中国科学技术大学

本科毕业论文

Stable-SPAM：如何在 4 比特精度下实现比 16 比特 Adam 更稳定的训练

作者姓名：	胡皓天
学　　号：	PB21000098
专　　业：	电子科学技术
导　　师：	金西教授
完成时间：	2025 年 4 月 20 日

摘 要

本篇论文对近年来提出的多种用于 4 比特训练的优化器进行了全面评估，发现低比特精度会加剧对学习率的敏感性，且常常导致梯度范数不稳定，从而在较高学习率下引起训练发散。在这些优化器中，近期提出的 SPAM（具备动量重置与对尖峰敏感的梯度裁剪机制）在多个比特精度设置下表现最佳，但仍难以稳定梯度范数，因此需要精心调整学习率。

为了解决这些问题，本文[①]提出了 `Stable-SPAM`，其引入了增强的梯度归一化与裁剪技术。具体而言，`Stable-SPAM` 具有以下三个关键特性：（1）通过跟踪尖峰梯度的历史最大值，自适应地更新其裁剪阈值；（2）基于整个梯度矩阵的历史 l_2 范数统计对其进行归一化；（3）继承了 SPAM 的动量重置机制，周期性地重置 Adam 优化器的一阶与二阶动量，从而缓解尖峰梯度的积累问题。

大量实验证明，`Stable-SPAM` 能够有效稳定 4 比特大语言模型训练中的梯度范数，并在性能上优于 Adam 和 SPAM。值得注意的是，使用 `Stable-SPAM` 训练的 4 比特 LLaMA-1B 模型，其困惑度 (Perplexity) 比使用 Adam 训练的 BF16 精度 LLaMA-1B 模型高出最多 2 点。此外，在两者均为 4 比特训练的情况下，`Stable-SPAM` 能以大约一半的训练步数达到与 Adam 相同的损失值。代码已开源，地址为：https://github.com/TianjinYellow/StableSPAM.git。

关键词：大语言模型；4 比特训练；LLM 预训练；优化器，机器学习

[①]本篇毕业论文的工作基于作者在 UT-Austin 的实习成果，已投稿至双盲评审的会议。论文版权归 VITA 研究团队共同所有，未经允许不得使用。

ABSTRACT

This paper comprehensively evaluates several recently proposed optimizers for 4-bit training, revealing that low-bit precision amplifies sensitivity to learning rates and often causes unstable gradient norms, leading to divergence at higher learning rates. Among these, SPAM, a recent optimizer featuring momentum reset and spike-aware gradient clipping, achieves the best performance across various bit levels, but struggles to stabilize gradient norms, requiring careful learning rate tuning. To address these limitations, we propose `Stable-SPAM`, which incorporates enhanced gradient normalization and clipping techniques. In particular, `Stable-SPAM` (1) adaptively updates the clipping threshold for spiked gradients by tracking their historical maxima; (2) normalizes the entire gradient matrix based on its historical l_2-norm statistics; and (3) inherits momentum reset from SPAM to periodically reset the first and second moments of Adam, mitigating the accumulation of spiked gradients. Extensive experiments show that `Stable-SPAM` effectively stabilizes gradient norms in 4-bit LLM training, delivering superior performance compared to Adam and SPAM. Notably, our 4-bit LLaMA-1B model trained with `Stable-SPAM` outperforms the BF16 LLaMA-1B trained with Adam by up to 2 perplexity. Furthermore, when both models are trained in 4-bit, `Stable-SPAM` achieves the same loss as Adam while requiring only about half the training steps. Code is available at https://github.com/TianjinYellow/StableSPAM.git.

Key Words: LLMs; LLM Pre-training; 4bit training; Machine Learning; Optimizer

目 录

符 号 说 明

a	The number of angels per unit area
N	The number of angels per needle point
A	The area of the needle point
σ	The total mass of angels per unit area
m	The mass of one angel
$\sum_{i=1}^{n} a_i$	The sum of a_i

第一章　绪论

第一节　课题研究的背景及意义

近年来，研究者提出了多种先进的优化器，声称在大语言模型（Large Language Models, LLMs）训练中，相较于广泛使用的 Adam 优化器具有更优的性能，或能在降低计算和内存成本的同时实现相当的效果。由于 LLM 模型规模巨大，如何降低 Adam 优化器的内存占用成为了该方向研究的核心目标之一[1-6]。另一项研究重点则是解决 LLM 训练过程中的不稳定性问题。例如，Huang et al.[7] 提出了 SPAM 优化器，引入了动量重置机制与对尖峰敏感的梯度裁剪（SpikeClip），以缓解损失突增带来的负面影响。Zhao et al.[8] 则分析了多种优化器在 BF16 精度下对超参数的稳定性表现。目前这些优化器主要是在 BF16 精度这一现实训练设置下进行评估的[9-10]。

随着近年来 LLM 向 FP8 和 FP4 等低比特精度迁移趋势的加快（其显著节省了计算和存储成本)[11-14]，一个关键问题在于，这些优化器在低比特精度下是否依然能够保持其效果与稳定性。若要使这些新提出的优化器在实际中具有经济性，其在低比特精度下的训练过程应具备与高精度训练时类似的鲁棒性，尤其是在超参数选择方面。

第二节　本文的主要工作及章节安排

本文对包括 Adam[15]、Adafactor[1]、Adam-mini[3] 以及 SPAM[7] 在内的多种最新优化器进行了全面评估，重点分析其在权重和激活均为 4 比特精度的训练场景下，对学习率选择的有效性与鲁棒性。我们的研究揭示了以下几个关键发现：

所有评估的优化器在 4 比特训练中对学习率选择都表现出更高的敏感性，尤其在较大学习率下容易发生训练发散，如图 1.2 所示。

1. SPAM 在多个比特精度设置下始终取得最低的评估损失，但需要精细调整学习率。相比之下，Adafactor 对学习率的鲁棒性出人意料地好，甚至在这方面优于 Adam。

2. 图 2.1 展示的训练动态分析显示，相较于 BF16 精度，4 比特训练过程中梯度范数极度不稳定，且频繁出现尖峰。这种现象常引发损失突增，甚至在较高学习率下导致训练直接发散。

3. 虽然 SPAM 中引入的 SpikeClip 技术在一定程度上缓解了 4 比特训练中梯度范数的不稳定问题，但仍不足以完全避免训练发散，如图 2.2 所示。

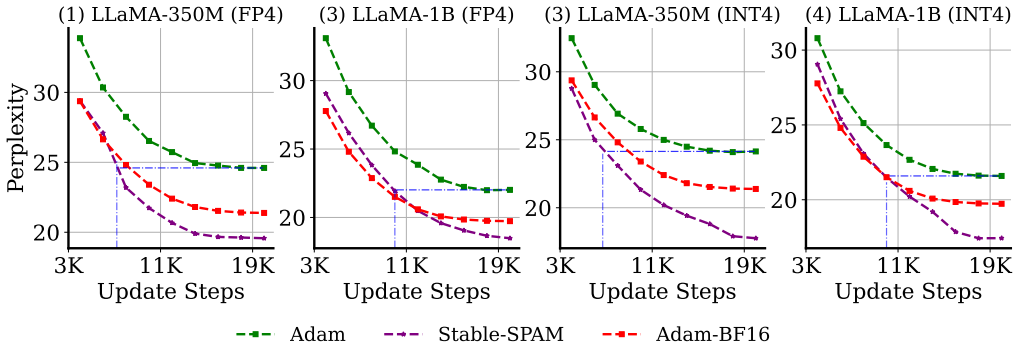

图 1.1 **4-bit LLM 训练性能对比**。所有实验均基于 C4 数据集，并在 LLaMA-130M/350M/1B 模型上进行。**Adam-BF16** 表示使用 Adam 在 BF16 精度下训练模型。结果报告了验证集上的困惑度。

