

Langchain-chat-0.2.9绑定到公司企业微信

一.首先先申明一点是想在公司企微中绑定Langchain知识库第一点是要用到公网IP,这里我推荐大家使用阿里云的ECS服务器。

1.租阿里云的服务器的教程如下:

网址如下: [阿里云云服务器, 质优价更低 (aliyun.com)](https://cn.aliyun.com/daily-act/ecs/activity_selection?from_alibabacloud=&utm_content=se_1015259433)
登录自己的阿里云账号



2.点击控制台首页,再点击左上角三个横杠,找到云服务器ECS



3.进入带ECS服务器后开始创建实例,这边选择按量付费,尽量不选择抢占式实例(当库存不足会释放实例),实例中选GPU/FPGA/ASIC,找自己合适的算卡,我这边就以我的为例我要部署的是glm3的模型,使用的是A10的显卡,显存是24GB。

付费类型

包年包月
先付费后使用，价格优惠

按量付费
先使用后付费，按需开通

抢占式实例
较按量付费最高可省90%

使用须知

按量付费实例不支持预留实例券

地域

华南 1 (深圳)

华南 3 (广州)

如何选择地域

使用须知 实例创建之后地域将无法更改，不同地域的实例之间内网互不相通，距离实例所在地域越近，对实例访问速度越快

网络及可用区

[默认]vpc-7xvcb6542toaurdlsmxwf

广州 可用区 A | [默认]vsw-7xvryvgmd0x9qo4mr1l9j | 网络: 172.24.32.0/20

如何选择可用区

[创建专有网络](#)

[创建交换机](#)

☐ 指定主网卡主私网IP地址

实例和镜像

实例

最近使用规格

全部规格

如何选择实例

筛选 选择 vCPU 选择内存 模糊搜索规格名称 I/O 优化实例 查看更多规格参数

架构

X86 计算

Arm 计算

GPU/FPGA/ASIC

弹性裸金属服务器

高性能计算

全部分类

全部分类

全部分类

全部分类

全部分类

规格族	实例规格	vCPU	内存	GPU/FPGA	GPU显存	可售可用区	架构	参考价格
GPU 计算型 gn7i	ecs.gn7i-c8g1.2xlarge	8 vCPU	30 GiB	1 * NVIDIA A10	1 * 24 GB	14个可用区	GPU/加速	¥9.5326/时
GPU 计算型 gn7i	ecs.gn7i-c16g1.4xlarge	16 vCPU	60 GiB	1 * NVIDIA A10	1 * 24 GB	14个可用区	GPU/加速	¥10.0934/时

4. 镜像选择用的是公共镜像Alibaba Cloud Linux操作系统，版本选择的是Alibaba Cloud Linux 3.2104 LTS 64位，勾选安装GPU驱动，CUDA版本选择12.0.1，Driver 版本 525.105.17，CUDNN 版本 8.9.1.23，云盘大小设置成100GiB。

实例特性 FastGPU 是一款阿里云推出的 GPU 集群 极速创建工具，为您轻松获取 IaaS 层的计算、存储、网络等资源，一键部署分钟级创建 GPU 集群，[了解更多](#)

镜像

最近使用镜像

公共镜像

自定义镜像

共享镜像

云市场镜像

社区镜像

Alibaba Cloud Linux

Anolis OS

CentOS

Windows Server

SUSE Linux

Ubuntu

Alibaba Cloud Linux 3.2104 LTS 64位

☒ 免费安全加固 云服务器加载基础安全组件，提供网站漏洞检查、云产品安全配置检查、主机登录异常告警等安全功能，并可以通过云安全中心统一管理

☒ 安装 GPU 驱动 安装相对耗时约10-20分钟，实例启动时间会更长并伴有自动重启

☐ AIACC 训练加速

☐ AIACC 推理加速

CUDA 版本 12.0.1 / Driver 版本 525.105.17 / CUDNN 版本 8.9.1.23

镜像特性 该镜像支持使用ecs-user作为默认用户，[了解更多](#)

展开

存储

系统盘

如何选择云盘

类型	容量	数据	IOPS	性能	操作
ESSD云盘	100	GiB	1	3000	PL0 (单盘IOPS性能上限1万) <input checked="" type="checkbox"/> 随实例释放

☐ 加密

云盘类型 (ESSD云盘) 在当前可用区已购买 0 GiB，还可购买 133020 GiB，[提升配额](#)

云盘性能 不同云盘性能不同，[各云盘性能指标](#)

云盘容量 云盘创建总大小会受到配额限制 [查看详情](#)

5. 带宽和安全组，一定要勾选分配公网IPv4地址，新建安全组在开通IPv4端口/协议中将SSH，HTTPS,HTTP勾选上，登录我这边用的是自定义密码登录

带宽和安全组

公网 IP

☒ 分配公网 IPv4 地址
系统会分配公网 IP，也可采用更加灵活的弹性公网 IP 方案，了解 [如何配置并绑定弹性公网 IP 地址](#)>

带宽计费模式

按使用流量

按固定带宽

带宽峰值

1

2

3

5

10

50

100

Mbps

—

5

+

Mbps

阿里云免费提供最高 5Gbps 的恶意流量攻击防护。了解更多> | [提升防护能力](#)>

安全组

已有安全组

新建安全组

如何配置安全组

安全组名称

自定义安全组-20240221

安全组类型

普通安全组

企业级安全组

普通安全组和企业级安全组的对比差异>

开通IPv4端口/协议

☒ SSH (TCP: 22)
☒ HTTPS (TCP: 443)
☐ MySQL (TCP: 3306)
☐ Redis (TCP: 6379)

☐ Telnet (TCP: 23)
☐ MS SQL (TCP: 1433)
☒ RDP (TCP: 3389)
☒ ICMP (IPv4)

☒ HTTP (TCP: 80)
☐ Oracle (TCP: 1521)
☐ PostgreSQL (TCP: 5432)

