Assignment 1 report

毛川

2025 年 4 月 27 日

1 Tokenizer

1.1 算法实现和训练

1.1.1 简述 BPE 算法

Byte Pair Encoding, 基于统计方法合并最常见的字符或子词对，逐步构造更紧凑的词表。具体过程为 1. 将单词拆分成字节级单位（也是一个字符）；2. 统计字符对出现频率；3. 合并出现频率最高的字符对，得到新字符；4. 将合并产生的新字符加入词表；5. 循环以上过程直到词表的大小达到指定大小。

1.1.2 训练 Tokenizer 的流程

训练 Tokenizer 实际上是为了得到一个字典，字典的 key 为 byte 或 byte 组合，value 为 token 编号，具体流程：

- 1. 将输入的文本以字节为单位进行切分，创建初始词汇表，由于每个字节只能表示 0-255，所以词汇表的大小不超过 256。
- 2. 统计每个字符对出现的频率。
- 3. 将出现频率最大的字符对组成一个新的字符 (merge) 将这个字符分配一个新的编号（在简单的 BPE 算法中从 256 开始），然后将训练文本中的所有该字符对替换为生成的字符 (replace)。
- 4. 重复多次 merge 和 replace 操作直到词表达到上限。

依据以上步骤得到的字典是一个双射，在之后 encode 操作中，仅仅需要根据字节查询编号（优先选择可以合并的字节对，从 256 开始向后遍历）；在 decode 过程中只需要根据 Token 的编号查询到对应的字节。

1.1.3 使用《北京大学研究生手册（2023 版）》训练 tokenizer

训练数据在本地，调用 train 函数，得到 vocab 字典。将 bytes 转换为十六进制字符串后保存在 `JSON` 文件中。

图 1: vocab data

1.1.4 对比 encode 再 decode 后的 manual.txt

完全一致

1.1.5 对比 huggingface transformer 中 tokenizer 和自己训练的 tokenizer

- 英文：gpt2 的 Tokenizer 将这段英文 encode 为 186 个 token，而我的 Tokenizer 将这段英文 encode 为 943 个 token。
- 中文：gpt2 的 Tokenizer 将这段中文 encode 为 306 个 token，而我的 Tokenizer 将这段中文 encode 为 118 个 token。

关于 token 数目，发现我们训练的 tokenizer 能更好地 encode 中文，encode 后的长度不足 gpt2 的一半，但是对英文的 encode 长度为 gpt2 的 5 倍。很显然这是因为我们在训练自己的 tokenizer 时使用的语料基本都是中文。而在英文文本的训练较少，我的 tokenizer 多数时候可能“不认识”英语单词，所以将许多单词拆分为一个一个的字母。

针对具体的 token，分别在英文挑选了 4 个有代表性单词和中文语料中挑选了 3 个词语，得到以下结果：表 1, 2

表 1: 测试 encode 中文

words	**博士**	**论文**	**专家**
gpt-2 tokenizer	[39355, 248, 18803]	[164, 106, 118, 23877, 229]	[10310, 241, 22522, 114]
my tokenizer	[409, 366]	[368, 355]	[429, 491]

表 2: 测试 encode 英文词汇

words	**in(IN)**	**OF**	**China**	**MANUAL**
gpt-2 tokenizer	[259]([19238])	[1268]	[14581]	[10725, 25620]
my tokenizer	not merge(not merge)	[635]	b’China’	[77, 604, 442, 76]

GPT-2 阻止把单词和标点等不同类的 character encoder 为一个 Token，对空格特殊处理。而我们的 tokenizer 没有实现。这里通过对比两种 tokenizer 对具体中文和英文词汇的 encode 结果。结论如下：

- 同样是使用 utf-8 将中文转化为字节，然后基于 BPE 算法训练编码器，我的 tokenizer 将词汇编码为更少的 token，而编号的大小也比 gpt2 的 tokenizer 小，原因是我们的训练语料基本都是中文。例如：“文”，utf-8 编码为 `xe6,x96,x87`（十进制为 230,150,135），我们的 tokenizer encode 为 [284, 135]，而词汇“论文”encode 为 [368, 355]，可见对中文的编码效率非常高。
- 由于我们的语料中语料中的英文词汇出现非常少（且众多都是大写），对于一些简单的单词，我们并没有将其中的字母 encode 为一个 token，例如（“in”，“China”），在多个测试的单词中仅仅将“OF”encode 为一个 token，因为在训练文本中“OF”出现的频率比较高（POSTGRADUATE MANUAL OF PEKING UNIVERSITY）。但是 gpt2 的 tokenizer 由于有大量的英语语料的训练，可以将这两个词语分别 encode 为一个 token。在 gpt2 的 tokenizer 中相同单词但是类似“IN”，“OF”这样的全部大写的字母被 encode 为序号较大的 token，可能得原因是它们出现的频率没有小写单词高，在标题等文本中往往单独出现或是一些单词的中缀。
- 另外，我还发现：虽然我的 tokenizer 的对英语语料的编码能力差，但是对与“MANUAL”这个出现频率非常高的单词，在训练 tokenizer 时内部形成了对字符的 merge（AN-604，UA-442）。这可以进一步说明，如果扩大我们的 tokenizer 的 vocab size 并在更多的英语语料上进行训练，就可以对英文单词更高效的 encode。

1.2 回答问题

- 1. `ord()` 查看字符的 Unicode，`chr()` 查看 Unicode 对应的字符。ord(' 北')=21271；ord(' 大')=22823；chr(22823)= 大；chr(27169)= 模；chr(22411)= 型。
- 2. vocab size 大小的影响：size 大时，更多的字符被编码为一个 Token，这样可以减小 encoded 文本的长度，有助于处理长文本同时加速推理。但是弊端为需要足够多的训练数据，储存 vocab 需要较多的空间。size 小时，适用于形式复杂的语言（韩语），但是弊端为失去语义的完整性，Transformer 处理更长的输入需要消耗更多的时间。
- 3. 不能处理非常简单的字符串操作任务的原因：可能是这个字符串中的多个字符被 encode 为一个 token, LLm 只能看到这个 token，看不到具体的细节。例如翻转任务，反转字符串需依赖字符级解码，而模型仅能基于 token 概率生成，缺乏字符级控制。如果 prompt 修改为先逐个打印字符串中的字符，然后用相反的顺序打印就可能正确实现字符串翻转。
- 4. 非英语语言（例如日语）上表现较差的原因：首先与 LLM 的训练数据有一定关系，现有的许多 LLM 使用大量英语语料训练。此外，也与 tokenizer 的训练有关：相比于英语中的单词，韩语中的“chunk”被 tokenizer 被分割为多个 token, ，所以对同样语义的句子，韩语被 encode 后的 token 数量更多，增加了模型的负担。
- 5. LLM 在简单算术问题上表现不好：大于 3 位的数字会被 merge 为一个 Token，LLM 不能直接获取到精确的数字。
- 6. 在编写 Python 代码时遇到比预期更多的困难：有的 Tokenizer （gpt2）将每一个空格作为一个 token，python 中有过多的 space，浪费上下文的空间。为了提高 coding 能力，希望输入更多的上下文，一些 Tokenizer 会压缩空格，但是输入的缩进信息会损失，缩进是 python 中的重要信息。
- 7. 遇到字符串 `<|endoftext|>` 时突然中断的原因：`<|endoftext|>` 是 tokenizer 中的 Special tokens，代表序列的终结。若输入包含此标记，模型可能误判为生成结束。
- 8. 关于“SolidGoldMagikarp”的问题时 LLM 会崩溃：如 SolidgoldMagikarp 等词在在训练集中从未出现，相应的 Neuron 从未激活，在初始化阶段是任意的，从未更新过。所以输入后会产生错误输出。
- 9. 使用 LLM 时应该更倾向于使用 `YAML` 而不是 `JSON`：`YAML` 格式文件的储存密度比 `JSON` 高，相同信息量下，`YAML` 文件体积更小且容错性更强。
- 10. 不是端到端的原因：LLM 处理的原始文本需经 tokenizer 转换为模型可处理的离散 token，而生成的 token 序列需通过相同 tokenizer 解码为自然文本。Tokenizer 通常独立于模型训练（如 BPE 算法），导致上下游割裂。

2 GPT-2 模型构建实验报告

2.1 Commit 记录

2.2 实验目标

实现基于 PyTorch 的 GPT-2 模型训练流程，覆盖模型架构设计、训练数据处理、分布式训练优化等关键环节，验证模型在文本生成任务中的性能。但由于算力限制，最初在 `TinyShakespeare` 上实现小批次的训练，当数据集更换为 `fineweb` 且训练模式改为 8 块 GPU 分布式训练后本地将无法进行合适批次的训练，仅实现相应的代码。实现时间为 1 星期（不考虑报告完成时间约 11 小时），commit 记录见 2 和 3。