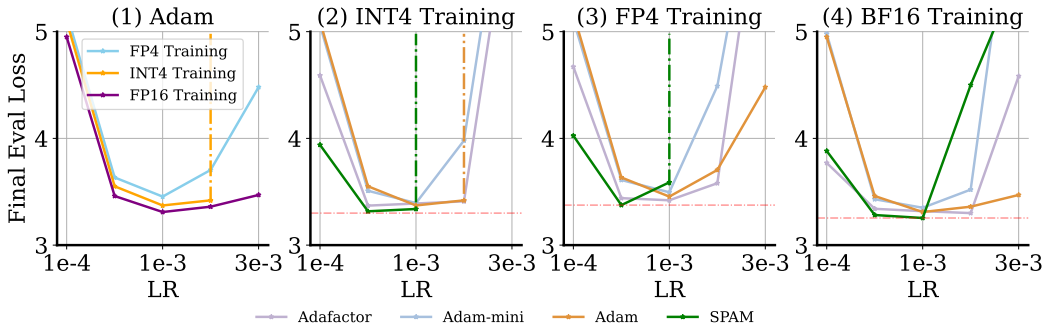

图 1.2 **在 C4 上训练 LLaMA-130M 模型时，不同学习率下的最终验证损失**。虚线垂直线表示学习率再提升后模型无法继续训练（即训练损失变为 NaN）。红色虚线表示模型在各组实验中的最佳性能。

尽管 SPAM 对学习率选择较为敏感，但其在各个比特精度设置下均能取得最低评估损失，因此为进一步改进提供了良好基础。在此基础上，本文提出了 `Stable-SPAM`，以解决低精度 LLM 训练中存在的不稳定性问题。Stable-SPAM 保留了 SPAM 的优秀性能[①]，并显著提升了训练稳定性，代表了低比特优化方法的一个重要进展。

具体而言，除继承 SPAM 原有的动量重置机制外，`Stable-SPAM` 还引入了两个关键技术：其一是自适应尖峰感知裁剪（Adaptive Spike-Aware Clipping, `AdaClip`），可对尖峰梯度进行动态裁剪；其二是自适应梯度范数归一化（Adaptive Gradient Norm, `AdaGN`），基于梯度范数的历史 l_2 统计对整个梯度矩阵进行归一化处理。我们的分析表明，这些增强机制能有效稳定 4 比特训练过程中的梯度范数，整体性能优于 Adam 和 SPAM。值得注意的是，使用 `Stable-SPAM` 训练的 4 比特 LLaMA-1B 模型，其性能优于使用 Adam 训练的 BF16 精度 LLaMA-1B

[①]此外，表 4.5 的实验结果也显示，本文提出的技术同样能够提升其他优化器的表现。

模型。此外，当两者均在 4 比特精度下训练时，`Stable-SPAM` 所需的训练步数仅为 Adam 的约一半，便能达到相同的损失水平。

Stable-SPAM 同样提升了 MoE 模型的训练稳定性。我们使用 Stable-SPAM 和 Adam 优化器，在 BF16 精度下训练了 OLMoE-1B-7B 模型[16] 共 3,000 步。下表的结果显示，Stable-SPAM 在整个训练过程中均显著优于 Adam，这表明 Stable-SPAM 有效增强了 MoE 的训练稳定性。

表 1.1　在总计 3,000 步更新下，OLMoE 模型的验证损失。Adam 和 Stable-SPAM 的学习率均设为 $\mathrm{lr} = 1 \times 10^{-4}$。

优化器	第 1,000 步	第 2,000 步	第 3,000 步
Adam	5.43	4.73	4.65
Stable-SPAM	5.28	4.25	4.11

Stable-SPAM 通过稳定异常梯度来改善训练过程，这一现象同样出现在非量化训练（例如 BF16）中。已有研究[17-18] 记录了 BF16 语言模型训练中的梯度和损失突增现象，这也解释了为何 Stable-SPAM 能优于 BF16 下的 Adam。为了进一步验证其在语言建模之外的普适性，我们在强化学习（MuJoCo）和时间序列预测（天气预报）任务中开展了额外实验，特别是在天气预测训练集中人为地引入了 10% 的异常数据以加剧训练不稳定性。如下面的表格所示，Stable-SPAM 在这些任务中依然稳定地优于 Adam，体现出其在多种任务中的广泛有效性。

表 1.2　在三个 MuJoCo 环境（HalfCheetah、Ant、Hopper）中的最终测试奖励

优化器	HalfCheetah	Ant	Hopper
Adam	5276.1 ± 1542.9	3835.6 ± 759.5	2447.5 ± 1037.9
Stable-SPAM	6762.6 ± 1414.2	4907.6 ± 954.6	3435.1 ± 1178.3

表 1.3　天气预测任务中的最终测试损失。报告的是 10 次重复实验的平均测试损失。异常数据通过向 10% 随机选取的输入值添加高斯噪声生成：$X = X + \mathcal{N}(0, \mathrm{Severity} \times \max(X))$，其中 X 为输入，S 为严重程度。

优化器	$S = 0$	$S = 2$	$S = 5$
Adam	0.151	0.2003	0.346
Stable-SPAM	0.150	0.186	0.316

第二章　4 比特训练稳定性探索

近期已有多项研究工作[7-8,19,19-20] 针对大语言模型（LLM）训练中的稳定性问题展开了探讨，涵盖了诸如学习率不稳定、梯度尖峰（gradient spikes）以及损失突增（loss spikes）等现象。在本节中，我们在 4 比特 LLM 训练环境下进一步分析了不同优化算法的稳定性表现。

遵循[8,19] 所描述的实验设置，我们使用从 1e-4 到 3e-3 范围内的一系列学习率来评估各优化器的最终性能。本次评估涵盖了两种广泛使用的优化器：Adam[15] 和 Adafactor[1]，以及两种近期提出的新方法：Adam-mini[3] 和 SPAM[7]。

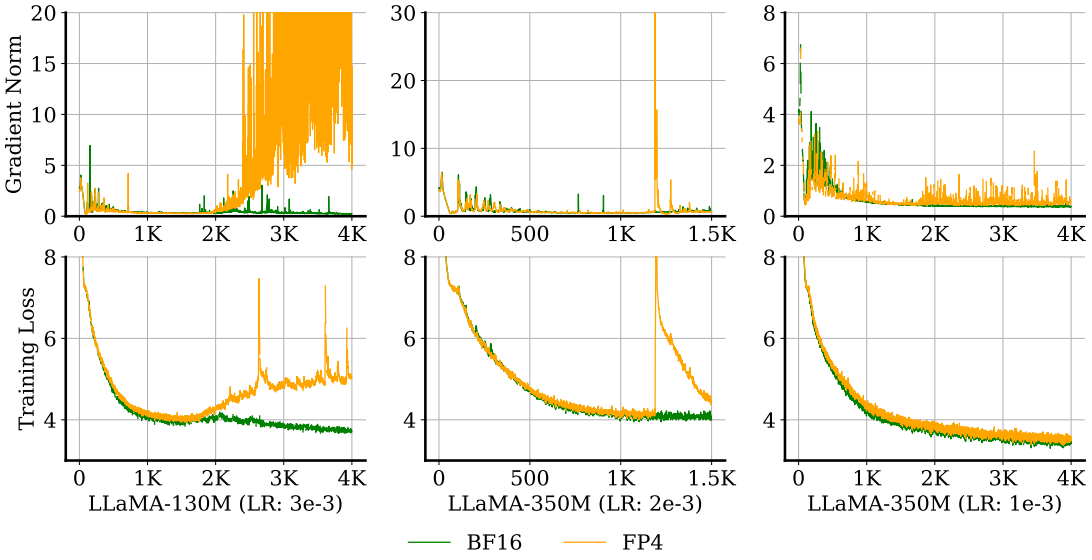

图 2.1　**在 BF16 与 FP4 精度下，使用不同学习率训练时的损失与梯度范数变化（Adam 优化器）**。实验在相同的训练配置下，基于 LLaMA-130M 和 350M 模型进行。

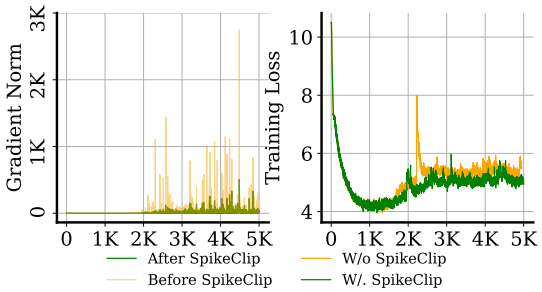

图 2.2　**SpikeClip[7] 对训练稳定性的影响**。左图：应用梯度突变剪裁前后的梯度范数对比。右图：使用与不使用梯度突变剪裁时的训练损失对比。模型采用基于 LLaMA-130M 和 C4 的 Adam 优化器进行训练。

