**15分钟汇报模板**

**Slide 1｜（30–40秒）**

* “大家好，我是刘畅。非常荣幸能在这里向大家汇报我的大作业成果。今天我要分享的主题是 **‘对扩散式语言模型Dream-7B的监督微调’**。 在整个过程中，我参考和使用了Gemini Pro2.5的帮助 ，所有的代码和实现细节都开源在了我的GitHub仓库中，欢迎大家访问。”

**Slide 2｜（1–1.5分钟）**

* 首先，我来介绍一下任务目标。核心是使用S1K数据集，对一个70亿参数的开源扩散模型Dream-7B，进行监督微调。这个任务最大的挑战在于，Dream-7B是一个文档稀缺、非标准的‘黑盒’模型，需要进行大量的探索。项目成功的标准有三点，都必须在**5天**和**100元**的算力成本限制下完成：” 在5天和100元的算力成本内，跑通整个训练流程。微调后的模型必须在行为上有所改善，相比原始模型能看出有意义的对比。模型需要展现出初步的逻辑推理能力，输出要逼近标准答案。
* 核心技术上，考虑到显存和效率，我采用了QLoRA进行4-bit量化微调，并使用了PEFT和transformers等主流库。大家可以看到，最终的成本精确控制在了94.92元，成功达成了预算目标。这里展示了我设置的关键训练超参数和LoRA配置，比如学习率是2e-5，LoRA的秩r设为16。

**Slide 3–4｜（~3分钟）**

* 当然，整个过程并非一帆风顺，我遇到了大量的报错。这里是我部分调试历程的记录。从一开始就遇到了模型加载失败的问题。在经过与AI的合力分析之后，我定位问题在于作者并没有遵循Hugging Face的标准命名规范于是我查阅了作者的源代码在获得了这个确切的类名之后才正确加载了这个模型。内核崩溃的问题也很有意思，我遇到这个问题一开始无论怎么试都会报错，后来我才发现，我最初启动的云实例是“无卡模式”这是一个有点滑稽的错误。在准备训练的过程中，遇到了张量类型不匹配的问题我的思路是直接修改了我的预处理方程，在数据处理的最后一步手动将tokenizer强制转换为匹配的类型。还有爆显存问题，我花了很长时间才搞明白它其实是一个内存碎片化导致的问题，我在最顶上加了一行expandable\_segments:True就解决了。后面我就去改用qlora了，下一个问题是 DreamModel 作为一个自定义模型它缺少 peft 和 transformers 库里训练流程所必需的一个方法于是我就自定义了一个简化版的函数添加到了模型的实例上。在我测试训练好的模型的能力时遇到了API签名不匹配的问题。这个问题也挺棘手。它也是dreammodel作为一个自定义模型导致的，huggingface的.generate与dreammodel的diffusion generate之间存在参数传递的冲突。于是我写了一个适配器创建了全新的方法，它拥有一个huggingface标准的接口，而在内部它会把 关键字参数重新打包成diffusion generate所期望的位置参数。

**Slide 5｜（~3分钟）**

* 为了选择最优的微调方案，我对比了‘全参数微调’和‘QLoRA’。这是它们在训练集上的学习曲线。大家可以看到：…读ppt

**Slide 6-7｜（~2分钟）**

* 那么，微调的效果究竟如何呢？坦白说，**微调有效，但并未达到我最初最理想的指标**。我们来看两个对比案例。左侧是实验组，右侧是对照组。（念ppt）,下一页。
* 同样，对于一个开放性问题‘你怎么看待人类’，（念ppt）

**Slide 8｜（1分钟）**

* 面对不完美的成果，最重要的不是气馁，而是 **分析背后的原因**。（念ppt）找到问题的根源，比得到一个完美的结果更重要。

**Slide 9｜**

* 为了验证我的假设，我设计了一组严谨的 对比实验，控制变量是关键：我保持所有训练代码、数据和超参数不变，只替换基础模型。实验结果非常清晰：两个Dream系列模型，质量都没有显著提升。而Qwen2模型，则表现出了显著更优的效果。这初步验证了我的第一个假设——基础模型的能力至关重要。”