弹性网卡

主网卡

交换机

vsw-7xrvyvgmd0x9qo4mr1l9j

☒ 自动分配 IP 地址

☒ 随实例释放

辅助网卡

已有辅助网卡

新建辅助网卡

请选择辅助网卡

创建弹性网卡

IPv6

当前交换机 vsw-7xrvyvgmd0x9qo4mr1l9j 尚未开通 IPv6，[立即开通](#)

弹性网卡 | IPv6 (选项)

管理设置

登录凭证

密钥对

自定义密码

创建后设置

密钥对安全强度远高于常规自定义密码，可以避免暴力破解威胁，建议您使用密钥对创建实例

登录名

☒ root

☐ ecs-user

root 具有操作系统的最高权限，使用 root 作为登录名可能会导致安全风险，建议您使用 ecs-user 作为登录名。前往[了解更多](#)>

登录密码

购买实例规格

当前所选实例规格: [提升规格](#)

使用期限

配置概要

付费类型

地域

可用区

网络类型

专有网络

交换机

实例规格

镜像

系统盘

公网带宽

安全组

弹性网卡

登录凭证

标签

实例名称

实例释放保护

元数据访问

自定义数据

☐ [云助手](#)

6.下面开始创建实例（账户内至少100RMB），创建之后就开始系统会自己装对应的CUDA版本，接下来自己创建conda环境，这里申明一下ECS服务器得自己创建conda环境，创建conda环境指令如下：

阿里云ECS服务器配置conda环境

- 1.在Miniconda官网上找到适用于Linux的安装包链接，然后使用wget命令下载：
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
- 2.运行安装脚本
bash Miniconda3-latest-Linux-x86_64.sh
- 3.选择安装路径：
for example : /opt/miniconda3
- 4.激活conda 环境：
source ~/.bashrc
- 5.测试conda 是否安装成功
conda --version
- 6.创建并激活新的conda环境
conda create -n xxxx python==版本号
- 7.激活conda 环境
conda activate xxxx
- 8.查看是否创建成功
conda env list

7.开始部署Langchain-chatchat0.2.9，这里有个技巧是阿里云这边用git clone是无法使用的，我们可以使用scp -r 文件绝对路径（C:/Users/Administrator/Desktop/知识库备份）root@公网IP:放在服务器文件的路径（home），将文件上传之后开始部署Langchai项目。（备注：-r当上传整个文件的时候要用的，如果上传单个文件是不需要-r的）forexample：scp -r F:/chatgpt-on-wechat-master root@8.136.105.209:/var/www/html

1.langchain-chatchat-0.2.9下载地址: <https://github.com/chatchat-space/Langchain-Chatchat>

```
1. # 安装全部依赖
$ pip install -r requirements.txt
$ pip install -r requirements_api.txt
$ pip install -r requirements_webui.txt
```

2. 下载大模型chatglm3-6b, embedding模型bge-large-zh这里可以采用阿里的modelscope上下载

(1). vim 创建一个.py

```
vim demo.py
```

(2). 文件中输入下面两行代码:

```
# 从modelscope上下载模型
from modelscope.hub.snapshot_download import snapshot_download
model_dir = snapshot_download('Jerry0/m3e-base', cache_dir='./model',
revision='master')
```

(3). 运行.py文件, 开始下载。

```
python demo.py
```

(4). 下载完之后就开始更改模型路径, 更改的地址在/langchain-chatchat/config/model_config.py

3. 初始化知识库和配置文件

```
$ python copy_config_example.py
$ python init_database.py --recreate-vs
```

4. 或者还想采用多卡部署 以Qwen-72B-Chat-Int4为例 (最低需要的显存为48GB)

下面是多卡部署的教程:

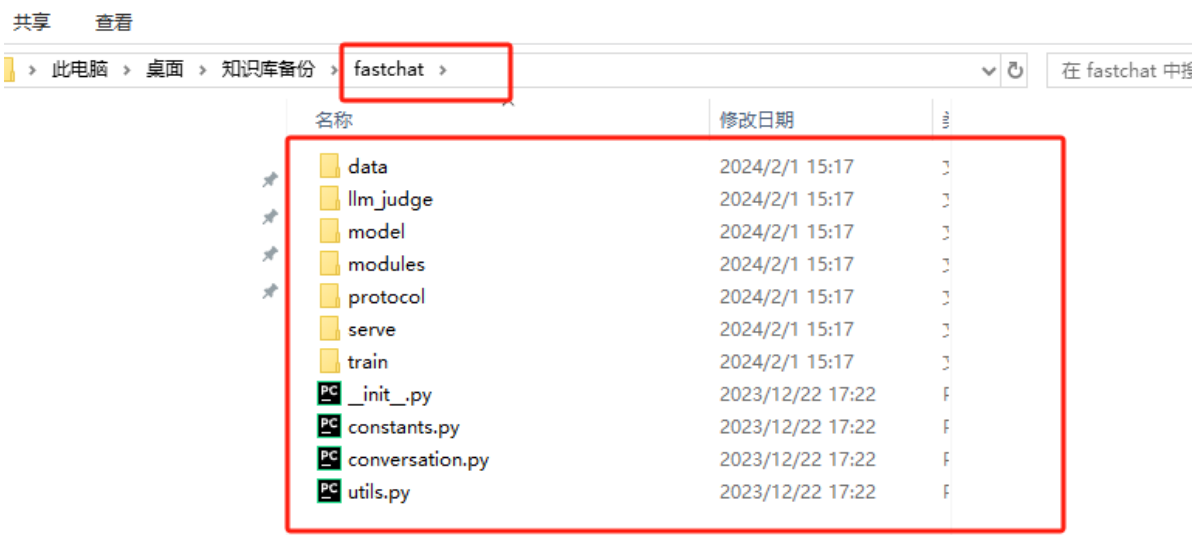
在Langchain-Chatchat/configs/server_config.py找到47行下, 将这三行放开

```
"gpus":None,
"num_gpus":3,
"nsx_gpu_memory":22GiB
```

接下来修改104行将chatglm3替换成Qwen-72B-Chat-Int4, 其它的操作就按下面的修改就行

接下来就是跳坑环节

1. 当执行python init_database.py --recreate-vs执行python init_database.py --recreate-vs, 会有一些坑最大的就是'ModuleNotFoundError: No module named 'fastchat'', 这个解决的办法是‘在终端执行命令pip3 install "fschat[model_worker,webui]"', 当执行完还在报错的话还需要将fastchat文件放到根目录下