此外，我们在整个 4 比特精度训练过程中持续监控全局梯度范数与训练损失。全局梯度范数定义如下：$\sqrt{\sum_{i=0}^{N} \|g_i\|_2^2}$ 其中 N 表示模型的层数，g_i 表示第 i 层的梯度。

所有实验均在 LLaMA-130M 和 350M 模型上，使用 C4 数据集进行，结果展示于图 1.2 和图 2.1 中。我们观察到如下现象：

一、低比特训练的学习率稳定性较差

如图 1.2 所示，随着学习率的增大，4 位精度训练的最终评估损失显著上升，而 BF16 精度训练在不同学习率下则表现出更为稳定的性能。这表明，在学习率方面，4 比特训练更加敏感且稳定性更差。

二、低比特训练更容易出现损失尖峰和梯度范数尖峰

图 2.1 展示了 LLaMA-130M 和 LLaMA-350M 模型在 BF16 与 FP4（E1M2）精度下，使用不同学习率训练时的训练损失与梯度范数变化曲线。可以观察到，BF16 精度训练过程整体较为平稳，而 FP4 精度训练则频繁出现显著的损失尖峰，且无论模型大小如何，均会出现类似现象。此外，这些损失尖峰总是伴随着梯度范数的剧烈爆发。

三、SPAM 在 4 比特训练中表现最佳但对学习率非常敏感

如图 1.2 所示，SPAM 在 INT4 或 FP4 设置中使用最优学习率时，在多个优化器中取得了最低的验证损失。然而，当学习率上升时，其验证损失往往会迅速上升，甚至发散为 NaN。

此外，我们对 SPAM 中提出的尖峰裁剪技术（SpikeClip）在训练过程中的梯度范数与损失曲线进行了监控。SpikeClip 利用梯度的二阶矩来检测并抑制异常梯度，其具体表达形式如下：

$$g_i = \text{sign}(g_i) \cdot \sqrt{\theta V_i} \quad \text{满足} \quad \frac{g_i^2}{V_i} > \theta \tag{2.1}$$

其中，g_i 为第 i 个梯度分量，V_i 为对应的二阶动量，θ 为预定义阈值（其原始论文中默认使用 5000）。

我们的实验证明，SpikeClip 在一定程度上可以缓解损失尖峰问题，但并不能完全阻止训练发散。一种可能的解释是，SpikeClip 是逐元素操作的，所使用的阈值可能偏高；若所有梯度分量同时整体升高，该机制仍可能保留较大的整体梯度范数。由于其仅关注个别异常值的裁剪，未能有效处理整体偏大的梯度情况。

这一问题也在图 2.2 中得到了验证，图中显示即便应用了 SpikeClip，梯度范数依然保持在较高水平。

第三章　Stable-SPAM

为了解决 4 比特精度大语言模型（LLM）训练过程中的不稳定性问题，本文提出了一种稳定化的尖峰感知 Adam 优化器——Stable-SPAM。除继承自原始 SPAM 的动量重置机制外，Stable-SPAM 还引入了两项关键技术：自适应梯度范数归一化（AdaGN）与自适应尖峰感知裁剪（AdaClip）。这两项技术将在下文中进行详细说明。伪代码见附录 三。

第一节　自适应梯度范数归一化（AdaGN）

如图 2.1 和图 2.2 所示，训练过程中损失的尖峰现象以及训练发散通常与梯度范数的剧烈上升同时出现，这与已有研究[7,20] 的发现一致。为应对这一问题，本文提出了 AdaGN 方法，通过基于梯度历史 l_2 范数统计信息对其进行自适应缩放，从而稳定训练过程中的梯度。

我们借鉴 Adam 优化器的思想，维护梯度范数的一阶与二阶动量的滑动平均值，以追踪其动态变化。具体计算过程如下：

$$g_{\text{norm}} = |g_t|^2 \tag{3.1}$$

$$m_{\text{norm}} = \gamma_1 \cdot m_{\text{norm}} + (1-\gamma_1) \cdot g_{\text{norm}} \tag{3.2}$$

$$v_{\text{norm}} = \gamma_2 \cdot v_{\text{norm}} + (1-\gamma_2) \cdot g_{\text{norm}}^2 \tag{3.3}$$

$$\hat{g}_t = \frac{g_t}{g_{\text{norm}}} \cdot \frac{m_{\text{norm}}}{\sqrt{v_{\text{norm}}} + \epsilon} \tag{3.4}$$

其中，$\hat{g}t$ 表示归一化后的梯度，γ_1 与 γ_2 分别为一阶与二阶动量的衰减系数，ϵ 为数值稳定性常量。通过将当前梯度 g_t 按照其历史均值 $m$norm 与平方均值 $\sqrt{v_{\text{norm}}}$ 之间的比例进行缩放，AdaGN 有效抑制了梯度范数的剧烈波动。

需要注意的是，由于梯度范数 g_{norm} 是针对每一层而言的一个标量，因此 AdaGN 引入的参数开销几乎可以忽略（每层仅需两个额外的标量参数）。

第二节　自适应尖峰感知裁剪（AdaClip）

不同于 SPAM[7] 所采用的固定阈值尖峰裁剪方法，本文提出了自适应裁剪机制 AdaClip。其核心思想是：通过跟踪梯度分量的历史最大值，动态调整裁剪阈值，而非依赖预设的静态常数。

设 g_t 为第 t 步的梯度，我们首先计算出当前梯度张量中各分量绝对值的最大值 $g_{\max}$，再使用指数滑动平均更新裁剪阈值 $T_{\text{threshold}}$，并将所有超出该阈值的梯度分量按比例缩放。具体操作如下所示：

$$g_{\max} = \max_i(|g_t[i]|), \tag{3.5}$$

$$T_{\text{threshold}} = \gamma_3 \cdot T_{\text{threshold}} + (1-\gamma_3) \cdot g_{\max}, \tag{3.6}$$

$$\text{Mask}_{\text{spikes}} = (g_t > T_{\text{threshold}}), \tag{3.7}$$

$$g_t[\text{Mask}_{\text{spikes}}] = \frac{g_t[\text{Mask}_{\text{spikes}}]}{g_{\max}} \times T_{\text{threshold}}. \tag{3.8}$$

其中，$\gamma_3 \in [0,1]$ 控制滑动平均的响应速度。γ_3 越大，$T_{\text{threshold}}$ 对新的尖峰响应越慢，从而更新更平滑；反之则更新更快，适应性更强。

第三节　动量重置（MoRet）

与 Huang et al.[7] 一致，Stable-SPAM 采用动量重置机制（MoRet），周期性地重置 Adam 优化器中一阶与二阶动量的累计值。这一机制的作用在于消除尖峰梯度对动量历史造成的长效干扰。

由于 Adam 使用指数滑动平均来记录梯度历史信息，若某一步骤中出现极大的尖峰梯度，其数值将会在较长时间内持续影响优化器状态，甚至引发训练不稳定[7]。通过每隔固定步数 ΔT 对动量项进行重置，MoRet 能有效消除异常梯度的累积影响，从而实现更加稳定、可靠的训练过程。

表 3.1　**在 C4 数据集上，不同优化器用于 LLaMA 模型的 INT4 与 FP4 训练的比较。**

	INT4 Training			**FP4 Training**		
	130M	**350M**	**1B**	**130M**	**350M**	**1B**
Adam	26.4	24.14	21.59	28.9	24.59	22.01
Adam+GradClip	26.30	21.64	19.74	28.27	20.84	20.25
Adafactor	25.11	20.45	20.65	26.89	20.53	20.03
SPAM	25.03	20.19	19.98	26.78	20.35	19.74
Stable-SPAM	**24.33**	**17.76**	**17.42**	**26.31**	**19.49**	**18.48**
Adam (BF16)	24.53	21.38	19.73	24.53	21.38	19.73
Training Tokens	2.2B					

