'LangChain integrates with many providers', and a section titled 'Partner Packages' which lists 20 providers in a bulleted format.

Providers

- Anthropic
- AWS
- Google
- Hugging Face
- Microsoft
- OpenAI
- More >

Components

- Chat models >
- LLMs >
- Embedding models** ▾

Embedding models

- AI21 Labs
- Aleph Alpha
- Anyscale
- AwaDB
- Azure OpenAI
- Baichuan Text Embeddings
- Baidu Qianfan
- Bedrock
- BGE on Hugging Face
- Bookend AI
- Clarifai
- Cloudflare Workers AI
- Cohere
- DashScope
- DeepInfra
- EDEN AI
- Elasticsearch
- Embaas
- ERNIE
- Fake Embeddings

INFO

If you'd like to write your own integrations, see [Contributing integrations](#).

LangChain integrates with many providers.

Partner Packages

These providers have standalone packages for testing.

- AI21
- Airbyte
- Amazon Web Services
- Anthropic
- Astra DB
- Cohere
- Elasticsearch
- Exa Search
- Fireworks
- Google
- Groq
- IBM
- MistralAI
- MongoDB
- Nomic
- Nvidia
- OpenAI
- Pinecone
- Robocorp
- Together AI
- Upstage
- Voyage AI

05、 Vector Stores

The screenshot shows the LangChain website's 'Integrations' page. The top navigation bar includes 'Components' (highlighted with a red box), 'Integrations' (highlighted with a red box), 'Guides', 'API Reference', and 'More'. The left sidebar menu lists various categories: 'Microsoft', 'OpenAI', 'More', 'Components' (highlighted with a red box), 'Chat models', 'LLMs', 'Embedding models', 'Document loaders', 'Document transformers', and 'Vector stores'. Under 'Vector stores', a list of providers is shown, with the entire list highlighted by a red box: ActiveLoop Deep Lake, Alibaba Cloud OpenSearch, AnalyticDB, Annoy, Apache Doris, Astra DB, Atlas, AwaDB, Azure Cosmos DB, Azure AI Search, Bagel, Baidu Cloud ElasticSearch VectorSearch, Baidu VectorDB, Apache Cassandra, Chroma, Clarifai, ClickHouse, Couchbase, DashVector, Databricks Vector Search, DingoDB, and DocArray HnswSearch. The main content area on the right features an 'INFO' box with a link to 'Contributing integrations', a statement 'LangChain integrates with many', and a 'Partner Package' section. This section lists various providers: AI21, Airbyte, Amazon Web Services, Anthropic, Astra DB, Cohere, Elasticsearch, Exa Search, Fireworks, Google, Groq, IBM, MistralAI, MongoDB, Nomic, Nvidia, OpenAI, Pinecone, Robocorp, Together AI, and Upstage.

LangChain

Components Integrations Guides API Reference More

Microsoft
OpenAI
More

Components

Chat models
LLMs
Embedding models
Document loaders
Document transformers
Vector stores

ActiveLoop Deep Lake
Alibaba Cloud OpenSearch
AnalyticDB
Annoy
Apache Doris
Astra DB
Atlas
AwaDB
Azure Cosmos DB
Azure AI Search
Bagel
Baidu Cloud ElasticSearch VectorSearch
Baidu VectorDB
Apache Cassandra
Chroma
Clarifai
ClickHouse
Couchbase
DashVector
Databricks Vector Search
DingoDB
DocArray HnswSearch

INFO

If you'd like to write your own [Contributing integrations](#).


LangChain integrates with many

Partner Package

These providers have standalone testing.

- AI21
- Airbyte
- Amazon Web Services
- Anthropic
- Astra DB
- Cohere
- Elasticsearch
- Exa Search
- Fireworks
- Google
- Groq
- IBM
- MistralAI
- MongoDB
- Nomic
- Nvidia
- OpenAI
- Pinecone
- Robocorp
- Together AI
- Upstage

06、 Retrievers

 LangChain

Components

Integrations

Guides

API Reference

More

Providers

Anthropic

AWS

Google

Hugging Face

Microsoft

OpenAI

More

Components

Chat models

LLMs

Embedding models

Document loaders

Document transformers

Vector stores

Retrievers

Activerloop Deep Memory

Amazon Kendra

Arcee

Arxiv

Azure AI Search

Bedrock (Knowledge Bases)

BM25

BREEBS (Open Knowledge)

Chaindesk

ChatGPT plugin

Cohere reranker

Cohere RAG

DocArray

Dria

ElasticSearch BM25

Elasticsearch

! INFO

If you'd like to write your own [Contributing integrations.](#)

LangChain integrates with many

Partner Packages




These providers have standard or testing.

- AI21
- Airbyte
- Amazon Web Services
- Anthropic
- Astra DB
- Cohere
- Elasticsearch
- Exa Search
- Fireworks
- Google
- Groq
- IBM
- MistralAI
- MongoDB
- Nomic
- Nvidia
- OpenAI
- Pinecone
- Robocorp
- Together AI
- Unstage

③ LangChain API总结

01、LangChain总结

Ecosystem

-  [LangSmith](#): Tracing and evaluating your language model applications and intelligent agents to help you move from prototype to production.
-  [LangGraph](#): Creating stateful, multi-actor applications with LLMs, built on top of (and intended to be used with) LangChain primitives.
-  [LangServe](#): Deploying LangChain runnables and chains as REST APIs.
 - [LangChain Templates](#): Example applications hosted with LangServe.



THANK YOU