《自然语言处理》最终大作业报告

基于 LLaMA-Factory 开源项目跑通一个 Chat 机器人

姓　　名 王崇骁

专　　业 计算机科学与技术

学　　号 2301950

学　　院 计算机科学与工程学院

2023 年　　**11** 月　　**22** 日

一，实验报告背景介绍与课堂内容总结

本节针对课堂内容与实验中设计的语言大模型进行了知识背景介绍与这几节课的课堂内容和 PPT 总结。

训练阶段介绍

1.1 指令微调介绍

以往的预训练-微调范式，在多领域很大规模的无标签数据集（TB 级别的 TOKEN 数据集）上预训练，在特定下游任务的小规模标注数据集（几万条数据）上微调，当下游任务变换时，只需要在下游任务对应的数据集上重新训练，就可以得到一个在下游任务上表现良好的模型。

然而预训练-微调范式存在一些工程上无法忽视的弊端，比如 1）泛化能力较弱，对于没见过的分类或领域知识效果不佳。2）存储开销大，需要为每一个下游任务存储一个完整的参数副本。3）成本开销大，重新训练一个大模型、收集下游任务数据集，多副本切换部署。

图 1 语言大模型的几个阶段

由于生成式预训练的目标是根据前文生成下一个单词（续写），因此仅预训练的大语言模型无法理解人类指令，就会导致虽然模型拥有丰富的通用知识，但因无法理解人类指令而无法为人类提供很好的帮助。

指令微调（instruction tuning）是指在自然语言格式的多任务数据集上有监督的方式微调预训练后的大语言模型的方法。和传统的 Pretrain-finetune 范式相比，指令微调的过程不限于某个特定任务数据集。和 prompting 方式相比，指令微调存在参数更新的过程。同时在多种任务指令数据上学习，提升模型在未见任务上的泛化能力，目的不是从指令数据中学习知识，而是学习如何按照人类指令回答问题，指令微调主要包括下面三个步骤：1. 获取基座模型；2. 收集指令数据集；3. 微调

但由于全量参数微调虽然效果好但耗费大量 GPU 资源，刚刚介绍的指令微调虽然与全量指令微调方式不同，但微调效果略逊于全量参数微调。于是引出了高效指令微调方式 LoRA（Low-Rank Adaptation of Large Language Models），用 LoRA 训练的好处在于，要训练的参数量大大降低，效果与全量微调相当，同时，存储也只需要存储 LoRA 矩阵，降低存储量，并且无额外推理延迟，推理时只需要提前算出 BA 的值，加回预训练模型即可。

1.2 奖励建模

在大语言模型完成 SFT 监督微调后，下一阶段是构建一个奖励模型来对问答对作出得分评价。奖励模型源于强化学习中的奖励函数，能对当前的状态刻画一个分数，来说明这个状态产生的价值有多少。在大语言模型微调中的奖励模型是对输入的问题和答案计算出一个分数。输入的答案与问题匹配度越高，则奖励模型输出的分数也越高。

奖励模型（RM 模型）将 SFT 模型最后一层的 softmax 去掉，即最后一层不用 softmax，改成一个线性层。RM 模型的输入是问题和答案，输出是一个标量即分数。由于模型太大不够稳定，损失值很难收敛且小模型成本较低，因此，RM 模型采用参数量为 6B 的模型，而不使用 175B 的模型。

奖励模型的训练数据是人工对问题的每个答案进行排名，如下图所示对于每个问题，给出若干答案，然后工人进行排序，而奖励模型就是利用排序的结果来进行反向传播训练。奖励模型的损失函数采用 Pairwise Ranking Loss。

1.3 强化学习

大语言模型完成奖励模型的训练后，下一个阶段是训练强化学习模型（RL 模型），也是最后一个阶段。大语言模型微调中训练 RL 模型常采用的优化算法是 PPO（Proximal Policy Optimization，近端策略优化）算法，即对设定的目标函数通过随机梯度下降进行优化。在 LLama-factory 中有 DPO 和 PPO 强化学习方法可以选择，本节主要介绍其中的 PPO 强化学习方法的背景以及原理。

近端策略优化是一种深度强化学习算法，用于训练智能体在复杂环境中学习和执行任务。通过智能体的训练，使得其在与环境的交互中能够最大化累积回报，从而达成指定任务目标。这里的智能体在大语言模型中指的就是 RL 模型。

RL 模型的初始模型采用 SFT 微调之后的大语言预训练模型。训练 RL 模型的数据集只需要收集问题集（Prompt 集），不需要对问题进行标注。问题集通过 RL 模型生成答案文本，然后将问题和答案输入上一步训练的 RW 模型进行打分，来评价生成的文本质量，而训练 RL 模型的目标是使得生成的文本要在 RW 模型上获得尽可能高的得分。

将初始语言模型的微调任务建模为强化学习（RL）问题，需要定义策略（policy）、动作空间（action space）和奖励函数（reward function）等基本要素。策略就是基于该语言模型，接收 prompt 作为输入，然后输出一系列文本（或文本的概率分布）；而动作空间就是词表所有 token 在所有输出位置的排列组合；观察空间则是可能的输入 token 序列（即 prompt），为词表所有 token 在所有输入位置的排列组合；而奖励函数则是上一阶段训好的 RM 模型，配合一些策略层面的约束进行的奖励计算。

通过强化学习的训练方法，迭代式的更新奖励模型（RW 模型）以及策略模型（RL 模型），让奖励模型对模型输出质量的刻画愈加精确，策略模型的输出则愈能与初始模型拉开差距，使得输出文本变得越来越符合人的认知。

大语言模型训练中的 PPO 强化学习：1.在大语言模型训练中，强化学习模型架构与 SFT 监督微调的模型一样，2.RLHF 中训练强化学习模型阶段不需要标注问题的答案 3.RLHF 中的初始策略就是 SFT 模型

二，实验内容与结果分析

下图是实验 LLama-factory 的界面总图，可以观察到有很多参数可以选择，本节主要分为三个内容进行实验，分别对应三个阶段，每个阶段都会对参数进行调节和比对，并给出实验结果图像以及结果信息。