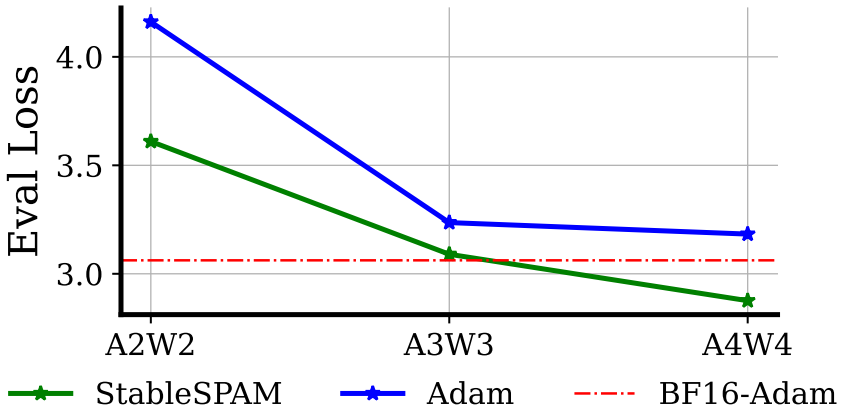

图 3.1 **StableSPAM 在极低精度训练下的表现**。实验使用 C4 数据集上的 350M 模型进行。**BF16-Adam** 表示模型使用 Adam 优化器在 BF16 精度下进行训练。报告的是验证集上的最终损失。

第四章　本文实验设置与结果分析

第一节　实验设置

为了验证所提出的 `Stable-SPAM` 的有效性，我们在 C4 数据集上对不同规模的 LLaMA 模型进行了大量实验。

一、基线方法（Baselines）

我们选取了五种流行的优化器作为基线方法，包括 Adam[15]、Adafactor[1]、Lion[2]、Adam-mini[3] 和 SPAM[7]。其中，Adam 和 Adafactor 是经过广泛验证和广泛使用的优化器，而 Adam-mini 和 SPAM 则是近期提出的新方法。此外，我们还将梯度裁剪（GradClip）[21] 与 Adam 结合作为额外的基线方法进行对比。

二、实验设置（Experimental Setup）

参考[4,22]，我们在参数规模从 60M 到 1B 的 LLaMA 架构上进行训练。所有模型均采用 RMSNorm[23] 和 SwiGLU 激活函数[24]。对于每个模型规模，我们在各方法中保持相同的超参数配置，仅调整学习率。具体来说，我们对每个优化器的学习率从 1×10^{-4} 到 1×10^{-3} 进行网格搜索，步长为 2×10^{-4}。按照[7,20]中的设置，我们将 GradClip 基线的裁剪阈值设为 1。对于 Adafactor，我们采用原始论文[1]中的超参数设置，即 $\epsilon_1 = 10^{-30}$、$\epsilon_2 = 10^{-3}$ 和 $d = 1.0$。SPAM 的超参数配置参考[7]，将重置间隔设置为 500，学习率预热步数设置为 150，GSS 阈值设置为 5000。对于 `Stable-SPAM`，在 4 比特 LLM 训练中，我们设定 $\gamma_1 = 0.7$、$\gamma_2 = 0.9$、$\theta = 0.999$；在 BF16 训练中，则设定 $\gamma_1 = 0.85$、$\gamma_2 = 0.9999$ 和 $\gamma_3 = 0.999$。更详细的任务设置和超参数信息请参见附录 一。

三、4 比特 LLM 训练的表现

为评估在 4 比特 LLM 训练中的表现，我们采用 FP4（E1M2: 1 位指数，2 位尾数）和 INT4（4 位整数）两种感知量化训练策略进行实验。C4 数据集上不同规模 LLaMA 模型的训练曲线如图 1.1所示，最终困惑度结果汇总于表 3.1中。

我们观察到与 BF16 训练相比，4 比特训练会导致显著的性能下降。如表 3.1所示，BF16（Adam）与 INT4/FP4（Adam）之间的困惑度差距在所有模型规模中均超过 1.5，凸显了低精度训练带来的挑战。图 1.1表明，在 4 比特场景下，始终显

著优于 Adam，甚至超过了 16 位 Adam 的表现。表 3.1进一步展示了优于其他先进优化器，如 Adafactor 和 SPAM。在所有基线方法中，引入 GradClip 有助于降低困惑度，而 Adafactor 与 SPAM 的表现均优于仅使用 GradClip 的方法。能在仅使用一半训练 Token 的情况下达到与 Adam 相同的性能。如图 1.1所示，在大约一半训练步骤内达到了与 Adam 相同的困惑度。值得注意的是，在更大规模模型（如 LLaMA-350M 和 LLaMA-1B）上的表现尤为出色，展现出在大规模训练场景中的潜力。这可能是因为大模型的低精度训练更容易出现不稳定问题[25]，因此像这样具备稳定特性的训练方法尤为受益。

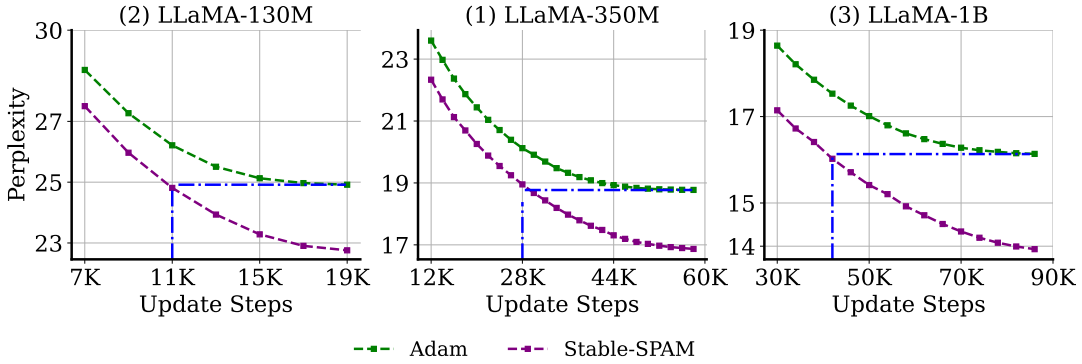

图 4.1 **Performance of BF16 training with various model sizes.** Experiments are based on LLaMA models trained on C4 Dataset.

四、极低精度训练的表现

为评估在极低精度训练下的表现，我们在 LLaMA-350M 上进行了实验，采用了 A2W2（INT2）、A3W3（INT3）和 A4W4（INT4）配置。最终验证损失如图 3.1所示。结果表明，在所有低精度设置中，始终优于 Adam，并且在 INT3 训练下的表现甚至与 BF16-Adam 相当。

五、BF16 精度 LLM 训练的表现

为进一步评估的有效性，我们在多个不同规模的 LLaMA 模型上进行了标准 BF16 训练实验，实验基于 C4 数据集。训练曲线和最终困惑度分别如图 4.1 和表 4.1 所示。表 4.1 显示，在所有模型规模上都表现出优越的性能，显著超过排名第二的优化器。

此外，图 4.1 显示，在 LLaMA-350M 和 LLaMA-1B 上，在仅使用一半甚至更少训练步数的情况下即可达到与 Adam 相同的性能，验证了在 BF16 LLM 训练中能够以更少的训练 Token 达到相同的效果。上述结果表明，的优势不仅适用于低精度训练，也适用于标准的 BF16 训练。

表 4.1 **BF16 精度训练中不同优化器的比较**。报告的是困惑度（Perplexity）。

Optimizer	60M	130M	350M	1B
Adam-mini	34.10	24.85	19.05	16.07
Adam	34.09	24.91	18.77	16.13
Adam + GradClip	33.33	24.88	18.51	15.22
Adafactor	32.57	23.98	17.74	15.19
SPAM	30.46	23.36	17.42	14.66
gray!20 `Stable-SPAM`	**28.84**	**22.21**	**16.85**	**13.90**
Training Tokens	1.1B	2.2B	6.4B	11.6B