**Slide 10-11**

* **这里我们快速看一下修复了数据预处理代码后，Dream-7B Base模型的表现。很遗憾，即使使用了Github上官方推荐的预处理逻辑，结果依然没有本质改变。（读ppt）**

**Slide 12**

* **接下来是它的指令微调版本，Dream-7B Instruct。情况稍好一些。可以看到，它拥有了遵循指令和进行Chain of thought的回答模式。相比Base模型，逻辑性有了很大提升。但在微调后，它的核心推理能力依然没有被‘唤醒’，只是学会了更复杂的回答‘模板’。这说明，对于一个能力不足的模型，单纯的微调可能只是在‘刷墙’，而没有改变房子的结构。”**

**Slide 13**

* **最后，我们来看 Qwen2-7B Instruct 的表现，（读ppt）这有力地证明了，基础模型的能力不足，其次才是数据量不够。**

**Slide 14**

* **虽然没有达到理想指标，但是这只会让我想要做得更好。**
* **通过这个复杂的项目，我收获了三点宝贵的经验，这也是我认为比项目结果本身更重要的财富。（读ppt）**

**Slide 15**

* **读ppt。**

**QA环节**

***为什么选QLoRA而不是全参*（用你那张“震荡vs平滑收敛”曲线作答）**

***为什么Dream-7B提升有限*（先底模能力，再数据/预处理，最后给下一步计划）**

***如果给你再两天/再200元预算会做什么*（扩大数据×5；加验证集与早停；补自动化评测脚本）。**

**问题 1 (技术深度): “您的报告提到，您通过‘猴子补丁’解决了模型缺少 prepare\_inputs\_for\_generation 方法的问题 。能详细描述一下您发现并解决这个问题的完整思路吗？以及您认为这种方法有什么潜在风险？”（首先，在发现问题方面，当我尝试用PEFT库的 get\_peft\_model 函数来包装Dream-7B模型时，程序直接报错，提示 AttributeError: 'DreamModel' object has no attribute 'prepare\_inputs\_for\_generation' 。这个报错非常明确，就是说PEFT库期望基础模型有一个特定的接口方法，但Dream-7B这个非标模型并没有实现它。在解决思路上，我的第一反应是去阅读PEFT库的源代码，理解这个缺失的方法到底起了什么作用。通过阅读源码我发现，这个方法其实是一个相对简单的预处理器，它的主要功能是在模型生成（generation）之前，将 input\_ids 和其他关键词参数（比如 attention\_mask）打包成一个字典格式。它的逻辑并不复杂。最后，在具体实现上，我没有去改动PEFT库的源码，因为这会破坏环境的稳定性。我选择了一种侵入性最小的方案，也就是‘猴子补丁’。我定义了一个独立的、与PEFT期望接口完全相同的函数，然后动态地将这个函数绑定到我加载的模型实例上。这样一来，当PEFT库调用这个方法时，模型实例就能正确响应了。这是一种在不修改外部库源码的前提下，适配非标接口的常用工程技巧。”“至于潜在风险，我完全清楚。第一，这种补丁是临时的，如果PEFT库未来版本更新，改变了这个方法的内部逻辑或签名，我的补丁可能会失效。第二，我的实现是最小化的，可能没有覆盖到一些复杂的边缘情况。但在这次限时5天的项目中，这是一种在工程可行性和理论完备性之间做出的最高效的权衡。”**

**问题 2 (战略思维): “您在对比实验中选择了Qwen2 ，但它是一个自回归模型，和扩散模型Dream-7B的架构不同。为什么选择它作为参照物？这个架构差异给您的实验带来了什么挑战？”**

**中文回复草稿:**

**“这是一个非常好的问题，确实是我在设计实验时重点考虑过的一点。”**

**“**

**我选择Qwen2的核心目的，是为了验证我的核心假设：即项目结果不理想，根本原因在于Dream-7B这个基础模型的能力太弱 。为了验证这一点，我需要一个‘高能力’的参照系作为对比。在目前的开源社区，要找到一个公认的、高质量的、7B规模的**

