AI 大模型开发工程师
之基于私有模型的知识库项目改造

讲师：李希沅

目录

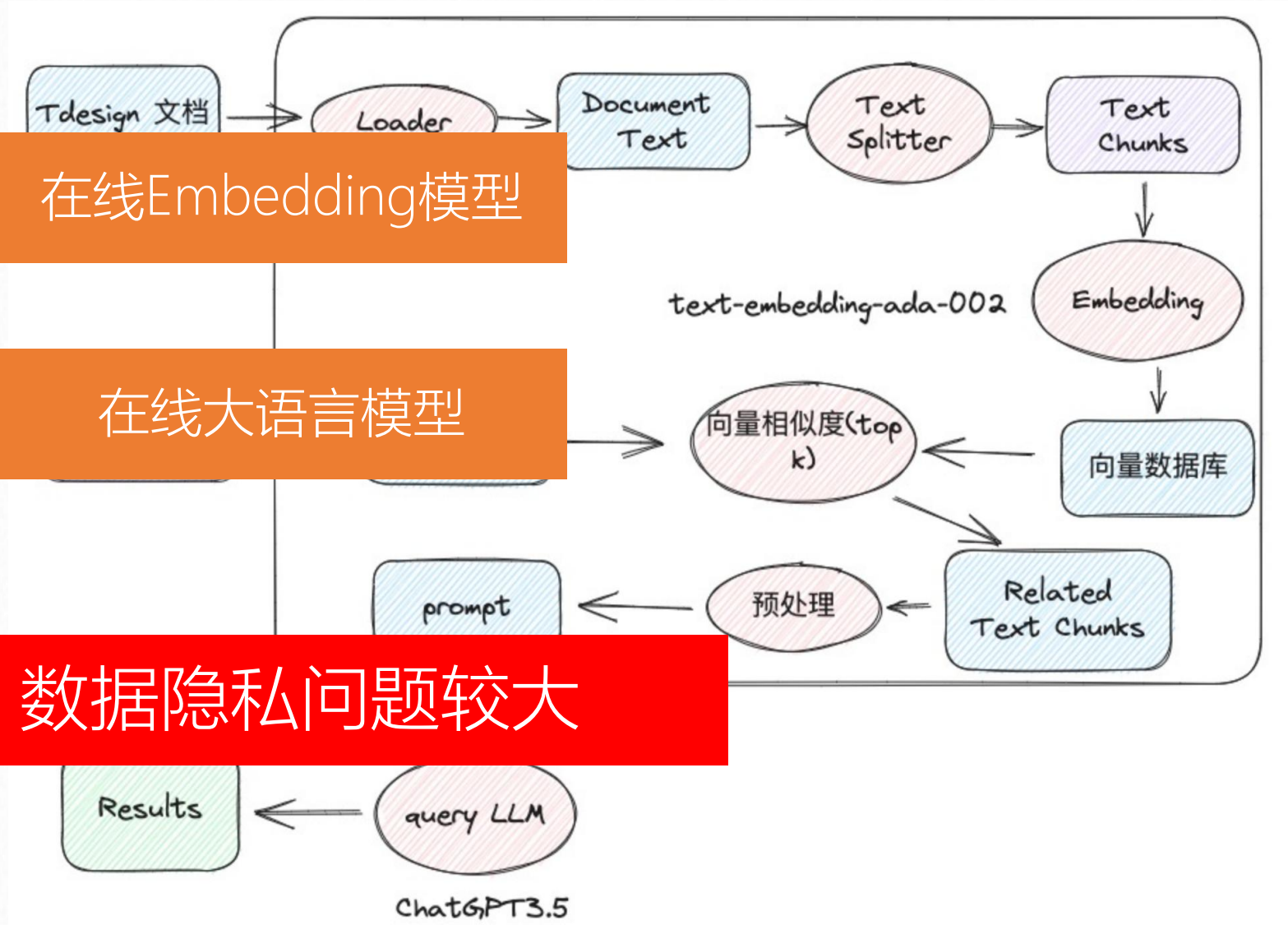
1 基于私有模型的本地知识库架构改造

01、CVP架构模式回顾

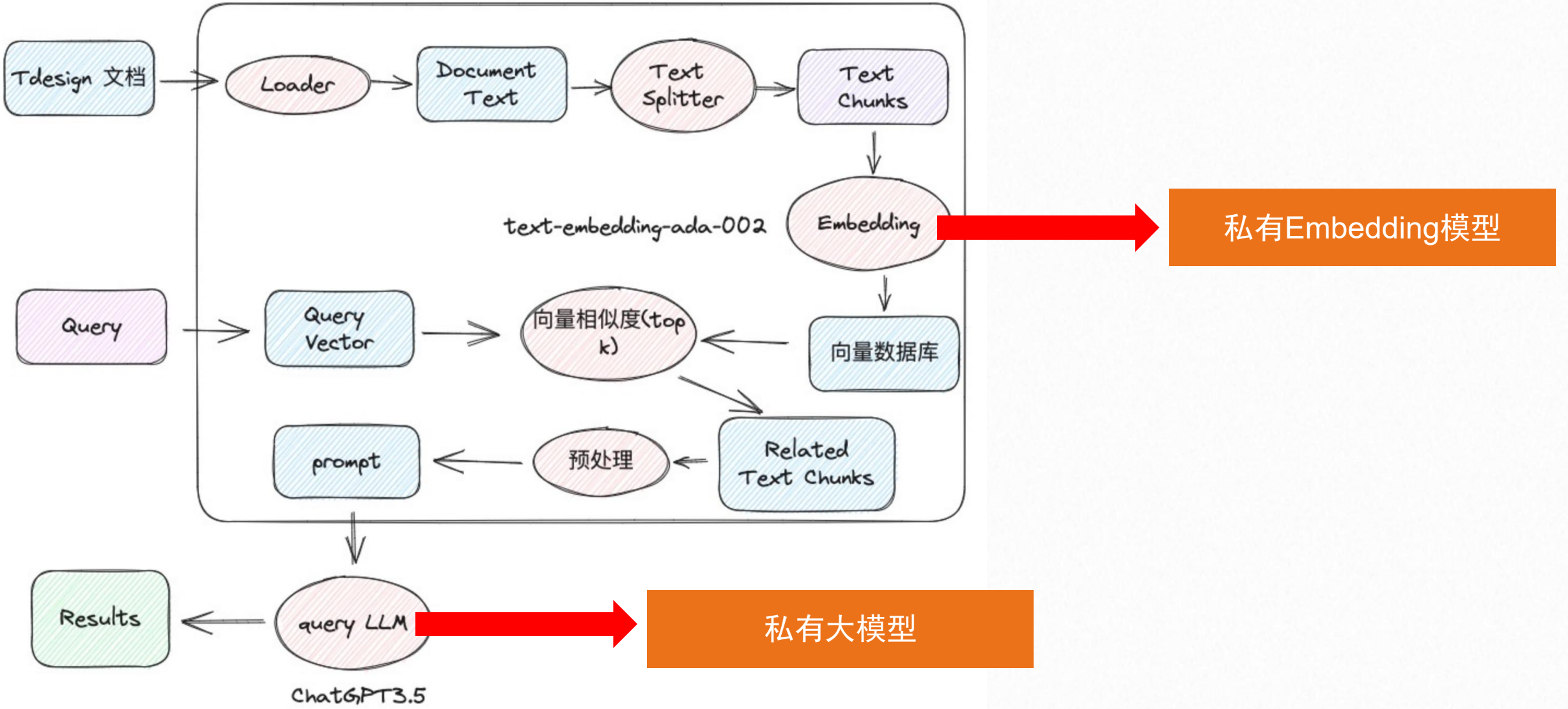
1、知识数据向量化

2、知识数据召回

3、查询返回结果



02、RAG（retrieval augmented generation）



03、ChatGLM3-6B模型

ChatGLM3 是智谱AI和清华大学 KEG 实验室联合发布的新一代对话预训练模型。ChatGLM3-6B 是 ChatGLM3 系列中的开源模型，在保留了前两代模型对话流畅、部署门槛低等众多优秀特性的基础上，ChatGLM3-6B 引入了如下特性：

