

智能搜索问答方案（V3 版本）部署指引

一、部署文件上传：

- a. 海外区域：打开 cloud9，上传部署文件到 cloud9，并解压部署文件
- b. 国内区域：新建 EC2 后，上传部署文件到 EC2，并解压部署文件，执行以下脚本，安装 npm 和 node：
 - `curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.3/install.sh | bash`
 - `nvm install node`

二、通过部署 cdk 部署 lambda、opensearch(或 Kendra)和 Notebook 服务

1、修改部署配置，打开 deployment/cdk.json 文件

1. 如果需要部署 Kendra，修改 `search_engine_kendra` 为 `true`，`search_engine_opensearch` 为 `false`。
2. 如果只部署智能问答功能，`selection` 参数仅保留 `"langchain_processor_qa"` 选项，其余选项去除。

2、 打开 Terminal，进入 deployment 目录，依次运行以下命令部署

1. `pip install -r requirements.txt`
2. `export AWS_ACCOUNT_ID=替换实际参数`
3. `export AWS_REGION=替换实际参数`

以下两项需要在 EC2 环境执行，cloud9 环境跳过：

- (1)`export AWS_ACCESS_KEY_ID=替换实际参数`
- (2)`export AWS_SECRET_ACCESS_KEY=替换实际参数`

4. `cdk bootstrap`
5. `cdk deploy --all`

三、部署向量模型和开源大语言模型

中文：

1. 在 Amazon Sagemaker Notebook, 找到 SmartSearchNotebook
2. 通过 `Embedding_Model/chinese_bge_large_zh/chinese_bge_large_zh_deploy.ipynb` 部署中文向量模型
3. 部署大语言模型：
进入 `LLM_Model/llm-baichuan`, 运行 `main.ipynb`

英文：

1. 在 SageMaker->Notebook->Notebook instances, 找到 SmartSearchNotebook, 点击 Open Jupyter

2. 部署向量模型：运行

Embedding_Model/chinese_bge_large_en/english_bge_large_zh_deploy.ipynb

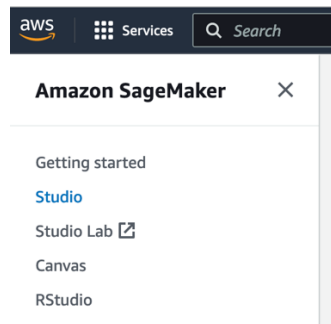
3. 部署大语言模型

vicuna1.5:

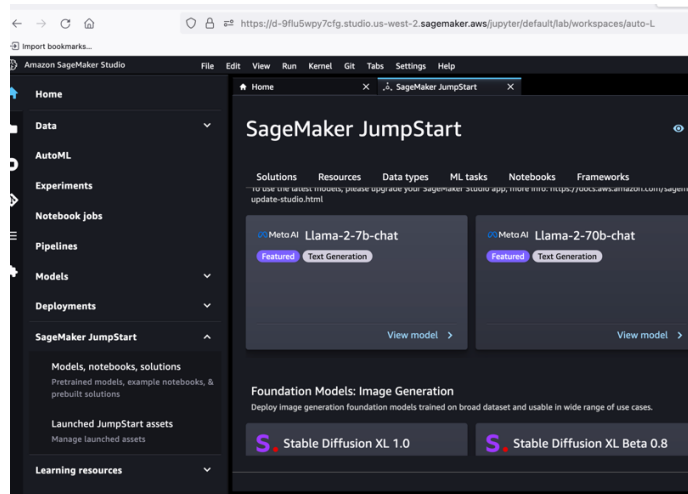
- (1) 打开 LLM_Model/llm-english/code/inference.py, 把 37 行的 LLM_NAME 的值改为 "lmsys/vicuna-7b-v1.5"
- (2) 运行 LLM_Model/llm-english/llm_english_sagemaker_byos.ipynb 部署模型

llama2(可选):

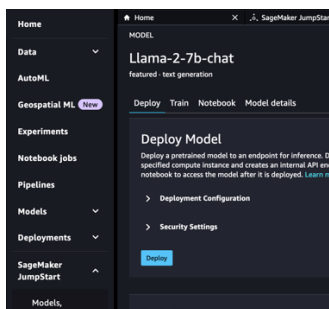
- (1) 在 Sagemaker 页面打开 Studio



- (2) 进入 Studio 后, 打开 SageMaker JumpStart, 点击 "Llama-2-7b-chat"