**扩散语言模型其实非常困难，这个领域远不如自回归模型成熟。而Qwen2是一个公认的、SOTA级别的自回归模型，把它作为性能的‘软上限’来进行比较，最能有力地说明问题。”**

**“**

**这个选择确实带来了挑战。最大的挑战在于如何确保变量控制的公平性。虽然底层架构不同，但我保证了实验的其他所有条件完全一致：使用了完全相同的数据集、相同的QLoRA微调代码和超参数 。这么做是为了证明，即使在最优的外部条件下，Dream-7B的提升也远不如一个强大的基础模型。”**

**“**

**在结果解读上，我也很谨慎。我承认，Qwen2的优异表现部分可能得益于其自回归架构更适合这类数学推理任务。但两者之间展现出的巨大性能鸿沟——从‘token坍塌’到‘逻辑完备’——已经远远超出了架构差异所能解释的范畴，从而有力地印证了我的核心结论：基础模型的能力是决定项目成败的关键瓶颈 。”**

**问题 3 (反思与成长): “回顾整个项目，如果您可以从头再来一次，您会选择做哪一件不同的事？”**

**中文回复草稿:**

**“如果能重来一次，我会在项目最开始，用半天时间对Dream-7B这个基础模型进行一次‘能力审计’（Capability Audit）。”**

**“这次项目中，我是在微调遇到困难、结果不理想后，才通过对比实验倒推出基础模型能力不足的结论。这个过程虽然严谨，但相对被动。”**

**“更优化的流程是，在第一天就设计一组覆盖不同能力的测试集，比如基础问答、数学推理、代码生成等，然后直接测试未经微调的Dream-7B、Dream-7B Instruct以及Qwen2等模型，对它们的基准能力有一个清晰的量化认知。”**

**“**

**这样做的好处有两点：第一，可以让我更早地形成‘基础模型是短板’的假设，从而在后续的调试中，能更快地将问题定位到模型本身，而不是过度怀疑我的代码。第二，这个审计结果本身就是项目一个非常有价值的交付成果，能够更有力地指导我们未来的技术选型。这体现了我提到的从项目中学会的‘系统性的工程调试思维’ 。”**

**问题 2: Dream的结构特殊性**

**“扩散式LLM与自回归在掩码/预处理上的本质差异是什么？你修的preprocess具体避免了哪些失配？能否展示一条样本从‘原始→模型输入’的对照？”**

**中文回复草稿:**

**“这个差异是两者生成范式的根本区别。”**

**“本质差异在于注意力机制的范围：”**

* **“自回归模型（Autoregressive），比如GPT和Qwen，采用的是\*\*‘因果掩码’（Causal Mask）\*\*。在预测第N个词时，它只能看到前面N-1个词的信息。它的天性是‘从左到右，依次预测下一个’。”**
* **“扩散式模型（Diffusion LLM），比如Dream，其灵感来源于图像去噪，采用的是类似BERT的\*\*‘双向掩码’（Bi-directional Mask）\*\*。它的输入是一个‘被破坏’的序列，模型的目标是利用上下文（包括前后文）来‘修复’或‘还原’出原始序列。它的天性是‘总览全局，填补残缺’。”**

**“我修复的preprocess，就是为了避免将‘自回归的训练方式’错误地用在‘扩散式模型’上。我推测，最初的预处理方式可能只是简单地将问题和答案拼接，然后让模型去预测答案部分，但这没有发挥出扩散模型双向注意力的优势。修正后的预处理函数，严格遵循了模型预训练时的\*\*‘去噪’格式\*\*，确保了微调任务和预训练任务在输入格式和注意力掩码上的一致性，避免了这种根本性的失配。”**