2. `AttributeError: "NoneType" object has no attribute 'conjugate'`, 按照官方说这个报错的解决办法是把embedding模型换成m3e-base, 我试过倒按照这样倒是也可以跳过这个报错信息

3. 我这边提供一个完整的版本号包:

accelerate	0.21.0
addict	2.4.0
aiofiles	23.2.1
aiohttp	3.9.3
aiosignal	1.3.1
aliyun-python-sdk-core	2.14.0
aliyun-python-sdk-kms	2.16.2
altair	5.2.0
annotated-types	0.6.0
anyio	3.7.1
async-timeout	4.0.3
attrs	23.2.0
backoff	2.2.1
beautifulsoup4	4.12.3
blinker	1.7.0
Brotli	1.1.0
cachetools	5.3.2
certifi	2023.7.22
cffi	1.16.0
chardet	5.2.0
charset-normalizer	3.3.2
ci-info	0.3.0
click	8.1.7
cmake	3.28.1
coloredlogs	15.0.1
configobj	5.0.8
configparser	6.0.0
crcmod	1.7
cryptography	41.0.2
dataclasses-json	0.5.14
dataclasses-json-speakeasy	0.5.11
datasets	2.17.0
Deprecated	1.2.14
deprecation	2.1.0
dill	0.3.8

dirtyjson	1.0.8
distro	1.9.0
einops	0.7.0
emoji	2.10.1
etelemetry	0.3.1
faiss-cpu	1.7.4
fastapi	0.99.1
filelock	3.13.1
filetype	1.2.0
Flask	3.0.2
flatbuffers	23.5.26
frontend	0.0.3
frozenset	1.4.1
fsspec	2023.10.0
gast	0.5.4
gitdb	4.0.11
GitPython	3.1.41
greenlet	3.0.3
h11	0.14.0
h2	4.1.0
hpack	4.0.0
httpcore	1.0.2
httplib2	0.22.0
httpx	0.26.0
huggingface-hub	0.20.3
humanfriendly	10.0
hyperframe	6.0.1
idna	3.6
importlib-metadata	6.11.0
isodate	0.6.1
itsdangerous	2.1.2
jinja2	3.1.3
jmespath	0.10.0
joblib	1.3.2
jsonpatch	1.33
jsonpath-python	1.0.6
jsonpointer	2.4
jsonschema	4.21.1
jsonschema-specifications	2023.12.1
langchain	0.0.352
langchain-community	0.0.20
langchain-core	0.1.23
langdetect	1.0.9
langsmith	0.0.87
lit	17.0.6
llama-index	0.9.46
looseversion	1.3.0
lxml	5.1.0
markdown-it-py	3.0.0
markdownify	0.11.6
MarkupSafe	2.1.5
marshmallow	3.20.2
mdurl	0.1.2
modelscope	1.12.0
mpmath	1.3.0

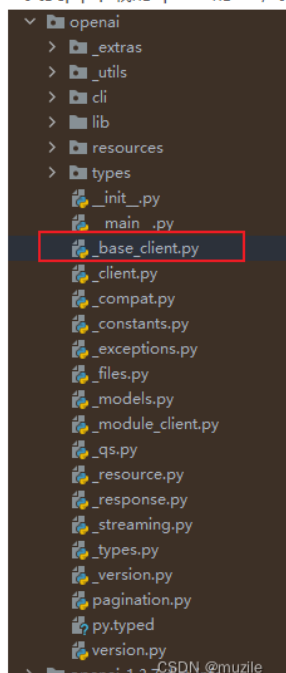
multidict	6.0.5
multiprocess	0.70.16
mypy-extensions	1.0.0
nest-asyncio	1.6.0
networkx	3.2.1
nibabel	5.2.0
nipy	1.8.6
nltk	3.8.1
numpy	1.26.4
nvidia-cublas-cu11	11.10.3.66
nvidia-cublas-cu12	12.1.3.1
nvidia-cuda-cupti-cu11	11.7.101
nvidia-cuda-cupti-cu12	12.1.105
nvidia-cuda-nvrtc-cu11	11.7.99
nvidia-cuda-nvrtc-cu12	12.1.105
nvidia-cuda-runtime-cu11	11.7.99
nvidia-cuda-runtime-cu12	12.1.105
nvidia-cudnn-cu11	8.5.0.96
nvidia-cudnn-cu12	8.9.2.26
nvidia-cufft-cu11	10.9.0.58
nvidia-cufft-cu12	11.0.2.54
nvidia-curand-cu11	10.2.10.91
nvidia-curand-cu12	10.3.2.106
nvidia-cusolver-cu11	11.4.0.1
nvidia-cusolver-cu12	11.4.5.107
nvidia-cusparse-cu11	11.7.4.91
nvidia-cusparse-cu12	12.1.0.106
nvidia-nccl-cu11	2.14.3
nvidia-nccl-cu12	2.19.3
nvidia-nvjitlink-cu12	12.3.101
nvidia-nvtx-cu11	11.7.91
nvidia-nvtx-cu12	12.1.105
onnxruntime	1.17.0
openai	1.6.1
opencv-python	4.9.0.80
oss2	2.18.4
packaging	23.2
pandas	2.2.0
pathlib	1.0.1
pdf2image	1.16.3
pdfminer.six	20221105
pdfplumber	0.10.3
Pillow	10.0.1
pip	23.3.1
platformdirs	4.2.0
protobuf	4.25.2
prov	2.0.0
psutil	5.9.8
pyarrow	15.0.0
pyarrow-hotfix	0.6
pyclipper	1.3.0.post5
pycparser	2.21
pycryptodome	3.20.0
pydantic	1.10.14
pydantic_core	2.16.2