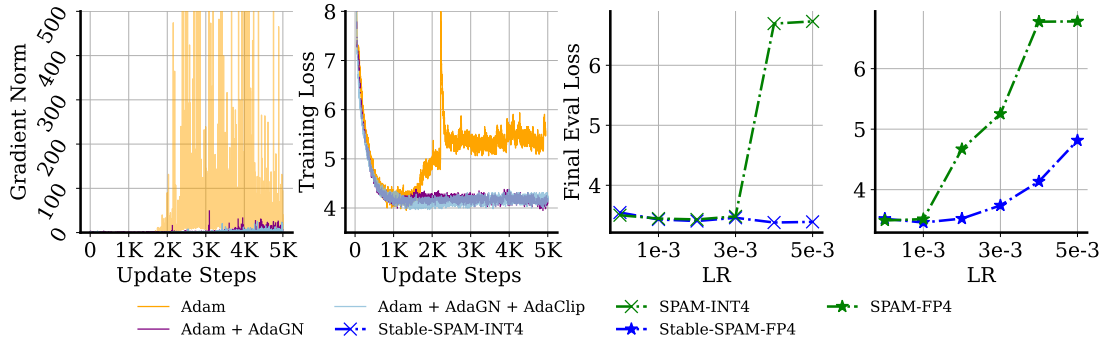

图 4.2 **AdaGN 与 AdaClip 对稳定 FP4 LLM 训练的作用**。左两图为 LLaMA-130M（学习率 = 3e-3），右两图为 LLaMA-60M。

六、MoE 模型训练表现

Stable-SPAM 同样提升了 MoE 模型的训练稳定性。我们使用 Stable-SPAM 和 Adam 优化器，在 BF16 精度下训练了 OLMoE-1B-7B 模型[16] 共 3,000 步。下表的结果显示，Stable-SPAM 在整个训练过程中均显著优于 Adam，这表明 Stable-SPAM 有效增强了 MoE 的训练稳定性。

表 4.2 在总计 3,000 步更新下，OLMoE 模型的验证损失。Adam 和 Stable-SPAM 的学习率均设为 $\mathrm{lr} = 1 \times 10^{-4}$。

优化器	第 1,000 步	第 2,000 步	第 3,000 步
Adam	5.43	4.73	4.65
Stable-SPAM	5.28	4.25	4.11

七、强化学习任务中训练表现

Stable-SPAM 通过稳定异常梯度来改善训练过程，这一现象同样出现在非量化训练（例如 BF16）中。已有研究[17-18] 记录了 BF16 语言模型训练中的梯度和

损失突增现象，这也解释了为何 Stable-SPAM 能优于 BF16 下的 Adam。为了进一步验证其在语言建模之外的普适性，我们在强化学习（MuJoCo）和时间序列预测（天气预报）任务中开展了额外实验，特别是在天气预测训练集中人为地引入了 10% 的异常数据以加剧训练不稳定性。如下面的表格所示，Stable-SPAM 在这些任务中依然稳定地优于 Adam，体现出其在多种任务中的广泛有效性。

表 4.3　在三个 MuJoCo 环境（HalfCheetah、Ant、Hopper）中的最终测试奖励

优化器	HalfCheetah	Ant	Hopper
Adam	5276.1 ± 1542.9	3835.6 ± 759.5	2447.5 ± 1037.9
Stable-SPAM	6762.6 ± 1414.2	4907.6 ± 954.6	3435.1 ± 1178.3

表 4.4　天气预测任务中的最终测试损失。报告的是 10 次重复实验的平均测试损失。异常数据通过向 10% 随机选取的输入值添加高斯噪声生成：$X = X + \mathcal{N}(0, \text{Severity} \times \max(X))$，其中 X 为输入，S 为严重程度。

优化器	$S = 0$	$S = 2$	$S = 5$
Adam	0.151	0.2003	0.346
Stable-SPAM	0.150	0.186	0.316

第二节　与其他优化器的集成

虽然 `AdaGN` 与 `AdaClip` 是专为设计的，但我们也探讨了它们是否可以与其他优化器兼容。为此，我们将 `AdaGN` 和 `AdaClip` 应用于两种近期提出的优化器：Lion[2] 和 Adam-mini[3]。我们分别对 Lion 与 Adam-mini 原始版本，以及与 `AdaGN` 和 `AdaClip` 组合的版本进行了对比实验，实验设定为 4 比特训练，模型为 LLaMA-60M 和 LLaMA-130M，数据集为 C4。

表 4.5 的结果显示，在 LLaMA-60M 与 130M 下，`AdaGN` 和 `AdaClip` 在 INT4 和 FP4 训练场景中均能稳定提升 Lion 与 Adam-mini 的性能。值得注意的是，在 LLaMA-130M 的 INT4 训练中，Lion 的困惑度提升高达 5.88，而在 LLaMA-60M 的 FP4 训练中，Adam-mini 的提升为 1.72。这些改进充分说明了 `AdaGN` 与 `AdaClip` 的通用性与有效性。

第三节　对训练稳定性的影响

为验证我们提出的 `AdaGN` 和 `AdaClip` 在提升 LLM 训练稳定性方面的有效性，我们进行了如下两方面的分析：

首先，我们比较了三种设置下的训练损失与梯度范数曲线：仅使用 Adam、

表 4.5 **AdaGN 与 AdaClip 在 Lion 与 Adam-mini 优化器上的表现**。实验基于 LLaMA-60M/130M 的 4 比特训练。

Optimizers	INT4 训练		FP4 训练	
	60M	**130M**	**60M**	**130M**
Lion	39.36	35.28	39.89	34.20
Lion+AdaGN+AdaClip	**38.49**	**29.40**	**36.75**	**31.63**
Adam-mini	34.84	29.79	36.37	32.95
Adam-mini+AdaGN+AdaClip	**34.61**	**29.65**	**34.65**	**32.39**
Training Tokens	1.1B			

使用 Adam + AdaGN、以及使用 Adam + AdaGN + AdaClip。实验采用 FP4 精度，在 LLaMA-130M 上进行，学习率为 3e-3。如图 4.2 所示，单独使用 Adam 时，训练损失存在发散趋势，梯度范数频繁震荡；加入 AdaGN 后，训练损失开始收敛，梯度范数显著减小；在此基础上引入 AdaClip，进一步平滑了梯度范数和损失曲线。

其次,我们展示了在不同学习率(从 5×10^{-4} 到 5×10^{-3})下的最终模型表现,实验基于 LLaMA-60M, 分别采用 FP4 和 INT4 设置。如图 4.2 所示, Stable-SPAM 展现出更为平稳的性能曲线，说明其在不同学习率下依然具有较高稳定性。

上述结果充分表明, 所提出的 AdaGN 与 AdaClip 方法能够显著提升 LLM 训练的稳定性与一致性。

第四节　消融实验（Ablation Study）

为验证中三个组成部分——MoRet、AdaGN 和 AdaClip——的有效性，我们进行了全面的消融实验。具体来说，我们采用了两种方法:

(1) 我们将 MoRet、AdaGN 和 AdaClip 逐步引入到 Adam 优化器中，评估它们在 FP4 与 BF16 两种训练设置下的独立和组合性能提升;

(2) 我们将 AdaClip 替换为 SpikeClip[7]，将 AdaGN 替换为 GradClip[21]，进一步分析我们所提出组件的独特贡献。

表 4.6 总结了实验结果，主要观察如下:

MoRet 在 FP4 和 BF16 设置下均能稳定提升性能;

在 FP4 训练中，单独使用 AdaGN 提升有限，但与 AdaClip 结合后显著降低了最终困惑度;

相反，在 BF16 设置下，AdaGN 单独就能带来较大提升，加入 AdaClip 后收益较小。这种差异可能源于 FP4 训练中更频繁出现极端梯度尖峰现象，因而更需要 AdaClip 来有效修正偏移的更新方向。

最后，将 AdaClip 替换为 SpikeClip[7]、AdaGN 替换为 GradClip[21] 会导致困惑度上升，进一步验证了我们提出的 AdaGN 和 AdaClip 的有效性。

表 4.6 **Stable-SPAM 的消融实验**。实验基于 LLaMA-60M 和 C4 数据集。

Optimizer	FP4	BF16
Adam	35.47	34.09
Adam + MoRet	32.40	31.47
Adam + MoRet + AdaClip	31.97	30.29
Adam + MoRet + AdaGN	32.26	28.96
gray!20 Adam + MoRet + AdaGN + AdaClip ()	**31.40**	**28.84**
Adam + MoRet+AdaGN+SpikeClip[7]	32.01	28.90
Adam + MoRet+GradClip[21]+AdaClip	31.95	29.87
gray!20 Adam + MoRet+AdaGN+AdaClip ()	**31.40**	**28.84**
Training Tokens	1.1B	

