**Fate-LLM 应用**

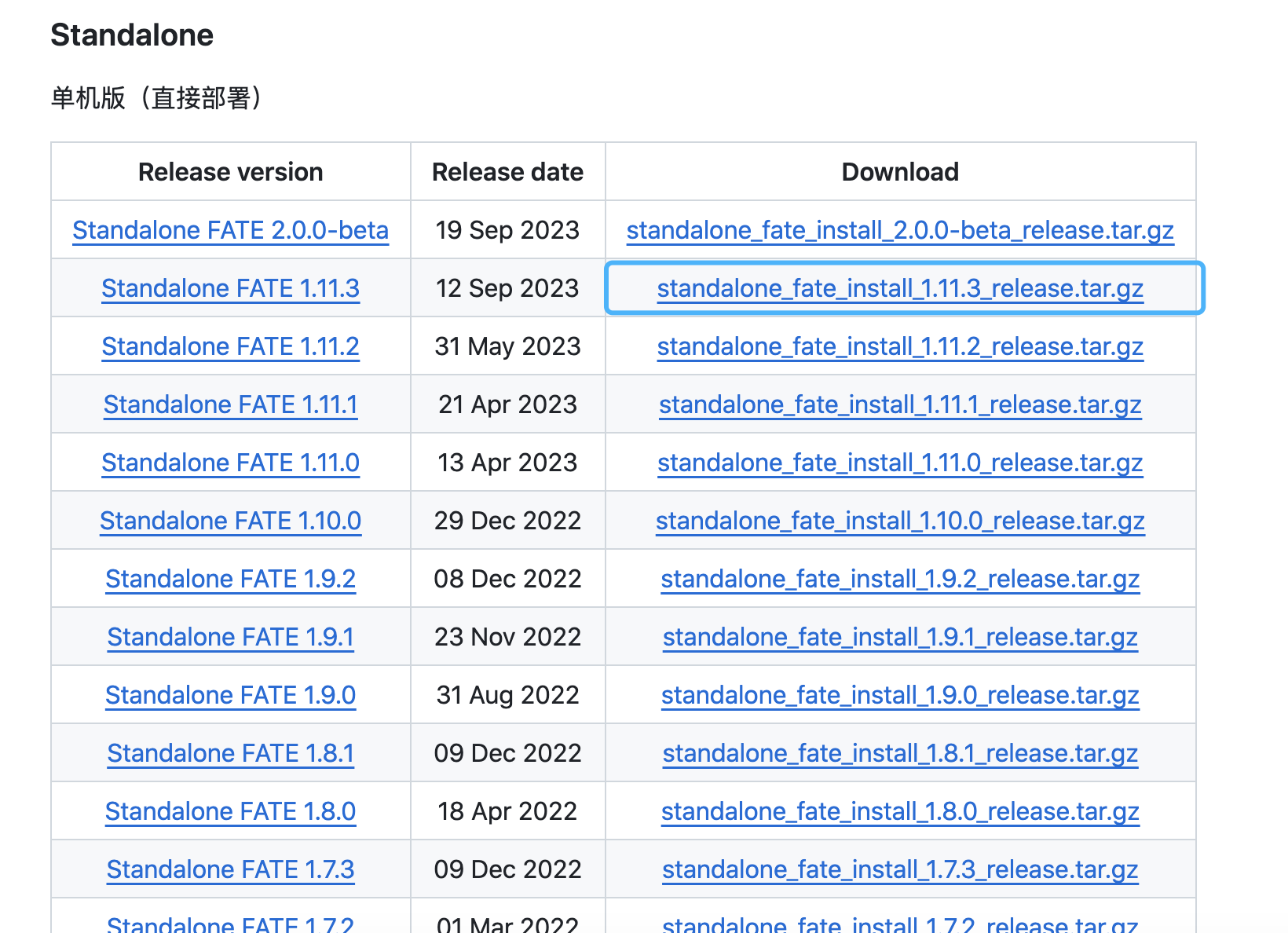
**一、源码单机部署**

**Fate+LLM安装**

1. **源码下载**

1.1 离线**安装包下载**：

https://github.com/FederatedAI/FATE/wiki/Download



**1.2 在线下载：**

|  |
| --- |
| Bash export version=1.11.4  wget https://webank-ai-1251170195.cos.ap-guangzhou.myqcloud.com/fate/${version}/release/standalone\_fate\_install\_${version}\_release.tar.gz; tar -xzvf standalone\_fate\_install\_${version}\_release.tar.gz |