pydeck	0.8.1b0
pydot	2.0.0
Pygments	2.17.2
PyMuPDF	1.23.24
PyMuPDFb	1.23.22
pyparsing	3.1.1
pypdfium2	4.27.0
python-dateutil	2.8.2
python-decouple	3.8
python-iso639	2024.2.7
python-magic	0.4.27
python-multipart	0.0.8
pytz	2024.1
pytz-deprecation-shim	0.1.0.post0
pyxnat	1.6.2
PyYAML	6.0.1
rapidfuzz	3.6.1
rapidocr-onnxruntime	1.3.11
rdflib	7.0.0
referencing	0.33.0
regex	2023.12.25
requests	2.31.0
rich	13.7.0
rpds-py	0.17.1
safetensors	0.4.2
scikit-learn	1.4.0
scipy	1.12.0
sentence-transformers	2.2.2
sentencepiece	0.1.99
setuptools	68.2.2
shapely	2.0.3
shortuuid	1.0.11
simplejson	3.19.2
six	1.16.0
smmap	5.0.1
sniffio	1.3.0
socksio	1.0.0
sortedcontainers	2.4.0
soupsieve	2.5
SQLAlchemy	2.0.25
sse-starlette	2.0.0
starlette	0.27.0
streamlit	1.29.0
streamlit-aggrid	0.3.4.post3
streamlit-chatbox	1.1.11
streamlit-feedback	0.1.3
streamlit-modal	0.1.2
streamlit-option-menu	0.3.6
strsimpy	0.2.1
sympy	1.11.1
tabulate	0.9.0
tenacity	8.2.3
threadpoolctl	3.2.0
tiktoken	0.6.0
tokenizers	0.13.3

toml	0.10.2
tomli	2.0.1
toolz	0.12.1
torch	2.0.1
torchvision	0.17.0
tornado	6.4
tqdm	4.66.1
traits	6.3.2
transformers	4.28.1
triton	2.0.0
typing	3.7.4.3
typing_extensions	4.9.0
typing-inspect	0.9.0
tzdata	2023.4
tzlocal	4.3.1
unstructured	0.12.4
unstructured-client	0.18.0
urllib3	2.2.1
uvicorn	0.23.2
validators	0.22.0
watchdog	4.0.0
websockets	11.0.3
werkzeug	3.0.1
wheel	0.38.4
wrapt	1.16.0
xxhash	3.4.1
yapf	0.40.2
yaml	1.9.4
zipp	3.17.0

当遇到这个报错ERROR: APIConnectionError: Caught exception: Connection error.的时候的解决办法，这是因为对于openai==1.6.1新版本中源码修改了，

1.找到pip下载的openai的Lib，找到_base_client.py这个文件



2.找到base_client.py文件中的BaseClient类，把init中原本的self.proxies=proxies修改为图片中的内容

```

class BaseClient(Generic[_HttpClientT, _DefaultStreamT]):
    _client: _HttpClientT
    _version: str
    _base_url: URL
    max_retries: int
    timeout: Union[float, Timeout, None]
    _limits: httpx.Limits
    _proxies: ProxiesTypes | None
    _transport: Transport | AsyncTransport | None
    _strict_response_validation: bool
    _idempotency_header: str | None
    _default_stream_cls: type[_DefaultStreamT] | None = None

    def __init__(
        self,
        *,
        version: str,
        base_url: str | URL,
        _strict_response_validation: bool,
        max_retries: int = DEFAULT_MAX_RETRIES,
        timeout: float | Timeout | None = DEFAULT_TIMEOUT,
        limits: httpx.Limits,
        transport: Transport | AsyncTransport | None,
        proxies: ProxiesTypes | None,
        custom_headers: Mapping[str, str] | None = None,
        custom_query: Mapping[str, object] | None = None,
    ) -> None:
        self._version = version
        self._base_url = self._enforce_trailing_slash(URL(base_url))
        self.max_retries = max_retries
        self.timeout = timeout
        self._limits = limits
        self._proxies = {'http': 'http://localhost:7890', 'https': 'http://localhost:7890'}
        self._transport = transport
        self._custom_headers = custom_headers or {}

```

CSDN @muzile_

3.最后在自己的测试文件中加入这两行代码

```

import os
os.environ["http_proxy"] = "http://localhost:7890"
os.environ["https_proxy"] = "http://localhost:7890"
123

```

```

import os

from openai import OpenAI

os.environ["http_proxy"] = "http://localhost:7890"
os.environ["https_proxy"] = "http://localhost:7890"

system_prompt = open("./prompt.txt", "r", encoding='UTF-8').read()
API_KEY = open("./APIKEY", "r").read().strip("/n").strip()

client = OpenAI(api_key=API_KEY)

stream = client.chat.completions.create(
    model="gpt-3.5-turbo-1106",
    messages=[{"role": "system", "content": system_prompt}],
    stream=True
)

for chunk in stream:
    if chunk.choices[0].delta.content is not None:
        print(chunk.choices[0].delta.content, end="")

```

CSDN @muzile_

这里还有一个坑我这边就是当整个服务启动之后，上传文档做做embedding的时候不成功的报错，错误如下：

知识库中包含源文件与向量库，请从下表中选择文件后操作				
文档加载器	分词器	文档数量	源文件	向量库
tx		0	✓	×

当传入文档做embedding向量的时候会出现RuntimeError: Directory 'static/'does not exist这个报错，解决的办法是安装两个指令是pip uninstall fitz,pip install pymupdf,pip install pdf2image==1.16.3,pip install pdfminer.six==2021105,pip install pdfplumber==0.10.3

解决fitz模块报错RuntimeError: Directory 'static/' does not exist

原创

xiaoyurainzi

于 2022-12-01 15:44:39 发布

阅读量7.7k

★ 收藏 14

👍 点赞数 15

版权

分类专栏: 遇到的问题及解决方法 文章标签: python

 遇到的问题及解决方法

专栏收录该内容

0 订阅 9 篇文章 订阅专栏

报错

fitz模块报错RuntimeError: Directory 'static/' does not exist

原因

使用Python处理PDF文档时，需要使用fitz模块。由于Python 3.8以上版本与fitz有兼容问题，会出现以下错误信息：RuntimeError: Directory 'static/' does not exist

