基于CKKS函数型Bootstrapping的融合式同态LayerNorm方案设计

陈智罡[1]，宋新霞[2]，Liqun Chen[3]
（1.宁波财经学院人工智能学院，浙江 宁波315175；2.浙江万里学院大数据与软件工程学院，浙江 宁波315100；3.英国萨里大学计算机科学系，Guildford, Surrey萨里，GU27XH）

摘要：随着全同态加密（FHE）在隐私保护机器学习中的广泛应用，基于CKKS方案的同态Transformer推理成为研究热点。然而，Transformer中重要的层归一化（LayerNorm）组件包含倒数平方根运算，传统方法采用Goldschmidt迭代实现该运算，需额外消耗6-8层乘法深度，成为制约推理效率的关键瓶颈。本文提出一种基于CKKS函数型Bootstrapping的融合式同态LayerNorm计算方案。核心思想是在Bootstrapping的EvalMod阶段，将恢复线性值的反正弦多项式替换为直接输出倒数平方根的复合多项式，从而在自举过程中"免费"获得倒数平方根的计算结果，在输入满足校准边界且不触发可选软裁剪的默认设置下，实现倒数平方根计算的零额外乘法深度。此外，为解决正弦函数的周期别名风险，本文提出了由预缩放和统计边界构成的默认范围守卫机制，并讨论了在分布外输入下可选的软裁剪备用策略，并严格证明了在该机制下输入方差经缩放后始终落入正弦函数的单调区间内，从根本上保证计算正确性。最后对于多项式拟合，采用Remez算法在切比雪夫基上求解31阶极小极大逼近，逼近误差达10-11量级，远低于CKKS自举的固有噪声。实验结果表明，相比Goldschmidt迭代节省6层乘法深度，单次LayerNorm计算加速1.26倍，峰值内存较Goldschmidt(k=3) 节省16.7%。在BERT-Base模型的SST-2数据集上，实现了92.1% 的同态分类准确率，与明文推理（92.5%）的差距仅为0.4%。本文方案有效消除了倒数平方根计算的额外深度消耗，在性能和资源效率方面均明显优于传统Goldschmidt迭代方法。该研究为归一化函数在加密域中的高效计算提供了新思路，为同态加密Transformer推理的实际部署提供了关键支撑，具有重要的理论意义与应用价值。

关键词：全同态加密；CKKS全同态加密；层归一化；函数型Bootstrapping；Transformer；隐私保护推理
中图分类号：TP309
文献标志码：A
doi：10.11959/j.issn.1000

Design of a Fused Homomorphic LayerNorm Scheme Based on CKKS Functional Bootstrapping

Chen Zhigang[1], Song Xinxia[2], Chen Liqun[3]
1. College of Artificial Intelligence, Ningbo University of Finance and Economics, Ningbo, 315175, China
2. School of Big Data and Software Engineering, Zhejiang Wanli University, Ningbo 315100, China
3. Department of Computer Science, University of Surrey, Surrey GU2 7XH, UK

Abstract: With the widespread adoption of fully homomorphic encryption (FHE) in privacy-preserving machine learn-

收稿日期：XXXX–XX–XX；修回日期：XXXX–XX–XX
通信作者：宋新霞，sxxx@zwu.edu.cn
基金项目：国家自然科学基金资助项目（No.62571283）；浙江省自然科学基金资助项目（No.LMS26F020025）；宁波市自然科学基金资助项目（No.2025J185）；宁波财经学院高级别科研培育项目（No.1320263702）
Foundation Items: The National Natural Science Foundation of China (No.62571283), Zhejiang Provincial Natural Science Foundation of China (No.LMS26F020025), Ningbo Natural Science Foundation of China (No.2025J185), High-level Scientific Research Incubation Project of Ningbo University of Finance & Economics (No.1320263702)