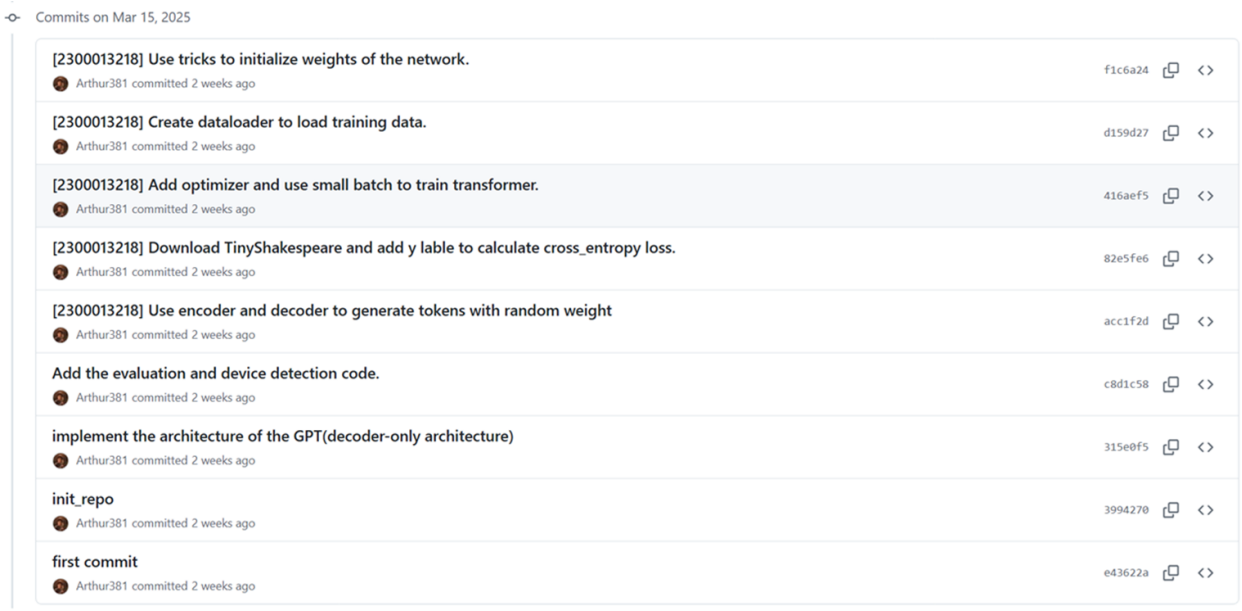


图 2: commit on Mar 15

2.3 框架搭建和小规模训练

首先查看 OpenAI 发布的 GPT-2 的网络参数，参考网络的结构命名自己的网络架构。GPT-2 是 decode-only 的 Model，所以不需要实现 cross-attention。

1. **基础架构实现**

   - 实现 Decoder-only Transformer 结构，包含 12 层多头注意力机制，每一层我们实例化一个 `block` 类。在 12 个 `block` 中，每个 `block` 的结构为：LayerNorm-Self-Attention-LayerNorm-MLP，每个 MLP 的内部结构为：Linear(up-projection)-GeLU-Linear(down-projection)。在 12 个 `block` 之前设置 token embedding 和 position embedding，在 12 个 `block` 之后设置 Language Model Head 来得到每个 token 的对数概率，最终实现 next token generation 的功能。
   - **Evaluation**：直接测试 pretrained GPT-2：使用 Hugging face 的 Transformer，模型的参数最初选择加载 GPT2 原有的参数，在特定的 prompt 下查看模型的输出，虽然是小模型，但是输出有一定的语义信息。
   - **设置 Device**：检测设备如果有 gpu 时将 device 设置为 cuda。

2. **调试和小规模训练**

   - 删除原有的 huggingface 上的 pretrained GPT-2 的参数，以 random 的参数测试模型的输出，同时得到 5 个 batch，理所当然模型输出混乱的 token(garbage)。在模型输出下一个 token 时，挑选出概率最大的 k 个 token，然后随机从中挑选出一个 token，将这个 token 拼接在原有的 prompt 后，继续作为模型输入，直到达到 `max_length` 的长度。
   - **下载和处理训练数据**：从 huggingface 上下载了 `TinyShakespeare` 数据集，这是训练语言模型的一个优质数据集，这里利用了一个 trick 来设计 x 和 y。只需要将字符串 x，向后平移一个单位得到 y（之后 `view` 成 `batch_num`×`sequence_length`），通过交叉熵计算 loss。在之后的训练中即使多个 GPU 分布式训练，扩大了 `batch_num`，仍然使用这种方法设置训练数据。
   - **使用优化器**：采用 AdamW，AdamW 结合了 Adam 优化器和带有权重衰减的 SGD。除了一阶动量、二阶动量和自适应学习率，AdamW 在参数更新时衰减权重，即 $w_{t+1} = w_t + \Delta w_t - \eta\lambda w_t$，可以防止模型过拟合。

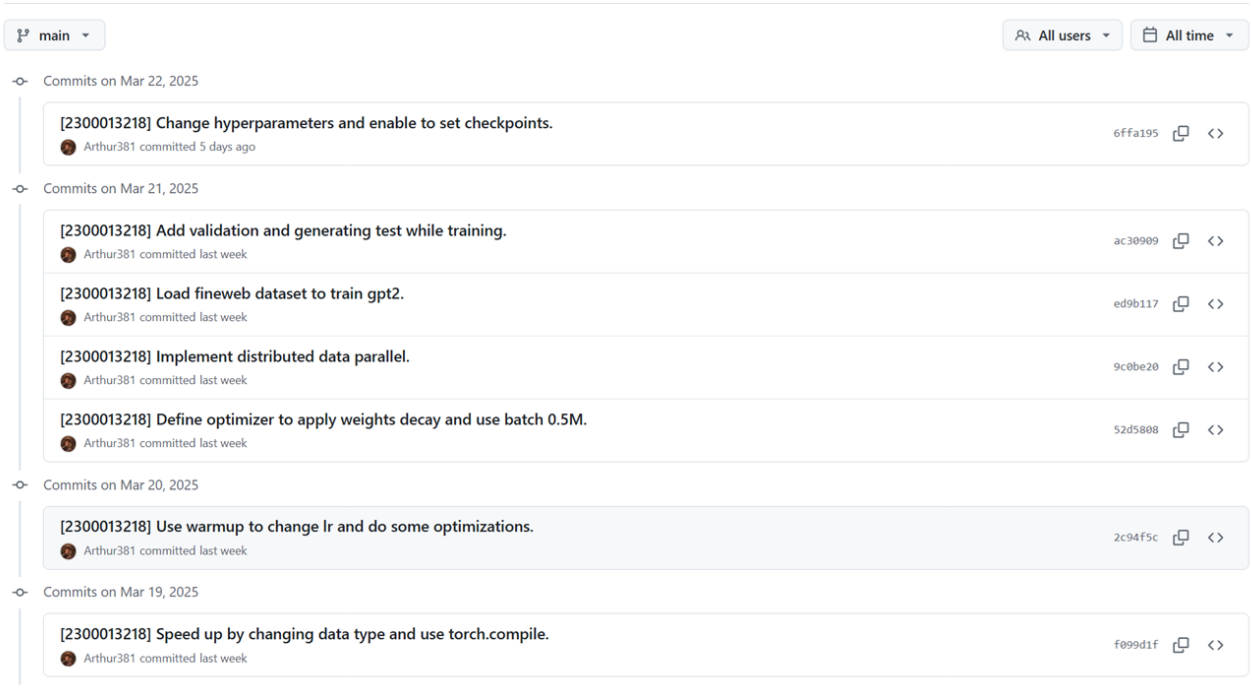
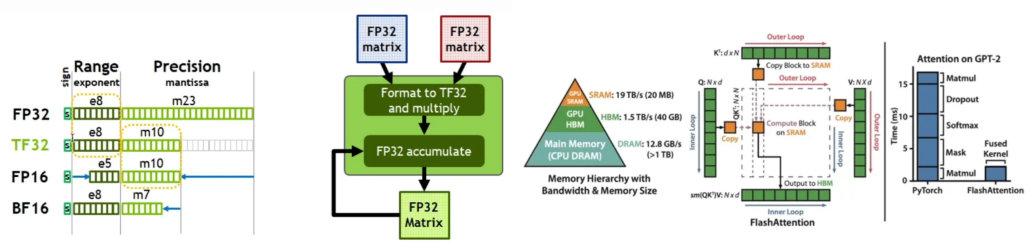


图 3: commit from Mar 19 to Mar 22

3. **数据加载器设计**

- 支持动态批处理，每次从 `current_position` 处取出 $B \times T$ 长度的字符串作为一个 `step` 的训练语料。
- 在加载训练数据时使用 CPU 处理 token 和 text，节约 GPU 的内存资源。