解决办法

卸载fitz模块，安装pymupdf模块

执行的命令如下：

```
pip uninstall fitz
pip install pymupdf
```

参考资料

这个解决办法是安装pip install unstructured指令

```
2024-02-20 15:34:33,814 - faiss_cache.py[line:38] - INFO: 已将向量库 ('personnel', 'bge-large-zh') 保存到磁盘
INFO: 127.0.0.1:34930 - "POST /knowledge_base/delete_docs HTTP/1.1" 200 OK
2024-02-20 15:34:33,815 - _client.py[line:1027] - INFO: HTTP Request: POST http://127.0.0.1:7861/knowledge_base/delete_docs "HTTP/1.1 200 OK"
2024-02-20 15:36:04,799 - utils.py[line:289] - INFO: RapidOCRPDFLoader used for /home/Langchain-Chatcat-0.2.9/knowledge_base/personnel/content/员工手册-2024版.pdf
2024-02-20 15:36:04,800 - utils.py[line:371] - ERROR: ValueError: 从文件 personnel/员工手册-2024版.pdf 加载文档时出错: unstructured package not found, please install it
with 'pip install unstructured'
2024-02-20 15:36:04,800 - faiss_cache.py[line:38] - INFO: 已将向量库 ('personnel', 'bge-large-zh') 保存到磁盘
INFO: 127.0.0.1:58930 - "POST /knowledge_base/upload_docs HTTP/1.1" 200 OK
2024-02-20 15:36:04,801 - _client.py[line:1027] - INFO: HTTP Request: POST http://127.0.0.1:7861/knowledge_base/upload_docs "HTTP/1.1 200 OK"
INFO: 127.0.0.1:58938 - "POST /knowledge_base/search_docs HTTP/1.1" 200 OK
2024-02-20 15:36:10,443 - _client.py[line:1027] - INFO: HTTP Request: POST http://127.0.0.1:7861/knowledge_base/search_docs "HTTP/1.1 200 OK"
2024-02-20 15:36:12,564 - utils.py[line:289] - INFO: RapidOCRPDFLoader used for /home/Langchain-Chatcat-0.2.9/knowledge_base/personnel/content/员工手册-2024版.pdf
2024-02-20 15:36:12,564 - utils.py[line:371] - ERROR: ValueError: 从文件 personnel/员工手册-2024版.pdf 加载文档时出错: unstructured package not found, please install it
with 'pip install unstructured'
2024-02-20 15:36:12,565 - faiss_cache.py[line:38] - INFO: 已将向量库 ('personnel', 'bge-large-zh') 保存到磁盘
INFO: 127.0.0.1:58954 - "POST /knowledge_base/update_docs HTTP/1.1" 200 OK
2024-02-20 15:36:12,565 - _client.py[line:1027] - INFO: HTTP Request: POST http://127.0.0.1:7861/knowledge_base/update_docs "HTTP/1.1 200 OK"
INFO: 127.0.0.1:58970 - "POST /knowledge_base/search_docs HTTP/1.1" 200 OK
2024-02-20 15:36:12,756 - _client.py[line:1027] - INFO: HTTP Request: POST http://127.0.0.1:7861/knowledge_base/search_docs "HTTP/1.1 200 OK"
```

还有一个报错是libXext.so.6: cannot open shared object file: No such file or directory，解决的办法是

安装：sudo yum install libXext

最后当运行python startup.py -a 成功之后，出现这样的界面就算成功了

服务端运行信息:

OpenAI API Server: http://127.0.0.1:20000/v1

Chatchat API Server: http://127.0.0.1:7861

Chatchat WEBUI Server: http://0.0.0.0:8503

=====Langchain-Chatchat Configuration=====

Collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.

You can now view your Streamlit app in your browser.

URL: http://0.0.0.0:8503

二.知识库的代码封装

```
import json
from flask import Flask, request, jsonify
import requests
from config import conf, load_config

app = Flask(__name__)
debug=False
# 目标API的URL
TARGET_API_URL = 'http://127.0.0.1:7861/chat/knowledge_base_chat'

@app.route('/v1/chat/completions', methods=['POST'])
def api_adapter():
    # 接收到的请求数据
    incoming_data = request.json

    # 根据原始请求格式转换数据为目标API所需格式
    transformed_data = {
        "query": incoming_data['messages'][-1]['content'],
        "knowledge_base_name": "gouzi", #更改知识库
        "top_k": incoming_data['top_p'],
        "score_threshold": 0.8,
        "history": incoming_data['messages'][:-1],
        "stream": False,
        "model_name": "openai-api",
        "temperature": incoming_data['temperature'],
        "max_tokens": 0,
        "prompt_name": "gouzi"
    }

    # 向目标API发送请求
    response = requests.post(TARGET_API_URL, json=transformed_data)

    # 处理响应
    if response.ok:
        response_content = response.json()['answer']
        response_data={}
        response_data["usage"] = {"total_tokens": len(response_content),
"completion_tokens": len(response_content)}
        response_data["choices"]= [{"message": {"content": response_content}}]
        print(response_content)
        return jsonify(response_data)
    else:
        return jsonify({"error": "Failed to process request"}), 500
```

```
if __name__ == '__main__':
    app.run(debug=0, port=5002)
```

三.wechat项目的下载

地址如下<https://github.com/lewisliuyi/chatgpt-on-wechat>，下载好之后开始安装依赖项

(1) 克隆项目代码：

```
git clone https://github.com/zhayujie/chatgpt-on-wechat
cd chatgpt-on-wechat/
```

(2) 安装核心依赖 (必选)：能够使用 `itchat` 创建机器人，并具有文字交流功能所需的最小依赖集合。

```
pip3 install -r requirements.txt
```

(3) 拓展依赖 (可选，建议安装)：

```
pip3 install -r requirements-optional.txt
```

(4) 这个项目的最关键的配置就是配置 `config.json` 文件，首先我们得通过官当给的复制出最终生效的 `config.json` 文件，

```
cp config-template.json config.json
```