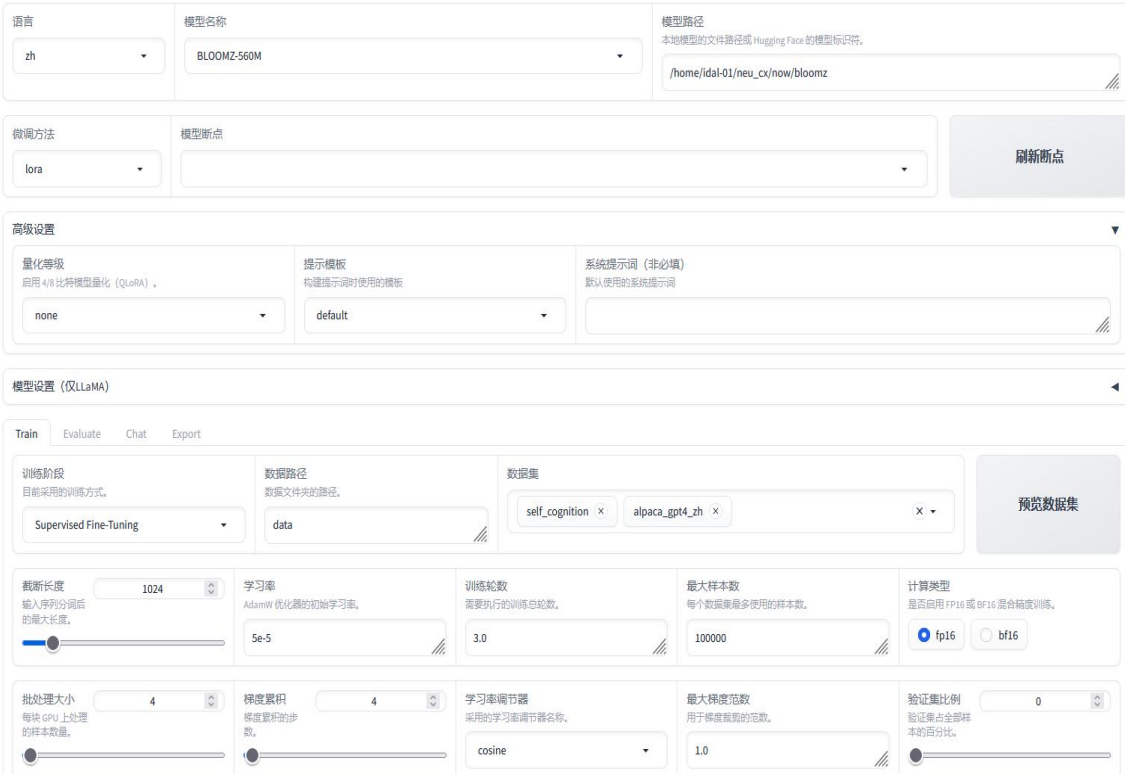


图 2：LLama-factory 界面

2.1 指令微调

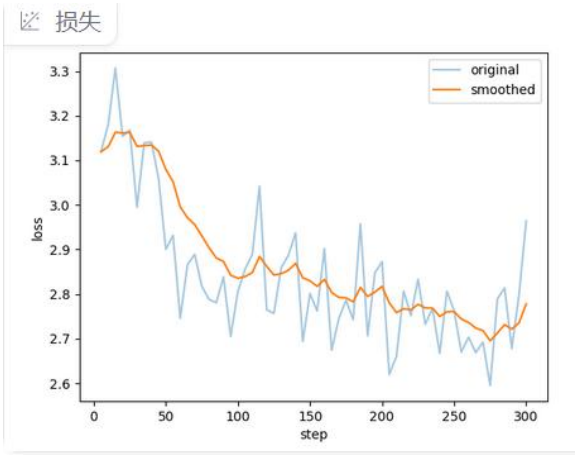
2.1.1 使用微调方法的不同：

1）冻结头部，训练尾部 ；2）全部训练； 3）lora 微调方法，使用低秩近似来降低权重矩阵的维度，从而减少模型中可训练参数的数量，以达到加速大型语言模型的训练，并且具有更低的内存消耗。

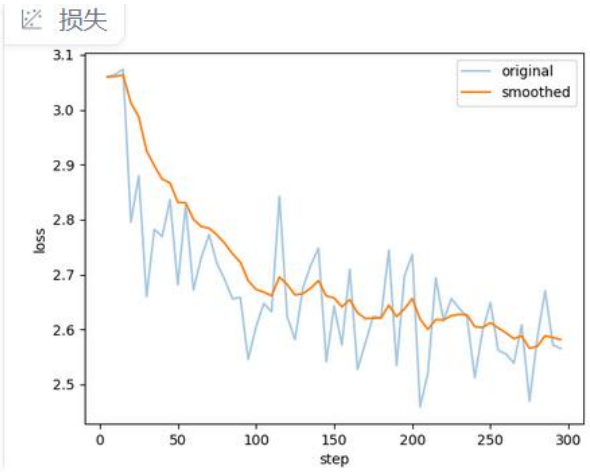
1.微调方法的不同对应图像（300-steps）

1. **Freeze:**BLOOMZ-560m 无法冻结训练，因为其 config 文件中没有 layer 层数，但能进行 full 或 lora 微调方式。

2. **Full：**



3. **Lora**



2.预计运行时间表格

策略	预计运行时间
Freeze	无
Full	1h-26min
Lora	1h-14min

3.总结与分析

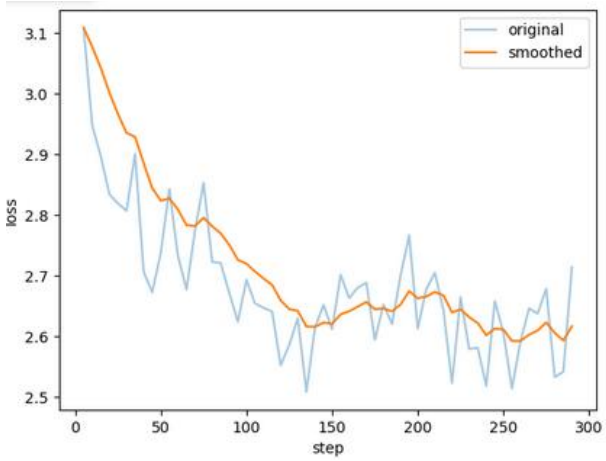
lora 相较于 full 微调方法，通过使用低秩近似来降低权重矩阵的维度，从而减少模型中可训练参数的数量，加速大型语言模型的训练，并且 loss 降低的也更迅速，因此优化策略为采用 lora 微调方法。