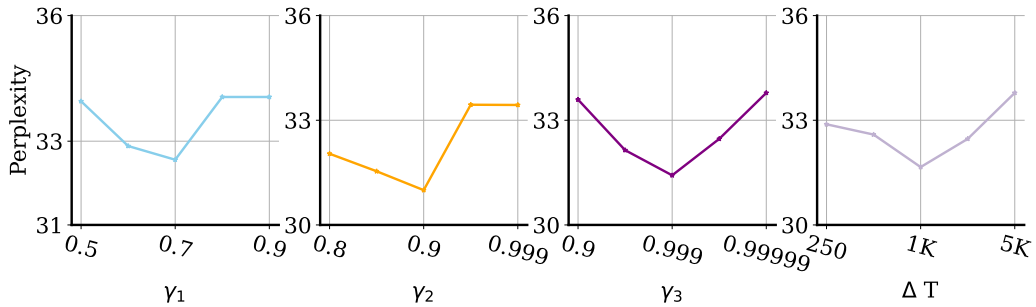

图 4.3 **超参数分析（Hyper-parameter Analysis）**。实验使用 LLaMA-60M 和 C4 数据集，在 FP4 训练设定下进行，总训练 Token 为 1.1B。

第五节 超参数分析（Hyper-Parameter Analysis）

引入了四个超参数：γ_1、γ_2、γ_3 和 ΔT，用于扩展 Adam 的功能。其中，γ_1 和 γ_2 类似于 Adam 中的 β_1 和 β_2，用于控制一阶矩 m_{norm} 和二阶矩 v_{norm} 的平滑程度。较大的 γ_1 与 γ_2 会使更新更加平滑，更加依赖历史梯度范数的统计值来调节当前梯度范数。

γ_3 则用于决定识别尖峰梯度的阈值，其值越大，阈值变化越平滑、越保守，从而导致更多梯度被归类为尖峰梯度。

为了研究这些超参数的影响，我们绘制了在不同超参数值下的最终困惑度曲线：γ_1 从 0.5 到 0.9，γ_2 从 0.8 到 0.999，γ_3 从 0.9 到 0.999，$\boldsymbol{\Delta T}$ 从 250 到 5000。实验基于 LLaMA-60M，在 C4 数据集上以 FP4 设置训练 1.1B Token。

如图 4.3 所示，过小或过大的超参数值都会导致性能下降。然而，这些超参数具有直观的解释，使得它们的调节过程相对简单，并且通常只需少量调整即可。在本论文中，我们采用 $\gamma_1 = 0.7$、$\gamma_2 = 0.9$、$\gamma_3 = 0.999$ 和 $\boldsymbol{\Delta T} = 1000$ 作为默认配置，已在所有 4 比特训练任务中展现良好效果。

第五章　总结与相关工作

第一节　相关工作

大语言模型训练的不稳定性。大语言模型（LLM）训练过程中的不稳定性问题，常表现为损失突增（loss spike）和灾难性发散（catastrophic divergence）[26-27]，已引发大量关于训练稳定性技术的研究。这些方法大致可以分为三类：（1）梯度预处理、（2）结构修改、（3）初始化策略。

梯度预处理方法通常在优化初期通过缩放和裁剪梯度来稳定训练。例如，梯度裁剪（gradient clipping）[21] 是一种广为人知的技术，它将梯度范数整体缩放至一个固定值。随后，Adafactor[1] 提出将裁剪对象从原始梯度改为参数更新。近期，SPAM[7] 利用历史梯度统计信息检测并裁剪异常梯度。然而，这些方法普遍存在一个问题：需要手动设定阈值。

在结构层面，Xiong et al.[28] 发现 Post-LayerNorm（Post-LN）会放大梯度，在较大学习率下易引发不稳定，而 Pre-LayerNorm（Pre-LN）能够保持梯度范数，从而实现更稳定的训练。Embed LayerNorm（Embed LN）对嵌入层进行归一化[29]，但可能影响模型性能[30]；Embed Detach[31-32] 则通过截断梯度缓解损失突增。DeepNorm[33] 通过缩放残差连接稳定超深模型，而 αReparam[34] 则利用谱归一化参数化方式防止注意力熵塌缩。

初始化策略则为模型提供了补充的稳定性优势。Scaled Embed[20] 可稳定 LayerNorm 的梯度，而 Scaled Initialization[35] 通过 $\mathcal{N}(0, \sqrt{2/(5d)}/\sqrt{2N})$ 分布降低方差。Fixup[36-37] 则完全去除 LayerNorm，启发了无归一化（norm-free）架构。尽管这些方法不断得到改进，但训练稳定性仍是大语言模型发展的关键挑战之一。

低精度大模型训练。低精度训练[38-42] 近年来成为提升训练计算效率和内存效率的有力手段。其中，FP16[43] 与 BF16[44] 是当前最广泛使用的低精度格式。为了进一步提高效率，8 比特训练逐渐受到关注。例如，LM-FP8[13] 支持 FP8 精度训练，而[25] 指出，随着训练规模扩大（如超过 250B tokens），激活值中的异常值（activation outliers）问题愈发严重，挑战了低比特数据格式的表达范围。

为了解决这一问题,[25] 提出一种平滑策略，而[45] 则引入 Hadamard 变换来缓解激活异常值的影响。此外，数据格式的选择也对训练表现有重要影响。INT8 是目前最广泛支持的低精度格式，而 FP8 在 NVIDIA Hopper GPU 架构中得到了专门支持。MX 格式[46] 虽具有更强的表达能力，但目前硬件支持尚不广泛。

在本工作中，我们重点研究了低精度训练中存在的稳定性问题，并通过优化器设计提出了改进方法。我们的方法可与现有技术兼容，为提升低精度训练的稳定性提供了互补的解决方案。

第二节　结论

本文系统研究了大语言模型在 4 比特量化训练中面临的训练不稳定性问题。我们发现，尽管低精度训练大幅降低了内存和计算成本，但它也显著增强了对学习率的敏感性，并提高了梯度与损失突增的发生概率。

为了解决上述问题，我们提出了 Stable-SPAM 优化器，该方法融合了三项关键技术：AdaClip、AdaGN 和 MoRet。在不同规模的 LLaMA 模型上进行的实证研究表明，Stable-SPAM 不仅有效提升了 4 比特训练的稳定性，还在性能上优于现有优化器，甚至在某些情况下超过了 BF16 的训练效果。

此外，我们进一步验证了这些稳定化策略的广泛适用性，AdaClip 和 AdaGN 等组件在 Lion 和 Adam-mini 等优化器中同样展现出良好效果。

参 考 文 献

[1] SHAZEER N, STERN M. Adafactor: Adaptive learning rates with sublinear memory cost[C]//International Conference on Machine Learning. PMLR, 2018: 4596-4604.

[2] CHEN X, LIANG C, HUANG D, et al. Symbolic discovery of optimization algorithms[J]. Advances in neural information processing systems, 2024, 36.

[3] ZHANG Y, CHEN C, LI Z, et al. Adam-mini: Use fewer learning rates to gain more[A]. 2024.

[4] ZHAO J, ZHANG Z, CHEN B, et al. Galore: Memory-efficient llm training by gradient low-rank projection[A]. 2024.

[5] ZHANG Z, JAISWAL A, YIN L, et al. Q-galore: Quantized galore with int4 projection and layer-adaptive low-rank gradients[A]. 2024.

[6] MA C, GONG W, SCETBON M, et al. Swan: Preprocessing sgd enables adam-level performance on llm training with significant memory reduction[A]. 2024.

[7] HUANG T, ZHU Z, JIN G, et al. Spam: Spike-aware adam with momentum reset for stable llm training[A]. 2025.

[8] ZHAO R, MORWANI D, BRANDFONBRENER D, et al. Deconstructing what makes a good optimizer for language models[A]. 2024.

[9] TOUVRON H, MARTIN L, STONE K, et al. Llama 2: Open foundation and fine-tuned chat models[A]. 2023.

[10] LI S, LIU H, BIAN Z, et al. Colossal-ai: A unified deep learning system for large-scale parallel training[C]//Proceedings of the 52nd International Conference on Parallel Processing. 2023: 766-775.

[11] LIU A, FENG B, XUE B, et al. Deepseek-v3 technical report[A]. 2024.

[12] LEE J, BAE J, KIM B, et al. To fp8 and back again: Quantifying the effects of reducing precision on llm training stability[A]. 2024.

[13] PENG H, WU K, WEI Y, et al. Fp8-lm: Training fp8 large language models[A]. 2023.