(a) float data type

(b) Flash Attention

图 4: Consider data representation and memory hierarchy

2.4 训练初步优化和加速

1. **初始化权重**

- **共享参数**：token embedding 和 Language Model Head 两个权重矩阵使用相同的参数。理论可行性：语言模型的输入和输出都涉及到词汇表。Embedding 层学习到的表示已经是最优的词表示，将其用于输出层符合对偶性原理。直观上：相同的 token 具有相同的 embedding。embedding 相当于做了查找工作，利用 token 的索引得到嵌入向量，token embedding 和 Language Model Head 的操作有一种互逆的关系。通过将两个权重矩阵共享相同的内存空间，在实现同步训练的同时，大大减少了参数量和内存的消耗（768×50257）。

- **维持稳定方差**：在初始化 GPT-2 网络的权重参数时，需要考虑方差的变化，主要是因为深度神经网络的层数增加时，方差可能会逐层放大或缩小，导致梯度消失或爆炸的问题。由于每个 `block` 的输入为 768 维（embedding 向量），GPT-2 以 $\frac{1}{\sqrt{768}}$ 作为基础标准差，这是 Xavier 初始化的一个变体，保证前向传播时激活值的方差不会随层数增加或减少太多（实验中 $std = 0.02$）。此外，对 `block` 中的层，在原有标准差的基础上再乘 $\frac{1}{\sqrt{2*n\_layer}}$。

2. **训练加速**

- 改变数据类型：使用精度更小的数据类型来完成计算。python 默认的数据类型为 `float32`，这里对于矩阵的乘法修改为 `FT32`。如图 4 (a) 在减少小数位的长度时，FP32 转换到 TF32 只需要做截断操作，且在做矩阵乘法时精度损失很小。
- 使用 `torch.compile` 实现图编译功能，自动优化模型执行，提升推理和训练速度，同时减少内存占用。但是由于显卡的问题，我的电脑无法使用 `torch.compile`
- **Flash Attention**：利用 online softmax 的思想，通过将多个操作（如 $\mathbf{QK}^T$ 计算、Softmax 归一化以及与 $\mathbf{V}$ 的乘法）融合到单个 CUDA 核函数中，有效减少了 kernel 启动开销和中间数据传输，从而更高效地利用 GPU 并行计算能力，特别是在大 batch size 或大模型参数的情况下，能显著提高计算效率。具体原理如图4(b)。
- 调整参数：基于 GPU 的二进制计算特性，将参数调整为 2 的幂次。改变 vocab 的大小，发现在参数量增大的同时未影响 token embedding 和 position embedding 效果，推测在网络末端的分类器上可能会影响效果，但是实际 Language Model Head 上激活的 token 非常少，很多 token 的概率都为 0，所以也可以正确输出。

3. **Batch Normalization**

- 归一化梯度：在某一个 batch 带来很大 loss 的情况下，通过批归一化使参数不被梯度影响过大。但是添加了 Batchnorm 会使模型的训练明显变慢

2.5 进一步优化

初始化参数后能保证一定的训练效果，以上通过若干步骤实现训练加速，目前已经可以进行训练。但是我们尝试通过进一步优化提升训练效果：

- **动态学习率调整 (Learning Rate Schedule)**: 训练初期使用 Warm-up 机制逐步增大学习率，以稳定梯度并提高收敛效果。代码中的设置方式为：

$$\eta_t = \begin{cases} \eta_{\max} \times \frac{\text{iter\_time}+1}{T_{\text{warmup}}}, & \text{if iter\_time} < T_{\text{warmup}} \\ \eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\pi \cdot \frac{\text{iter\_time} - T_{\text{warmup}}}{T_{\max} - T_{\text{warmup}}}\right)\right), & \text{if } T_{\text{warmup}} \leq \text{iter\_time} \leq T_{\max} \\ \eta_{\min}, & \text{if iter\_time} > T_{\max} \end{cases} \quad (1)$$

- **应用 Fused AdamW 优化器与权重衰减策略**：为了高效调整超参数，这里使用融合实现的 AdamW 优化器，并手动定义参数分组，其中 `decay_params` 收集所有二维及以上的权重（如线性层和嵌入矩阵）以应用权重衰减（weight decay），而 `nodecay_params` 主要收集一维参数（如偏置项和 LayerNorm），不进行权重衰减，以避免影响模型稳定性。Fused 优化器通过将多个计算操作融合到同一个 CUDA kernel，减少内存读写与 kernel 启动开销，从而提高 GPU 计算效率，并在大模型训练中显著加速。

- **梯度累积 (Gradient Accumulation) 调整 Batch Size**：由于超参数的设置依赖于 batch size，我们希望 batch size 设为 $0.5M$，但受限于 GPU 计算资源，采用 Gradient Accumulation 以在多个小

batch 之间累积梯度，延迟参数更新。需注意，梯度累积的 Loss 计算实际上是 Loss 的求和，而交叉熵损失函数默认使用 mean($B \times T$) 进行归一化，因此在更新时需考虑累积的 `grad_accu_steps` 数，以保持梯度的正确缩放。

2.6 分布式训练

1. **实现数据并行**

   - 实现 DDP（Distributed Data Parallel），通过参数：`ddp_rank` (进程 ID)，`ddp_local_rank` (本节点 GPU 的 ID)，来控制 8 块 A100，即有 8 个 python 解释器在 read 相同的代码文件，只是他们的 rank 不同。让 master process 来完成 print，check 等额外工作。之前说明为了实现 gpt2 官方训练时的 batch size（0.5M）利用了梯度累积，此处实现在 Dataloader 中分割用于训练的数据，不同处理器就不会使用相同的数据，进而实现 8 块 GPU 在卡间传递梯度，分配一个 batch。但是在数据并行的过程中不明确各个 GPU 运行的 batch 在训练语料中的先后次序。
   - 梯度同步策略：更新梯度使用的 Loss 需要对 batch size 作平均，所以这一使用了一些 tricks 来实现多 GPU 数据并行对 loss 的处理，即 all-reduce 异步通信。利用 `require_backward_grad_sync` 函数实现只有在累积的最后一步才同步梯度，减少 GPU 之间的通信开销。原因是：如果在 PyTorch 中调用 `loss.backward()` 多次后再调用 `optimizer.step()` 会导致梯度在 `model.parameters()` 中累积多次，从而影响优化过程。

2. **加载 `fineweb` 数据集**

   - 由于 GPT-3 的数据集未公布，这里采用 FineWeb 数据集。FineWeb 是一个高质量的网络文本数据集，其内容具有很高的教育价值。该数据集的质量评估标准是由大语言模型判定的，确保了数据的教育性和可靠性。

2.7 验证与测试

1. 区分训练阶段和评估阶段

   - 在训练阶段开始时调用 `model.train()`，将模型设置为训练模式。在此模式下：BatchNorm 层使用当前 batch 的统计信息（均值和方差）进行归一化；Dropout 层会按照设定的概率随机丢弃神经元，起到正则化作用。
   - 在评估/推理阶段开始时调用 `model.eval()`，将模型设置为评估模式。在此模式下：BatchNorm 层使用训练阶段积累的全局统计信息（运行均值和方差）；Dropout 层会保留所有神经元，不进行随机丢弃。
   - 这种区分确保了模型在训练和推理时的行为一致性，并保证评估结果的可靠性

2. **Evaluation**

   - **评估验证集 loss**：每 100 步输出验证集的 loss，查看模型是否过拟合。
   - **储存 checkpoint**：每 5000 步在 TinyShakespeare 数据集计算困惑度（Perplexity）
   - 每 250 步令 master process 推理并输出，prompt 设置为"I'm a large language model"。发现随着训练轮数的增加，模型的输出语义性更强。

3. **调整超参数**：根据文献中的超参数设计 `warmup steps` 和 `max_steps`。

   - 结合 GPT-3 的参数，一个 epoch 需要训练 10B tokens；batch size 设置为 2^{19}(524288，约为 0.5 M)。

- 由于有公式 `warmup_steps=total_tokens/batch_size`，此时设置为 $375M/2^{19} = 715$。
- 实验中训练了 4 个 epoch，每个 epoch 的数据量选取为：`max_steps*total_batch_size=10B`，因此可以反推出 `max_steps=19073`。

4. **mask 处理**：对 context + option 进行编码，并构造出 mask 矩阵来标记 context 和 option 的区别。每个 (context, option_i) 对单独编码和评分，确保模型对每个选项的评估仅依赖 context 而非其他选项。Mask 可强制模型学习 context 到正确 option 的映射，避免因输入结构差异引入偏差。
5. 后续可以继续使用 `llm.c` 对训练进行加速。

