作业1报告

2300010850 程飞林

1. Tokenization

1.1 实现BPE，训练Tokenizer

1. BPE算法流程

- 将文本语料拆分为单个字符组成的序列。
- 统计当前词汇中所有相邻字符对的频率。
- 找到出现频率最高的字符对，并将其合并为一个新符号。
- 重复以上步骤若干次（直到达到预设的合并次数或词汇表大小），构建词表和合并规则。

2. 基于BPE算法训练Tokenizer的流程

- 收集大规模文本语料，进行基本预处理。
- 利用BPE算法基于收集到的语料进行训练，得到词汇表和合并规则。
- 从而利用合并规则来编码，词汇表来解码。

3. **我的实现**:具体的需查看代码

- 功能函数

```
def compute_bigram_statistics(token_ids, counter)
"""
计算相邻的token的bigram统计信息
返回Counter对象，key是bigram元组，值是该bigram的出现次数
"""
def replace_bigram(token_ids, bigram, bigram_id)
"""
将token序列中的所有指定bigram替换为新的bigram_id
返回: 替换后的新token序列
"""
def replace_control_character(s)
"""
替换控制字符为Unicode转义序列
"""
def render_token(t)
"""
将字节token渲染为可读字符串
"""
```

- Tokenizer类核心方法

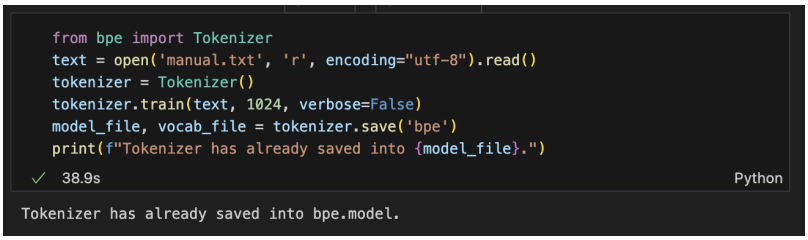
```
def train(self, text, vocab_size, verbose=False):
"""训练BPE分词器
```

```
text: 训练文本
vocab_size: 目标词汇表大小
verbose: 是否打印训练过程
"""
def encode(self, text):
"""
根据存储的合词规则编码文本为token id序列
"""
def decode(self, ids)
"""
根据词汇表解码token id序列为文本
"""
def save(self, file_prefix):
"""
保存模型和词汇表导文件
"""
def load(self, model_file)
"""
从文件加载模型
"""
```

4. 实现结果

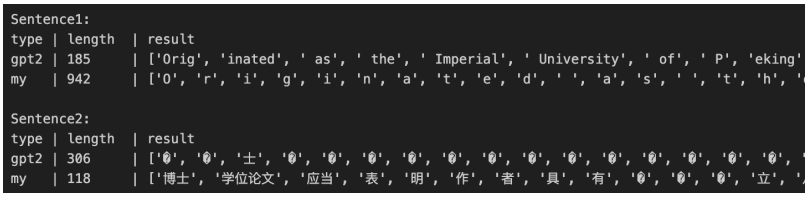
结果在bpe/test.ipynb中均有呈现

- 用《北京大学研究生手册》训练



- 检查encode和decode manual.txt的结果

- 两个句子的tokenize结果
  - 对于句子1，gpt2 tokenizer得到的明显短于我训练的，从具体token可以看出，gpt的结果中有很多以词根词缀等形式出现，而我训练的基本上都是单个字符出现。因为我训练的语料基本都是中文，而gpt2接触了大量的英文，从而能够得到更多的英文字符组合。
  - 对于句子2，gpt2 tokenizer得到的明显长于我训练的，我得到的token很多在训练预料中出现，也正是因为第二个句子词汇在manual.txt中出现频率很高，从而很多被记录在词汇表中。



