1.运行步骤

官方示例给出的Meta-Llama-3-8B-Instruct这个大模型运行时需要大约16g的内存!

1.1创建一个新的conda环境

【强烈建议创新环境，用自己之前的环境会出版本兼容问题】

尽量创建python版本：3.8 – 3.11

```
conda create -n myenv python=3.9 -y
conda activate myenv
```

1.2 安装依赖包

安装vllm包：

如果自己电脑、服务器的cuda版本>=12.1都可以用这个命令：

```
# Install vLLM with CUDA 12.1.
pip install vllm
```

其他版本的安装参考：https://docs.vllm.ai/en/latest/getting_started/installation.html

安装需要的其它包：

```
pip install -r requirements.txt
pip install -r requirements_eval.txt
```

1.3 导入大模型

在models文件夹下的dummy_model.py文件中，可以定义自己设计的大模型（假设名字为DummyModel）；

然后在user_config.py文件中，把DummyModel赋给UserModel，例如：

```
from models.dummy_model import DummyModel
UserModel = DummyModel
```

在示例代码里直接调用了Meta-Llama-3-8B-Instruct模型：

```
from models.vanilla_llama3_baseline import Llama3_8B_ZeroShotModel
UserModel = Llama3_8B_ZeroShotModel
```

需要下载Meta-Llama-3-8B-Instruct模型的参数文件，保存路径为models/meta-llama/Meta-Llama-3-8B-Instruct

1）在models文件夹下新建文件夹：models/meta-llama

1）如果自己的服务器或电脑可以访问外网，直接参考docs/download-baseline-model-weights.md从Hugging Face网站下载，这个过程可能会出一些问题

2）国内网站：https://gitee.com/hf-models/Meta-Llama-3-8B-Instruct/tree/main

或者https://modelscope.cn/models/LLM-Research/Meta-Llama-3-8B-Instruct/files

注意：因为模型参数文件很大，建议用git lts命令下载：

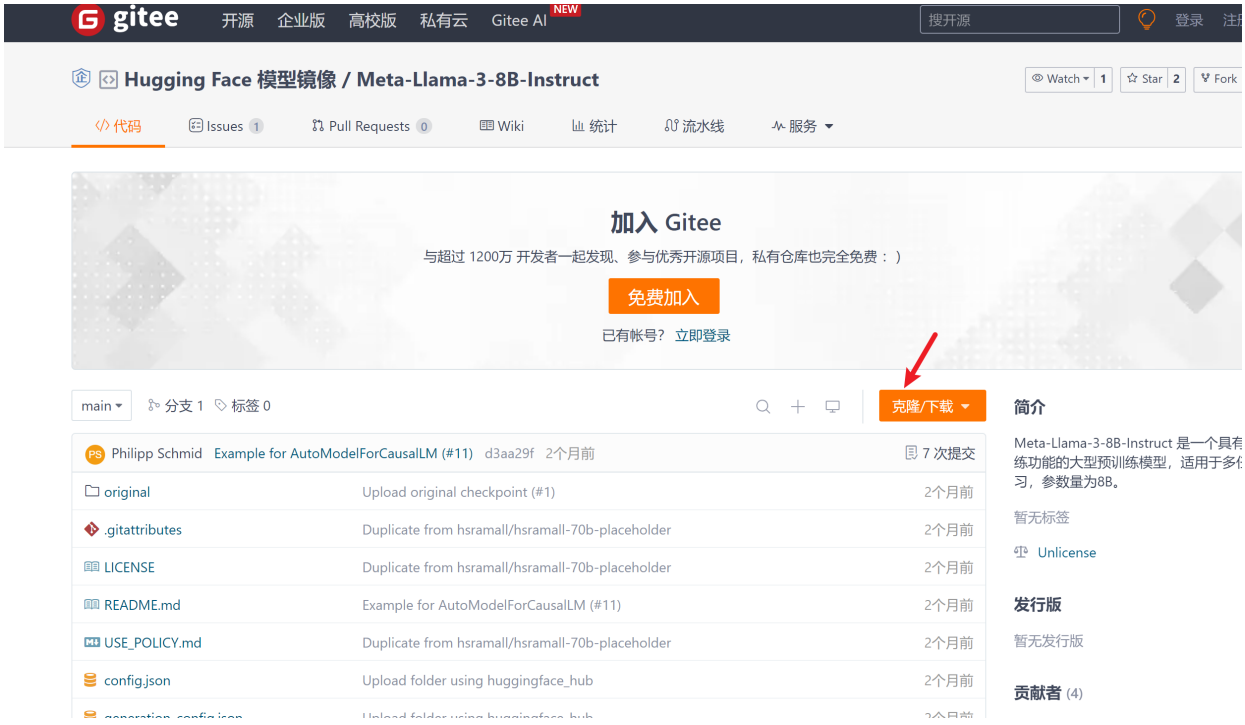
进到models/meta-llama文件夹下，右击打开git bash

先输入：

```
git lfs install
```

再输入：

```
git lfs clone https://www.modelscope.cn/LLM-Research/Meta-Llama-3-8B-Instruct.git
```



1.4 其它文件下载

下载sentence-transformers相关的模型：https://public.ukp.informatik.tu-darmstadt.de/reimers/sentence-transformers/v0.2/

代码里local_evaluation.py里的评估函数get_evaluation_methods()会用到，如果运行时提示访问不了Hugging Face，可以从上面的网站手动下载好，再把这里的参数改为model_path="模型保存的路径"

```python
# Define and return evaluation methods
def get_evaluation_methods():
    """
    Get evaluation methods including accuracy, sentence transformers, and other metrics.

    Returns:
    - A dictionary mapping metric names to their respective evaluation functions.
    """
    return {
        "accuracy": metrics.calculate_per_sample_accuracy,
        "hit rate@3": metrics.calculate_hit_rate_3,
        "rougel": metrics.calculate_rougel,
        "sent-transformer": lambda generated_text, reference_texts: metrics.calculate_cosine_similarity(
            generated_text=generated_text,
            reference_texts=reference_texts,
            model_name="all-MiniLM-L6-v2",
        ),
        "multilingual-sent-transformer": lambda generated_text, reference_texts: metrics.calculate_cosine_similarity(
            generated_text=generated_text,
            reference_texts=reference_texts,
            model_name="paraphrase-multilingual-MiniLM-L12-v2",
        ),
```

1.5.运行函数入口

```
python local_evaluation.py
```

1.6其他说明

- 提供的实例代码直接使用了预训练好的Meta-Llama-3-8B-Instruct模型，没有微调过，因此需要自己设计微调的方式；
- 可以参考官网给出的三个相关的额外信息数据集和别的数据集，调用大模型生成更多的样本，这里也会给一些生成的样本，但质量可能不是非常高

External Resource

Since we **do not provide large-scale training datasets**, all solutions have to rely heavily on external resources. We would like to highlight that all solutions submitted to this challenge should be based on resources (e.g. datasets and models) that are **publicly available**. Submissions should **not contain proprietary data or model checkpoints**. Participants can paraphrase or extend upon existing datasets (e.g. manual labeling, or labeling/generation with GPT), but should **make their extended datasets** available after the competition.

We list some public resources that may be helpful to your solutions.

- ECInstruct: An instruction tuning dataset also based on Amazon raw data.
- Amazon-M2: A multi-lingual Amazon session dataset with rich meta-data used for KDD Cup 2023.
- Amazon-ESCI: A multi-lingual Amazon query-product relation dataset used for KDD Cup 2022.

```
sudo apt install git-lfs
```