2.8 实验分析

在这个实验中仅仅使用了少量数据模型在一些简单的测试集上的正确率已经超过了 GPT-2，以下是可能的原因：

1. **使用高质量数据**：GPT-2 使用多领域混合数据（含代码、数学等），但未严格过滤低质量内容。而我们使用的 FineWeb 经过精细清洗，保留高教育价值的文本（如学术、技术文档），数据信噪比显著提升。
2. **训练效率优化**：仅需 100 亿 token（1 个 epoch）即可超越 GPT-2，说明：
   - 数据质量 > 数据数量：优质文本可高效传递语言规律。
   - 模型未过拟合：单 epoch 训练验证了学习过程的稳定性。
3. **算法与工程改进**：
   - 现代训练技术（如更好的优化器、初始化策略）弥补了数据量差距。
   - 硬件效率提升（如 FlashAttention）使小数据训练也能充分挖掘模型潜力。
4. GPT-2 的通用性导致其在特定任务（如文本理解）上分散了能力，而我们的模型只能完成特定任务。

3 LoRA

3.1 LoRA 策略与步骤

在 LoRA 出现之前，常见的方法如**全参数微调（Full Fine-tuning）**，需要对整个预训练模型的权重进行更新。这导致了巨大的参数开销，每当更换下游任务时，都需要重新训练并保存一套完整的参数，计算和存储代价极高。

为了缓解这一问题，一些工作尝试**在神经网络中插入新的适配层（Adapter Layers）**，通过引入小规模的可训练参数来适应新任务。然而，这种方法往往需要**增加模型深度**，在推理时引入额外的延迟，尤其对于需要模型分片并行的场景，推理效率问题更加突出。此外，还有一些工作探索了**迁移学习**方向，如优化输入层的激活形式或插入新的轻量模块，以减少微调开销。

在此背景下，有研究提出：*模型在适应新任务时，权重变化本身具有较低的“内在秩”（intrinsic rank）*。这意味着，只需要对权重矩阵做低秩调整，就能有效地完成任务适应，而无需更新大量参数。

LoRA（Low-Rank Adaptation）正是基于这一假设提出的。其核心优势在于：

- 预训练模型参数可以**完全冻结**，不同任务只需引入小型的 LoRA 模块（即低秩矩阵 A 和 B）。
- 多任务之间可以**共享同一个预训练模型**，仅需替换对应任务的 LoRA 模块即可，显著降低存储开销和任务切换成本。
- 通过低秩分解有效减少了可训练参数数量，兼顾了效率与性能。

3.2 运行源代码

- 助教提供的原始代码在 Colab 上使用自带的 50 条数据时可以正常训练，但当增大数据量时，出现了无法找到合适 Batchsize 的问题。原因在于代码中使用了 `auto_find_batch_size=True`，该参数会在显存不足时不断减小 Batchsize，从而导致训练过程不稳定。
  鉴于算力有限（本地笔记本 4060 GPU，租用 3090 GPU），我将训练数据量修改为 1024 条（原数据集为 2999 条），在保证一定训练效果的同时，能在有限时间内尝试不同的超参数配置。为了进一步节省显存并使训练更稳定，我采用了每次前向传播 Batchsize 为 1，每累计 8 次梯度后进行一次反向传播的设置。

- 为了方便助教查看，本次实验的基础运行结果（不要求复现）分别保存在 `lora_imdb_review.ipynb` 和 `Lora_alpaca.ipynb` 文件中。Loss 曲线与模型微调完成后的测试生成结果均可在对应文件中查阅，本文也对关键结果进行了总结。

- Loss 曲线整体呈下降趋势，表明训练有效，具体数据见图 5。

152	3.939000
160	3.696600
168	3.738100
176	3.683500
184	3.644100
192	3.681600
200	3.673400
208	3.757500
216	3.594500
224	3.672900
232	3.733500
240	3.591400
248	3.654000
256	3.675200

(a) IMDB 数据集上的微调结果

152	1.558600
160	1.447700
168	1.577300
176	1.565400
184	1.618900
192	1.502300
200	1.532400
208	1.587700
216	1.635100
224	1.515700
232	1.517300
240	1.491300
248	1.395200
256	1.509300

(b) Alpaca 数据集上的微调结果

图 5: 提交文件中保留了两个数据集下，使用默认超参数（$r = 4, \alpha = 1$）的微调结果示例

- 原代码中提供了一个测试 prompt：”I love this movie because”，下列是两个模型在该输入下的输出示例：

  `IMDB Review 3000` 微调后模型输出：
  I love this movie because it is so good. I have seen the first two movies and loved them, but now I'm going to see The Last Ship of Men again!
  The last time we saw a film about an underwater world was in 2009 when they were making their debut on TV with ”The Greatest Show Ever” (the one that aired at 9pm).

  可以看到，经过大量**影评数据**微调后，模型输出中记住了不少影评中的细节内容，但句式连贯性与 Foundation Model 相比有所下降。后续章节中，我们将通过调整超参数，尝试获得更好的生成效果。

`Alpaca` 微调后模型输出：

I love this movie because it is a story of passion, adventure and determination. It has an unexpected ending that leaves you wanting more than ever to know what will happen next or even better yet?" The film was nominated for the Academy Award in both categories. It received positive reviews from critics who loved its climax as well like others have already done before.

使用 Alpaca 微调的目的是进行**指令微调（Instruction Fine-tuning）**，后续章节在设置超参数时，将进一步分析该模型在 Instruction Prompt 下的表现。

3.3 实验设计

3.3.1 设计思路

由于 LoRA 的实现原理中：

$$W' = W + \Delta W = W + \frac{\alpha}{r} BA$$

其中 $B \in \mathbb{R}^{\text{indim} \times r}, A \in \mathbb{R}^{r \times \text{outdim}}$，所以实验中主要探究 r 和 α。对两个不同的数据集，首先固定 $\alpha = 1$，尝试不同的 r，综合 loss 和模型在相同 prompt 下的生成结果，得到针对特定数据集最优的 r，然后固定 r，尝试不同的 α，就可以得到最优的 r, α 组合。

3.3.2 我的代码框架

1. 根据原有代码实现了 `exp_lora.py` 文件进行批量训练，有以下一个特点：
   - 通过调整 Batchsize 和梯度积累，保证训练稳定性的同时修复显存占用高的问题。
   - 处理 Alpaca 数据集中的数据，使用 tokenizer encode 并设置 label。
   - 使用 **tensorboard** 实时记录 loss 曲线，同时可以在不同 setting 下更加直观地比较参数的影响。
   - 更改储存 tensor event 文件和 checkpoint 的路径，通过命令行读取参数，交互更加便捷。
   - 在每次训练后分别使用普通 prompt 和一个 instruction prompt 简要评估生成结果。
2. 实现了 `run_exp.py`：在无监控情况下通过命令行自动运行 `exp_lora.py`，尝试不同参数并保存模型输出。
3. 实现了 `test_lora.py`：设计了若干不同类别的 prompt 对训练好的 checkpoint 评估生成结果。

3.3.3 确定基础超参数

在实验展开前，我认为源代码中的 Learning rate（3e-2）和 epoch num（2）已经可以达到一定的效果，又尝试了 Batchsize=16 和 Batchsize=8（实际上是改变梯度积累的次数）。在 imdb review 3000 数据集中的 loss 曲线如图 6。

每次训练 steps 的数量为 $\frac{1024 \times 2}{\text{batchsize}}$，通过 tensorboard 每 8 个 Step 记录一次 Loss，发现 Batchsize=8 和 Batchsize=16 时 loss 收敛的值接近。由于训练速度相同，为了使 loss 曲线更加精确，以下的实验设置中都采用 Batchsize=8。