1. **更强大的基础模型：** ChatGLM3-6B 的基础模型 ChatGLM3-6B-Base 采用了更多样的训练数据、更充分的训练步数和更合理的训练策略。在语义、数学、推理、代码、知识等不同角度的数据集上测评显示，**ChatGLM3-6B-Base 具有在 10B 以下的基础模型中最强的性能。**
2. **更完整的功能支持：** ChatGLM3-6B 采用了全新设计的 Prompt 格式，除正常的多轮对话外。同时原生支持工具调用（Function Call）、代码执行（Code Interpreter）和 Agent 任务等复杂场景。
3. **更全面的开源序列：** 除了对话模型 ChatGLM3-6B 外，还开源了基础模型 ChatGLM3-6B-Base、长文本对话模型 ChatGLM3-6B-32K。以上所有权重对学术研究**完全开放**，在填写问卷进行登记后**亦允许免费商业使用。**

开源的

高性能的

可商业使用的

低成本可部署的

支持中文的

04、Embedding模型选择

MTEB排行榜：https://huggingface.co/spaces/mteb/leaderboard

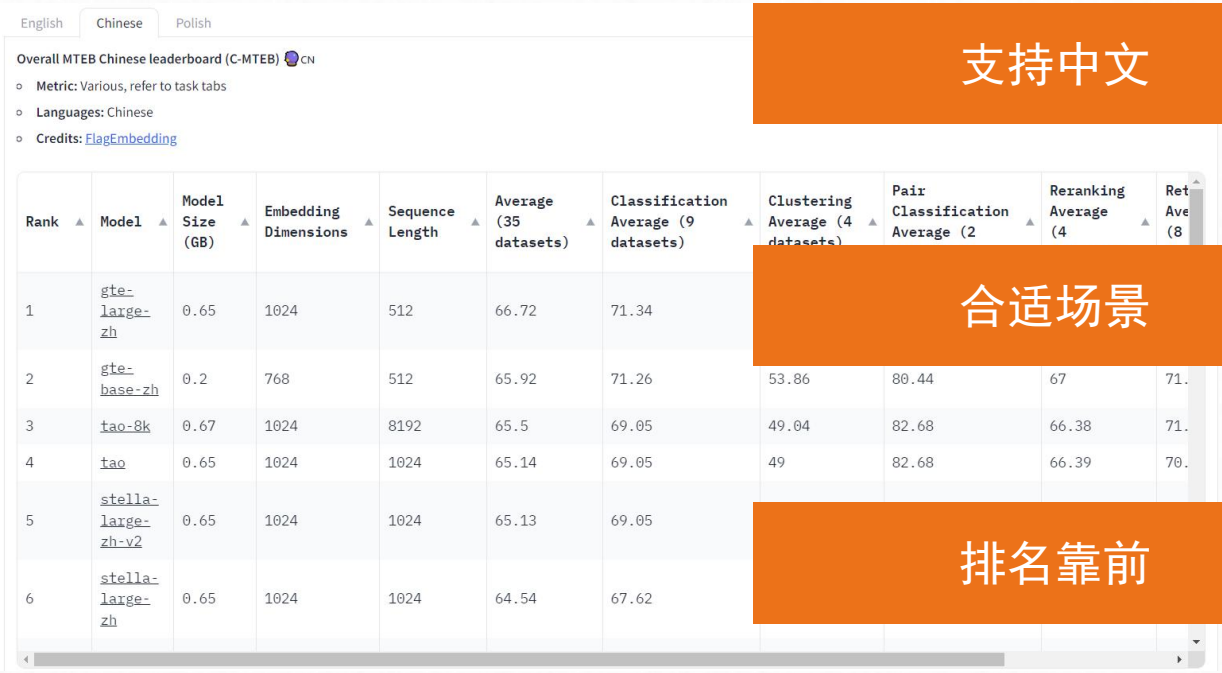
MTEB 是衡量文本嵌入模型在各种嵌入任务上性能的重要基准

开源的

支持中文

合适场景

排名靠前



Rank	Model	Model Size (GB)	Embedding Dimensions	Sequence Length	Average (35 datasets)	Classification Average (9 datasets)	Clustering Average (4 datasets)	Pair Classification Average (2	Reranking Average (4	Ret Ave (8
1	gte-large-zh	0.65	1024	512	66.72	71.34				
2	gte-base-zh	0.2	768	512	65.92	71.26	53.86	80.44	67	71.
3	tao-8k	0.67	1024	8192	65.5	69.05	49.04	82.68	66.38	71.
4	tao	0.65	1024	1024	65.14	69.05	49	82.68	66.39	70.
5	stella-large-zh-v2	0.65	1024	1024	65.13	69.05				
6	stella-large-zh	0.65	1024	1024	64.54	67.62				

- shibing624/text2vec-base-chinese模型，是用CoSENT方法训练，基于hfl/chinese-macbert-base在中文STS-B数据训练得到，并在中文STS-B测试集评估达到较好效果，运行examples/training_sup_text_matching_model.py代码可训练模型，模型文件已经上传HF model hub，中文通用语义匹配任务推荐使用

