《深度学习中的最优化方法》期末项目

项目要求

1. 本期末项目为小组合作的形式，同学们可自行组队，每组最多 3 人（以组队文档为准）。期末项目共有 10 个选题，每个选题限额 6 组，组队汇总表将会在 12 月 11 日 20:00 放出，选题先到先得，特殊情况可联系助教。选题结束后，仍未完成组队的同学可联系助教协助组队。

2. 本期末项目需提交**一份项目报告**与**一套实验代码**。

3. 请各组在 2025 年 1 月 17 日 23:59 前将项目报告与实验代码压缩发送至公邮 dlopt_2024@163.com，邮件主题命名为“【期末项目】第 X-X 组-姓名 1（-姓名 2-姓名 3）”（例：“【期末项目】第 3-1 组-张优化-李通义-王传播”，附件命名为“第 X-X 组-选题编号-姓名 1（-姓名 2-姓名 3）.zip/.rar/.7z...”.

4. 项目报告要求：

   (a) 建议使用 LaTeX 排版，也可使用 word 排版，请将终稿转为 PDF 格式提交。建议项目报告 5 页以上，30 页以下，杜绝无意义的字数累积。

   (b) 请使用中文作为主要语言撰写项目报告，不可进行大段的英文叙述。规范引用，杜绝“照搬照抄”现象。

   (c) 请在报告开篇写明每位成员的分工与贡献。

   (d) 项目报告需要汇报人将项目背景与目标、相关调研的文献综述、自行实验的结构设计、参数选择、实验结果等内容整合为一个有机整体。

   (e) 如实汇报实验结果（其有可能与理论相悖），不得捏造结果。

   (f) 报告简明扼要、逻辑清晰，不要在正文中大量粘贴程序代码。

   (g) 报告撰写帮助：对已有工作的调研可以参考综述的写法，同时加入自己的深入理解；动手实践的部分可梳理自己实验探究的整体逻辑，指出哪些内容是自己独立完成的，实验的结果如何解释，又是如何验证既有结论的；最后可以总结这类方法目前的主要应用场景、目前存在哪些问题、以及对未来前景的展望等。无论如何，选题明确、思路新颖、逻辑清晰、详略得当的报告有助于取得更好的成绩。

5. 实验代码要求：

   (a) 除主要代码外，请在代码文件夹内建立“readme”文件以简述代码的运行逻辑，建议使用 Jupyter Notebook（.ipynb 文件）展示代码的运行结果。

   (b) 代码组织规范（考虑 Pycharm 的格式化工具）、依赖关系清晰。详尽的注释有助于提高成绩。无法正常运行的程序会影响成绩。

(c) 使用开源代码或仓库需要如实声明，有借鉴嫌疑且未声明来源的部分可能会影响成绩评价。

(d) 对于小型数据集或自己整理的中小型数据集（50MB 以下），可以将其随代码一并打包。而对于微调等大存储需求的项目，只需提交相关代码，声明使用的官方数据集及开源模型即可，无需提交微调后的模型。如确实有必要（譬如自己整理的大型数据集或与官方数据集有差异的数据集、自己预训练的中型模型等）可考虑上传至北大网盘并将共享链接随邮件发出。

6. 完成度要求：所有题目均包含三个任务，任务一着重阅读与调研；任务二着重代码与实践；任务三则是总结与提高。无论如何，任务一与任务二应当尽可能完成，才能保证获得应有的分数；如果有余力，可尝试探究包括但不限于任务三中的拓展方向，以获得更好的分数反馈。

7. 算力需求：部分题目可能需要较强的算力需求，同学们需要自行解决。如有需要，可联系代码侧助教寻求算力获取方式建议，但我们不会直接提供算力支持。

8. 其他问题：如有上述未涵盖的问题，请及时询问教师或助教。

以下是每个选题的具体要求，祝同学们圆满完成任务!