1.2 回答问题

1. 使用ord()函数查看unicode，使用chr()函数转换为字符。北：21271，大：22823。 22823:大, 27169:模, 22411:型。
2. 大词表、小词表的好处和坏处：
    - 大词表：好处：编码效率高，语义保真度高；坏处：内存占用高，模型体积较大，泛化能力可能较弱，训练速度慢（输出层的softmax参数量与vocab size成正比）。
    - 小词表：好处：计算效率高，泛化能力可能更强；坏处：更多词未学习，语义信息分割不准确。
3. LLM基于分词（Token）处理，而非直接操作字符，故而无法进行一些简单的字符串操作
4. 训练数据中非英语语料占比低，分词器对非英语优化不足。并且大多数分词器都是基于拉丁语系设计的。
5. 算术需要符号逻辑推理，而LLM是概率模型，仅通过模式匹配生成结果，缺乏数学结构的内部表示。
6. GPT-2训练数据中代码占比少且质量远不及现代为代码优化的模型，此外代码需严格语法逻辑，而GPT-2没有经过专门的指令微调。
7. 该字符串是预训练时的特殊分隔符（标记文本边界），模型可能被训练为见到它即停止生成。
8. 原因是分词器在处理这个特定、罕见的字符串时，会将其分解成一个非常不寻常且在训练数据中极少共同出现的 token 序列。当模型处理这个“怪异”的序列时，其内部的激活值（activations）可能会出现极端异常，例如变得非常大或非常小，导致数值不稳定（如NaN，最终引发计算错误并使程序崩溃。
9. YAML更人类可读（支持注释、无需引号），而JSON严格格式可能被LLM误解析（如多余逗号）。
10. LLM中有tokenization这一步的存在，数据需要经过预处理（tokenizer）和后处理（detokenizer），这是独立于核心模型之外的，模型本身的端点实质上是Token ID序列。

2. LLM Implementation

git commit 截图：

```
commit 074b4de42ab10687db0980b10c993d7e2d5fb18d
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 22:33:29 2025 +0800

    [2300010850] add checkpoint

commit a7a99382ea894c9d4d7c12970d0734bd20b05a34
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 21:17:42 2025 +0800

    [2300010850] add logging & HellaSwag dataset

commit d65191cfe1bc9b2e3dab8fa20f06480760cc1849
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 20:00:36 2025 +0800

    [2300010850] move up the sampling code into the main loop

commit 3e45661e876afcdf3e639be77f34c79aaf91484b
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 19:47:29 2025 +0800

    [2300010850] add validation split

commit 5d3ae731b426222c4bacf1b171904bf993c0e237
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 19:39:25 2025 +0800

    [2300010850] switch to fineweb-edu

commit 76db0407861aa6e9d30ac317dbd948e7e98f670e
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 18:52:37 2025 +0800

    [2300010850] add DistributedDataParallel training

commit 23632b2a19db07ab2564fc942b3682fec7b0f947
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 17:07:28 2025 +0800

    [2300010850] add gradient accumulation

commit 4cb332021ef89ef61c9410ea78926b4288c8c009
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 16:48:38 2025 +0800

    [2300010850] add weight decay for 2D params, and add fused AdamW
```

```
commit a64dcec56d7d95cb691ac8a5e0274866979af31b
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 16:01:46 2025 +0800

    [2300010850] add learning rate scheduler

commit 9803bd58e4e134b41ace30c4ef8870ad834bc91c
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 15:48:44 2025 +0800

    [2300010850] AdamW params and grad clipping set

commit da36d476e686271bf374dfff7de30c0b7c8309c3
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 15:39:31 2025 +0800

    [2300010850] change print type

commit e1323c82424bf2db83cf5997bd49a6898719c234
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 15:35:31 2025 +0800

    [2300010850] change vocab size

commit a4e0435ca27ffb22268e65a236aed5804a6e2ff0
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 15:34:26 2025 +0800

    [2300010850] switch to flash attention

commit d7851ce95e121d7bdadbe2f57a5bd604e3c86000
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 15:12:09 2025 +0800

    [2300010850] add torch compile

commit 53a64141e0c009bb5603e435581318775ee34f83
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 15:04:05 2025 +0800

    [2300010850] add bfloat16

commit aed6952a4342996ac2d1acc46cad2bbc3095d342
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 14:53:06 2025 +0800

    [2300010850] add TensorFloat32 matmuls

commit 7a48187034e97d441bf32e916a8c62d25c4021a7
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 14:31:15 2025 +0800

    [2300010850] gpt2 initialization

commit 5c6ae7d4894d2c42c1704217b7c50c8bfcd3eac3
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 14:16:18 2025 +0800

    [2300010850] add a little dataloaderlite class

commit 8f9f0b7a48ba7e73c6f338dbed87566ee30ed146
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 13:54:10 2025 +0800

    [2300010850] a little loop train

commit 20ff68559cddc805faa39fb29669d94b7097a5f0
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 13:46:54 2025 +0800

    [2300010850] add loss to forward function for gpt class

commit 79527b06a51b4cc598384b40cc08a18d580c2b31
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 13:17:17 2025 +0800

    [2300010850] using shakespeare as an example

commit 38ea36607792f36e5b4a517c2755d6642d12b12c
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 12:54:59 2025 +0800

    [2300010850] generate from the model, autodetectd device and random model generation

commit ae420e7d8cdb4e1f40a00b687bfcd6492a7c68d0
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 11:58:55 2025 +0800

    [2300010850] add forward function for GPT class

commit 3d865f03cafc7bf1f9052d6316efaff2794fa717
Author: cflyuke <cflyuke@gmail.com>
Date:   Wed May 7 11:56:36 2025 +0800

    [2300010850] init commit
```