1.3 **环境检查**：

|  |
| --- |
| Bash lsof -i:8080 lsof -i: 9360 netstat -apln|grep 9380 |

**1.4 环境安装**

|  |
| --- |
| Bash cd standalone\_fate\_install\_${version}\_release bash bin/init.sh init |

**1.5 启动**

|  |
| --- |
| Bash bash bin/init.sh status bash bin/init.sh start bash bin/init.sh status |

**1.6 服务启动或停止【辅助说明--直接略过】**

|  |
| --- |
| Bash  cd $project\_base  source $project\_base/bin/init\_env.sh    cd $project\_base/fateflow  bash bin/service.sh $1   cd $project\_base/fateboard  bash service.sh $1    cho "usage: $0 {start|stop|status|init}" |

**1.7 安装fate-client**

|  |
| --- |
| Bash source /data/standalone\_fate\_install\_1.11.3\_release/env/python/venv/bin/activate  cd /data/standalone\_fate\_install\_1.11.3\_release/fate/python/fate\_client python setup.py install  #初始化fate flow client cd /data/standalone\_fate\_install\_1.11.3\_release flow init -c conf/service\_conf.yaml  ##或者 ### pip install fate-client=1.10.0 ### flow init ip:9380  # 初始化pipeline ，大模型后续用的到 pipeline init --ip 127.0.0.1 --port 9380 |

**1.8 fate测试**

|  |
| --- |
| Bash source bin/init\_env.sh  flow test toy -gid 10000 -hid 10000 # success to calculate secure\_sum, it is 2000.0  fate\_test unittest federatedml --yes # 单元测试 |

**1.9 结合LLM模块**

Deploy FATE-Standalone version with 1.11.3 <= version < 2.0, then copy directory /FATE\_LLM/python/fate\_llm to {fate\_install}/fate/python/fate\_llm

|  |
| --- |
| Bash source /data/standalone\_fate\_install\_1.11.3\_release/env/python/venv/bin/activate |

**ChatGML环境安装**

**Step 1:下载源码和模型文件**

打开 https://cloud.tsinghua.edu.cn/d/fb9f16d6dc8f482596c2/ 下载模型文件

下载后的模型文件上传到 ChatGLM-6B/model 中

|  |
| --- |
| Bash git clone https://github.com/THUDM/ChatGLM-6B.git mkdir ChatGLM-6B/model cd ChatGML-6B pip install -r requirements.txt  环境依赖 # 运行微调需要 4.27.1 版本的 transformers pip install rouge\_chinese nltk jieba datasets -i https://pypi.tuna.tsinghua.edu.cn/simple |

**step 2： 下载transformer相关内容**

**安装Git LFS**

从https://github.com/git-lfs/git-lfs/releases 下载对应的git-fls上传服务器

|  |
| --- |
| Bash tar -xzvf git-lfs-linux-amd64-v3.3.0.tar.gz cd git-lfs-3.3.0 sudo ./install.sh  ##另一种安装方式 # git lsf install |

**下载模型到ChartGML-6B中，避免在线加载模型**

下载LLM模型方法1：

|  |
| --- |
| Python # 下载LLM模型 git clone https://huggingface.co/THUDM/chatglm-6b cd chatgml-6b git checkout v1.1.0 mv chatglm-6b ChatGLM-6B/  # 下载 Embedding 模型 git clone https://huggingface.co/GanymedeNil/text2vec-large-chinese /root/autodl-tmp/projects/ChatGLM-6B/text2vec  # 模型需要更新时，可打开模型所在文件夹后拉取最新模型文件/代码 git pull |