1 Variants of Adam Optimizer in Deep Learning Training

Adam 优化器及其变种在深度学习模型的训练中具有广泛的应用。本次任务将围绕几种 Adam 变种展开详细研究，并对其性能进行复现与比较。

- **任务要求：**

  1. 阅读与 Adam 优化器变种相关的文献，包括但不限于 Adam-mini, AdamW, AdaFactor, Sophia 等算法。建议调研这些优化器的设计动机、数学原理、优化过程及其适用场景；比较这些算法的独特优势、劣势及适配的任务类型；提出可能的改进方向等。
  2. 在指定任务和数据集（如 CIFAR-10 分类任务）上，使用上述几种优化器分别训练一个中等规模的深度神经网络（如 ResNet18）。比较不同优化器在模型收敛速度、最终性能（如准确率）、内存占用及计算效率上的表现。提供详细的实验方法、超参数设置和实验结果分析。
  3. 如有余力，可探究其他与 Adam 优化器相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R1-1] Loshchilov I. Decoupled weight decay regularization[J]. arXiv preprint, arXiv:1711.05101, 2017.

[R1-2] Shazeer N, Stern M. Adafactor: Adaptive learning rates with sublinear memory cost[C]//International Conference on Machine Learning. PMLR, 2018: 4596-4604.

[R1-3] Liu H, Li Z, Hall D, et al. Sophia: A scalable stochastic second-order optimizer for language model pre-training[J]. arXiv preprint, arXiv:2305.14342, 2024.

[R1-4] Zhang Y, Chen C, Li Z, et al. Adam-mini: Use fewer learning rates to gain more[J]. arXiv preprint, arXiv:2406.16793, 2024. [R1-5] Kingma D P. Adam: A method for stochastic optimization[J]. arXiv preprint, arXiv:1412.6980, 2014.

2 Mixed Precision Training

在模型训练中，为了提升计算效率，并有效减小内存开销，常使用混合精度训练（Mixed precision training）策略。本项目中，将对混合精度训练方法做初步了解，并对某些典型混合精度训练方法进行简单尝试。

- **任务要求：**

  1. 广泛阅读与混合精度训练相关的文献，深入理解不同混合精度训练方法或范式的引入动机、算法结构、数学原理、算法独特优势与潜在缺陷。
  2. 在至少两类模型上（至少一类模型的参数量不少于 100M）尝试进行混合精度训练（大模型可做微调任务）。比较在不同 batch size 下，全精度和混合精度算法占用内存、训练时间、training loss 与 test accuracy 之间的差异；测试 loss scaling 对混合精度训练的影响，并考虑 loss scaling 策略对混合精度训练的重要性，探究在去掉 loss scaling 的情况下，混合精度训练的表现情况如何。
  3. 如有余力，可探究其他与混合精度相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

  [R2-1] Automatic mixed precision (AMP) 库教程：`https://pytorch.org/tutorials/recipes/recipes/amp\_recipe.html`

3 Blockwise Pretraining or Finetuning

在大模型的训练和微调中，为了解决内存开销问题，近期的研究中将 Block coordinate descent（BCD）方法用在了大模型的训练和微调中。本项任务将探索 BCD 在大模型微调中发挥的作用。

- **任务要求：**

  1. 广泛阅读与 BCD 微调相关的文献，深入理解 BCD 相关算法范式和动机、算法结构、数学原理、算法的独特优势与潜在缺陷。（例如 BCD 在大模型预训练和微调中可以节省 activation 的内存开销,BCD 的收敛性质类似于 full finetune 等等）
  2. (a) 选定相同的模型（如 llama，roberta 等等），在相同的下游数据集上测试 LISA[R3-1], BAdam[R3-2], 以及 full finetune 的内存使用情况和测试集微调准确率，并结合之前作业中 BCD 方法的 memory 分析验证内存使用情况是否合理；
     (b) 选定一个特定模型，探索在给定下游微调数据集时，学习率和 layer 采样间隔对于验证集准确率的影响（模型和数据集的选取可以参考 LISA 和 BAdam 文中实验）；
     (c) 探索如果 BCD 的采样方式改变（例如采用 random 的采样或者 importance sampling 的方式），微调性能是否有增强。
  3. 如有余力，可探究其他与 BCD 或部分微调相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

  [R3-1] Rui Pan, Xiang Liu, Shizhe Diao, et al. LISA: Layerwise Importance Sampling for Memory-Efficient Large Language Model Fine-Tuning[J]. arXiv preprint arXiv:2403.17919, 2024.