各个commit所完成的作用以及我的学习思考

1. `init commit`: 完成因果自注意机制层，多层感知机类，并将它们包装为块。再初步创建GPT2，创建类方法用于选择和加载gpt2参数。
2. `add gorward function for GPT class`: 为GPT2类完善forward方法，即整个GPT2所包含的各层，得到每个位置预测结果。
3. `generate from the model, autodetectd device and random model generation`:
    - 尝试导入模型生成句子
    - 自动检测可用设备
    - 再完全随机地生成句子。

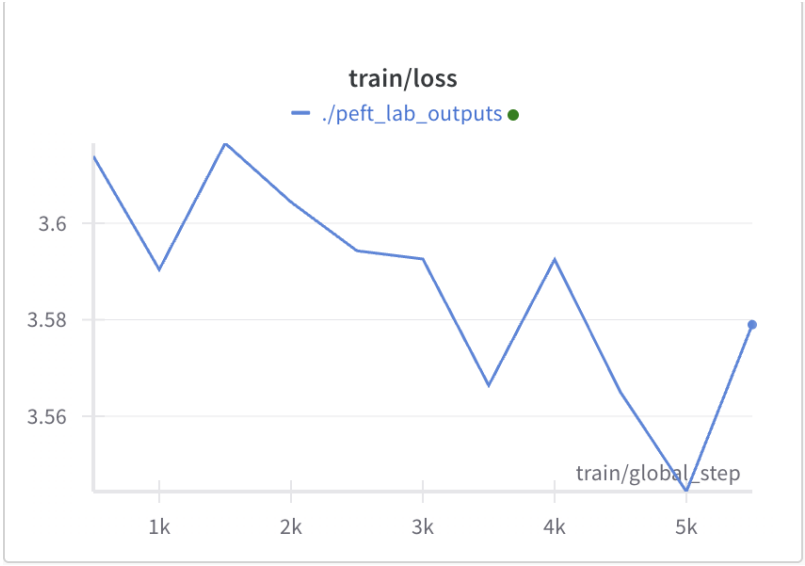
4. `using shakespeare as an example`:下载莎士比亚文集，并截取部分变为若干个block，用来试着生成5个例句。平常也可以用这种小的文集用来debug。
5. `add loss to forward function for gpt class`: 在forward function中计算结果的交叉熵损失。
6. `a little loop train`: 添加优化器，并且用文集进行训练。
7. `add a little dataloaderlite class`: 添加一个dataloader类，提供batch生成器。
8. `gpt2 initialization`:
    - 为词嵌入矩阵和最后一层的线性映射矩阵共享参数。
    - 神经网络权重初始化，并且对不同的层采用了不同的初始化策略。
9. `add TensorFloat32 matmuls`:控制float32矩阵乘法的计算精度，并记录下每秒钟处理的token数量，用于比较速率。
10. `add bfloat16`: 将部分操作（如矩阵乘法、卷积）转换为 bfloat16, 加速计算，避免数值溢出/下溢问题。
11. `add torch compile`: 添加模型编译优化功能，通过融合操作、优化内存访问等方式加速训练。
12. `switch to flash attention`: 将attention层转换为flash attention，通过改善内存访问机制加速训练过程。
13. `change vocab size`: 改变词汇表的大小，适配gpt2
14. `change print type`: 改善打印信息
15. `AdamW params and grad clipping set`: 为adam优化器添加参数，并且进行梯度裁剪，防止梯度爆炸
16. `add learning rate scheduler`: 设定lr的变化函数
17. `add weight decay for 2D params, and add fused AdamW`: 完成优化器配置函数，为模型设置分组权重衰减，并且启用参数优化的内核融合技术，加速训练
18. `add gradient accumulation`: 对梯度进行收缩累计，从而在不增加显存的同时，实现更大的batchsize
19. `add DistributedDataParallel training`: 实现多节点，多gpu的数据并行训练
20. `switch to fineweb-edu`: 切换到gpt2的训练数据集
    - 完成fineweb-edu数据集加载脚本。
    - 改变训练集到fineweb-edu数据集，改变数据集的拆分，加载等函数
21. `add validation split`: 添加评估数据集，在训练循环中添加评估代码。
22. `move up the sampling code into the main loop`: 将sample代码移动到训练主循环中
23. `add logging & HellaSwag dataset`:
    - 添加log file，对训练过程进行记录。
    - 添加hellaswag数据集进行评估训练模型的优劣，与原始GPT2进行对比
24. `add checkpoint`:
    - 添加checkpoint，对训练得到模型的进行实时存储，防止意外
    - 去除冗余代码，改进代码、使用数据类型等。