- (3) 点击 Deploy 部署



四、数据导入

1. 在 S3 查找 intelligent-search-data-bucket-ACCOUNT-REGION 分桶，新建 source_data 目录，并上传文件。可以在 cloudWatch 的 Log groups 查找 langchain_processor_dataupload，查看文件导入情况。成功导入会输出文件导入时间和文件名

```
File import takes time: 0:00:27.171235
```

```
Complete the import of the following documents: ['tmp/基于预训练语言模型的行业搜索的应用和研究.docx']
```

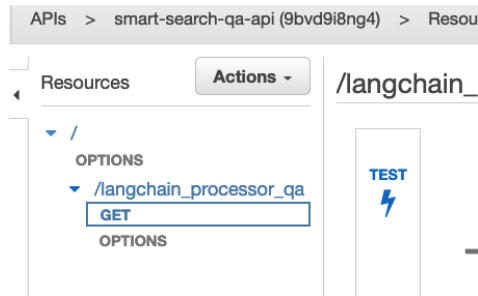
2. 在 Notebook 文件导入：

S3 数据导入不能更改数据导入的 index 和 language 等参数，建议使用 Notebook 导入，相比 S3 数据导入更灵活。

 - (1) 在 SageMaker->Notebook->Notebook instances，找到 SmartSearchNotebook，点击 Open Jupyter
 - (2) 打开 data_load/data_load.ipynb，在第 3 个代码框中，修改 index、language 和 EMBEDDING_ENDPOINT_NAME 参数。其中 index 为导入数据的 index 名称，可以自定义，language 为导入文件的语言，EMBEDDING_ENDPOINT_NAME 为上面部署的向量模型的 endpoint name，在 Sagemaker ->Inference -> Endpoints 页面可以找到
 - (3) 在 isearchjupyter 文件夹下新建一个文件夹，上传需要导入的文件
 - (4) 在第 3 个代码框中，修改 local_file 参数，替换为需要导入的文件路径。依次运行代码框以导入文件。

五、数据验证

1. API 测试，进入 API Gateway，点击 smart-search-qa-api，在 Resources 下点击 GET，再点击旁边的 TEST，进入测试页面



2. 在 Query Strings 下输入查询文本 query=你好&task=chat:

Query Strings

{langchain_processor_qa}

query=你好&task=chat

3. 稍等约 10 多秒后，右边输出 Status:200 表示运行成功

Status: 200

4. 点击左边栏的 Stages -> Prod，找到右边的 Invoke URL 并拷贝

Invoke URL: <https://9bvd9i8ng4.execute-api.us-west-2.amazonaws.com/prod>

5. 进入 gradio-web 目录，点击 gradio-webpage.py，将 Invoke URL 替换 invoke_url 变量。同时配置 index、embedding_endpoint、llm_endpoint 参数，如果只部署了一个语种的 embedding model 和 llm，仅需配置相应的参数。
6. 安装 gradio: pip install -r requirements.txt
7. 运行 gradio-webpage.py，运行后生成一个本地 URL 地址和公共 URL 地址，公共 URL 地址能分享给其他人测试

```
(/home/ec2-user/environment/web_test/python3.9_env) Admin:~/environment/web_test $ python gradio_test.py
Running on local URL: http://127.0.0.1:7860
Running on public URL: https://4d23740206bfdb0c0d.gradio.live
```

测试页面的有效时间是 72 小时，页面过期后需重新生成

8. 在本地浏览器打开公共 URL 地址后测试

AWS Intelligent Q&A Solution Guide

Question Answering

Summarize

Task

☒ Knowledge base Q&A ☐ Chat

Query

What is Amazon Elastic Transcoder?

