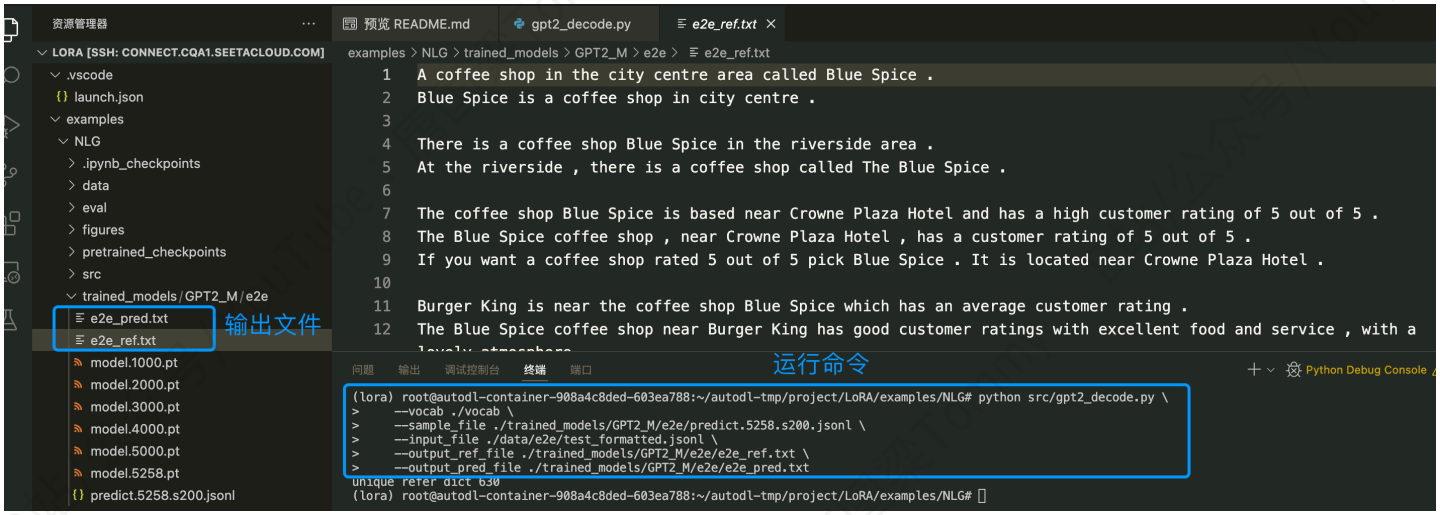
```
python src/gpt2_decode.py \
    --vocab ./vocab \
    --sample_file ./trained_models/GPT2_M/e2e/predict.5258.s200.jsonl \
    --input_file ./data/e2e/test_formatted.jsonl \
    --output_ref_file ./trained_models/GPT2_M/e2e/e2e_ref.txt \
    --output_pred_file ./trained_models/GPT2_M/e2e/e2e_pred.txt
```



第7步：在E2E测试集上进行评估

```
python eval/e2e/measure_scores.py ./trained_models/GPT2_M/e2e/e2e_ref.txt
./trained_models/GPT2_M/e2e/e2e_pred.txt -p
```

踩坑记录

运行上面的命令后，抛出Error。如下图：

```
Traceback (most recent call last):
  File "/root/autodl-tmp/project/LoRA/examples/NLG/eval/e2e/measure_scores.py", line 380, in <module>
    evaluate(data_src, data_ref, data_sys, args.table, args.header, args.sys_file, args.python)
  File "/root/autodl-tmp/project/LoRA/examples/NLG/eval/e2e/measure_scores.py", line 234, in evaluate
    coco_eval = run_coco_eval(data_ref, data_sys)
  File "/root/autodl-tmp/project/LoRA/examples/NLG/eval/e2e/measure_scores.py", line 323, in run_coco_eval
    coco_eval.evaluate()
  File "/root/autodl-tmp/project/LoRA/examples/NLG/eval/e2e/pycocoevalcap/eval.py", line 36, in evaluate
    gts  = tokenizer.tokenize(gts)
  File "/root/autodl-tmp/project/LoRA/examples/NLG/eval/e2e/pycocoevalcap/tokenizer/ptbtokenizer.py", line 54, in tokenize
    p_tokenizer = subprocess.Popen(cmd, cwd=path_to_jar_dirname, \
  File "/root/miniconda3/envs/lora/lib/python3.10/subprocess.py", line 971, in __init__
    self._execute_child(args, executable, preexec_fn, close_fds,
  File "/root/miniconda3/envs/lora/lib/python3.10/subprocess.py", line 1863, in _execute_child
    raise child_exception_type(errno_num, err_msg, err_filename)
FileNotFoundError: [Errno 2] No such file or directory: 'java'
```