```
# 加载embedding
embedding_model_dict = {
    "thenlper-base": "thenlper/gte-base-zh",
    "ernie-base": "nghuyong/ernie-3.0-base-zh",
    "text2vec": "GanymedeNil/text2vec-large-chinese",
    "text2vec2": "uer/sbert-base-chinese-nli",
    "text2vec3": "shibing624/text2vec-base-chinese",
}
```

05、改造后的技术选型

类型	对应技术
LLM模型	ChatGLM3-6B
Embedding模型	text2vec-base-chinese或者更多
LLM应用开发框架	LangChain
向量数据库	FAISS/pinecone/Milvus
前端框架	streamlit/gradio

❷ 基于私有模型的本地知识库资源准备

01、资源评估

推理的GPU资源要求

简单测试样例的实际测试数据

量化等级	生成 8192 长度的最小显存
FP16	15.9 GB
INT8	11.1 GB
INT4	8.5 GB

镜像	PyTorch 2.0.0 Python 3.8(ubuntu20.04) Cuda 11.8 更换
GPU	RTX 4090(24GB) * 1 升降配置
CPU	12 vCPU Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz
内存	90GB
硬盘	系统盘: 30 GB

1. NVIDIA Pascal架构的GPU，如TitanXp，GTX 10系列等。 这类GPU缺乏低精度的硬件加速能力，但却具备中等的单精度算力。由于价格便宜，适合用来练习训练小模型(如Cifar10)或调试模型代码。
2. NVIDIA Volta/Turing架构的GPU，如GTX 20系列, Tesla V100等。 这类GPU搭载专为低精度(int8/float16)计算加速的TensorCore, 但单精度算力相较于上代提升不大。我们建议在实例上启用深度学习框架的混合精度训练来加速模型计算。 相较于单精度训练，混合精度训练通常能够提供2倍以上的训练加速。
3. NVIDIA Ampere架构的GPU，如GTX 30系列，Tesla A40/A100等。 这类GPU搭载第三代TensorCore。相较于前一代，支持了TensorFloat32格式，可直接加速单精度训练 (PyTorch已默认开启)。但我们仍建议使用超高算力的float16半精度训练模型，可获得比上一代GPU更显著的性能提升。
4. 寒武纪 MLU 200系列加速卡。 暂不支持模型训练。使用该系列加速卡进行模型推理需要量化为int8进行计算。 并且需要安装适配寒武纪MLU的深度学习框架。
5. 华为 Ascend 系列加速卡。 支持模型训练及推理。但需安装MindSpore框架进行计算。