3.3.4 Prompt 设计

在每次训练后使用以下两个 prompt 测试模型的生成结果。

- Without instruction: *"I love this movie because".*

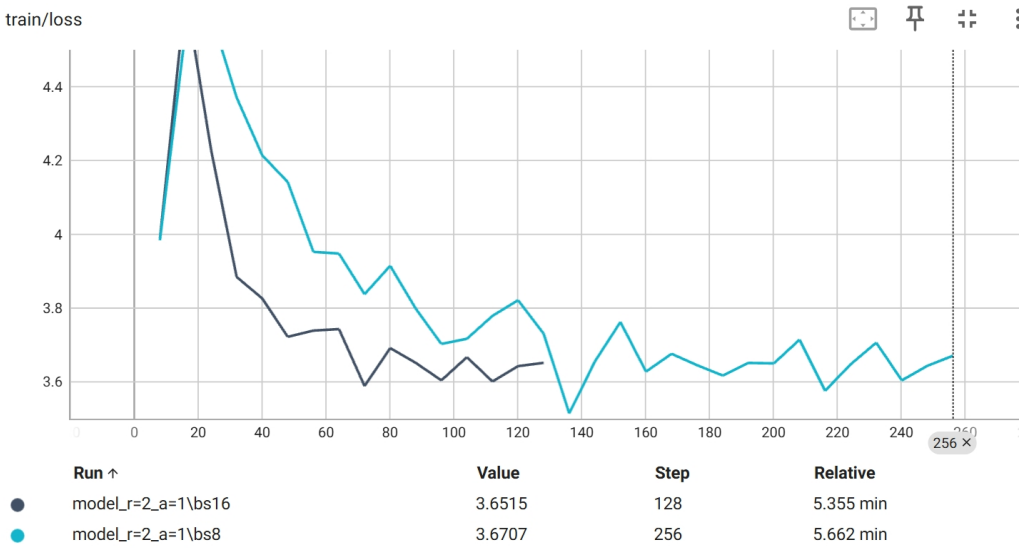


图 6: 尝试不同 Batchsize

- With instruction: *"Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request. Instruction: Read the movie review provided below and summarize the author's opinion in one sentence. Input: The movie had a stunning visual style and a compelling performance from the lead actor, but the plot felt disjointed and the pacing was slow. Response:".*

由于篇幅限制，我们在之后通过生成结果分析超参数的优劣时：对于 `imdb review 3000` 仅比较 without instruction，针对 `Alpaca` 仅使用 with instruction。

3.4 Dataset *imdb_review_3000*

3.4.1 Lora 超参数 r

分别尝试了 r=1,2,4,8,16,32 。Loss 曲线如图 7

在 Loss 曲线中，不同 r 取值对应的 Loss 收敛情况非常接近，甚至曲线的震荡幅度也较为相似。我们推测，原因可能是微调数据集中部分 sequences 中的 token 较难预测，而部分 token 则较容易预测，因此带来了小幅度且与 r 关系不大的波动。

然而，我们观察到在 $r = 1$ 和 $r = 32$ 时，训练初期 Loss 的震荡幅度明显较大。进一步分析可得：

- 当 r 过小时（如 $r = 1$），LoRA 矩阵的学习能力受限，导致模型在训练过程中容易忘记已学到的知识，表现出不稳定的震荡；
- 当 r 过大时（如 $r = 32$），加到 Foundation Model 权重矩阵上的扰动 ΔW 较大，容易破坏原模型的参数结构，导致输出不稳定。

由于最终 Loss 水平接近，我们进一步通过生成结果（without instruction）来比较不同 r 的效果，最终选择了表现最优的 $r = 4$。在该设置下，模型生成的文本没有明显的语义断裂，语言自然连贯，主题清晰完整，且未出现无关概念的引入。具体生成示例见表 3。

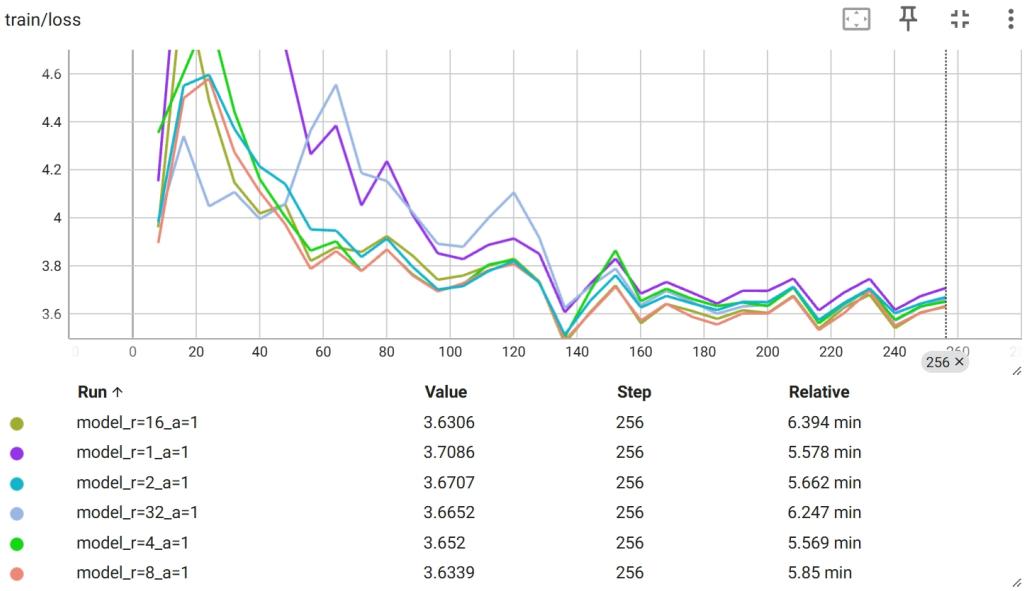


图 7: imdb $\alpha = 1$ 时改变 r

进一步分析模型在相同 Prompt 下的输出特征，我们尝试解释为何 $r = 4$ 表现最佳。当 r 较大（如 $r = 16$ 或 $r = 32$）时，LoRA 训练得到的矩阵具有更强的“记忆能力”，在生成中引入了大量无关的细节信息。这些细节可能源自训练语料，但在实际回答中反而导致了内容分散、语义连贯性下降，破坏了 Foundation Model 原本的整体表达能力。

表 3: imdb 在 $\alpha = 1$ 时不同 r 取值下的输出

r 取值	**Generation**
r=1	I love this movie because it is a very good story. I think the main character, who has been in jail for years and never got out of his cell was really nice to me as well! The film also had some great acting from both actors (the first one being my favorite actor) but most importantly they did their best work together making sure that everyone felt like parting company with each other!! The ending made you feel so happy about what happened on day two!!! It makes your heart sing!!!!
r=2	I love this movie because it is so good. I have seen the other films and they are all great, but these two movies make me want to watch them again! This film was a little bit different from most of my favorite horror flicks; it's not like you see some old pictures that were just made for TV or something...this one has more than enough gore in its own right (and that's why I'm glad).
r=4	**I love this movie because it is so well made and the acting. I have to say that, in my opinion (and of course everyone else) it's one helluva film! The story line was very interesting as were all characters except for a few minor ones which are just**
r=8	I love this movie because it is so good. I have seen a lot of movies that are bad, but not these two films! The acting was great and the storyline really makes you want to see more than just one person in life (and that's what made me like them). The music plays well too - it's very jazzy with lots going on between songs which make your ears ring as they sing along...

r 取值	Generation
r=16	I love this movie because it is so funny and I think it's the best film of all time. It was a great idea to make an action comedy about two guys who are in their 20's, but they have no money or any other means for living anymore than by working as prostitutes at night bars (they're not even married).
r=32	I love this movie because it is so good. I have seen the film many times and loved every time! The acting was excellent, especially in a very bad situation where you are trying to save your life from an evil man who has been stalking her for years (and she doesn't even know what he does).

3.4.2 Lora 超参数 α

在固定 $r = 4$ 后尝试找到最好的超参数 α，训练的 loss 曲线如图 8，模型的输出见表 4

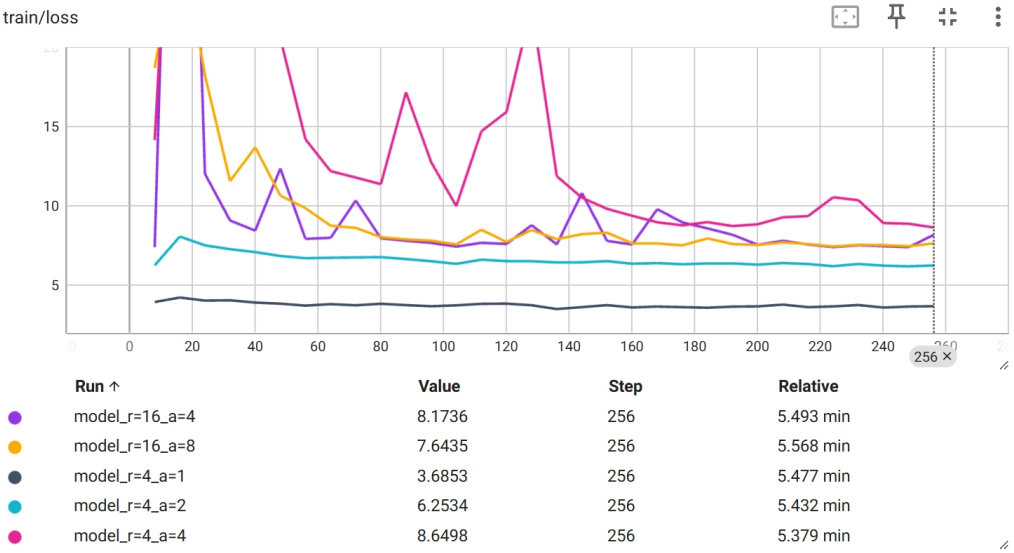


图 8: imdb r=4 和 16 时改变 a

表 4: imdb 在 r=4 时不同 α 取值下的模型输出结果

α 取值	Generation
a=1	**I love this movie because it is so well made and the acting. I have to say that, in my opinion (and of course everyone else) it's one helluva film! The story line was very interesting as were all characters except for a few minor ones which are just**
a=2	I love this movie because I was a good. The film is the first, but it would have been to be not just really like that you think and my lot of your story in their time for an real films!
a=4	I love this movie because. akan, the it- is a in to an and I but The or that one some mere((who there after he even was too his her very-of their with when ará adar/ it's ilé of story... are both game Pember