2.1.2 量化等级的不同：

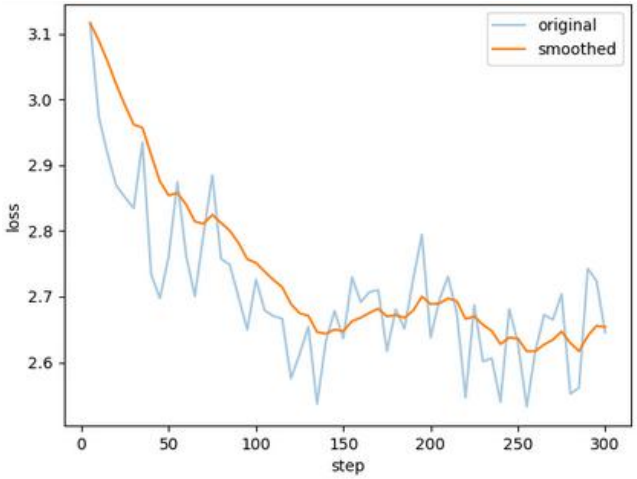
分别采用 4/8bit，权重参数量化为 4 位或 8 位的整数，减少模型的内存占用和计算需求，从而提高模型的推理速度和效率，但缺点在于较低精度的表示可能无法完全捕捉原始模型中的细节和复杂性，可能会导致一定程度的性能下降，在 lora 微调方法下实验。

1.量化等级不同对应的图像:

4bit:



8bit:



2.预计运行时间表格

量化等级	预计运行时间
4bit	1h-43min
8bit	3h-24imn

3.总结与分析

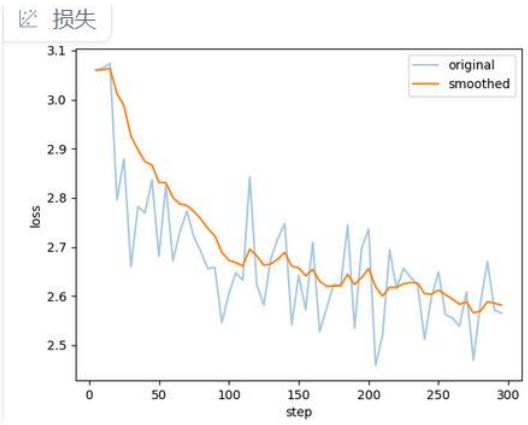
较低的量化等级可以在一定程度上提高训练速度，因为它减少了计算和存储需求。但是，过低的量化等级可能会导致精度损失过大，从而降低模型的性能,观察 loss 图看出采用 4bit 和 8bit 的量化等级在前 300epoch 类似，曲线近似不稳定，并进行波动，并且 loss 下降并不明显，因此在后续不采用量化等级。

2.1.3 提示词模版的不同：

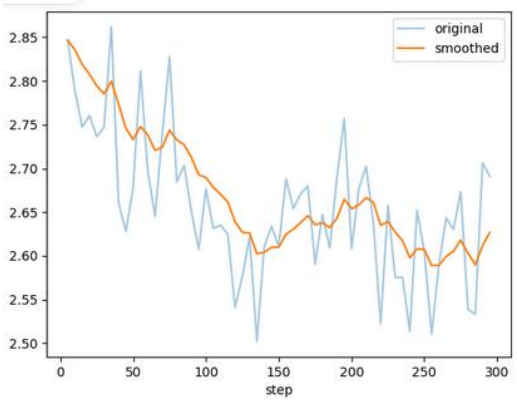
通过不同模版去调整和优化聊天模型的行为和性能，有多种模版，可以让模型更有偏向的去训练和回答问题。这里我选用了 default, alpaca, vicuna 三个模版进行比对。

1.图像：

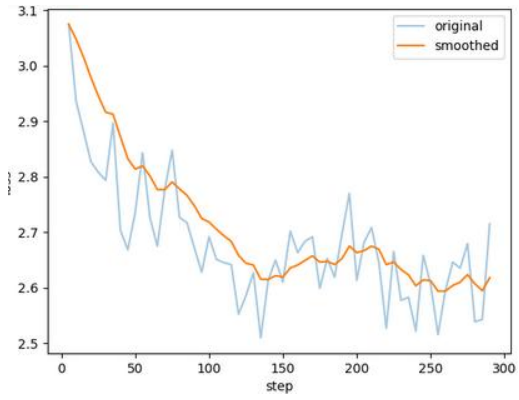
default：



alpaca：



vicuna：



2.预计运行时间表格

模版	预计运行时间
default	1h-14min
alpaca	1h-12min
vicuna	1h-20min

3.总结与分析

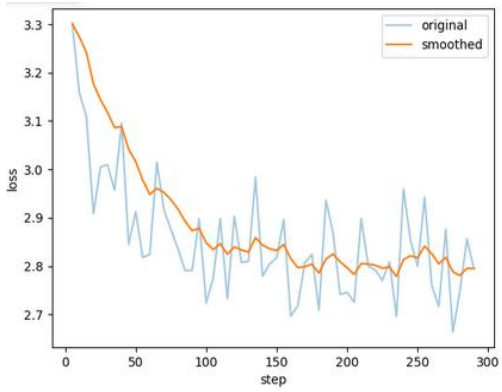
模版的选择最后体现在的是和机器人沟通情况，但根据图像还是可以观察出，加入模版后模型的 loss 波动幅度较大，说明模型在接触更新的知识内容，对新的知识较为敏感，但如果优化步数更多会趋于稳定，模型也会更好的适应该模版。