[R3-2] Qijun Luo, Hengxu Yu, XiaoLi, et al. BAdam: A Memory Efficient Full Parameter Optimization Method for Large Language Models[J]. arXiv preprint arXiv:2404.02827, 2024.

4 Zero-order Pretraining or Finetuning

在大模型预训练和微调过程中，为了降低内存的占用，可以使用零阶梯度近似来代替实际的梯度计算，从而省去反向传播的过程及相应的内存占用。本项目中，将对以 mezo 为基础的零阶优化应用进行初步了解，并对其中经典的零阶优化方法进行简单尝试。

- **任务要求：**

  1. 广泛阅读与零阶优化相关的文献，深入理解不同零阶优化方法的动机、算法结构、数学原理、算法独特优势与潜在缺陷。
  2. (a) 在经典深度学习模型（如 CNN, resnet）上比较 sgd, mezo、deepzero 等方法在 mnist, cifar10 等数据集上的表现，探究不同的超参数如学习率、光滑半径对模型收敛的影响。。

     (b) 在相同大模型、微调任务及数据集下，比较 mezo、mezo-svrg、lozo 三者的性能差异。
  3. 如有余力，可探究其他与零阶优化相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R4-1] Gautam T, Park Y, Zhou H, et al. Variance-reduced zeroth-order methods for fine-tuning language models[J]. arXiv preprint arXiv:2404.08080, 2024.

[R4-2] Chen Y, Zhang Y, Cao L, et al. Enhancing Zeroth-order Fine-tuning for Language Models with Low-rank Structures[J]. arXiv preprint arXiv:2410.07698, 2024.

[R4-3] Malladi S, Gao T, Nichani E, et al. Fine-tuning language models with just forward passes[J]. Advances in Neural Information Processing Systems, 2023, 36: 53038-53075.

[R4-4] Chen A, Zhang Y, Jia J, et al. Deepzero: Scaling up zeroth-order optimization for deep model training[J]. arXiv preprint arXiv:2310.02025, 2023.

5 Subspace Optimization for LLM's Pretraining and Finetuning

子空间优化（subspace optimization）在大语言模型的预训练和微调中具有重要的作用。在本次任务中，我们会对大语言模型中的子空间优化技术进行详细的调研，并实际复现几种典型算法。

- **任务要求：**

  1. 广泛阅读基于子空间优化方法的大模型预训练与高效微调文献，调研各种子空间预训练与微调方法，（包括但不限于 GaLore [R5-1], GoLore [R5-2], FLora [R5-3], ReLora [R5-4], SLTrain [R5-5] 算法）。理解这些算法提出的动机、算法结构、数学原理、算法独特优势与潜在缺陷。
  2. 在相同任务和数据集下，对比上述多种子空间方法的实际性能表现，并在报告中详细介绍实验方法与实验结果。
  3. 如有余力，可探究其他与子空间优化、低秩优化相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R5-1] Zhao, Jiawei, et al. “Galore: Memory-efficient llm training by gradient low-rank projection.” arXiv preprint arXiv:2403.03507 (2024).

[R5-2] He, Yutong, et al. “Subspace Optimization for Large Language Models with Convergence Guarantees.” arXiv preprint arXiv:2410.11289 (2024).

[R5-3] Si, Chongjie, et al. “FLoRA: Low-Rank Core Space for N-dimension.” arXiv preprint arXiv:2405.14739 (2024).