(5) 然后在 `config.json` 中填入配置，以下是对默认配置的说明，可根据需要进行自定义修改,这是我这边配置的文件具体得根据自己的情况进行一个修改（请去掉注释）： **

```
{
    "channel_type": "wechatcom_app",                #企微名称
    "model": "gpt-3.5-turbo",                        # 模型名称，支持 gpt-3.5-turbo,
    gpt-3.5-turbo-16k, gpt-4, wenxin, xunfei
    "open_ai_api_key": "",                            #open_ai_key可以不用填写
    "open_ai_api_base": "http://localhost:5001/v1",    #运行知识库封装好的地址生成出的url
    地址
    "text_to_image": "dall-e-2",
    "voice_to_text": "openai",
    "text_to_voice": "openai",
    "proxy": "",
    "hot_reload": false,
    "image_create_prefix": [
        "画"
    ],
    "speech_recognition": true,                        # 是否开启语音识别
    "group_speech_recognition": false,                # 是否开启群组语音识别
    "voice_reply_voice": false,
    "conversation_max_tokens": 2500,                  # 是否开启群组语音识别
    "expires_in_seconds": 3600,
    "character_desc": "你是ChatGPT，一个由OpenAI训练的大型语言模型，你旨在回答并解决人们的
    任何问题，并且可以使用多种语言与人交流。",
    "temperature": 0.1,
    "top_p": 1,
```

```
"subscribe_msg": "感谢您的关注!\n这里是ChatGPT,可以自由对话.\n支持语音对话.\n支持图片输入.\n支持图片输出,画字开头的消息将按要求创作图片.\n支持tool、角色扮演和文字冒险等丰富的插件.\n输入{trigger_prefix}#help 查看详细指令。",
"wechatcom_corp_id": "", #企微ID
"wechatcomapp_port": 80, #端口号(阿里云只开放了80端口,这里封装的时候得封装成80端口才能访问,autodl开放的端口号是60端口)
"wechatcomapp_agent_id": "", #企微AgentId
"wechatcomapp_secret": "", #企微Secret
"wechatcomapp_token": "", #企微Token
"wechatcomapp_aes_key": "" #企微EncodingAESKey
}
```

(6) 配置好config文件之后,这里有个坑就是先后顺序的问题。在Token 和 EncodingAESKey 先在企微中随机生成出来,企微中不要着急保存,执行下面的启动命令,成功执行完之后再到企微中保存,要不然连接不成功一定要细看这句话要不然配置不成功。

```
python app.py
```

(7) 正常启动之后的界面如下

```
[INFO][2024-02-21 14:58:49][plugin_manager.py:88] - Scanning plugins ...
[INFO][2024-02-21 14:58:49][plugin_manager.py:41] - Plugin Finish_v1.0 registered, path=./plugins/finish
[INFO][2024-02-21 14:58:49][plugin_manager.py:41] - Plugin Godcmd_v1.0 registered, path=./plugins/godcmd
[INFO][2024-02-21 14:58:49][plugin_manager.py:41] - Plugin Hello_v0.1 registered, path=./plugins/hello
chatgpt-tool-hub version: 0.4.6
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin tool_v0.4 registered, path=./plugins/tool
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin linkai_v0.1.0 registered, path=./plugins/linkai
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin Role_v1.0 registered, path=./plugins/role
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin Banwords_v1.0 registered, path=./plugins/banwords
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin Keyword_v0.1 registered, path=./plugins/keyword
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin Dungeon_v1.0 registered, path=./plugins/dungeon
[INFO][2024-02-21 14:58:50][plugin_manager.py:41] - Plugin BDunit_v0.1 registered, path=./plugins/bdunit
[INFO][2024-02-21 14:58:50][godcmd.py:194] - [Godcmd] 因未设置口令,本次的临时口令为1879.
[INFO][2024-02-21 14:58:50][godcmd.py:210] - [Godcmd] initied
[INFO][2024-02-21 14:58:50][keyword.py:40] - [keyword] {}
[INFO][2024-02-21 14:58:50][keyword.py:42] - [keyword] initied.
[INFO][2024-02-21 14:58:50][role.py:69] - [Role] initied
[INFO][2024-02-21 14:58:50][dungeon.py:56] - [Dungeon] initied
[INFO][2024-02-21 14:58:50][hello.py:24] - [Hello] initied
[INFO][2024-02-21 14:58:50][finish.py:23] - [Finish] initied
http://0.0.0.0:80/
```