结合 Loss 曲线和表格可以进一步分析：当 r 较小时，增加缩放因子 α 会引起 Loss 的迅速上升，说明此时模型的缩放稳定性较差。具体而言，在最优超参数 $r = 4$ 的条件下，$\alpha = 1$ 时微调效果最佳，收敛

速度较快且最终 Loss 最低，而 $r = 16$ 时比较稳定。

推测原因如下：对于参数量较小的模型，由于单个 Query-Key-Value 对应的 LoRA 矩阵规模受限，无法充分学习到足够有组织的知识，反而可能学习到较为杂乱的信息。在这种情况下，若进一步增大 α，即增大 ΔW 对原权重矩阵 W 的扰动幅度，会导致模型原有结构被破坏，使得连简单 token 的预测也变得困难，最终影响微调效果，导致 Loss 明显上升。

3.4.3 其他超参数

接下来，我尝试修改了其他训练过程中的超参数，结果如图 9 所示。其中，在增大训练数据量（至 2048 条样本）后，也就是让模型见到更多的“题目”，最终的 loss 收敛程度与原有设置相比差异不大。我认为这可能是由于在 LoRA 参数、基础模型和数据集确定后，语言模型在自回归微调任务上已经接近其性能下界（约为 3.5）。

此外，我还尝试了减小学习率（Learning rate），发现在训练初期 loss 更加稳定，且在相同步数下显著低于 base model。如果增加训练的迭代次数，较小的学习率可能会进一步提升模型性能。

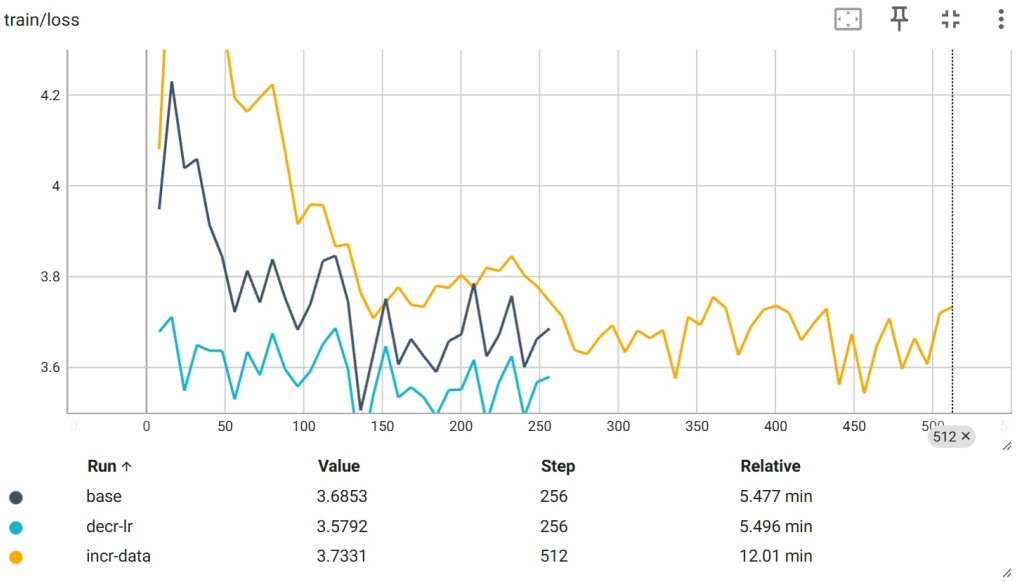


图 9: imdb a=1 时改变 r

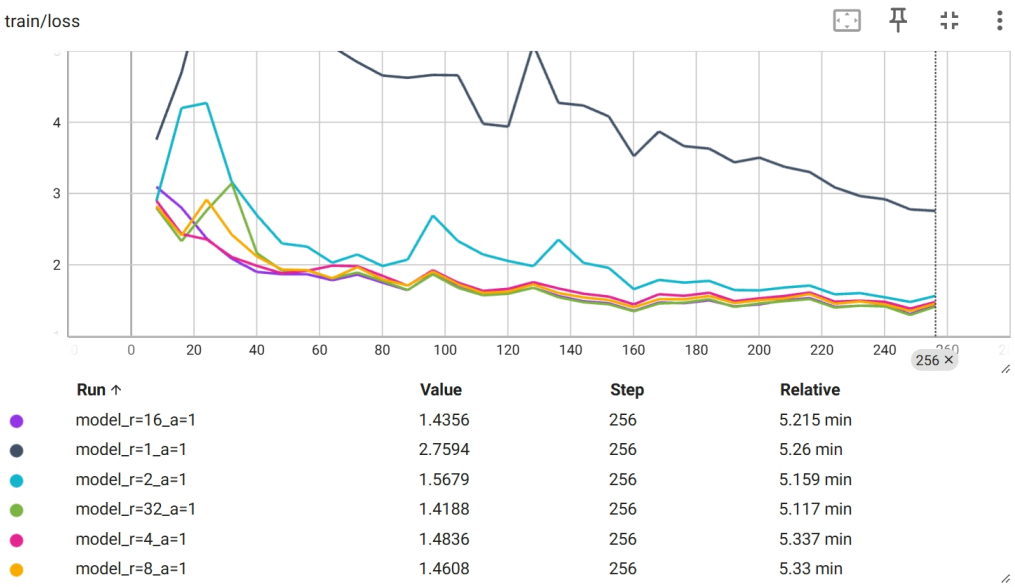
3.5 Dataset *alpaca*: Instruction fine-tuning

3.5.1 数据集的处理方式

查看 `tatsu-lab/alpaca` 数据集可知，每条数据包含 `instruction`、`input`、`output` 和 `text` 四个字段。其中，`text` 是一个固定模板下，将 prompt、instruction、input 和 output 拼接而成的完整字符串。在训练模型时，我们将 `text` 作为输入，通过 tokenizer 进行编码，并手动设置对应的 label。最初的尝试中，训练时 loss 无法收敛，并导致了梯度爆炸。经过分析发现：由于我们的训练目标仅是让模型正确生成 `Response` 部分（即 output 部分），在 `Response` 之前的内容（prompt、instruction 和 input）不应计入 loss。因此，我们将 `Response` 之前对应的 tokens 的 label 设置为 `-100`，以忽略这些部分对 loss 的贡献，从而使模型训练正常收敛。

3.5.2 Lora 超参数 r

与上一部分实验步骤相同：我们这里先固定 $\alpha = 1$，改变 r，loss 曲线如图 10，模型在 Instruction prompt 下的生成内容如表 5



Run	Value	Step	Relative
model_r=16_a=1	1.4356	256	5.215 min
model_r=1_a=1	2.7594	256	5.26 min
model_r=2_a=1	1.5679	256	5.159 min
model_r=32_a=1	1.4188	256	5.117 min
model_r=4_a=1	1.4836	256	5.337 min
model_r=8_a=1	1.4608	256	5.33 min

图 10: alpaca $\alpha = 1$ 时改变 r

表 5: alpaca 在 $\alpha = 1$ 时不同 r 取值下的输出

r 取值	**Generation**
r=1	- What does it feel good to be fun when you are excited about what he loves her favorite food or meal time at dinner. Foods were cooked for lunchtime meals on their home every day of your breakfast buffet before eating out during its delicious foods as wellness. I have been given by my mom's birthday party?
r=2	This film has been described as being both entertainingly complex yet engaging nonetheless! It also features some of my favorite characters - especially Tom Hanks who plays this role well; it will be interesting to see how they work together on their roles throughout its development process while still having fun moments at each stage.
r=4	This film has been described as "thriller" by critics who have found it to be soothingly entertaining; however its performances were not stellar it lacked; there are few moments of action or excitement and itś all about finding your way out!
r=8	I loved this film because it made me feel like I could never get enough of its characters' storyline or their actions without ending up being bored.
r=16	**I loved this film! It made me feel like I could watch it every minute of my life - even when there were no other options to get around or explore what else might be on screen than watching movies?**
r=32	This film has some of its own quirky moments as it struggles to keep up on schedule despite being set at random locations around Earth s most remote planet - yet this lackluster cast makes for interesting viewing only when you look closely into their eyes or even if they are not present during your visionary moment!