3. LoRA Fine-tuning

1. 运行记录：最原始的训练代码记录保留在`lora/Lora_imdb.ipynb`，后面在两个数据集上(包括Alpaca)调参数的训练记录保留在`lora/Lora.ipynb`上面

2. 首次训练结果：以下展示的是在最原始代码，完整数据集`noob123/imdb_review_3000`上面训练的结果

- loss曲线



- 模型生成结果

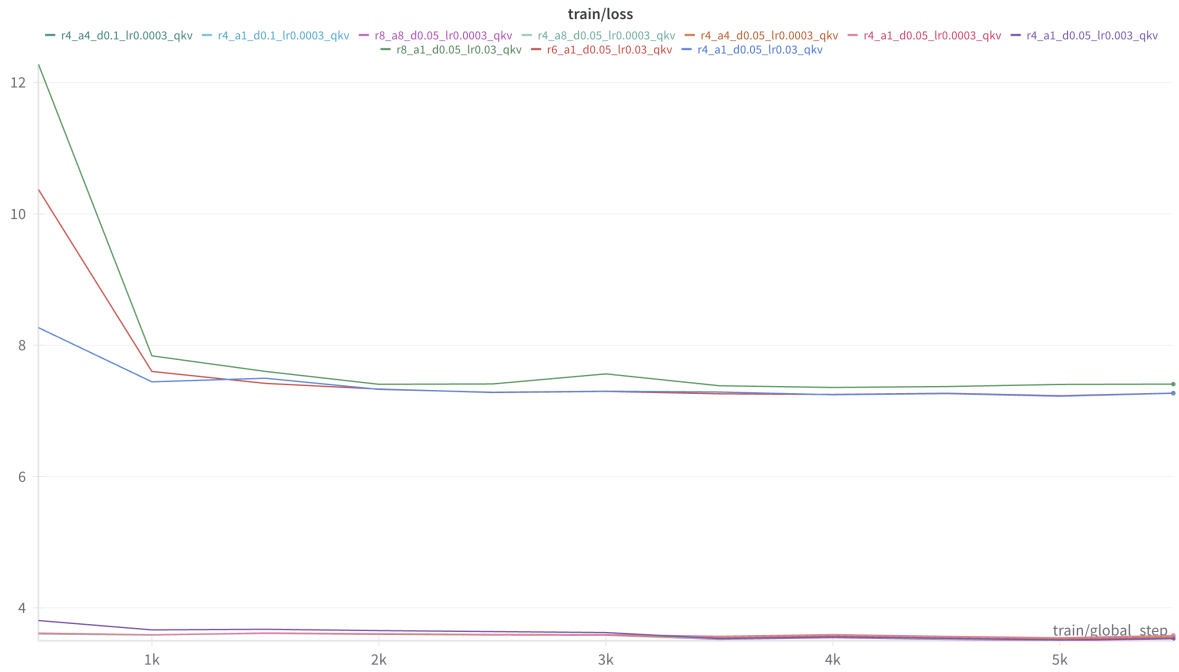
```
loaded_model = PeftModel.from_pretrained(foundation_model, peft_model_path, is_trainable=False)
input_sentences = tokenizer("I love this movie because", return_tensors="pt")
foundational_outputs_sentence = get_outputs(loaded_model, input_sentences, max_new_tokens=100)

print(tokenizer.batch_decode(foundational_outputs_sentence, skip_special_tokens=True))
```

```
["I love this movie because it is a very good one. I have seen the other movies and they are all great, but these two films were so different that i couldn't put
my finger on which was better than what we had in mind at first glance!<br /><br />The story of how an old man (who has been married for 30 years) falls ill with
cancer after being exposed to radiation from his wife's body during pregnancy...and then he gets sick again by eating food contaminated through her organ transpla
nts....this"]
```