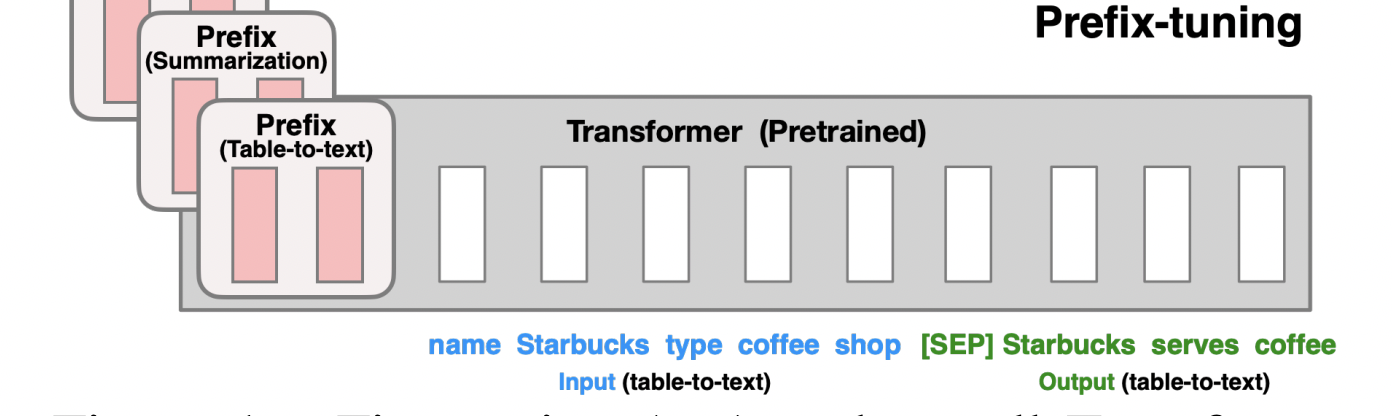
**将step1中ChatGLM-6B/model的pytorch\_model\* ice\_text.model移到 /root/autodl-tmp/projects/ChatGLM-6B/chatglm-6b目录下。**

|  |
| --- |
| Python mv /root/autodl-tmp/projects/ChatGLM-6B/model/pytorch\_model\* /root/autodl-tmp/projects/ChatGLM-6B/chatglm-6b mv /root/autodl-tmp/projects/ChatGLM-6B/model/ice\_text.model /root/autodl-tmp/projects/ChatGLM-6B/chatglm-6b |

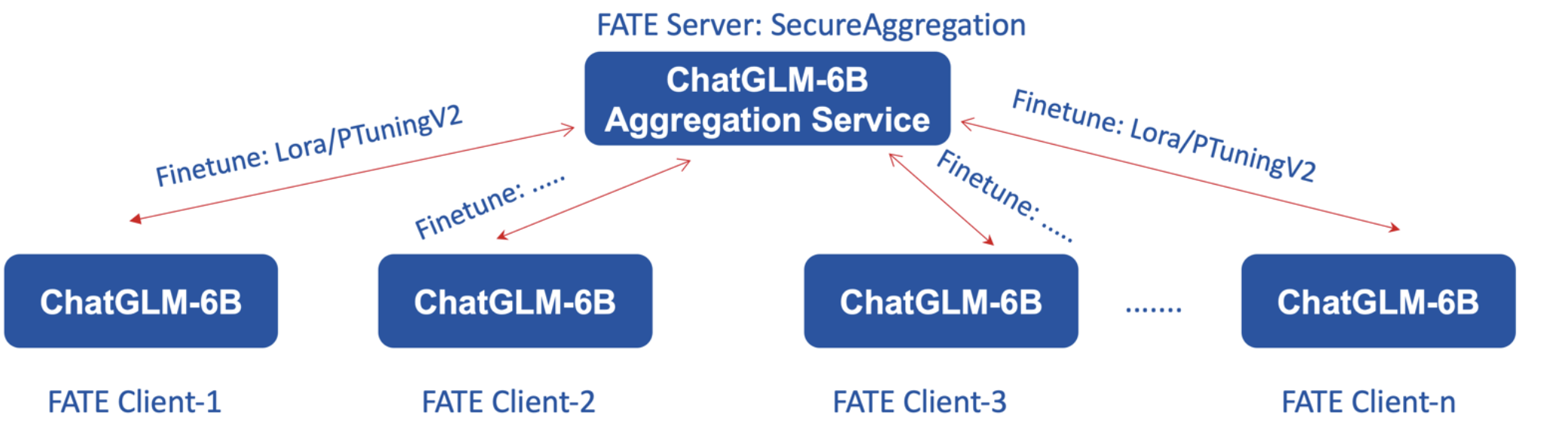
**二、联邦大模型与传统大模型对比**

**2.1 联邦大模型技术**

**P-Tuning v2原理**



Prefix-tuning（前缀微调）最开始应用在NLG任务上，由[Prefix, x, y]三部分构成，如上图所示：Prefix为前缀，x为输入，y为输出。Prefix-tuning将预训练参数固定，Prefix参数进行微调：不仅只在embedding上进行微调，也在TransFormer上的embedding输入每一层进行微调。