每块GPU应配备至少4~8核心的CPU

02、私有模型部署

获取工程

```
root@autodl-container-dd9f46bdad-dd54918f:~/glm# git clone https://github.com/THUDM/ChatGLM3
Cloning into 'ChatGLM3'...
remote: Enumerating objects: 469, done.
remote: Counting objects: 100% (234/234), done.
remote: Compressing objects: 100% (107/107), done.
remote: Total 469 (delta 154), reused 167 (delta 122), pack-reused 235
Receiving objects: 100% (469/469), 15.19 MiB | 12.18 MiB/s, done.
Resolving deltas: 100% (242/242), done.
root@autodl-container-dd9f46bdad-dd54918f:~/glm# cd ChatGLM3
root@autodl-container-dd9f46bdad-dd54918f:~/glm/ChatGLM3#
```

安装依赖

```
root@autodl-container-dd9f46bdad-dd54918f:~/glm/ChatGLM3# pip install -r requirements.txt
Looking in indexes: http://mirrors.aliyun.com/pypi/simple
Requirement already satisfied: protobuf in /root/miniconda3/lib/python3.8/site-packages (from -r requirement
Collecting transformers>=4.30.2
  Downloading http://mirrors.aliyun.com/pypi/packages/12/dd/f17b11a93a9ca27728e12512d167eb1281c151c4c6881d3
     | 2.3 MB 1.4 MB/s eta 0:00:04
```