3. 基于Imdb调试lora超参数

1. 主要调试的参数如下：
    - `r`：rank，即lora矩阵的秩
    - `lora_alpha`：缩放系数
    - `dropout`
    - `lr`
2. 训练的loss曲线如下，以下总共展示了10组超参数的结果：



3. 生成结果在`lora/lora_experiments_imdb.log`中有所展示，共有10组，以下仅展示了2组超参数的结果：

```
==================================================
Experiment at 2025-05-09 10:52:41
--------------------------------------------------
PARAMETERS:
{
    "r": 4,
    "lora_alpha": 1,
    "lora_dropout": 0.05,
    "learning_rate": 0.003
}
RESULTS:
Generated text:
I love this movie because it is so funny and I think it's the best
film ever made. It has a lot of good things about it: The acting,
especially by Tom Hanks (who plays an old man who was in his late
20's); all that great music from John Lennon to Michael Jackson...and
some really nice songs!<br /><br />The story: This one had me laughing
out loud for hours after watching it; but then when you get bored with
something like these movies or if you're just tired enough not wanting
Training metrics:
{'train_runtime': 349.1924, 'train_samples_per_second': 17.177,
'train_steps_per_second': 17.177, 'total_flos': 3922686368612352.0,
'train_loss': 3.604650484080631, 'epoch': 2.0, 'step': 5998}
==================================================
Experiment at 2025-05-09 10:39:06
--------------------------------------------------
PARAMETERS:
{
    "r": 6,
    "lora_alpha": 1,
    "lora_dropout": 0.05,
    "learning_rate": 0.03
}
RESULTS:
Generated text:
I love this movie because the., a of and to is it I in that was as The
for with but<br /><br film ( you not on have /> one be at about all
are his's who so like by from what out! good very an just)" really me
hasing he or story - some there myed great " they see... time if would
more were will no? up well been even we which get your their other
charactersly it's being hers how does than could any watching tooable
over bad
Training metrics:
{'train_runtime': 346.5137, 'train_samples_per_second': 17.31,
'train_steps_per_second': 17.31, 'total_flos': 3925234153488384.0,
'train_loss': 7.570473442319315, 'epoch': 2.0, 'step': 5998}
```

4. 分析：

    1. `lr`:是影响模型性能最重要的指标。在学习率很高在0.03时，可以看到结果基本都是乱序的单词和符号组合，随着学习率的下降到3e-4左右，文本明显改善，有连贯的句子和合理的内容，且逐渐

符合电影评论的规范。当学习率较高时，可以看到开始loss就很高，没有学习到有效的文本结构，并且步长过大也导致参数更新幅度过大，使优化过程“震荡”而无法收敛到最小点。

2. `r`:
    - 从较高的学习率时loss曲线下降的速度可以看出，较高的rank下降速度较快，这也符合lora矩阵的更新公式: `W = W_0 + rank / lora_alpha * AB`
    - 虽然更高的秩一般意味着传递的信息更多，从而表达效果更好，但这里可能由于数据集较简单、模式单一（关于电影评论）的缘故，在这里4，6，8三种rank下模型效果基本一致，在其他参数合适时，都有比较好的表现。
3. `lora_alpha`：在固定rank为4，考虑lora_alpha为1,4,8三种情况下，基本都有较好的表现。在`rank=8, lora_alpha=8`时，loss有细微提升，但生成句子的质量主观上感觉差不多。训练速度也无明显差异。
4. `dropout`: 由0.05到0.1带来的变化很小。dropout更多的是提高随机性，提高模型的泛化能力，但从单个句子的生成质量来说，很难有所改善。但在多次尝试下，所生成句子的多样性有所提升。