**2.2 传统微调部署**

# 禁用 W&B，如果不禁用可能会中断微调训练，以防万一，还是禁了吧  
export WANDB\_DISABLED=true

**2.2.1 参数调整**

修改 train.sh 和 evaluate.sh 中的 train\_file、validation\_file和test\_file为你自己的 JSON 格式数据集路径，并将 prompt\_column 和 response\_column 改为 JSON 文件中输入文本和输出文本对应的 KEY。可能还需要增大 max\_source\_length 和 max\_target\_length 来匹配你自己的数据集中的最大输入输出长度。并将模型路径 THUDM/chatglm-6b 改为你本地的模型路径。

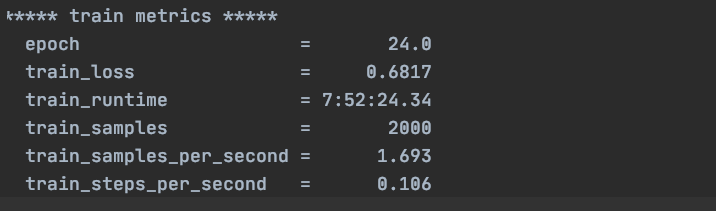
**train.sh 文件修改**

|  |
| --- |
| Shell PRE\_SEQ\_LEN=128 LR=2e-2  CUDA\_VISIBLE\_DEVICES=0 nohup python3 main.py \ --do\_train \ --train\_file train.json \ --validation\_file dev.json \ --prompt\_column question \ --response\_column report \ --overwrite\_cache \ --model\_name\_or\_path /root/autodl-tmp/projects/ChatGLM-6B/chatglm-6b \ --output\_dir output/adgen-chatglm-6b-pt-$PRE\_SEQ\_LEN-$LR \ --overwrite\_output\_dir \ --max\_source\_length 256 \ --max\_target\_length 256 \ --per\_device\_train\_batch\_size 1 \ --per\_device\_eval\_batch\_size 1 \ --gradient\_accumulation\_steps 16 \ --predict\_with\_generate \ --max\_steps 3000 \ --logging\_steps 10 \ --save\_steps 1000 \ --learning\_rate $LR \ --pre\_seq\_len $PRE\_SEQ\_LEN \ --quantization\_bit 4 >> train\_0522.out 2>&1 & |

train.sh 中的 PRE\_SEQ\_LEN 和 LR 分别是 soft prompt 长度和训练的学习率，可以进行调节以取得最佳的效果。P-Tuning-v2 方法会冻结全部的模型参数，可通过调整 quantization\_bit 来被原始模型的量化等级，不加此选项则为 FP16 精度加载。

**2000条数据微调用时 7:52:24.34**

**vi output/adgen-chatglm-6b-pt-128-2e-2/train\_results.json**



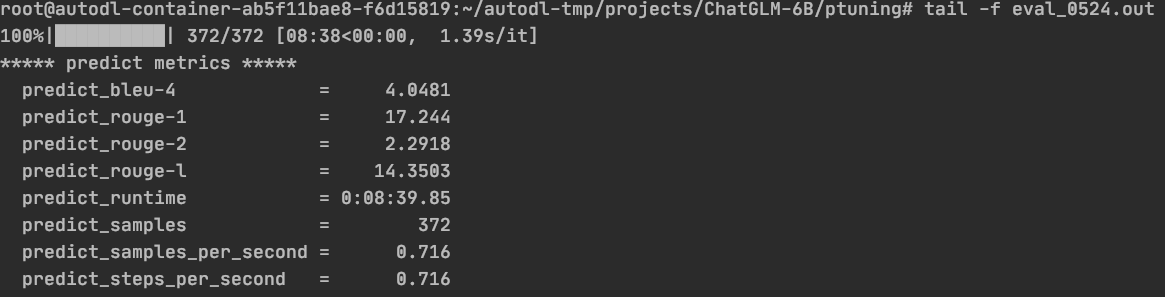
**evaluate.sh修改**

CHECKPOINT 实际就是 train.sh 中的 output\_dir。

|  |
| --- |
| Shell PRE\_SEQ\_LEN=128 CHECKPOINT=adgen-chatglm-6b-pt-128-2e-2 STEP=3000  CUDA\_VISIBLE\_DEVICES=0 nohup python3 main.py \  --do\_predict \  --validation\_file dev.json \  --test\_file dev.json \  --overwrite\_cache \  **--prompt\_column question \**  **--response\_column report \**  --model\_name\_or\_path /root/autodl-tmp/projects/ChatGLM-6B/chatglm-6b \  --ptuning\_checkpoint ./output/$CHECKPOINT/checkpoint-$STEP \  --output\_dir ./output/$CHECKPOINT \  --overwrite\_output\_dir \  --max\_source\_length 256 \  --max\_target\_length 256 \  --per\_device\_eval\_batch\_size 1 \  --predict\_with\_generate \  --pre\_seq\_len $PRE\_SEQ\_LEN \  --quantization\_bit 4 >> eval\_0524.out 2>&1 & |