私有模型测试

streamlit run web_demo2.py

03、个性化需求

个性化需求1: 学术资源加速

source /etc/network_turbo

个性化需求2：本地访问服务器端口

使用SSH将实例中的端口代理到本地，具体步骤为：

Step.1 在实例中启动您的服务（比如您的服务监听6006端口，下面以6006端口为例）

Step.2 在本地电脑的终端(cmd / powershell / terminal等)中执行代理命令：

ssh -CNg -L 6006:127.0.0.1:6006 root@123.125.240.150 -p 42151

其中root@123.125.240.150和42151分别是实例中SSH指令的访问地址与端口，请找到自己实例的ssh指令做相应替换。

6006:127.0.0.1:6006是指代理实例内6006端口到本地的6006端口。

（注：仅限于跟我一样用的AutoDL平台的场景）

个性化需求3: 修改streamlit的端口号为：6006

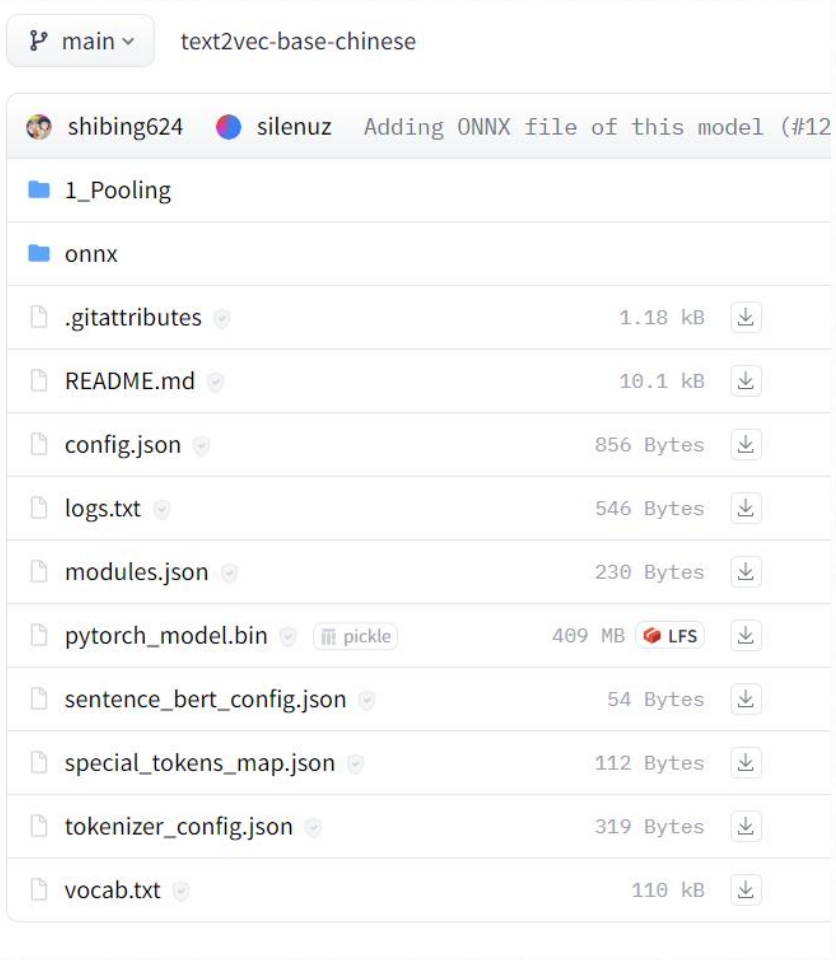
find / -name config.py

进入文件：/root/miniconda3/lib/python3.8/site-packages/streamlit/config.py

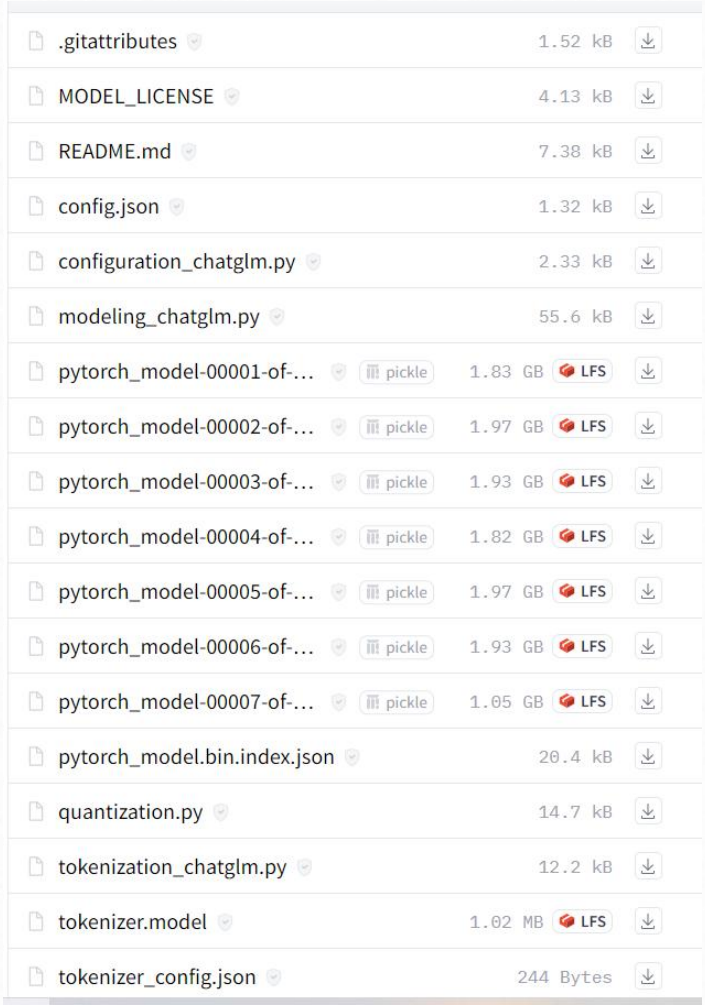
```
_create_option(
    "server.port",
    description="""
        The port where the server will listen for browser connections.""",
    default_val=6006,
    type_=int,
)
```