ing, homomorphic Transformer inference based on the CKKS scheme has become an active research topic. However, LayerNorm—an indispensable component in Transformers—requires computing the inverse square root , which is conventionally implemented via Goldschmidt iterations at a cost of 6–8 additional multiplicative depth levels, constituting a criticalbottleneck for inference efficiency. This paper proposes a fused homomorphic LayerNorm computation scheme based on CKKS functional bootstrapping. The key idea is to replace the arcsine recovery polynomial in the EvalMod stage of bootstrapping with a composite polynomial that directly outputs the inverse square root, thereby obtaining the inverse square root "for free" during the bootstrapping process. Under the default setting, in which the input satisfies the calibrated bound and the optional soft clamping is not triggered, this incurs zero additional multiplicative depth. To address the periodic aliasing risk of the sine function, we propose a low-depth range guard consisting of pre-scaling and statistical bounding, complemented by an optional soft-clamping fallback for out-of-distribution inputs, and rigorously prove that the scaled input variance is guaranteed to fall within the monotonic interval of the sine function, fundamentally ensuring computational correctness. The polynomial fitting employs the Remez algorithm to solve for a degree-31 minimax approximation on the Chebyshev basis, achieving an approximation error on the order of 10-11, well below the inherent noise of CKKS bootstrapping. Experimental results demonstrate that the proposed scheme saves 6 levels of multiplicative depth compared to Goldschmidt iteration, achieves a 1.26×speedup for single LayerNorm computation, and reduces peak memory usage by 16.7% relative to Goldschmidt(k=3). On the SST-2 dataset with BERT-Base, this paper attains a homomorphic classification accuracy of 92.1%, only 0.4% below plaintext inference (92.5%). The results confirm that the proposed scheme effectively eliminates the additional depth overhead of inverse square root computation without sacrificing inference accuracy, significantly outperforming traditional Goldschmidt iteration in both performance and resource efficiency. This work provides a novel approach for efficient normalization computation in the encrypted domain and offers key support for the practical deployment of homomorphic Transformer inference, carrying significant theoretical and practical value.

Key words: Fully Homomorphic Encryption, CKKS, LayerNorm, Functional Bootstrapping, Transformer, Privacy-Preserving Inference

0 引言

随着人工智能技术在医疗诊断、金融风控、自然语言处理等领域的深入渗透，用户数据的隐私保护问题日益突出。传统的机器学习服务模式要求用户将原始数据上传至云端服务器进行模型推理，这不可避免地引发隐私泄露风险。全同态加密（Fully Homomorphic Encryption, FHE）[1]为这一问题提供了理想的密码学解决方案：它允许在加密数据上直接执行任意计算，计算结果解密后与在明文上计算的结果一致，从而在整个推理过程中无需暴露原始数据。

在现有的FHE方案中，CKKS[2]因其对实数近似算术的原生支持而成为隐私保护机器学习的首选方案。与处理精确整数的BGV[3]、BFV[4]方案和处理布尔电路的TFHE[5]方案不同，CKKS方案将实数向量编码为多项式环元素，通过缩放因子控制精度，天然适合深度学习中大量浮点运算的需求。然而，CKKS的乘法运算会消耗模数链中的层级（Level），当层级耗尽时必须通过自举（Bootstrapping）操作刷新密文，而Bootstrapping本身消耗大量计算资源和乘法深度。

近年来，Transformer架构[6]已成为自然语言处理和计算机视觉领域的主导模型。从BERT[7]到GPT系列[8]，Transforme的应用范围不断扩展。在Transformer的每个注意力层和前馈层中，层归一化（LayerNorm）[9]是不可或缺的核心组件，其数学定义为：$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$其中$\mu$和$\sigma^2$分别为隐藏层向量的均值和方差，$\epsilon$为稳定性常数。其简化变体RMSNorm[10]省去均值计算但保留倒数平方根作为核心运算，近年来被LLaMA[11]等大语言模型广泛采用。

在CKKS环境下，LayerNorm的大部分计算，例如均值、残差、平方求和均可通过低深度的同态加法、乘法和旋转操作高效实现。然而，其核心的倒数平方根运算却构成了严重的计算瓶颈。由于CKKS的乘法深度资源由Bootstrapping[12-13]刷新后的剩余层级决定，每一层深度都弥足珍贵。目前计

算倒数平方根的主流方法是采用 Goldschmidt 迭代[14]：每次迭代消耗2层乘法深度，达到足够精度通常需要 $k \geq 3$ 次迭代，总计消耗6-8层乘法深度。这一开销在Bootstrapping后有限的深度预算中占比极高。对于12层的BERT-Base模型，仅LayerNorm中的倒数平方根计算就累计消耗数十层乘法深度，严重制约了推理效率。