2.1.4 尝试不同的数据集：

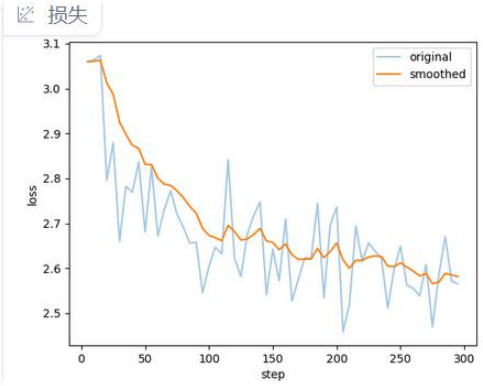
通过不同选择不同数据集去优化和微调模型。

1.图像

alpaca_zh+self_cognition+alpaca_gpt4_zh:



alpaca_gpt4_zh:



2.预计时间表格

数据集	预计运行时间
alpaca_zh+self_cognition+alpaca_gpt4_zh	2h-52min
alpaca_gpt4_zh	1h-14min

3.总结与分析

数据集对模型的影响是不言而喻的，数据集越多，模型学习的知识就越多，虽然需要时间较长，但大模型优势就在于此，模型参数大，数据集也大，这样才能得到一个精通任何知识的 chat 模型机器人，经过观察 loss 图像可以看出，在增加数据集数量后，运行时间成倍

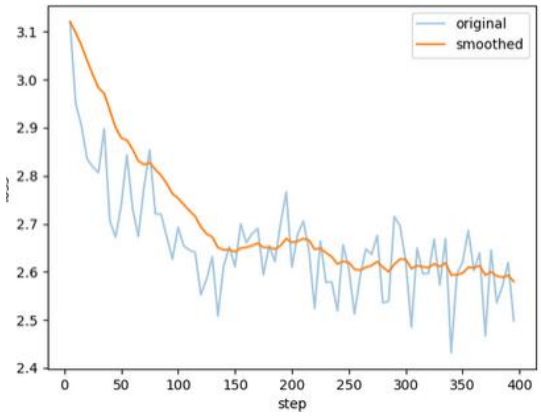
的变化，而且 loss 会相较于单数据集较高一些，还是在于模型在面对多数据集时，各数据集直接的内容可能有些许差异，因此会呈现如图像的 loss 变化。

2.1.5 调节模型的截断长度：

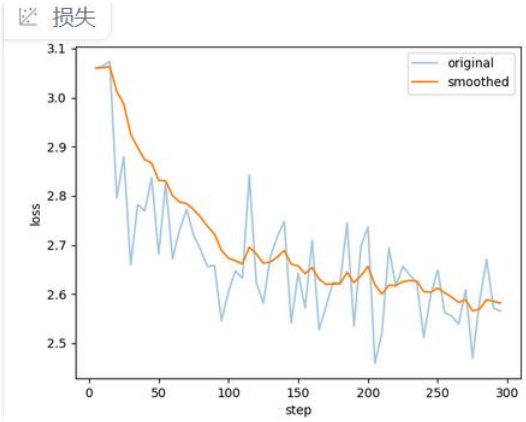
截断长度（truncation length）是指在处理输入文本时所采用的最大长度限制，较长的长度可以让模型更好的学习到上下文，但由于计算资源和模型内存的限制，大型语言模型通常无法处理非常长的文本序列，因此选择一个合适的截断长度是比较重要的，本节主要比较三个截断长度，并给出 loss 图像以及其预计运行时间的比较。

1.图像

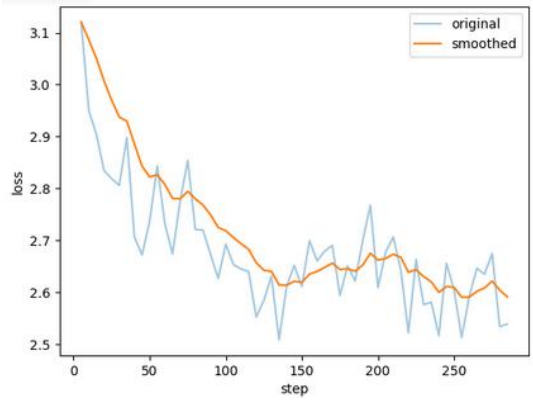
Length=512:



Length=1024:



Length=2048:



2.预计运行时间表格

截断长度	预计运行时间
512	1h-27min
1024	1h-14min
2048	1h-30min

3.总结与分析

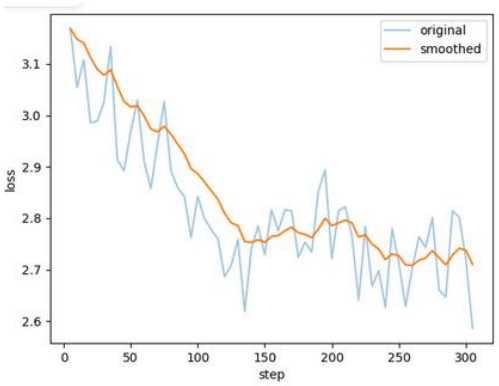
一个合适的截断长度是比较重要的,可以看出截断长度过长，运行速度慢，当截断长度过短，虽然运行速度快，但是其步数大，轮次多，总运行时间也不短，而且当截断长度过小，会导致模型无法正确学习上下文，导致 loss 不稳定而产生波动。

2.1.6 学习率的调节

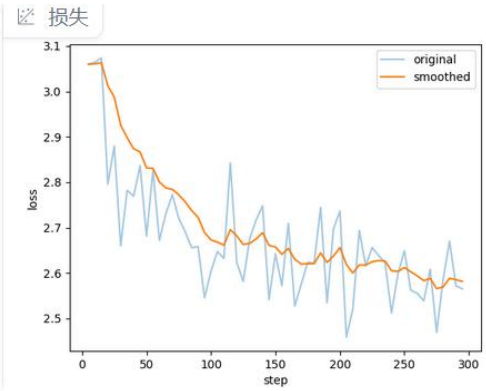
学习率决定了在训练过程中模型参数更新的步幅大小，选择较大的学习率模型收敛速度快，但可能导致波动较大，使得模型不稳定。选择较小的学习率会使模型收敛速度过慢，影响训练效率。

1. 图像

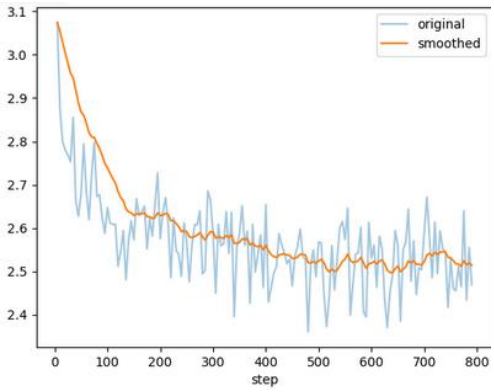
1e-5:



5e-5:



1e-4:



2. 预计运行时间表格

学习率	预计运行时间
1e-5	1h-25min
5e-5	1h-14min
1e-4	1h-15min

3. 总结与分析

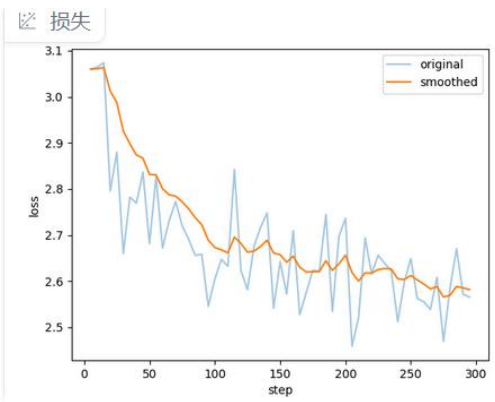
观察图像可以了解到大的学习率模型收敛速度快，但可能导致波动较大，使得模型不稳定。选择较小的学习率会使模型收敛速度过慢，影响训练效率。但其预计运行时间类似，因为学习率基本不会严重影响预计运行时间，其优化步数基本是一样的。

2.1.7 计算类型的调节

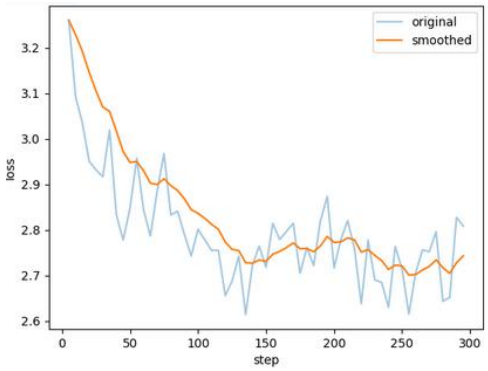
FP16 使用 16 位来表示浮点数，其中 1 位用于表示符号，5 位用于指数，剩下的 10 位用于表示尾数，BF16 同样使用 16 位来表示浮点数，其中 1 位用于表示符号，8 位用于指数，剩下的 7 位用于表示尾数，指数位较少，FP16 的数值范围相对较小，指数位较多，BF16 的数值范围相对较大，它们在精度和数值范围上有所不同，适用于不同的应用场景，需要调节比较其效果如何。

1. 图像

FP16：



BF16：



2.预计运行时间表格

计算类型	预计运行时间
FP16	1h-14min
BF16	1h-20min

3.总结与分析

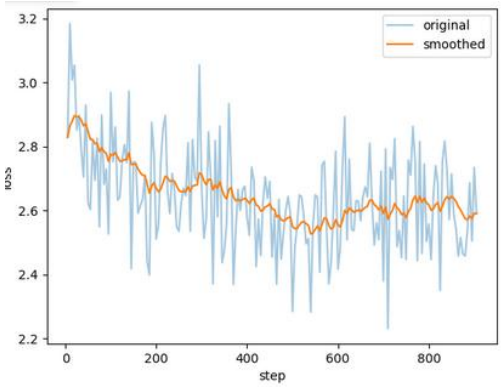
FP16 与 BF16 最大的区别就是在于精度和数值范围上有所不同，根据模型预计运行时间发现相差不大，说明数值范围影响对模型运行时间与速度较小，根据图像可以观察出 FP16 下降趋势更加明显，更容易收敛，且 FP 的尾数位更多，更精确，说明精度越高越容易收敛，类似的结论在量化等级中也有体现。

2.1.8 梯度累计的步数

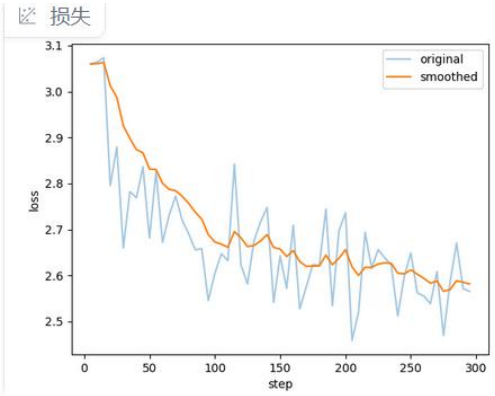
梯度累积是指在一个批次（batch）内先不更新参数，而是将多个批次的梯度进行累积，直到达到一定的步数或者满足一定的条件后再进行参数更新，由于训练过程中的批次较小或者计算资源有限时，单个批次的梯度可能不足以提供准确的方向信息进行参数更新，因此引入梯度累积的步数，但累计步数过大则会导致训练过程变慢，过小则会导致反向传播信息不准确，因此需要调节比较效果。

1. 图像

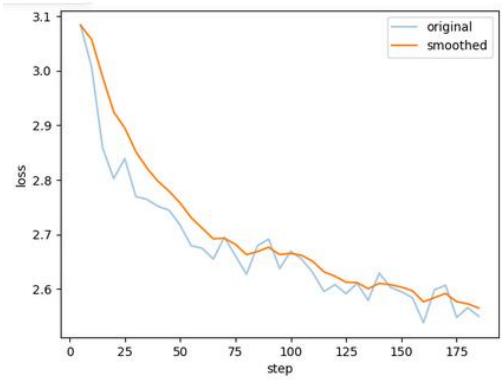
Gra-step=1:



Gra-step=4:



Gra-step=32:



2.预计运行时间表格

梯度累积步数	预计运行时间
1	1h-22min
4	1h-14min
32	1h-08min

3.总结与分析

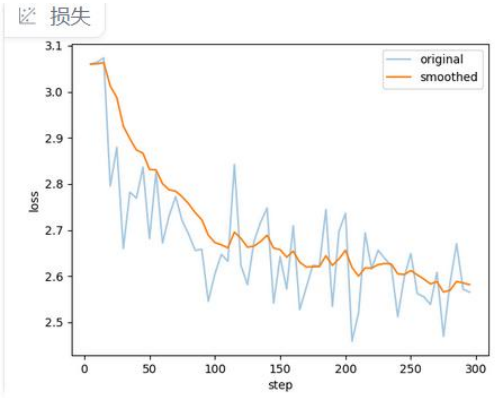
观察表格与图像可以得出结论，虽然预计运行时间相差不大，但是当累计步数过大时会导致训练过程每一个步长变慢，具体表现为开始到运行 300steps，累积步数越大所需时间就越长，但累积步数过小则会导致反向传播信息不准确，波动较大，因此当梯度累积步数为 4 时，训练速度和 loss 下降都取得了比较平衡的效果。

2.1.9 学习率调节器的选择和预热步数的大小

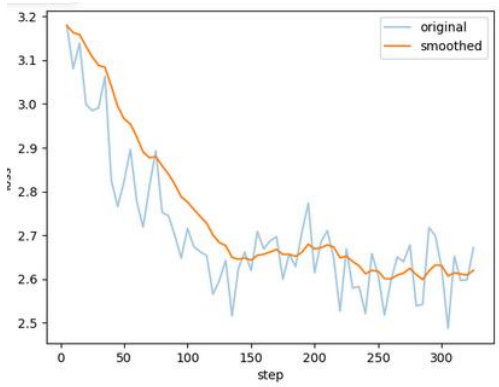
学习率调节器可以帮助平衡训练过程中的学习率大小，使得模型在不同阶段以适当的步伐进行参数更新。预热步数设置可以增加收敛速度避免过拟合，但又不能过大，避免训练过程中的震荡和不稳定性，因此需要进行调节比较，选出比较合适的预热步数。

1. 图像

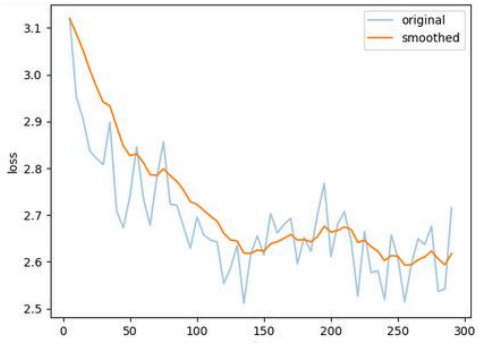
Cosine;0:



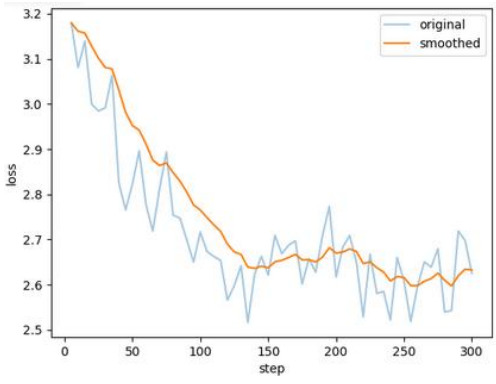
Cosine;50:



Linear;0:



Linear;50:



2.预计运行时间表格

学习率调节器和预热步数	预计运行时间
Cosine；0	1h-14min
Cosine；50	1h-16min
Linear；0	1h-20min
Linear；50	1h-21min

3.总结与分析

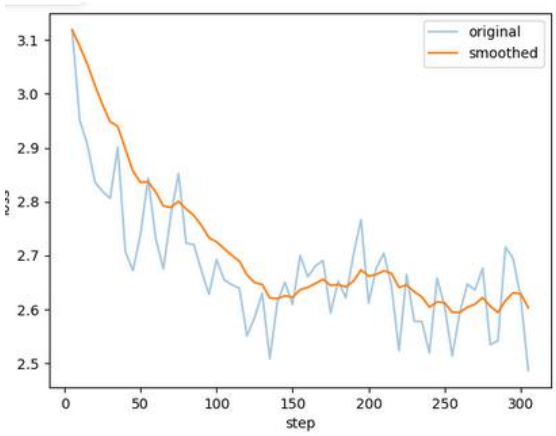
预热步数设置可以使学习率先上升使收敛速度加快，至将收敛时缩小学习率大小，慢慢寻找最优点，因此可以对比当设置预热步数时，模型在预热步数期间 loss 降低速度较快，但不影响预计运行时间，如图所示，在对比不同的学习率调节器时发现，余弦学习率调度器的效果要比线性学习率调度器效果稍微稳定一些，主要原因在于余弦学习率调度器通常用于训练深度神经网络，可以帮助模型更稳定地收敛并避免陷入局部最优点，而线性学习率调度器通常适用于简单的模型或者在训练初期需要较大学习率的情况。

2.1.10 最大梯度范数的选取

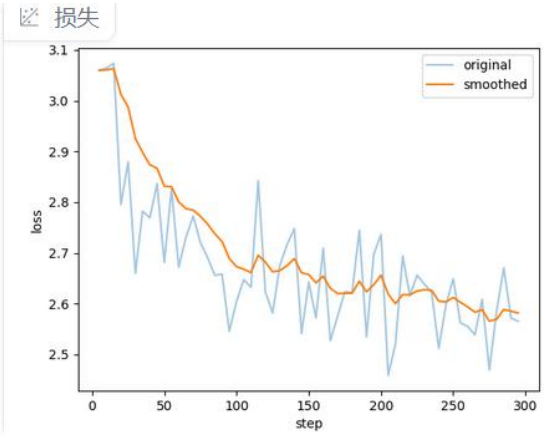
梯度裁剪用于控制梯度的大小，用于应对梯度爆炸（gradient explosion）的问题。当梯度值过大时，可能导致数值计算不稳定和模型训练困难。梯度裁剪通过限制梯度的范数（即梯度的 L2 范数）来避免梯度爆炸的情况。当梯度的 L2 范数超过该最大梯度范数阈值时，梯度会进行缩放，使其范数不超过阈值。这样可以确保梯度的大小在可接受的范围内，提高模型训练的稳定性。