**bash evaluate.sh**

执行完成后，会生成评测文件，评测指标为中文 Rouge score 和 BLEU-4。生成的结果保存在 ./output/adgen-chatglm-6b-pt-32-2e-2/generated\_predictions.txt。我们准备了 5 条推理数据，所以相应的在文件中会有 5 条评测数据，labels 是 dev.json 中的预测输出，predict 是 ChatGLM-6B 生成的结果，对比预测输出和生成结果，评测模型训练的好坏。如果不满意调整训练的参数再次进行训练。

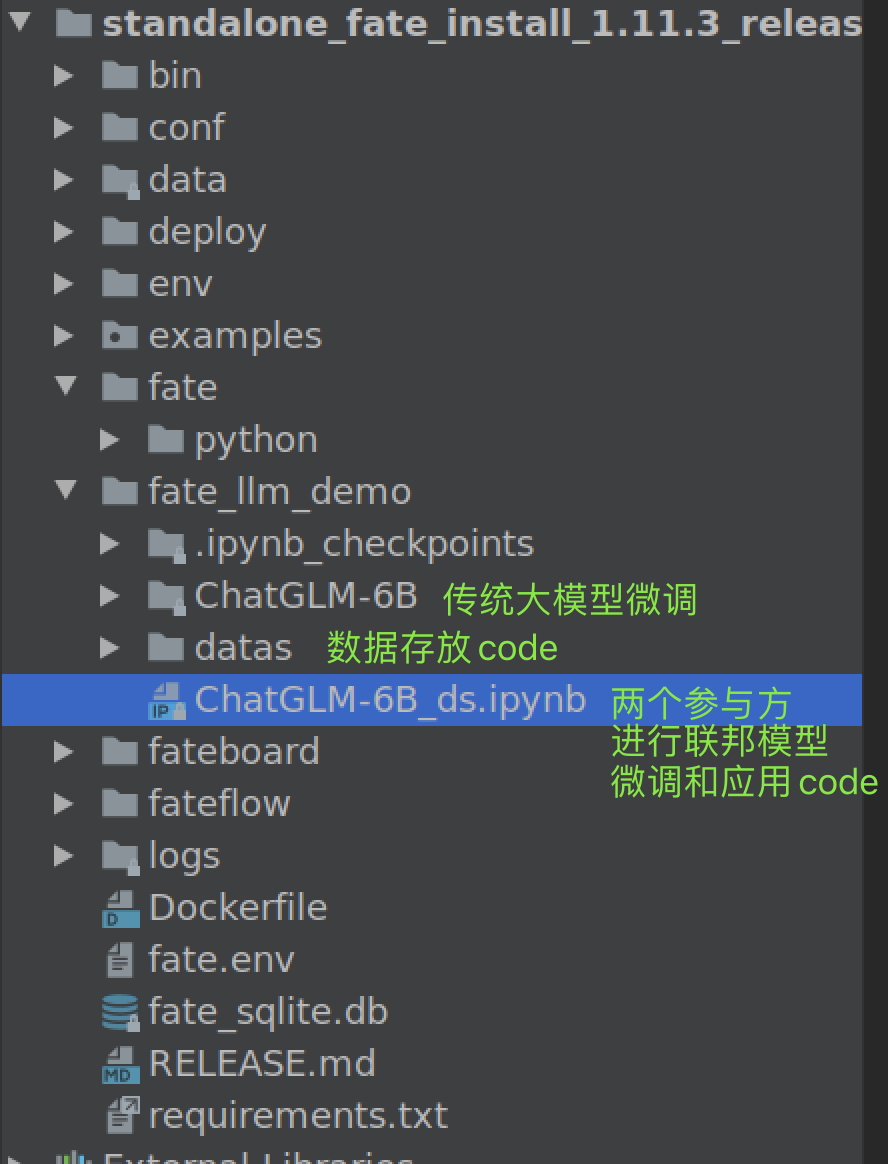