4. 使用最优超参数，基于Alpaca数据集进行lora微调

先前的最优超参数组合之一`r=4, lora_alpha=1, dropout=0.05, lr=3e-4`下，用`Alpaca`数据集进行微调，出现了训练不稳定的现象，训练的loss曲线在（第5点中）有展示。

- 从生成记录(`lora/lora_experiments_alpaca.log`)中可以看到，导入训练的loss爆炸前的checkpoint模型还是能正常生成流畅的句子的，但在loss爆炸后生成的就成了乱码了。

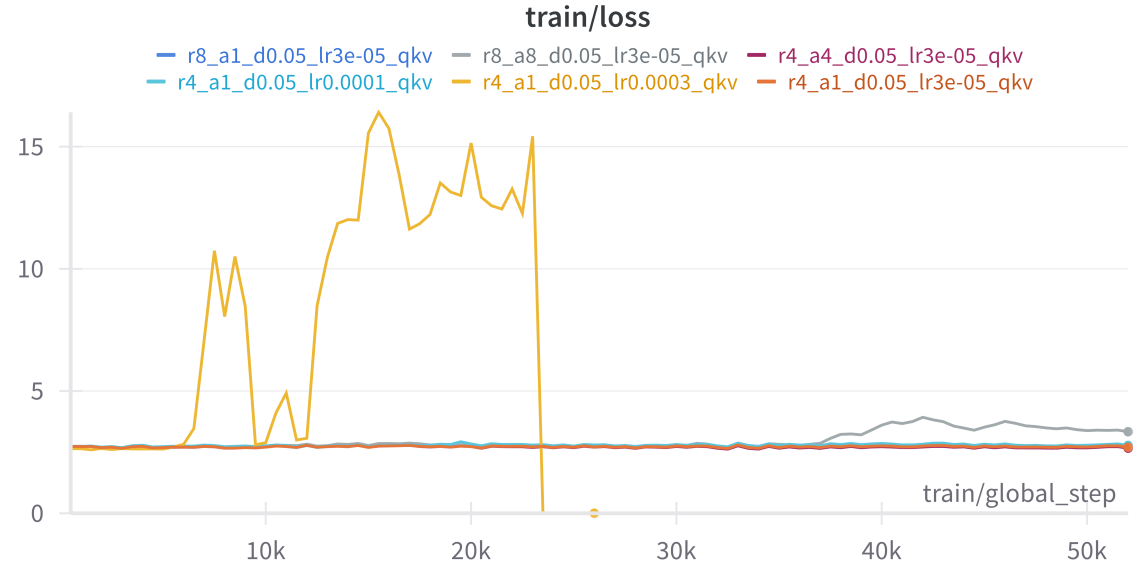
```
(checkpoint-4000)
I love this movie because it is so real. It shows us the struggles of
a child who has to deal with bullying and abuse in school, as well
like being bullied by his teacher or peers at home.  I loved that he
was able not only learn from these experiences but also get some
perspective on what makes him unique.
This film made me feel very connected towards my friends' lives too!
They are all different people than myself - they have their own
challenges which make them special.
It really touched every part i felt

(checkpoint-8000)
I love this movie because of / balikikliui!
```

- 降低学习率以后这样的情况就消失了，说明很有可能是学习率过大引起的。所以很有可能是lr过大导致过程震荡而无法正常收敛。

5. 基于Alpaca调试lora超参数

1. 使用`Alpaca`数据集做lora微调：
    - 调整超参数后的**loss曲线**如下，总共展示了6组。可知部分相同的超参数达到的效果不一样.