**问题 3: 稳定增益的最小方案**

**“若再给你24小时与20元预算，只在Dream体系内，你会做哪两件事来争取一个可量化的+X%改进？”**

**中文回复草稿:**

**“在如此严格的资源限制下，我会选择两个‘零训练成本’的、纯粹在推理（Inference）阶段进行优化的方案，来进行一次微小的消融实验。”**

**“第一件事：优化解码参数（Generation Parameters）。我会对测试集进行一次小型的网格搜索。重点调整temperature和top\_p这两个参数。我的假设是，对于数学推理这种需要精确性的任务，一个更低的temperature（例如从0.7降到0.2）可以减少模型随机‘幻觉’的概率，让输出更具确定性。我预期这可以直接在‘数值准确率’上带来1%-2%的提升。”**

**“第二件事：优化指令提示模板（Prompt Template）。既然微调已经让模型学会了‘思考链’（Chain of Thought），我可以通过更明确的指令来强化这种行为。我会测试2-3个不同的指令后缀，例如：”**

* **“基准: {question}”**
* **“变体A (强调格式): {question}\n请遵循以下步骤：1.列出公式 2.带入数值 3.计算结果。”**
* **“变体B (强调单位): {question}\n请注意单位换算，并给出最终答案的单位。” “我的假设是，变体A可以直接提升‘公式选择正确率’，而变体B能提升‘单位正确率’。这两项操作都无需重新训练，20元预算足够完成上百次推理测试，可以在24小时内找到一个最优的组合，争取到一个可量化的稳定增益。”**

**问题 4: 过拟合与token坍塌**

**“你观察到‘高频词坍塌’的触发条件与缓解手段？”**

**中文回复草稿:**

**“我观察到‘高频词坍塌’，比如输出大量‘of of of’，本质上是模型进入了低熵的恶性循环。它的触发条件通常是：”**

1. **“基础模型能力弱：模型本身对语言的理解很差，当遇到无法理解的、分布外的输入时，其内部状态会崩溃，只能输出最简单、最‘安全’的高频词。”**
2. **“解码策略单一：如果使用贪心搜索（Greedy Search），一旦模型陷入某个循环，就再也无法跳出。”**

**“缓解手段可以从训练和推理两个层面入手：”**

* **“训练时（治本）：可以通过正则化手段来提升模型的泛化能力。例如，\*\*权重衰减（Weight Decay）\*\*可以防止权重参数变得过大；\*\*标签平滑（Label Smoothing）\*\*可以防止模型对训练数据过拟合，从而在面对不确定性时更加鲁棒。这些都是降低模型‘过分自信’、避免其陷入思维定式的有效方法。”**
* **“推理时（治标）：最直接的手段是调整解码策略。弃用贪心搜索，改用采样方法，比如增大temperature或使用top\_p采样来增加输出的多样性。同时，设置一个大于1的\*\*重复惩罚（Repetition Penalty）\*\*参数，可以强行阻止模型连续输出相同的token，能非常有效地打破这种循环。”**

**问题 5: 安全性变化**

**“为什么微调后更‘守规矩’？S1K或指令模板是否暗含安全偏好？如何避免把与任务无关的风格/安全语气也学进去？”**

**中文回复草稿:**

**“微调后的模型更‘守规矩’，本质上是因为微调是一种‘行为克隆’（Behavioral Cloning）。S1K数据集的文风是高度格式化的、客观的、解决问题的风格。模型在学习解题方法的同时，也把这种严谨、安全的‘学者’人设（Persona）给学了进去。它为了在训练中获得更低的损失，会尽力模仿这种最‘安全’的表达方式。”**

**“S1K数据集和我的指令模板，确实暗含了强烈的安全偏好。这个偏好就是：专注、客观、避免主观意见。数据集中没有任何口语化、情绪化或争议性的内容，这本身就是一种筛选，让模型认为这种‘无感情、无立场’的风格才是正确答案。”**

**“要避免学进无关风格，也就是‘风格污染’，有几个方法：”**

1. **“数据集清洗：这是最根本的方法。在制作数据集时，就应严格剔除所有与核心任务无关的文体特征。比如，如果发现很多答案都以‘作为一个AI，我...’开头，就应该在训练前把这些前缀全部删掉。”**
2. **“使用‘去风格化’指令：在推理时，可以通过prompt明确要求模型不要产生无关的输出。例如，在指令末尾加上一句：‘请直接输出解题步骤和最终答案，不要包含任何解释性或对话性的语言。’”**
3. **“更前沿的技术：在学术界，也有像\*\*‘解耦表示’（Disentangled Representation）\*\*这样的研究，试图将模型的能力（比如推理）和它的风格（比如语气）在模型内部解耦开，从而在推理时可以独立控制，但这在工程上实现起来还比较复杂。”**