近期，RBOOT[15]提出了一种函数型Bootstrapping方法，其核心思想是将非线性函数（如ReLU）的计算融合进CKKS Bootstrapping的EvalMod阶段。通过将恢复线性值的 P_{arcsin} 多项式替换为直接计算目标函数的复合多项式，RBOOT实现了在Bootstrapping过程中"免费"获得ReLU的计算结果，避免了传统多项式近似[16]的高深度消耗。这一思想启发我们思考：能否将同样的融合策略应用于LayerNorm的倒数平方根计算？

与RBOOT中处理ReLU、Sign等分段函数不同，LayerNorm的倒数平方根函数并不能直接作为普通目标函数替换到EvalMod中。首先，$1/\sqrt{x}$ 在零点附近具有奇异性，且导数随 x 接近0而发散，若直接在包含零点或过近零点的区间上拟合，会导致多项式系数放大和噪声敏感性增强。其次，LayerNorm的输入是方差项，天然满足非负约束，而CKKS EvalMod的正弦域变量存在周期性和分支选择问题，若不约束输入范围，sin的周期别名会导致语义错误。最后，倒数平方根的输出尺度与预缩放后的方差尺度耦合，需要在复合函数中同时处理尺度恢复与稳定项。本文的贡献在于针对上述问题构造了带预缩放和统计边界的范围守卫，统一推导了正弦域复合目标函数，并分析了替换多项式对噪声传播的影响。

基于上述动机，本文提出一种将 $1/\sqrt{\sigma^2+\epsilon}$ 融合进CKKS Bootstrapping的新方法。本文的主要贡献包括：

1. 默认设置下零额外深度的融合式LayerNorm计算。我们在Bootstrapping的EvalMod阶段构造正弦域复合多项式 $P_{invsqrt}(t) \approx 1/\sqrt{\frac{q_0}{2\pi}\arcsin(t)+\epsilon'}$，将线性恢复与倒数平方根计算合并为一次多项式求值。在输入满足校准边界且不触发可选软裁剪的默认设置下，替换多项式与标准arcsin恢复多项式采用相同阶数和相同计算深度，因此不增加额外乘法深度。相比Goldschmidt迭代可节省6-8层深度。

2. 低深度范围守卫机制。本文采用预缩放与统计边界约束方差输入，使正弦域变量落在单调分支内，从而避免周期别名。对于分布外或统计保证不足的输入，本文进一步讨论可选软裁剪备用策略；该策略会额外消耗1-2层深度，不计入本文默认零额外深度方案。

3. 极低的实现代价。本方案仅需修改约50行代码（基于OpenFHEv1.1.2[17]），不改变Bootstrapping的整体结构。运行时计算量和内存占用与标准Bootstrapping完全一致。

1　背景知识

1.1　CKKS Bootstrapping原理

目前，CKKS方案是目前应用最广泛的支持近似算术的全同态加密方案。其核心特征是将实数（或复数）向量编码为多项式环 $R_Q = \mathbb{Z}_Q[X]/(X^N+1)$ 中的元素，并通过缩放因子 Δ 控制编码精度。CKKS方案支持同态加法、同态乘法以及旋转操作（实现槽位间的循环移位）。每次乘法运算消耗模数链中的一层（约 Δbits），当模数耗尽时需要通过Bootstrapping刷新密文以恢复计算能力。

CKKS Bootstrapping的目标是将密文的模数从接近耗尽的 q_{low} 提升回初始模数 Q，同时保持所加密消息的正确性。其核心挑战在于：模数提升会在明文中引入 q_{low} 的整数倍噪声，必须通过同态方式将其消除。标准CKKS Bootstrapping包含四个阶段：模数提升（ModRaise）、系数-槽域转换（CoeffsToSlots/SlotsToCoeffs）、模约减（EvalMod）以及输出阶段。其中模约减阶段是Bootstrapping的核心阶段。其目标是同态地计算模约减函数，消除模数提升阶段引入的 $q_{low} \cdot I$ 噪声项。由于模约减函数是不连续的，无法直接用多项式逼近，CKKS采用正弦逼近和反正弦恢复的两步策略来实现。表1给出了典型CKKS Bootstrapping的深度分解，其中 P_{arcsin} 为逼近arcsin的多项式（典型阶数31）。