04、建议下载私有模型到本地

https://huggingface.co/shibing624/text2vec-base-chinese



https://huggingface.co/THUDM/chatglm3-6b



3 基于私有模型的本地知识库代码落地

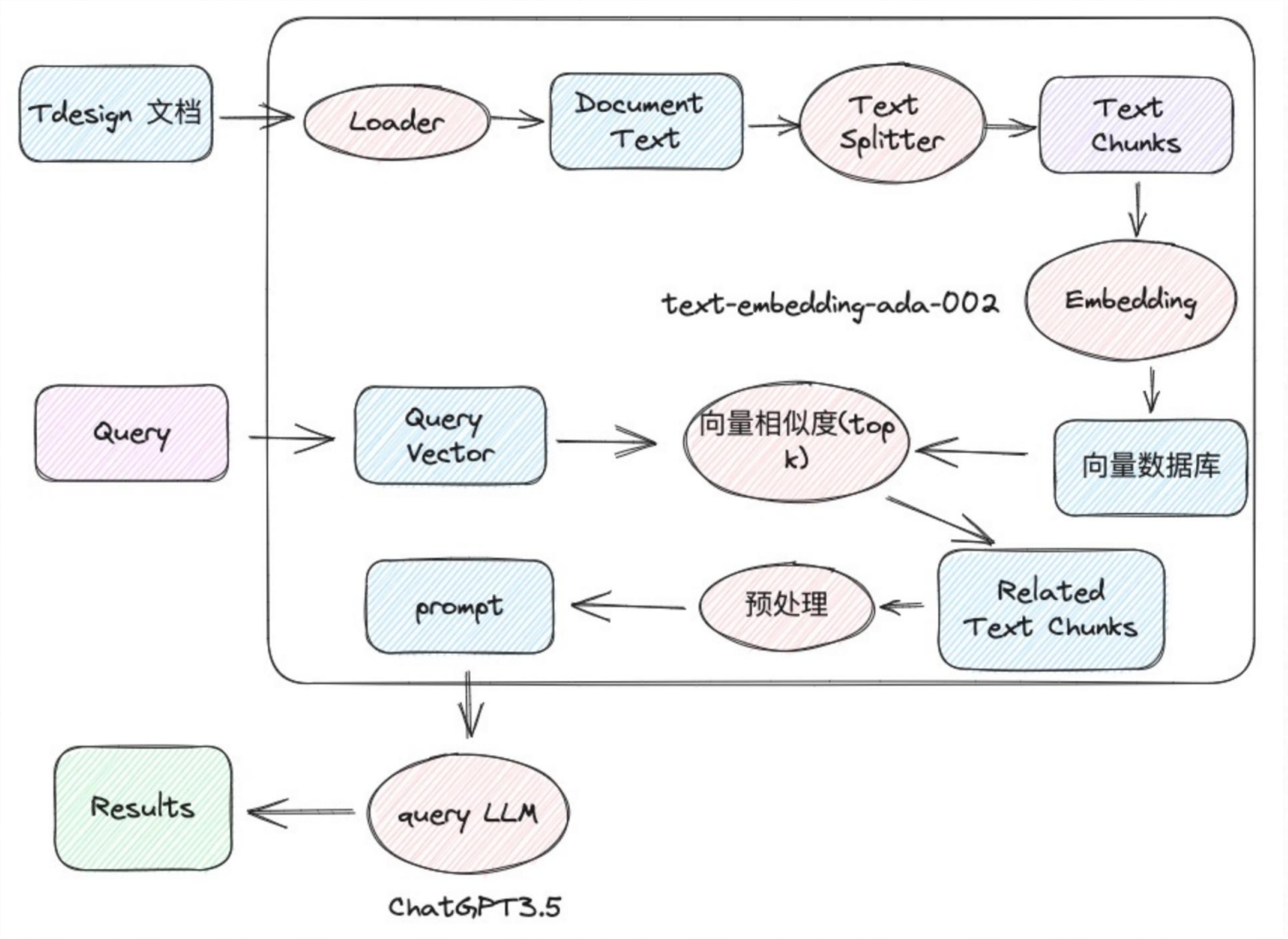
01、代码演示

Embedding模型

LLM模型

4 基于私有模型的本地知识库项目总结

01、项目总结



加入微调

Prompt方向优化

加入代码解释器

加入COT问题拆解步骤

加入模型审查流程

谢谢观看