**针对P2: 任务目标与技术栈**

**1. “你选择了QLoRA。很好，那你能否用简单的语言解释一下，QLoRA到底是如何在24G显存下微调7B模型的？它的核心思想是什么？”**

**回答：  
“这是一个非常核心的问题。在24G显存下微调7B模型（BF16下需要约14GB，加上梯度和优化器状态会远超），主要靠QLoRA的三个‘组合拳’：**

* **第一，4-bit 量化 (Q): 它使用了一种叫做NF4（Normal Float 4）的技术，将原始模型权重从16位压缩到4位，使7B模型在显存中只占用大约3.5GB。**
* **第二，LoRA (Low-Rank Adaptation): 我们并不训练这3.5GB的量化权重，而是在模型的关键层（如Attention矩阵）旁路添加微小的、可训练的低秩矩阵（Adapter）。训练时只更新这些Adapter，参数量极小。**
* **第三，分页优化器 (Paged Optimizers): 即使是训练Adapter，优化器状态（如AdamW的动量）在更新瞬间也可能产生显存峰值。分页优化器就像操作系统的虚拟内存，在显存不足时将优化器状态临时转移到CPU内存，从而避免OOM。**

**这三者结合，才使得在消费级显卡上微调大模型成为可能。”**

**2. “你的learning\_rate设为2e-5，lora\_alpha设为32。你是如何决定这些超参数的？如果效果不好，你会优先调整哪个？”**

**回答：  
“这些参数的选择是基于LoRA论文的推荐和社区的最佳实践。2e-5是一个相对保守的学习率，可以在让模型学习新知识的同时，最大限度地防止‘灾难性遗忘’。**

**lora\_alpha与r（这里是16）的比例（这里是2）起到一个缩放因子的作用。alpha/r越大，新学习到的权重对模型的影响越大。**

**如果效果不好，我会优先调整learning\_rate。因为学习率是影响训练稳定性和收敛速度最关键的因素。我会尝试1e-5或3e-5。如果依然欠拟合，我才会考虑增加r或lora\_alpha来增大适配器的容量。”**

**针对P3 & P4: 调试历程**

**3. “我们来聊聊Fig. 6的这个报错：AttributeError... prepare\_inputs\_for\_generation。当你看到这个报错时，你的思考过程是怎样的？”**

**回答：  
“好的。我的思考过程是这样的：**

1. **定位问题： 报错信息很明确，模型对象缺少 prepare\_inputs\_for\_generation 这个方法。我立刻意识到这是Hugging Face Trainer在调用model.generate()进行评估时需要的一个标准接口。**
2. **验证假设： 我查看了Dream-7B在Hugging Face上的模型仓库（modeling\_dream.py，如Fig.1所示），确认其自定义的模型类确实没有实现这个方法。这是一个‘非标’模型带来的典型问题。**
3. **寻找解决方案： 最彻底的办法是Fork仓库并提交PR，但这耗时太长。最快捷、侵入性最小的方法就是‘猴子补丁’（Monkey Patch）。**
4. **实施： 我参考了transformers库中标准模型的实现，写了一个prepare\_inputs\_for\_generation函数，并在运行时动态地将它‘嫁接’到模型实例上（如附录P17所示）。问题迎刃而解。”**