Session ID

Summit

Language

☐ chinese ☐ chinese-tc ☒ english

Model type

☐ claude2 ☒ titan(english) ☐ llama2(english) ☐ other

Prompt(must include {context} and {question})

基于以下已知信息，简洁和专业的来回答用户的问题，并告知是依据哪些信息来进行回答的。
如果无法从中得到答案，请说“根据已知信息无法回答该问题”或“没有提供足够的相关信息”，不允许在答案中添加编造成分。答案请使用中文。

问题: {question}
=====
{context}
=====
答案:

Search engine

☒ OpenSearch ☐ Kendra

OpenSearch index OR Kendra index id

Top_k of source text to LLM

Temperature parameter of LLM

Confidence score type

☒ query_answer_score ☐ answer_docs_score ☐ docs_list_overlap_score

Answer

Amazon Elastic Transcoder is a highly scalable, easy to use and cost effective way for developers and businesses to convert (or "transcode") video and audio files from their source format into versions that will playback on devices like smartphones, tablets and PCs.

Confidence

query_docs_score:0.018686099
query_answer_score:0.9793152440873676

Source

num:0 source:aws_fa... score:0.018686099
paragraph:Question: Q: What is Amazon Elastic Transcoder?Answer: Amazon Elastic Transcoder is a highly scalable, easy to use and cost effective way for developers and businesses to convert (or "transcode") video and audio files from their source format into versions that will playback on devices like smartphones, tablets and PCs.

num:1 source:aws_fa... score:0.014087653
paragraph:Question: Q: What can I do with Amazon Elastic Transcoder?Answer: You can use Amazon Elastic Transcoder to convert video and audio files into supported output formats optimized for playback on desktops, mobile devices, tablets, and televisions. In addition to supporting a wide range of input and output formats, resolutions, bitrates, and frame rates, Amazon Elastic Transcoder also offers features for automatic video bit rate optimization, generation of thumbnails, overlay of visual watermarks, caption support, DRM packaging, progressive downloads, encryption and more. For more details, please visit the Product Details page.

Url

https://pocflul6r4.execute-api.us-west-1.amazonaws.com/prod/langchain_processor_qa?query=What is Amazon Elastic Transcoder?&task=qa&language=english&embedding_endpoint_name=huggingface-inference-eb-bge-en&lm_embedding_name=model_type=bedrock&bedrock_api_url=https://bx2kc13y33.execute-api.us-east-1.amazonaws.com/prod/bedrock7&bedrock_model_id=amazon.titan-tg1-large&search_engine=opensearch&index=smart_search_qa_test_0826_en_3&top_k=2&cal_query_answer_score=true

Request time

0:00:05.117202

页面左边为参数输入部分，除 Query 外，其余参数可以不用输入，使用默认值。

其余参数说明如下：

1.Task:

(1)Knowledge base Q&A: 基于知识库的问答

(2)Chat: 聊天

2.Session ID

输入 Session ID 后，对话的问题和答案将会被记录下来，用于实现多轮对话功能。如要使用基于知识库的多轮对话，请输入一个随机的字符串作为 Session ID，如果要结束本轮对话，去除当前 Session ID 或换另外字符串作为新的 Session ID 即可。

3.Language

(1)chinese --- 简体中文

(2)chinese-tc --- 繁体中文

(3)english --- 英语

4.Model type

- (1)claude2: 目前支持调用 Bedrock 的 claude2 模型, 该模型同时支持中文和英文
- (2)Titan: 目前支持调用 Bedrock 的 Titan 模型,该模型目前只支持英文
- (3)llama2: 目前支持通过 jumpstart 部署的了 llama2 模型, 该模型建议只在英文场景使用
- (4)other: chatglm2 等中文大语言模型

5.Prompt

prompt 是提示词, 方案有使用默认提示词, 如需对提示词修改调优, 中文需要保留{question}和{context}字段, 修改其他字段。英文不需要{question}和{context}字段。

6.Search engine

方案支持 2 种搜索引擎, 分别为 OpenSearch 和 Kendra.