(8) 所有服务成功启动的界面如下:

```
2. root@iZwz92h2u0lonjq7subq2/home/Langchain-ChatChat-0.2.9
当前使用的分词器: ChineseRecursiveTextSplitter
当前启动的LLM模型: ['chatglm3-6b', 'zhipu-api', 'openai-api'] @ cuda
('device': 'cuda',
 'host': '0.0.0.0',
 'infer turbo': False,
 'model_path': '/home/model/zhipuAI/chatglm3-6b',
 'model_path_exists': True,
 'port': 20802)
('api_key': '',
 'device': 'auto',
 'host': '0.0.0.0',
 'infer turbo': False,
 'online_api': True,
 'port': 21001,
 'provider': 'ChatGLMWorker',
 'version': 'chatglm-turbo',
 'worker_class': <class 'server.model.workers.zhipu.ChatGLMWorker'>)
('api_base_url': 'https://api.openai.com/v1',
 'api_key': '',
 'device': 'auto',
 'host': '0.0.0.0',
 'infer turbo': False,
 'model_name': 'gpt-3.5-turbo',
 'online_api': True,
 'openai_proxy': '',
 'port': 20802)
当前Embeddings模型: m3e-base @ cuda

服务端运行信息:
OpenAI API Server: http://127.0.0.1:20800/v1
ChatChat API Server: http://127.0.0.1:7861
ChatChat WEBUI Server: http://0.0.0.0:8503
-----Langchain-ChatChat Configuration-----
Collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.
You can now view your Streamlit app in your browser.
URL: http://0.0.0.0:8503

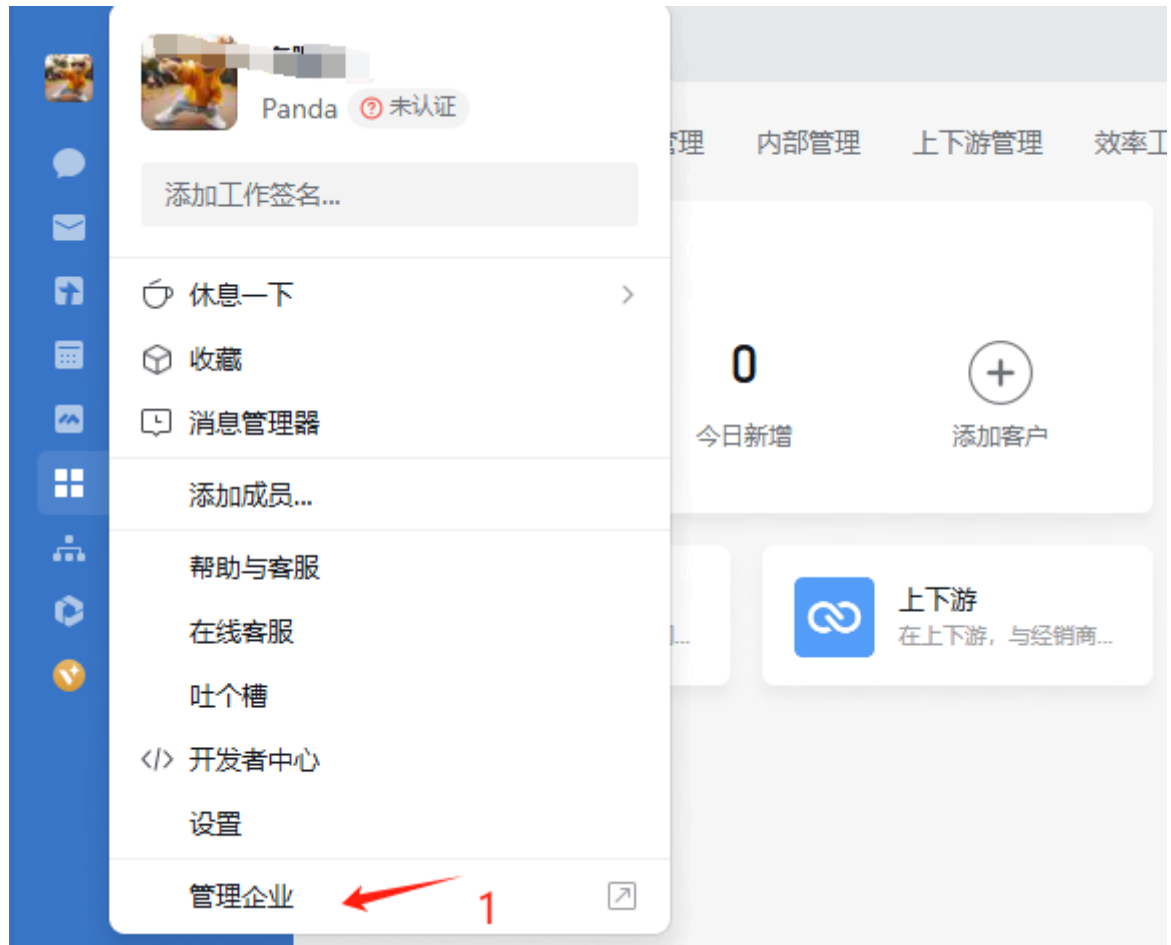
3. root@iZwz92h2u0lonjq7subq2/home/Langchain-ChatChat-0.2.9/server
Updates Information Summary: available
8 Security notice(s)
5 Moderate Security notice(s)
3 Low Security notice(s)
Run "dnf upgrade-minimal --security" to apply all updates. More details please refer to:
https://help.aliyun.com/document_detail/416274.html
Last login: Wed Feb 21 15:48:21 2024 from 118-31-243-54
(base) [root@iZwz92h2u0lonjq7subq2 ~]# cd ..
(base) [root@iZwz92h2u0lonjq7subq2 ~]# cd /home/Langchain-ChatChat-0.2.9/server/
(base) [root@iZwz92h2u0lonjq7subq2 server]# conda activate chatchat
(chatchat) [root@iZwz92h2u0lonjq7subq2 server]# python main.py
Serving Flask app 'main'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production W
tead.
 * Running on http://127.0.0.1:5002
 * Restarting with watchdog (inotify)
 * Debugger is active!
 * Debugger PIN: 118-377-289

4. root@iZwz92h2u0lonjq7subq2/home/Langchain-ChatChat-0.2.9
[INFO][2024-02-21 16:26:55][plugin_manager.py:41] - Plugin Godcmd_v1.0 registered, path=./plugins/g
[INFO][2024-02-21 16:26:55][plugin_manager.py:41] - Plugin Hello_v0.1 registered, path=./plugins/he
chatgpt-tool-hub version: 0.4.6
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin tool_v0.4 registered, path=./plugins/too
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin linkai_v0.1.0 registered, path=./plugins/li
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin Role_v1.0 registered, path=./plugins/ro
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin Banwords_v1.0 registered, path=./plugins/b
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin Keyword_v0.1 registered, path=./plugins/
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin Dungeon_v1.0 registered, path=./plugins/
[INFO][2024-02-21 16:27:13][plugin_manager.py:41] - Plugin BDunit_v0.1 registered, path=./plugins/b
[INFO][2024-02-21 16:27:13][godcmd.py:194] - [Godcmd] 因未设置口令,本次的临时口令为1478.
[INFO][2024-02-21 16:27:13][godcmd.py:210] - [Godcmd] initied
[INFO][2024-02-21 16:27:13][keyword.py:40] - [keyword] {}
[INFO][2024-02-21 16:27:13][keyword.py:42] - [keyword] initied.
[INFO][2024-02-21 16:27:13][role.py:69] - [Role] initied
[INFO][2024-02-21 16:27:13][dungeon.py:56] - [Dungeon] initied
[INFO][2024-02-21 16:27:13][hello.py:24] - [Hello] initied
[INFO][2024-02-21 16:27:13][finish.py:23] - [Finish] initied
http://0.0.0.0:80/
```