通过分析 loss 曲线和模型生成结果，在给定 $\alpha = 1$ 的情况下，LoRA 的最优超参数为 $r = 16$。此时，loss 曲线下降相对更加平稳，且模型生成的结果最符合 instruction 的要求（尽管尚未完全符合），语义连贯性也相对较高。

相反，当 $r = 1$ 时，模型基本未能理解 instruction 的含义，出现了答非所问的现象；当 $r = 2, 4$ 时，模

型已经注意到输出内容应与电影相关（Instruction 要求总结影评），但整体连贯性和相关性仍有欠缺——这一点并不容易做到，因为 prompt 中混杂了较多其他信息，容易干扰生成。

当 $r = 16$ 时，模型能够较好理解 instruction 的意图，并围绕影评进行合理的总结；而在 $r = 32$ 时，模型输出中出现了无关的语句，这可能是由于 LoRA 的秩过高，导致训练得到的增量矩阵具有过强的“记忆能力”，从而引入了过多与 instruction 无关的细节信息，反而干扰了主任务的生成。

3.5.3 Lora 超参数 α

此后，我们固定 $r = 16$，尝试不同的 α 取值，结果发现只有在 $\alpha = 1$ 时，模型才能更有效地降低 loss，如图 11 所示。

正如前文所述：当 LoRA 矩阵的秩较高时，缩放系数 α 的变化对模型训练的影响较小。这表明，当 r 较大时，LoRA 矩阵具备了学习和编码一定量语义信息的能力，缩放不会显著破坏其内部结构，因为这种语义信息是相对稳定和鲁棒的。

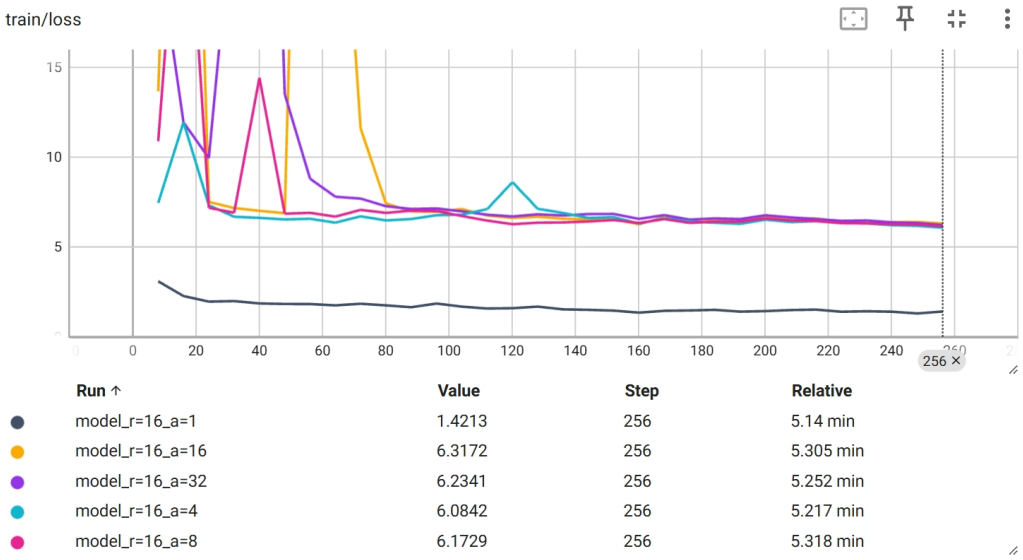


图 11: alpaca $r = 16$ 时改变 α

我们可以认为在 Alpaca 数据集下，最优超参数为 $r = 16, \alpha = 1$。

3.6 探究超参数：`LoraConfig.target_modules`

	Weight Type	$r = 1$	$r = 2$	$r = 4$	$r = 8$	$r = 64$
WikiSQL(±0.5%)	W_q	68.8	69.6	70.5	70.4	70.0
	W_q, W_v	73.4	73.3	73.7	73.8	73.5
	W_q, W_k, W_v, W_o	74.1	73.7	74.0	74.0	73.9
MultiNLI (±0.1%)	W_q	90.7	90.9	91.1	90.7	90.7
	W_q, W_v	91.3	91.4	91.3	91.6	91.4
	W_q, W_k, W_v, W_o	91.2	91.7	91.7	91.5	91.4

图 12: 原论文中的参数 r 与权重矩阵类别的关系

在 LoRA 的原论文中，r 是与微调时的 Weight Type 有关的，如图 12。由于算力有限，我们在最优的 r 和 α 下，尝试找到最优的 Weight Type。即进一步探究了不同 `target_modules` 设置对微调效果的影响，loss 曲线如图 13 所示。

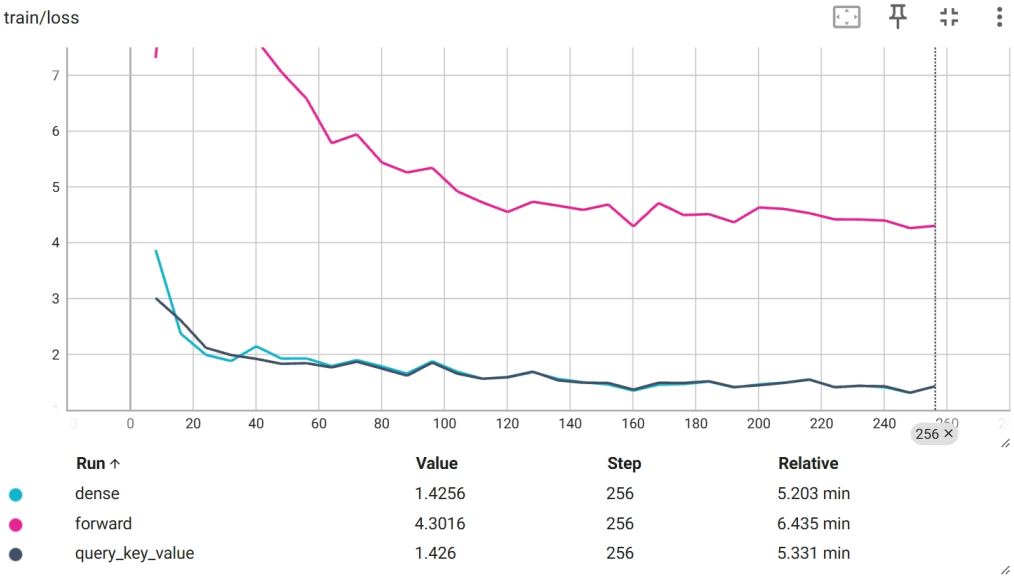


图 13: 权重矩阵类别对训练的影响

可以看到，当微调 QKV 或 dense 模块时，loss 曲线下降较快，且最终收敛至较低的水平；而当微调 MLP Linear 层时，loss 保持在较高的位置，训练效果较差。这一现象可以通过各模块在 Transformer 中的功能差异来解释：

- `query-key-value`：生成 Query、Key、Value 向量，计算注意力，微调它可以直接优化模型对输入信息的关注方式，显著提升任务完成度。
- `dense`：Attention 或 MLP 层的普通全连接层。微调时可以有效调整特征映射。
- `dense_h_to_4h` 和 `dense_4h_to_h`：MLP 的第一个线性层和第二个线性层，主要承担 MLP 内部的非线性扩展与压缩，微调这些层对下游任务的表征优化作用较弱，且可能引入噪声，导致 loss 上升。

因此，在有限参数量下，选择关注 Attention 层相关的模块进行 LoRA 微调，是更加高效且稳定的策略。

3.7 对比超参数

在之前的实验结果中，我们发现 `imdb-review-3000` 的最优超参数为 $r = 4, \alpha = 1$，而 Alpaca 数据集的最优超参数为 $r = 16, \alpha = 1$。在忽略数据集处理方式上的细微差异后，我认为 Alpaca 数据集最优超参数 r 更大的原因，主要在于 Instruction Fine-tuning 的任务特点。Instruction Fine-tuning 不仅要求模型理解输入指令，还需要在此基础上生成符合要求的内容，相比于简单的分类或续写任务，对模型学习能力的要求更高。因此，需要更大的 r 来增强 LoRA 模块的表示和记忆能力，从而捕捉更复杂、更抽象的指令-输出对应关系。

另一方面，`imdb-review-3000` 数据集的任务相对简单，主要是延续已有的语义或模仿影评风格进行生成，知识结构更集中，模式也更加固定。在这种情况下，较小的 r 已经能够满足模型的微调需求，过大的 r 反而可能引入不必要的噪声，影响微调效果。

综上，可以初步总结：任务复杂度越高、输出模式越丰富的微调目标，通常需要设置更大的 r，以提升微调过程中 LoRA 模块的表达能力；而对于模式较为单一、输出相对固定的任务，则适合采用较小的 r，以确保稳定、快速的收敛。