[R5-4] Lialin, Vladislav, et al. “Relora: High-rank training through low-rank updates.” The Twelfth International Conference on Learning Representations. 2023.

[R5-5] Han, Andi, et al. “SLTrain: a sparse plus low-rank approach for parameter and memory efficient pretraining.” arXiv preprint arXiv:2406.02214 (2024).

6 Memory-Efficient KV Cache

KV 缓存（Key-Value Cache）在大规模语言模型的推理过程中扮演着关键角色，通过存储 Transformer 的历史计算结果（Key 和 Value）来避免重复计算，从而加速推理过程。随着模型规模的不断扩大和序列长度的增加，KV 缓存所带来的内存和计算开销逐渐成为瓶颈。因此，许多学者从不同角度提出了多种 KV 缓存的改进方法，在提高推理效率的同时，减少内存消耗，并尽可能保持模型的性能。

- **任务要求：**

1. 广泛阅读 KV 缓存相关的文献，调研各种类型的 KV 缓存改进方法，理解这些方法提出的动机、算法结构、数学原理、算法独特优势与潜在缺陷。
2. 自行选定相同的语言模型，对比包括基于窗口的改进方法（如 StreamingLLM [R6-1]），基于量化的改进方法（如 KIVI [R6-2]），原始的 KV 缓存方法在内的至少 3 种 KV 缓存及其改进方法在一个相同下游任务（如 OpenbookQA [R6-3]）上的实际表现，评估不同方法在推理时间、内存占用和生成质量方面的差异，并在报告中详细介绍实验方法与结果。
3. 如有余力，可探究其他与 KV 缓存、高效推理相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R6-1] Guangxuan Xiao, et. al., Efficient Streaming Language Models with Attention Sinks, ICLR 2024

[R6-2] Zirui Liu, et. al., KIVI: A Tuning-Free Asymmetric 2bit Quantization for KV Cache, ICML 2024

[R6-3] Todor Mihaylov, et. al., Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering, EMNLP 2018

7 Transformer Pretraining

当今 Transformer 模型已成为众多大模型的首选架构，详细了解 Transformer 的结构细节有助于我们更好理解大模型的运作机理。该项目需要同学们从零搭建一个小规模 Transformer 模型，并训练其完成

简单的任务。

- **任务要求：**

  1. 了解基于传统 Transformer 架构的改进方案（如非线性激活函数的选取 [R7-1]、残差链接的位置 [R7-2]、相对位置编码 [R7-3]、稀疏注意力机制 [R7-4] 等），并探究其改进的效果。
  2. 使用 pytorch 库搭建一个简易的 Transformer 模型（可根据自己的算力资源选择合适的模型大小），可考虑使用上述的改进方式。随后训练其完成简单的任务，以下是一个推荐的可能比较好实现的任务：使用编码器+解码器的 Transformer 架构，自行准备若干古诗句作为训练语料，训练的目标是使模型能够根据一句古诗已有的前半句自由书写出后半句。模型性能不作为主要评分依据，鼓励使用 pytorch 单元模块逐步搭建而非直接使用集成的 Transformer 模块。
  3. 如有余力，考虑将该模型封装为一个古诗写作助手。注意自回归生成的实现方式，以及如何做到古诗的押韵。

- **参考文献：**

[R7-1] Shazeer, et al. " GLU Variants Improve Transformer," arXiv:2002.05202.

[R7-2] Xiong, et al. "On Layer Normalization in the Transformer Architecture," arXiv:2002.04745.

[R7-3] Shaw et al. "Self-Attention with Relative Position Representations," arXiv:1803.02155.

[R7-4] Child et al. "Generating Long Sequences with Sparse Transformers," arXiv:1904.10509.