[14] XI H, LI C, CHEN J, et al. Training transformers with 4-bit integers[J]. Advances in Neural Information Processing Systems, 2023, 36: 49146-49168.

[15] KINGMA D P. Adam: A method for stochastic optimization[A]. 2014.

[16] MUENNIGHOFF N, SOLDAINI L, GROENEVELD D, et al. Olmoe: Open mixture-of-experts language models[A/OL]. 2024. arXiv: 2409.02060. https://arxiv.org/abs/2409.02060.

[17] A A, et al. On the instability of bf16 training in language models[J]. Unspecified Journal, 2023.

[18] B A, et al. Gradient spikes in mixed-precision training: An empirical study[J]. Unspecified Journal, 2024.

[19] WORTSMAN M, LIU P J, XIAO L, et al. Small-scale proxies for large-scale transformer training instabilities[A]. 2023.

[20] TAKASE S, KIYONO S, KOBAYASHI S, et al. Spike no more: Stabilizing the pre-training of large language models[A]. 2023.

[21] GOODFELLOW I. Deep learning[M]. MIT press, 2016.

[22] LIALIN V, MUCKATIRA S, SHIVAGUNDE N, et al. Relora: High-rank training through low-rank updates[C]//The Twelfth International Conference on Learning Representations. 2023.

[23] SHAZEER N. Glu variants improve transformer[A]. 2020.

[24] ZHANG B, SENNRICH R. Root mean square layer normalization[J]. Advances in Neural Information Processing Systems, 2019, 32.

[25] FISHMAN M, CHMIEL B, BANNER R, et al. Scaling fp8 training to trillion-token llms[A]. 2024.

[26] CHOWDHERY A, NARANG S, DEVLIN J, et al. Palm: Scaling language modeling with pathways[J]. Journal of Machine Learning Research, 2023, 24(240): 1-113.

[27] MOLYBOG I, ALBERT P, CHEN M, et al. A theory on adam instability in large-scale machine learning[A]. 2023.

[28] XIONG R, YANG Y, HE D, et al. On layer normalization in the transformer architecture[C]//International Conference on Machine Learning. PMLR, 2020: 10524-10533.

[29] DETTMERS T, LEWIS M, SHLEIFER S, et al. 8-bit optimizers via block-wise quantization[A]. 2021.

[30] SCAO T L, WANG T, HESSLOW D, et al. What language model to train if you

have one million gpu hours?[A]. 2022.

[31] DING M, YANG Z, HONG W, et al. Cogview: Mastering text-to-image generation via transformers[J]. Advances in neural information processing systems, 2021, 34: 19822-19835.

[32] ZENG A, LIU X, DU Z, et al. Glm-130b: An open bilingual pre-trained model [A]. 2022.

[33] WANG H, MA S, DONG L, et al. Deepnet: Scaling transformers to 1,000 layers [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2024.

[34] ZHAI S, LIKHOMANENKO T, LITTWIN E, et al. Stabilizing transformer training by preventing attention entropy collapse[C]//International Conference on Machine Learning. PMLR, 2023: 40770-40803.

[35] NGUYEN T Q, SALAZAR J. Transformers without tears: Improving the normalization of self-attention[A]. 2019.

[36] ZHANG H, DAUPHIN Y N, MA T. Fixup initialization: Residual learning without normalization[A]. 2019.

[37] HUANG X S, PEREZ F, BA J, et al. Improving transformer optimization through better initialization[C]//International Conference on Machine Learning. PMLR, 2020: 4475-4483.

[38] WANG N, CHOI J, BRAND D, et al. Training deep neural networks with 8-bit floating point numbers[J]. Advances in neural information processing systems, 2018, 31.

[39] LIN J, ZHU L, CHEN W M, et al. On-device training under 256kb memory[J]. Advances in Neural Information Processing Systems, 2022, 35: 22941-22954.

[40] XI H, CAI H, ZHU L, et al. Coat: Compressing optimizer states and activation for memory-efficient fp8 training[A]. 2024.

[41] XI H, CHEN Y, ZHAO K, et al. Jetfire: Efficient and accurate transformer pretraining with int8 data flow and per-block quantization[A]. 2024.

[42] WORTSMAN M, DETTMERS T, ZETTLEMOYER L, et al. Stable and low-precision training for large-scale vision-language models[J]. Advances in Neural Information Processing Systems, 2023, 36: 10271-10298.

[43] MICIKEVICIUS P, NARANG S, ALBEN J, et al. Mixed precision training[A]. 2017.

[44] KALAMKAR D, MUDIGERE D, MELLEMPUDI N, et al. A study of bfloat16 for deep learning training[A]. 2019.

[45] ASHKBOOS S, NIKDAN M, TABESH S, et al. Halo: Hadamard-assisted lossless optimization for efficient low-precision llm training and fine-tuning[A]. 2025.

[46] ROUHANI B D, ZHAO R, MORE A, et al. Microscaling data formats for deep learning[A]. 2023.

[47] NIE Y, NGUYEN N H, SINTHONG P, et al. A time series is worth 64 words: Long-term forecasting with transformers[A]. 2022.

附录 A　补充材料

第一节　架构与超参数

我们在本节中介绍用于 4 比特和 BF16 预训练的 LLaMA 模型架构和超参数配置，参考 Zhao et al.[4], Lialin et al.[22]。表 A.1 列出了不同模型规模下的主要超参数。

所有模型的最大序列长度均设为 256, 批大小为 512, 即总计 131K tokens。我们在所有实验中均采用 2000 步的学习率预热（warmup), 并使用余弦退火（cosine annealing）策略将学习率衰减至初始值的 10%。

表 A.1　本文所使用的 LLaMA 模型配置。

参数量	隐藏层维度	中间层维度	注意力头数	层数
60M	512	1376	8	8
130M	768	2048	12	12
350M	1024	2736	16	24
1 B	2048	5461	24	32

对于每种模型规模（从 60M 到 1B），我们在 $1e{-}4$ 到 $1e{-}3$ 范围内以 2×10^{-4} 为步长调节学习率，最终通过验证集困惑度选择最优学习率。我们分别在表 A.2 和表 A.3 中报告了在 4 比特和 BF16 训练下的详细超参数配置。

表 A.2　本文中在 4 比特预训练实验下的超参数配置。

超参数	LLaMA-130M	LLaMA-350M	LLaMA-1B
学习率（LR）	$1e-3$	$4e-4$	$2e-4$
ΔT	1000	1000	1000
γ_1	0.7	0.7	0.7
γ_2	0.9	0.9	0.9
γ_3	0.999	0.999	0.999

表 A.3　本文中在 BF16 预训练实验下的超参数配置。

超参数	LLaMA-60M	LLaMA-130M	LLaMA-350M	LLaMA-1B
标准预训练（Standard Pretraining）				
学习率（LR）	$1e-3$	$8e-4$	$4e-4$	$2e-4$
ΔT	1000	1000	1000	1000
γ_1	0.85	0.85	0.85	0.85
γ_2	0.99999	0.99999	0.99999	0.99999
γ_3	0.999	0.999	0.999	0.999

第二节　时间序列预测任务

我们在时间序列预测任务上进行了额外实验。为了模拟梯度异常现象，我们在数据中以 10% 的概率引入异常点。实验在 Weather 时间序列数据集[①]上进行，采用 PatchTST[47] 模型，重复运行 10 次，实验结果如图 A.1 所示。

实验结果显示，随着异常数据强度（S）的增加，`Stable-SPAM` 相对于 Adam 的性能优势愈发显著；此外，`Stable-SPAM` 在所有设置中均优于 SPAM，进一步验证了我们方法的稳定性和有效性。

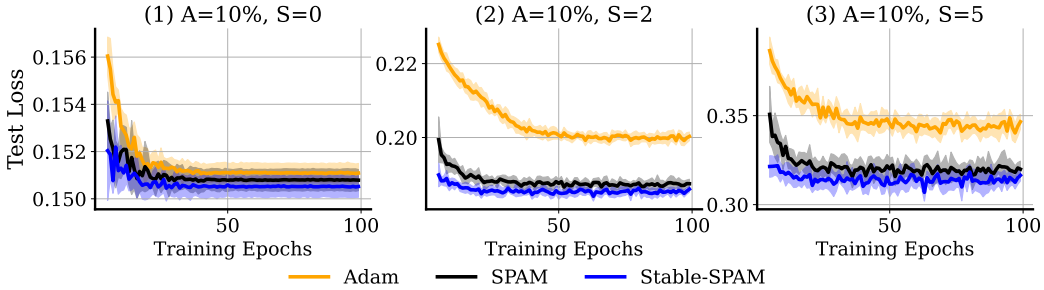

图 A.1　**Weather 时间序列数据集训练过程中的测试损失**。异常数据通过在 10% 随机选取的输入上添加高斯噪声生成，具体形式为：$X = X + \texttt{Gaussin}(0, S \cdot \texttt{Max}(X))$，其中 X 为输入，S 为异常强度。