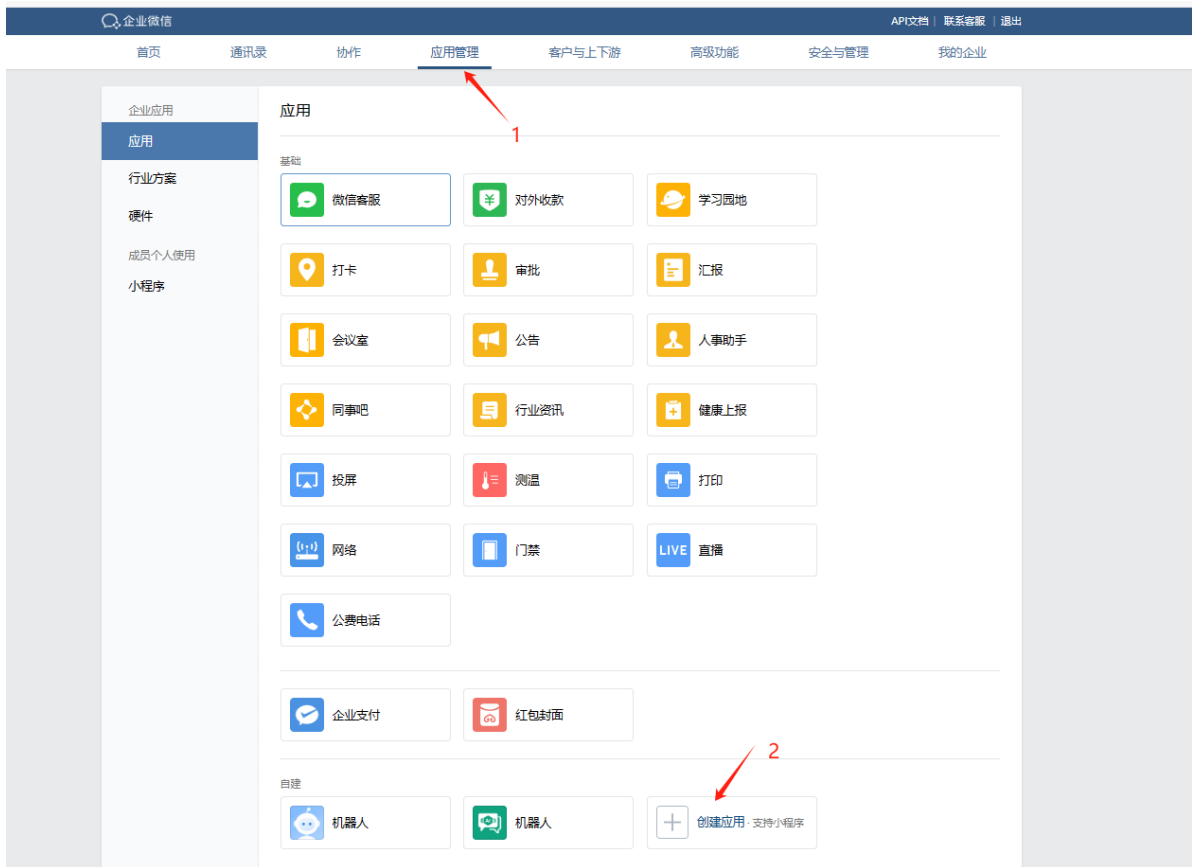
四.开始注册企业微信

一.先注册个人企业微信

1.点击个人头像然后左击有个管理企业，



2.然后进入到后台，在应用管理中的自建新建一个开发的应用



3.创建应用中按照提示创建应用logo,应用名称, 以及可见范围中的可见部门/成员（必填项）



4.创建好之后进入到应用中, 这里有几个点是需要了解, 方便后期使用, AgentId, Secret, 接收信息 (URL,Token,EncodingAESKey) ,企业可信IP





5.接下来得配置可信域名完成域名归属认证,得将企微的文件放置到对应的位置上去,当放置成功之后使用公网ip+文件名访问文件的内容,

(1) 我们可以使用Apache或者nginx,下面是Apache的操作指令



ChatGPT

在 CentOS/RHEL 系统上, Apache HTTP Server 被称为 `httpd`, 而不是 `apache2`。
请使用以下命令安装 Apache:

```
bash
sudo yum install httpd
```

然后, 您可以使用 `systemctl` 命令启动 Apache:

```
bash
sudo systemctl start httpd
```

如果您需要在系统启动时自动启动 Apache, 可以运行以下命令:

```
bash
sudo systemctl enable httpd
```

随后, 您可以使用浏览器访问服务器的公网 IP 地址, 以验证 Apache 是否已成功安装。
如果有其他问题或需要更多帮助, 请告诉我。



(2) 也可以用Nginx, Nginx的配置文件的文件位置一般在/etc/nginx/nginx.conf, 也可以将企微的配置文件放到这个位置下。无论你选择哪种方式, 确保文件可通过 `http://yourdomain.com/your_verification_file.txt` 访问, 以供企业微信服务器进行验证。完成验证后, 你可以将文件删除, 因为它仅用于验证域名控制权。**, 下载Nginx的指令如下: `sudo yum update,sudo yum install`

配置可信域名需完成域名归属认证 已验证

确定

取消

6.当前期工作做好以后下面就开始绑定URL地址这也是最难的, 它需要将公网ip解析到企业域名上, 直到openai回调成功之后的界面如下

<< 返回

API接收消息

保存成功

接收消息服务器配置

URL http://[redacted]/wxcomapp

Token 25tQ79Uw9Mk

EncodingAESKey bXlYmlxhmoB4P2LXoHoFcSt5bEN9BAH6P7czhoaB7r

编辑 删除

接收的消息事件类型

用户发送的普通消息, 自定义菜单操作, 审批状态通知事件, 外部联系人变更回调, 微信客服消息和事件, 支付和退款通知, 上下游变更回调, 直播状态变更回调, 会议室预定状态变更回调

7.当API接口回调成功之后，部署所有的都已解决，就可以在企微中间相对应的问题了,最后成功运行成功的界面