8 Sampling Strategy for SGD

在 SGD 算法中，由多样本构成的有限和是最常见的形式，而不同的采样方式对 SGD 算法的性质有不同的影响，我们将对其进行详细的探究。

- **任务要求：**

  1. 调研并深入理解不同的采样方式的效果及理论解释，以下是几类常见的采样方式：有放回均匀采样；重要性采样（[R8-2],[R8-3]）；分幕（epoch）固定顺序采样；分幕且每一幕进行重排（reshuffle）；分类别抽样（对于分类任务，在每一幕中无放回地集中采集每一类的数据，一类采集完成后再进入下一类）；等等。
  2. 对比不同的采样策略在实际问题中的效果，以下是推荐的任务：1）在某实际数据集（可在 LIBSVM [R8-1] 中自行挑选）上做 LASSO 任务；2）在 MNIST 上训练 VGGNet; 3）在 Cifar10 上训练 ResNet.
  3. 如有余力，可探究其他与 SGD 相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R8-1] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[R8-2] Kun Yuan et. al., Stochastic gradient descent with finite samples sizes, IEEE MLSP, 2016

[R8-3] Peilin Zhao and Tong Zhang, Stochastic Optimization with Importance Sampling for Regularized Loss Minimization, ICML 2015

9 Communication Efficient Distributed Training

随着数据量、模型规模的日渐庞大，单机训练模型已不能满足要求，我们必须要考虑多机的分布式并行训练。分布式训练中通信开销是主要需要考虑的因素，如何设计通信高效的分布式训练方式是一个重要的研究课题。课堂中我们讲解了分布式并行的三种策略，我们将在此做一深入探究。

- **任务要求：**

  1. 调研并深入理解数据并行、流水线并行 [R9-1]、张量并行 [R9-3] 的有关课题。着重探究数据并行中的通信开销节省方式（如去中心分布式训练 [R9-3]、压缩通信 [R9-4]、低频通信 [R9-5] 等）。
  2. 实践至少两种上述调研的具体算法，可在本地选用小规模问题做模拟实验，但需要在内存分配方式中体现分布式的理念（如 Python 中将不同节点的数据分配给不同对象的私有特征中，外部不可直接访问，只可通过对象内部定义的方法进行通信）。
  3. 如有余力，可探究其他与分布式训练相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R9-1] Huang et al. "GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism," arXiv:1811.06965.

[R9-2] Shoeybi et al. "Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism,". arXiv:1909.08053.

[R9-3] Shi et al. "EXTRA: An Exact First-Order Algorithm for Decentralized Consensus Optimization," arXiv:1404.6264.

[R9-4] Richtárik et al. "EF21: A New, Simpler, Theoretically Better, and Practically Faster Error Feedback," arXiv:2106.05203.

[R9-5] Jhunjhunwala et al. "FedVARP: Tackling the Variance Due to Partial Client Participation in Federated Learning," arXiv:2207.14130.

10 Speculative Decoding

投机采样（Speculative Decoding）在大规模语言模型的推理过程中展现出了巨大的潜力，通过并行运行两个模型，有望将推理速度提高 2-3 倍，同时保持模型输出的准确率。随着模型规模的不断扩大和推理任务的复杂性增加，投机采样所带来的计算效率提升逐渐成为研究热点。

- **任务要求：**

  1. 广泛阅读投机采样相关的文献，调研各种类型的投机采样改进方法，理解这些方法提出的动机、算法结构、数学原理、算法独特优势与潜在缺陷。
  2. 在相同的实验设定下，对比 [R10-1], CS Drafting [R10-2], LLMA [R10-3], REST[10-4] 在内的多种投机采样方法的性能差异，在报告中完整介绍实验参数和实验效果。
  3. 如有余力，可探究其他与投机采样、高效推理相关的具体课题；提出总结性、前瞻性的观点等。

- **参考文献：**

[R10-1] Fast Inference from Transformers via Speculative Decoding.

[R10-2] Cascade Speculative Drafting for Even Faster LLM Inference.

[R10-3] Draft & Verify: Lossless Large Language Model Acceleration via Self-Speculative Decoding.

[R10-4] REST: Retrieval-Based Speculative Decoding.