3.8 测试

3.8.1 加大训练规模

在确定好模型分别在 imdb 和 alpaca 数据集上的最优超参数后，我们可以利用这些超参数进行一次更大规模的训练：将训练的数据量增加到：

- `imdb-review-3000`：sample num=2048，epoch num=2，$r = 4, \alpha = 1$
- `Alpaca`：sample num=4096，epoch num=2，$r = 16, \alpha = 1$

得到更优的 checkpoint。由于训练时的 Loss 曲线与之前小数据训练时的 loss 曲线接近，这里就不再列举出来，仅仅评测模型的生成结果。

3.8.2 设计测试输入

训练输入分为四种：共 10 条

- 与电影有关的补全任务
  1. "I love this movie because",
  2. "A slow-burn thriller that",
  3. "The plot was boring, but"
- 普通的补全任务
  1. "Here are three tips for a good meal: 1 fresh ingredients. 2",
  2. "Write a poem about",
  3. "Pretend you are an alien visiting Earth. You would"
- Instruction prompt：最开始有一个统一的 prompt："Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request." 之后的内容 1,2 与电影有关，3,4 与电影无关。
  1. "Instruction: Read the movie review provided below and summarize the author's opinion in one sentence. Input: The movie had a stunning visual style and a compelling performance from the lead actor, but the plot felt disjointed and the pacing was slow. Response:",
  2. "Instruction:Write a comprehensive critique of the following movie review. Your response should include: an analysis of the review's strengths and weaknesses, a discussion of the reviewer's perspective, and a reflection on whether you agree or disagree with their assessment and why. Input: While the movie attempts to tackle complex philosophical themes, its heavy-handed symbolism and sluggish pacing often distract from the emotional impact. The lead actress delivers a powerful performance, but the dialogue sometimes feels unnatural, and the final twist is more confusing than clever. Response:",
  3. "Instruction:Given the list of tasks below, generate a prioritized to-do list for someone working from home. Input: - Reply to emails- Complete project report- Attend team meeting at 3 PM- Take a 30-minute walk- Review budget for Q2. Response:",
  4. "Instruction:Translate the following English sentence into French. Input:The sun is shining and the birds are singing. Response:"

3.8.3 测试结果

使用 3 个模型（Foundation Model，IMDB FT Model，Alpaca IFT Model），在如上的 10 个不同的 prompt 上生成结果见表 6。

表 6: 最终测试结果

编号	Foundation Model	IMDB FT	Alpaca IFT
I love this movie because	it is so funny and I am sure that my friends will enjoy too.	**it is so funny and I am a big fan of the characters. It was very well made, even though there were some scenes that didn't make me laugh at all!**	**it is so good! I am a fan of the film and will be returning to see more.**
A slow-burn thriller that	**is a must-see for fans of the genre.**	**is a little too much for the average viewer. The plot of this film was very interesting and I liked it!**	follows the The Dark Knight.
The plot was boring, but	the characters were not interesting.	**the characters were interesting. I liked it for being a bit more realistic than most of my other books and this one is not that bad at all.**	it akan terasa agak membing.
Here are three tips for a good meal: 1 fresh ingredients. 2	healthy options to eat, and 3 ways you can make it.	**healthy options, and 3 recipes to try out!**	-
Write a poem about	the weather.	the death of my father. I was born in England, and lived for many years on this island where we were all very happy to be alive.	the writer's writing in an essay on "self-criticism" by William H. Lawrence (1853–1954) was one of those writers who have been most influential and well-known for their work
Pretend you are an alien visiting Earth. You would	be able to see the planet and its inhabitants	**be a robot, but your body is human and can move around in the world.**	**be in the middle of it.**

Prompt	Foundation Model	IMDB FT	Alpaca IFT
Read the movie review provided below and summarize the author's opinion in one sentence. Input: The movie had a stunning visual style and a compelling performance from the lead actor, but the plot felt disjointed and the pacing was slow.	I don't know what to say about this film...	**I have to say this film has some very interesting things about it...but I don't think it's worth my time or money.**	This film did not have any of these features or perform to its target audience's needs as it lacked: well-design work by Viple made this feel unrealistic at best
Write a critique of the following movie review. Input: While the movie attempts to tackle complex philosophical themes, its heavy-handed symbolism and sluggish pacing often distract from the emotional impact. The lead actress delivers a powerful performance, but the dialogue sometimes feels unnatural, and the final twist is more confusing than clever.	The film was very well made in terms.	**The film has some very strong points... But it lacks emotion in this regard. I think it's worth noting for those who like movies about politics (and don't care if they're liberal)**	The positive feedbacks by your attitude toward this film was expressed in our reviews at first glance as it has been criticized for being too negative (the view).
Given the list of tasks below, generate a prioritized to-do list for someone working from home. Input: Reply to emails; Complete project report; Attend team meeting at 3 PM; Take a 30-minute walk; Review budget for Q2	**Complete Project Report**	**Complete Project Report - complete this**	This would be your weekly Q&A group - time-tested by 2 teams.

Prompt	Foundation Model	IMDB FT	Alpaca IFT
Translate the following English sentence into French. Input: The sun is shining and the birds are singing.	Voici une image de la lune et du soleil qui se brisent sur le ciel bleu clair que nous voyons depuis notre fenêtre.	A bird has sung to us.	**Il s'agit d'une question de culture et le soleil est en train d'être lu par les oiseaux qui chantent dans la nature.**

3.8.4 实验结论

从最终测试结果可以看出：

- **IMDB 微调效果**：对于与电影相关的 prompt，IMDB 微调后的模型在理解和总结影评内容上有明显提升，如在 summarization 和 critique 类任务中可以生成较为合理、连贯的答案。但是，由于训练数据主要限于影评文本，对于指令执行（如生成 to-do list、翻译任务等）能力不足，导致在非影评指令任务上输出偏离或胡言乱语。
- **Alpaca 微调效果**：经过 Alpaca 指令微调后的模型，在泛化指令（Instruction Following）能力上有明显提升，在任务型输入（如生成待办事项列表、角色扮演等）中表现更为合理。但在专门针对电影相关内容的理解和总结上，比 IMDB 微调模型稍弱，输出内容有时较浅显或不准确。

总体而言，尽管由于测试时 prompt 较复杂，加之模型规模相对较小，生成质量存在一定不足。我们仍然可以发现：专项微调（如 IMDB）可以有效提升模型在特定领域（影评）内的表现；指令微调（如 Alpaca）可以提升模型在广泛任务中的指令遵循和泛化能力。

随后，我从两个数据集中分别随机选择 512 条数据，测试 3 个不同模型在各自任务上的 Loss，结果如表 7 所示。

表 7: 最终模型的 Loss

数据集	Foundation Model	IMDB Fine-tuned	Alpaca IFT
imdb-review-3000	3.8278	3.6449	4.8368
Alpaca	4.6453	4.6256	2.1411

从表 7 可以看出，LoRA Fine-tuning 在两个数据集上都带来了直观的效果提升。尤其是在 Alpaca 数据集上，经过 Instruction Fine-tuning（IFT）后的模型相比 Foundation Model 有明显更低的 Loss。

这一结果可能归因于：Alpaca 数据集整体任务难度较低，但初始模型对指令（Instruction）理解有限。经过专门的指令微调后，模型在指令跟随能力上得到了显著提升，从而 Loss 在微调后下降幅度更大。但是也发现使用 Alpaca 微调后，模型对普通任务产生了部分遗忘。

综上，结合最终测试结果和在验证集上的 loss，可以看到微调带来了模型在预期任务上的效果提升。

3.9 补充

3.9.1 混合精度训练

在实验中，我尝试使用混合精度（fp16）进行训练，期望通过减少显存占用、加快计算速度来提升整体训练效率。然而发现：使用 fp16 后，loss 无法正常收敛，并在训练初期迅速出现梯度爆炸（grad_norm 异常增大）。在少量数据进行训练时也出现了较大的 loss，经过梯度裁剪可以解决问题。

3.9.2 安全性

在实验过程中，我惊奇地发现，微调后的模型生成了存在安全性风险的回答，例如：

The story revolves around these 2 men trying out different kinds o' sex - some pretty good ones like having them do something crazy with each other... and then when one gets bored he decides

这一现象表明，在已有基础模型上进行微调时，可能会破坏原有的安全性对齐。我的推测是：虽然原始模型在大规模数据上经过了对齐训练，能够较好地避免输出敏感或不当内容，但在微调过程中，即使只是小规模调整（如 LoRA 微调），如果训练数据中包含了少量敏感信息（如 IMDB 数据集中可能存在的片段），也有可能导致模型内部参数分布发生扰动，进而削弱原有的安全防护机制。