1. 图像

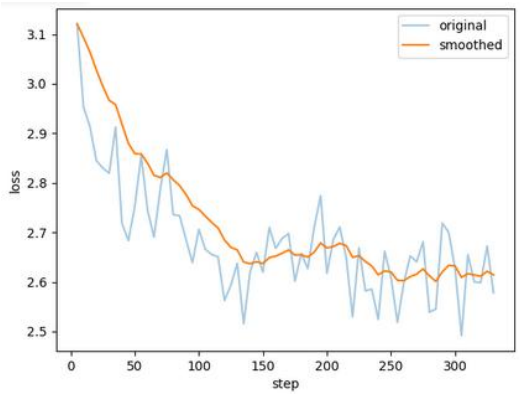
最大梯度范数为 0.1：



最大梯度范数为 1：



最大梯度范数为 100：



2.预计运行时间表格

最大梯度范数	预计运行时间
0.1	1h-5min
1	1h-14min
100	1h-25min

3.总结与分析

通过观察图像和查阅资料可知，当选取过大的最大梯度范数时会发生梯度爆炸，超过了数值稳定性的限制，导致训练不稳定，损失函数无法收敛，而当选择过小的最大梯度范数时，会导致梯度消失的问题，导致模型训练缓慢或无法收敛，虽然三个梯度范数在图像前 300steps 表现得并不明显，但应选择合适的最大梯度范数为 1 时，模型收敛效果会更好。

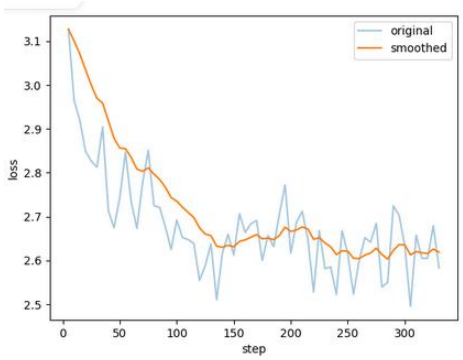
2.1.11.lora 参数的调节与设置

1）lora 的秩：秩的大小影响矩阵包含的信息大小，低秩则对信息进行压缩，但如果秩的选取过低，可能导致无法包含足够的信息，过大又无法起到压缩数据，凝聚信息的作用 2）lora 随机丢弃：和 Dropout 类似，可以使模型不易过拟合，增加模型的鲁棒性，但如果权重

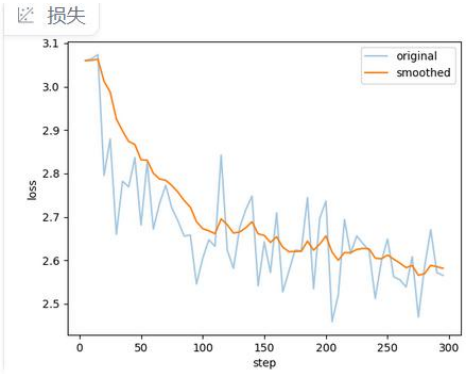
丢弃概率过高，可能导致模型训练困难。

1.图像

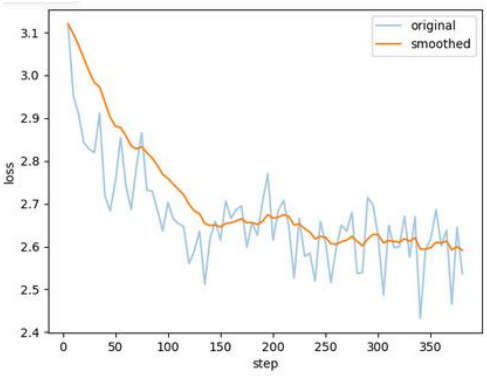
1；0.1：



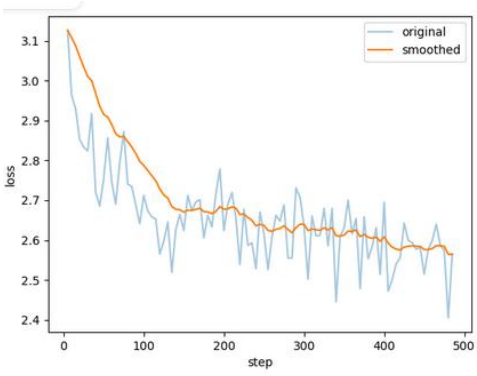
8；0.1：



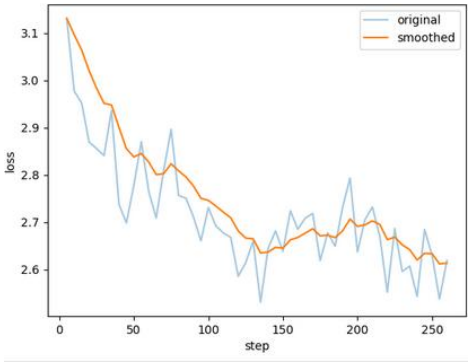
64；0.1：



8；0.6：



8；0.8：



2.预计运行时间表格

Lora 的秩与随机丢弃的概率	预计运行时间
1；0.1	1h-18min
8；0.1	1h-14min
64；0.1	1h-24min
8；0.6	1h-24min
8；0.8	2h-03min

3.总结与分析

根据图像观察可以得出，当丢弃概率过高时，图像抖动越明显，当丢失概率为 0.8 时，抖动最明显，因此要合理设置丢失概率，增加丢失概率会防止模型过拟合，但过大会造成训练过难，另外观察发现 lora 的秩越大，包含信息越多，导致运行时间会加长。

2.1.12.注意力机制的选择

在指令微调中，注意力机制有两个选项：1）flash_attn：Flash attention 是一种改进的自注意力机制，引入局部注意力和核化自注意力来提高计算效率和模型性能，旨在提高模型的效率和性能。当设置为 True 时，模型将使用 flash attention；当设置为 False 时，模型将使用传统的自注意力机制。2）shift_attn：Shift attention 是一种改进的自注意力机制，通过引入平移操作来增加位置信息的表示能力。当设置为 True 时，模型将使用 shift attention；当设置为 False 时，模型将使用传统的自注意力机制。