**4. “在Fig. 4中，你遇到了OutOfMemoryError（在全参微调中）。为什么QLoRA能避免？你能估算一下显存占用吗？”**

**回答：  
“是的。全参数微调的显存开销是巨大的。以7B模型（BF16）为例：**

* **模型权重：~14 GB**
* **梯度（与权重等大）：~14 GB**
* **优化器状态（AdamW需要2倍梯度）：~28 GB**

**总计需要约 56 GB 显存，在24G卡上必然OOM。**

**而QLoRA的开销则小得多：**

* **量化模型权重：~3.5 GB**
* **Adapter的梯度和优化器状态：极小（LoRA参数量通常只有全量的0.1%不到）。**

**所以QLoRA将显存需求从50+GB断崖式下降到了几GB，这是它能成功的关键。”**

**针对P5: 技术路线选择**

**5. “QLoRA有没有什么潜在的缺点？在什么情况下，全参数微调可能依然是更好的选择？”**

**回答：  
“QLoRA并非万能。它的主要缺点是引入了额外的超参数（r, alpha），增加了调试复杂性，并且由于4-bit量化，可能存在微小的精度损失。**

**在以下情况下，我会选择全参数微调：**

1. **资源充足： 有足够的A100/H100集群。**
2. **任务差异极大： 微调任务与预训练任务截然不同，需要模型进行彻底的‘换脑’。**
3. **数据量庞大且高质量： 当有数十万甚至数百万条高质量数据时，全参数微调才能充分发挥其强大的拟合能力。”**

**针对P8 & P9: 分析原因 & 对比实验**

**6. “为什么你选择的是Qwen2-7B-Instruct，而不是其他模型？”**

**回答：  
“我的选择基于两个原则：前沿性和可控性。**

1. **前沿性： Qwen2系列在当时代表了开源模型的顶尖水平（SOTA）。用它做对照，才能知道在当前技术条件下，这个任务的‘天花板’在哪里。**
2. **可控性： Qwen2-7B与Dream-7B在规模上一致，是理想的控制变量。而且我需要的是-Instruct版本，因为它已经具备了良好的指令遵循能力，这与我的SFT任务目标一致。”**

**7. “实验结果表明基础模型很重要。你认为是什么导致了Dream-7B和Qwen2的巨大差异？如果让你来设计Dream-8B，你会怎么做？”**

**回答：  
“差异的核心几乎可以肯定在于预训练数据的质量和多样性。Qwen和Llama的成功在于它们清洗和整合了万亿级别的多语言、多领域、包含大量代码和数学推理的高质量数据。Dream-7B作为一个扩散模型转语言模型的早期尝试，很可能在预训练语料的丰富度上有所欠缺。**

**如果让我设计Dream-8B，我会把90%的精力放在数据工程上：构建一个更大规模、清洗更彻底、并刻意增加逻辑推理和数学语料的预训练集。”**

**针对P13: 最终成果 (Qwen2)**

**8. “微调后的Qwen学会了主动提问（缺少温度T）。你认为它是如何从数学数据集中学会这种交互能力的？是涌现吗？”**

**回答：  
“我认为这更多是\*\*激活（Activation）\*\*而非从零学习的涌现。**

**S1K数据集中存在一些需要基于特定条件（如标准状况）解题的例子。Qwen2-7B-Instruct的基础模型本身就具有极强的逻辑完整性意识和对话能力。我的SFT训练（强调Let's think step by step）强化了它的推理链条。**

**当它在推理过程中发现‘温度’这个必要变量缺失时，它原有的能力被‘激活’了——它知道此时最合理的行动不是胡乱猜测，而是请求缺失的信息。这正是我们希望看到的‘对齐’行为。”**

**针对P15: 未来展望**

**9. “你提到了用BLEU评估。但S1K是数学题，BLEU合适吗？你会选择什么指标？”**

**回答：  
“您抓住了要害，BLEU确实不适合S1K。我在PPT中提到BLEU是用于评估‘文本与答案的相似性’的通用说法，对于数学题来说并不准确。**

**对于S1K，最合适的指标是 Exact Match (EM，精确匹配)，即最终答案是否与标准答案完全一致。**

**更进一步，为了评估推理过程，我会考虑：**

1. **步骤级验证： 使用正则表达式或更严格的解析器来检查Chain-of-Thought中的关键步骤是否正确。**
2. **LLM-as-a-Judge： 用一个更强大的模型（如GPT-4）来评判解题逻辑的合理性。”**