**部署微调后的模型**

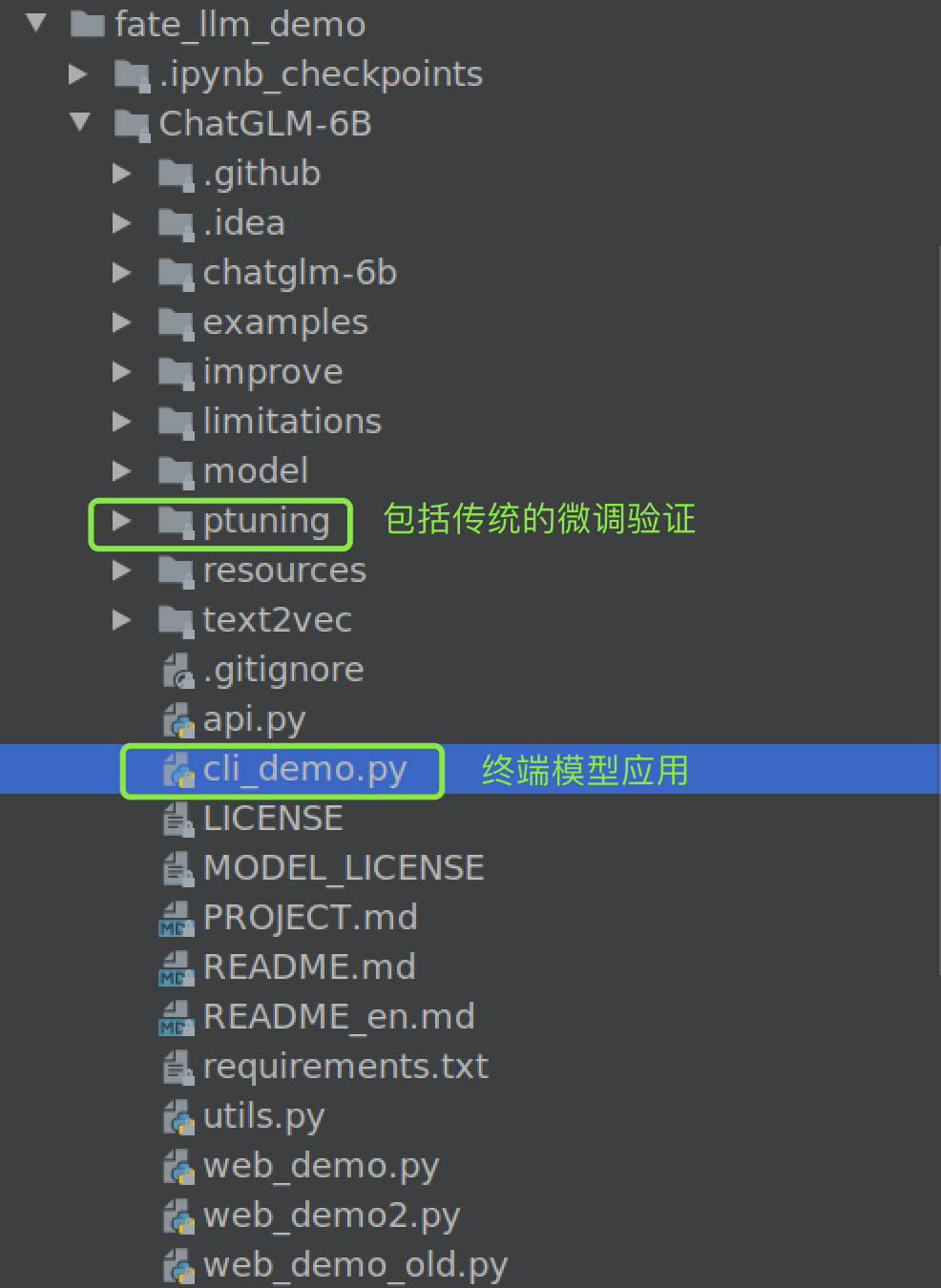
|  |
| --- |
| Shell CUDA\_VISIBLE\_DEVICES=0 python3 cli\_demo.py \ --model\_name\_or\_path /root/autodl-tmp/projects/ChatGLM-6B/chatglm-6b \ --ptuning\_checkpoint ptuning/output/adgen-chatglm-6b-pt-128-2e-2/checkpoint-3000 \ --pre\_seq\_len 128 |

**2.3 代码结构**

整体code:



传统大模型



**2.4 结果展示**