第三节　伪代码

算法 A.1 展示了 `Stable-SPAM` 优化器的伪代码实现。

[①] https://www.bgc-jena.mpg.de/wetter/

算法 A.1 `Stable-SPAM`

Input: A layer weight matrix $w \in \mathbb{R}^{m \times n}$, learning rate α, decay rates $\beta_1 = 0.9$, $\beta_2 = 0.999$, initial parameters w_0, $\gamma_1 = 0.7, \gamma_2 = 0.9$ for `AdaGN` and $\gamma_3 = 0.999$ for `AdaClip`, momentum reset interval ΔT, small constant $\epsilon = 1 \times 10^{-6}$, and total training steps T.

Output: Optimized parameters w_T.

while $t < T$ **do**

  $g_t \in \mathbb{R}^{m \times n} \leftarrow -\nabla_w \phi_t(w_t)$ `// Gradient of the objective at step t.`

  $g_{\max} \leftarrow \texttt{Max}(\texttt{abs}(g_t))$

  $T_{\text{threshold}} \leftarrow T_{\text{threshold}} \cdot \theta + (1 - \theta)\, g_{\max}$

  $\widehat{T}_{\text{threshold}} \leftarrow \dfrac{T_{\text{threshold}}}{1 - \theta^t}$ `// Bias correction for threshold`

  $\text{Mask}_{\text{spikes}} \leftarrow \left(\texttt{abs}(g_t) > \widehat{T}_{\text{threshold}}\right)$

  if $sum\left(Mask_{\text{spikes}}\right) > 0$ **then**

    $g_t[\text{Mask}_{\text{spikes}}] \leftarrow \dfrac{g_t[\text{Mask}_{\text{spikes}}]}{g_{\max}} \times \widehat{T}_{\text{threshold}}$

  end

  $g_{\text{norm}} \leftarrow \|g_t\|_2$

  $m_{\text{norm}} \leftarrow \gamma_1\, m_{\text{norm}} + (1 - \gamma_1)\, g_{\text{norm}}$

  $v_{\text{norm}} \leftarrow \gamma_2\, v_{\text{norm}} + (1 - \gamma_2)\, g_{\text{norm}}^2$

  $\widehat{m}_{\text{norm}} \leftarrow \dfrac{m_{\text{norm}}}{1 - \gamma_1^t}, \hat{v}_{\text{norm}} \leftarrow \dfrac{v_{\text{norm}}}{1 - \gamma_2^t}$ `// Bias-corrected norm estimates`

  $\text{adaptive\_norm} \leftarrow \dfrac{\widehat{m}_{\text{norm}}}{\sqrt{\hat{v}_{\text{norm}}} + \epsilon}$

  $g_t \leftarrow \dfrac{g_t}{g_{\text{norm}}} \times \text{adaptive\_norm}$

  if $Mod(t, \Delta T) = 0$ **then**

    $\boldsymbol{m} \leftarrow \texttt{zeros\_like}(\boldsymbol{m})$

    $\boldsymbol{v} \leftarrow \texttt{zeros\_like}(\boldsymbol{v})$

  end

  $\boldsymbol{m}_t \leftarrow \beta_1\, \boldsymbol{m}_{t-1} + (1 - \beta_1)\, g_t$

  $\boldsymbol{v}_t \leftarrow \beta_2\, \boldsymbol{v}_{t-1} + (1 - \beta_2)\, g_t^2$

  $\widehat{\boldsymbol{m}}_t \leftarrow \dfrac{\boldsymbol{m}_t}{1 - \beta_1^t}$ `// bias correction`

  $\hat{\boldsymbol{v}}_t \leftarrow \dfrac{\boldsymbol{v}_t}{1 - \beta_2^t}$ `// bias correction`

  $w_t \leftarrow w_{t-1} - \alpha\, \dfrac{\widehat{\boldsymbol{m}}_t}{\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon}$

  $t \leftarrow t + 1$

end

return w_T.

致　谢

谨以此文感谢四年来遇到的最好的你们。

首先，我想由衷地感谢我的爸爸妈妈。感谢你们为我倾注的全部心血，给予我生命、陪伴我成长。我会永远铭记你们的深情厚爱，愿你们健康长寿，平安喜乐。

我衷心感谢科大的金西老师，感谢您一直以来的悉心指导与支持！正因为有您的帮助，我才能顺利完成学业并顺利毕业。

感谢我在 UCSD 暑期科研期间遇到的丁雨霏老师，您全心投入的支持与富有洞见的指导，是我科研旅程中难得的宝贵经历；感谢 UT Austin 的汪张扬老师，是一次很特别的机会让我深入大语言模型这个领域；感谢 UCI 实习期间的黄思陶老师，和您合作的那段时间让我慢慢找到了真正喜欢的研究方向。同时也感谢所有在项目合作帮助过我的师兄、合作者：振宇、钟凯、tianjin、hezi、keyi、haocheng... 在此我还想感谢郅琛学长，在我准备转专业、出国最迷茫的时候，毫无保留全心全意地帮我想办法。衷心祝愿每一位老师和同学未来顺顺利利、每天都有好心情，愿大家都能过上自己喜欢的生活，收获属于自己的小确幸与精彩！

感谢科大四年来一路相伴的好友们[①]，一起度过的快乐瞬间将永远留在我的记忆里。无论我们今后身处何方，愿我们前程似锦，万里无忧！

感谢路过我青春的所有人们。感谢你们曾经来过。

我无数遍感谢他人的爱意和真诚，让我在很多时刻都得到了救赎。开心的日子是闪着光的。无论同行的人现在去哪了，都不应该被删除被遗忘。

最后，我想感谢自己，回首刚入学时的小胡，我真心感谢自己始终乐观上进，阳光自信，越来越好。希望小胡永远健健康康，天天开心，天天向上，万事顺遂！

2025 年 4 月

[①]太多就不一一列举惹，求放过

在读期间取得的科研成果

已发表论文

1. H. Xu, H. Hu, S. Huang, *Optimizing High-Level Synthesis Designs with Retrieval-Augmented Large Language Models*, in **Proc. IEEE LLM Aided Design Workshop (LAD)**, pp. 1–5, 2024.
2. H. Hu, K. Zhong, C. Pan, X. Xiao, *Ambiguity function shaping via manifold optimization embedding with momentum*, **IEEE Communications Letters**, vol. 27, no. 10, pp. 2727–2731, 2023.
3. T. Huang, H. Hu, Z. Zhang, G. Jin, X. Li, L. Shen, T. Chen, L. Liu, Q. Wen, *et al.*, *Stable-SPAM: How to Train in 4-Bit More Stably than 16-Bit Adam*, in **Proc. International Conference on Learning Representations (ICLR) SCOPE Workshop**, 2025.
4. H. Zhang, Y. Xu, H. Hu, K. Yin, H. Shapourian, J. Zhao, R. R. Kompella, *et al.*, *Optimizing Quantum Communication for Quantum Data Centers with Reconfigurable Networks*, in **Proc. International Symposium on Computer Architecture (ISCA)**, 2025.
5. K. Zhong, J. Hu, Z. Zhao, X. Yu, G. Cui, B. Liao, H. Hu, *MIMO radar unimodular waveform design with learned complex circle manifold network*, **IEEE Transactions on Aerospace and Electronic Systems**, vol. 60, no. 2, pp. 1798–1807, 2024.
6. K. Zhong, J. Hu, Y. Cong, G. Cui, H. Hu, *RMOCG: A Riemannian manifold optimization-based conjugate gradient method for phase-only beamforming synthesis*, **IEEE Antennas and Wireless Propagation Letters**, vol. 21, no. 8, pp. 1625–1629, 2022.