解决措施：

在系统上安装JAVA，

```
sudo apt update
sudo apt install default-jre
```

解决Error后，再次运行命令，输出如下：

```
(lora) root@autodl-container-908a4c8ded-603ea788:~/autodl-tmp/project/LoRA/examples/NLG#
(lora) root@autodl-container-908a4c8ded-603ea788:~/autodl-tmp/project/LoRA/examples/NLG# python eval/e2e/measure_scores.py ./trained_models/GPT2_M/e2e/e2e_ref.txt ./trained_models/GPT2_M/e2e/e2e_pred.txt -p
Running MS-COCO evaluator...
creating index...
index created!
Loading and preparing results...
DONE (t=0.00s)
creating index...
index created!
tokenization...
PTBTokenizer tokenized 132573 tokens at 506481.96 tokens per second.
PTBTokenizer tokenized 1141 tokens at 13036.07 tokens per second.
setting up scorers...
computing METEOR score...
METEOR: 0.017
computing Rouge score...
ROUGE_L: 0.031
computing CIDEr score...
CIDEr: 0.133
Running Py-MTEval metrics...
SCORES:
==============
BLEU: 5.4198
NIST: 0.0000
METEOR: 0.0165
ROUGE_L: 0.0312
CIDEr: 0.1326
```

指标含义：

1. BLEU

含义：
BLEU(Bilingual Evaluation Understudy) 是一种用于评估机器翻译和自然语言生成模型的精确度指标。它通过计算生成的文本与参考文本之间的 n-gram 匹配程度来评估生成文本的质量。

计算方法：

- 计算生成文本和参考文本中 n-gram 的精确匹配数。
- 计算不同 n-gram（通常是 1-gram 到 4-gram）的加权几何平均值。
- 结合长度惩罚项，惩罚生成的文本长度与参考文本长度不一致的情况。

公式：
$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$
其中：

- BP 是长度惩罚（Brevity Penalty）。
- w_n 是 n-gram 的权重（通常等权重）。

- p_n 是生成文本和参考文本之间 n-gram 的精确匹配比例。

2. NIST

含义：
NIST(NIST Machine Translation Evaluation) 是 BLEU 的变种，它不仅考虑 n-gram 的精确匹配，还考虑匹配 n-gram 的信息量（即罕见 n-gram 的匹配会得到更高的得分）。

计算方法：

- 计算生成文本和参考文本中 n-gram 的精确匹配数。
- 计算每个匹配 n-gram 的信息量。
- 结合长度惩罚项。

公式：
与 BLEU 类似，但对 n-gram 匹配的权重进行了加权处理，强调罕见 n-gram 的匹配。

3. METEOR

含义：
METEOR 通过考虑词形变化、同义词匹配和词序来评估生成文本的质量。

计算方法：

- 计算生成文本和参考文本之间的 unigram 精确匹配。
- 计算词形变化、同义词匹配的匹配数。
- 结合词序惩罚项。

公式：

$$\text{METEOR} = F_{\text{mean}} \cdot (1 - \text{Pen})$$

其中：

- F_{mean} 是精确度和召回率的调和平均。
- Pen 是词序惩罚项。

4. ROUGE_L

含义：
ROUGE_L 基于最长公共子序列（LCS）来评估生成文本的覆盖度，强调生成文本中有多少部分与参考文本的顺序匹配。

计算方法：

- 计算生成文本和参考文本之间的最长公共子序列。

- 结合精确度和召回率。

公式：

$$\text{ROUGE}_L = \frac{(1+\beta^2)\cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \beta^2 \cdot \text{Recall}}$$

其中：

- 精确度 (Precision) 和召回率 (Recall) 基于 LCS 计算。
- β 是精确度和召回率的权重系数。

5. CIDEr

含义：
CIDEr (Consensus-based Image Description Evaluation) 主要用于图像描述生成任务，通过评估生成文本与参考文本的共识度来评估生成文本的质量。

计算方法：

- 计算生成文本和参考文本之间的 TF-IDF 加权 n-gram 相似度。
- 结合多个参考文本的共识度。

公式：

$$\text{CIDEr} = \frac{1}{m} \sum_{i=1}^{m} \frac{\sum_{n=1}^{N} \text{CIDEr}_n(i)}{N}$$

其中：

- m 是参考文本数量。
- $\text{CIDEr}_n(i)$ 是第 i 个参考文本的第 n-gram 相似度。