7.OpenSearch index OR Kendra index id

如果 Search engine 选择为 OpenSearch, 该参数为 OpenSearch 的 index。如果 Search engine 选择为 Kendra, 该参数为 Kendra 的 index id。

8.Top_k of source text to LLM

查找与问题相关的文档数量, 查找到的相关文档将组合成 Prompt 后送到大模型推理。

9.Temperature

大语言模型的 temperature 参数

10.Confidence score type

计算答案置信度分数类型:

query_answer_scoer: 计算问题与答案的相似度

answer_docs_score:计算答案与相关文档的相似度

docs_list_overlap_score:计算问题相关文档与答案相关文档的重合率

右边为输出部分:

1.Answer

大模型根据问题生成的答案

2.Confidence

根据 Confidence score type 选择的计算类型, 输出答案置信度的计算结果

3.Source

根据问题找到的相关文档, 输出相关文档的来源标题、相关分数和具体文本内容。

4.Url

请求的 Url, 用于检查请求参数是否正确。

5.Request time

请求时长。

如果通过 API 调用, API 可用的参数为:

序号	参数名	默认值	说明
----	-----	-----	----

1	query	hello	输入的问题
2	task	qa	1.qa: 基于知识库的问答 2.chat: 基于模型的问答 3.summarize: 基于给定文档的创意文本生成
3	index	smart_search_qa_test	根据需求更改
4	language	chinese	1.chinese --- 简体中文 2.chinese-tc --- 繁体中文 3.english --- 英语
5	temperature	0.01	大模型的参数, 控制答案生成的随机性
6	embedding_endpoint_name	huggingface-inference-eb	向量模型的 endpoint 名字
7	llm_embedding_name	pytorch-inference-llm-v1	大语言模型的 endpoint 名字
8	search_engine	opensearch	支持 opensearch 和 kendra 共 2 种搜索引擎
9	kendra_index_id	kendra 部署后自动将 index id 同步为默认值	根据需求修改
10	session_id	无	多轮对话使用, 可以为随机字符串
11	prompt	可查看 lambda/longchain_processor_qa/prompy.py 文件	根据需求修改
12	chain_type	stuff	在 task=summarize 时使用, 可以使用 map_reduce 对文档分段处理后再合并处理
13	combine_prompt	可查看 lambda/longchain_processor_qa/prompy.py 文件	在 task=summarize and chain_type=map_reduce 时使用
14	top_k	2	根据需求修改
15	cal_query_answer_score	FALSE	是否计算 query_answer_score
16	cal_answer_docs_score	FALSE	是否计算 answer_docs_score
17	cal_docs_list_overlap_score	FALSE	是否计算 cal_docs_list_overlap_score
18	response_type	无	web_ui: 返回到 React 前端页面数据结构 其他: 返回到 gradio 前端页面数据结构
19	model_type	other	1.bedrock: 调用 bedrock 的大语言模型接口 2.llama2:调用 llama2 模型 3.other:调用其他模型
20	bedrock_api_url	无	调用 bedrock 服务的 API URL, 参数在 model_type=bedrock 时使用
21	bedrock_model_id	anthropic.claude-v2	1.anthropic.claude-v2: 调用 claud2 模型 2.amazon.titan-tg1-large: 调用 titan 模型 参数在 model_type=bedrock 时使用
22	bedrock_max_tokens	512	调用 bedrock 大语言模型时, 答案的最大生成 token 数

调用示例可以参考 gradio 页面的 Url。

六、多轮问答网页

需要验证多轮问答效果的场景, 可以使用 chatbot_demo.py 前端页面, 使用方式如下:

1. 打开 chatbot_demo.py, 修改 invoke_url、index、embedding_endpoint 和 llm_endpoint
2. 把 chatbot_demo.py 下载到本地电脑, 部署 streamlit:
pip install streamlit
3. 运行: streamlit run chatbot_demo.py, 本地浏览器自动打开页面, 可以在下面的输入框中输入问题:

AWS Intelligent Q&A Solution Guide

Models and parameters

Choose a language

☒ chinese
☐ chinese-tc
☐ english

Choose a task

☒ chat
☐ qa

Clear Chat History

Chatbot Demo

您好，请问有什么可以帮助您吗？

北京有什么好玩的地方？

北京有很多好玩的地方，例如故宫、长城、颐和园、天坛等。此外，北京还有许多美食和文化活动，如北京烤鸭、京剧、相声等。

上海呢？

上海有很多好玩的地方，例如外滩、东方明珠、南京路步行街、田子坊等。此外，上海还有许多美食和文化活动，如上海烤鸭、豫园、上海博物馆等。

Your message

4. 如果切换语言或任务参数，或重新起一个聊天 session，需要点击左下的“Clear Chat History”。