- **生成结果**在`lora/lora_experiments_alpaca.log`中有所展示，共有6组，以下仅展示了2组超参数的结果：

```
==================================================
Experiment at 2025-05-10 11:04:41
--------------------------------------------------
PARAMETERS:
{
    "r": 4,
    "lora_alpha": 1,
    "lora_dropout": 0.05,
    "learning_rate": 0.0003
}
RESULTS:
Generated text:
(checkpoint-4000)
I love this movie because it is so real. It shows us the struggles of
a child who has to deal with bullying and abuse in school, as well
like being bullied by his teacher or peers at home.  I loved that he
was able not only learn from these experiences but also get some
perspective on what makes him unique.
This film made me feel very connected towards my friends' lives too!
They are all different people than myself - they have their own
challenges which make them special.
It really touched every part i felt
(checkpoint-8000)
I love this movie because of / balikikliui!
Training metrics:
{'train_runtime': 4845.6681, 'train_samples_per_second': 21.463,
'train_steps_per_second': 10.732, 'total_flos': 2.526454676196557e+16,
'train_loss': 15.4244753732763364, 'epoch': 2.0, 'step': 52002}
==================================================
Experiment at 2025-06-15 05:04:01
--------------------------------------------------
PARAMETERS:
```

```
{
    "r": 8,
    "lora_alpha": 1,
    "lora_dropout": 0.05,
    "learning_rate": 3e-05
}
RESULTS:
Generated text:
I love this movie because it is so well made and the acting really
makes me feel like I am in a real life situation. It also has some
very touching moments that make you want to cry! The storyline of my
mom's struggles with depression, her relationship issues as she tries
hard not letting go or being too emotional when things get tough for
both sides but ultimately manages through them all together making us
laugh out loud at times.  This film definitely needs more screen time
than most movies I've seen before.
It was an amazing
Training metrics:
{'train_runtime': 3418.4031, 'train_samples_per_second': 30.425,
'train_steps_per_second': 15.212, 'total_flos':
1.6250848053313536e+16, 'train_loss': 2.7282228619679776, 'epoch':
2.0, 'step': 52002}
```

2. **结果分析**:

   - 最佳超参数：
     - Imdb: r=4, lora_alpha=1, lora_dropout=0.05, learning_rate=3e-4
     - Alpaca: r=8, lora_alpha=1, lora_dropout=0.05, learning_rate=3e-5
   - 数据集：
     - IMDB：非常狭窄，全部是关于电影评论的。语言风格和词汇相对固定，模型需要学习的模式比较单一。
     - Alpaca数据集：极其广泛，涵盖了常识问答、编程、数学、创意写作、翻译等数十个不同领域。指令和回答的格式、风格、长度和专业性千差万别。模型需要有很强的泛化能力来处理这种多样性。

3. **对具体LoRA超参数的影响**：

   从结果分析中可以看出，最优超参数主要差别为**r**和**lr**。

   r（Rank-秩）

   - r 决定了LoRA适配器的大小，即其参数量。它代表了适配器学习新知识的“容量”。
     - IMDB:
       - 任务简单，模式单一，不需要大幅改变模型行为。
       - 一个较小的 r (例如4) 通常就足够了。它有足够的容量来学习情感信号，同时避免在相对较小且单一的数据集上过拟合。
       - 使用过大的 r 反而可能引入不必要的噪声，导致性能下降。
     - Alpaca:
       - 任务复杂，需要学习全新的“指令遵循”能力，这需要对模型内部的注意力机制和知识表达方式进行更深刻的调整。

- 因此，通常需要一个更大的 r (例如8)。更大的容量才能捕捉和编码指令遵循这种复杂的行为模式。
- 如果 r 太小，适配器可能“学不会”或“学不好”如何遵循指令，导致模型只会重复预训练时的行为。

lr（learning rate）

- 学习率决定了模型在训练中更新权重的步长。
  - IMDB:
    - 任务目标明确，损失函数的“地形”相对平坦。
    - 可以容忍一个相对较高的学习率（例如3e-4）。模型可以较快地收敛到最优解。
  - Alpaca:
    - 任务非常精细和敏感。学习率过高很容易破坏基础大模型中已经存在的宝贵知识，导致“灾难性遗忘”，模型开始胡言乱语。
    - 因此，必须使用一个更低的学习率（例如3e-5），进行更“小心翼翼”的微调，确保在学习新技能的同时不破坏原有能力。