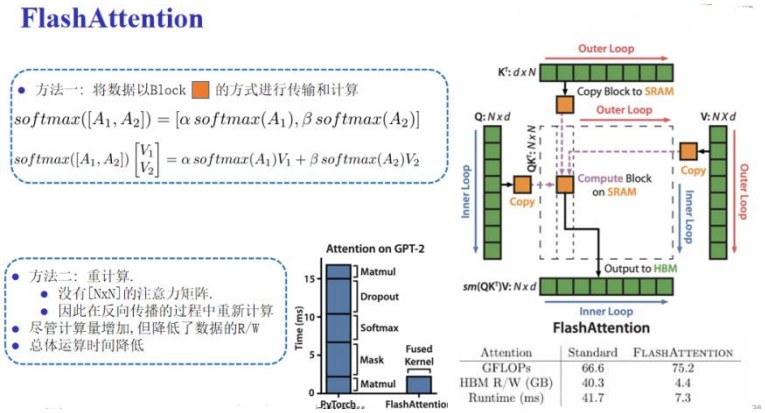


图 x：FlashAttention 两种计算法的讲解图

1. 预计运行时间表格

注意力机制	预计运行时间	loss
Flash：FALSE；Shift：FALSE	1h14min	2.59351
Flash：True；Shift：FALSE	1h55min	2.61512
Flash：FALSE；Shift：True	1h55min	2.612355
Flash：True；Shift：True	1h45min	2.61928

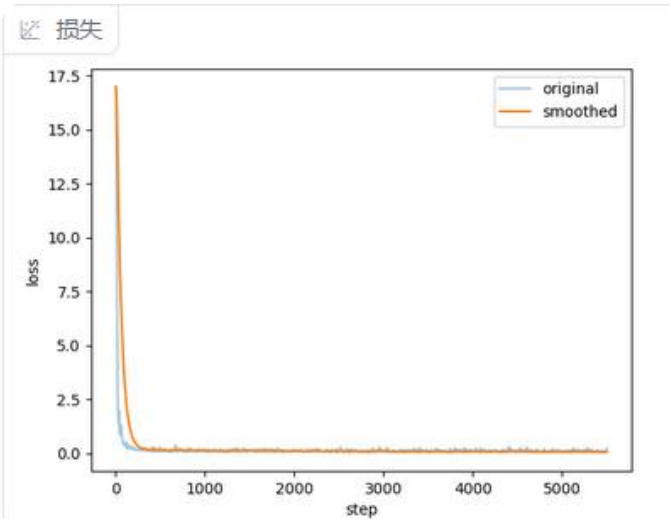
2.总结与分析

可以发现在使用传统注意力机制的情况下，预计运行时间相比于其他注意力机制要短很多，训练速度较快，且其 loss 在 300-step 时相差不大，因此选择传统注意力机制进行指令微调的效果要好很多。

2.2 奖励建模

在 2.1 节中详细介绍了各调节参数的背景和区别，本节不再赘述 2.1 节中的内容，主要分析在奖励建模阶段在参数最优情况下继续在指令微调的断点后训练过程截图并在 2.4 节中给出最终得到 chat 机器人的对话效果。

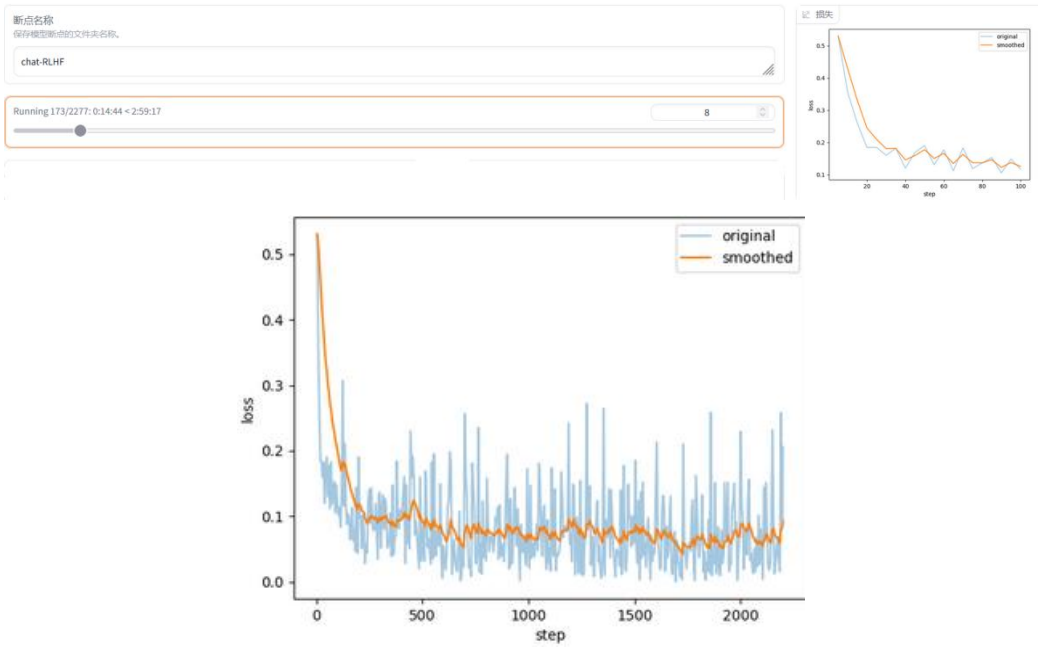
奖励建模的训练图像：



2.3 强化学习

在 2.1 节中详细介绍了各调节参数的背景和区别，本节不再赘述 2.1 节中的内容，主要分析在奖励建模阶段在参数最优情况下继续在奖励建模的断点后进行训练，并将训练过程截图，最后在 2.4 节中给出最终得到 chat 机器人的对话效果。

强化学习的训练图像：



2.4 Chat 机器人实测截图

三，总结

本篇实验报告基于 LLaMA-Factory 开源项目在 BLOOMZ-560M 基座上进行优化比较各参数以及各算法对模型的影响。

开篇对本文内容背景进行介绍和概括，为模型优化打好知识基础。在第二节中分阶段对模型进行优化，比较推理效率，使用模型量化、注意力优化等技术进行优化，并选择了一个语言模型基本流程从指令微调开始到强化学习结束，一个正向方向，比对探索优化属于自己 chat 模型机器人，虽然最终的 chat 机器人无法同目前开源的如 gpt 等回答的一样正确，但能做到基本的交流。

最后本文的代码链接如下，感谢各位助教学长课堂的分享知识经验以及老师的讲解教学，让我对语言大模型有一个进一步的认识与学习，收获颇丰！