1.2　函数型Bootstrapping（RBOOT）

RBOOT的核心思想是：既然Bootstrapping过程中必须计算一个多项式来恢复消息，为什么不将该多项式替换为直接输出目标函数值的多项式？具体而言，在EvalMod阶段的最后一步，标准Boot-

表1　典型CKKS Bootstrapping的深度分解

步骤	消耗深度	说明
CoeffsToSlots	4	层级预算配置
Sine逼近+倍角公式	7-8	63阶 + 2-3次倍角
P_{arcsin}计算	5	31阶，Estrin方案
SlotsToCoeffs	4	层级预算配置
总计	$L_{\text{boot}} \approx$ **19-21**	

strapping 计算：$P_{\text{arcsin}}(t) \approx \frac{q_0}{2\pi}\arcsin(t) \approx m$。RBOOT 将其替换为：$P_f(t) \approx f\left(\frac{q_0}{2\pi}\arcsin(t)\right) = f(m)$，即输出 $f(m)$ 而非 m，其中 f 为目标函数（如ReLU、Sign等）。

RBOOT的实现非常简洁，仅需将EvalMod最后一步的多项式系数从 P_{arcsin} 的系数替换为 P_f 的系数。Bootstrapping 的其他所有步骤（ModRaise、CoeffsToSlots、Sine 逼近、SlotsToCoeffs）均保持不变。这里的关键约束是替换多项式 P_f 的阶数不应超过原 P_{arcsin} 的阶数（典型值31），否则将增加额外的乘法深度。当目标函数 f 在拟合区间上足够平滑时，31阶多项式通常可以提供充足的逼近精度。

RBOOT对目标函数 f 有以下要求：1. 可多项式逼近。f 在拟合区间上可被有限阶多项式以足够精度逼近。2. 与正弦域兼容：复合函数 $g(t) = f(\frac{q_0}{2\pi}\arcsin(t))$ 在 $t \in [-1,1]$ 的子区间上可良好逼近。3. 阶数匹配：所需多项式阶数不超过深度预算（典型≤31阶）。

1.3　LayerNorm与RMSNorm

LayerNorm[9]是Transformer架构中的核心归一化组件，定义为：$\text{LayerNorm}(\mathbf{x}) = \boldsymbol{\gamma} \odot \frac{\mathbf{x} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \boldsymbol{\beta}$，其中 $\mathbf{x} = (x_1, x_2, \cdots, x_H) \in \mathbb{R}^H$ 为隐藏层向量，$\mu = \frac{1}{H}\sum_{i=1}^{H} x_i$ 为均值，$\sigma^2 = \frac{1}{H}\sum_{i=1}^{H}(x_i - \mu)^2$ 为方差，$\epsilon > 0$ 为数值稳定性常数（通常取 10^{-5} 或 10^{-6}）。$\boldsymbol{\gamma}, \boldsymbol{\beta} \in \mathbb{R}^H$ 为可学习的缩放和偏移参数。

RMSNorm[10] 是 LayerNorm 的简化变体，省去均值计算：$\text{RMSNorm}(\mathbf{x}) = \boldsymbol{\gamma} \odot \frac{\mathbf{x}}{\sqrt{\frac{1}{H}\sum_{i=1}^{H} x_i^2 + \epsilon}}$。两者的核心计算瓶颈均为倒数平方根 $1/\sqrt{\sigma^2 + \epsilon}$（或 $1/\sqrt{\text{RMS}^2 + \epsilon}$）。

在CKKS同态加密环境下，LayerNorm的各步骤有不同的计算难度，如表2所示。

目前在同态加密中计算 $1/\sqrt{x}$ 的主流方法是Goldschmidt迭代（又称倒数平方根迭代），其迭代公式为：

$$y_0 = x,\quad h_0 = x \cdot c_0, h_{i+1} = h_i \cdot \frac{3 - y_i \cdot h_i}{2}, y_{i+1} = y_i \cdot \frac{3 - y_i \cdot h_i}{2}$$

其中 c_0 为初始估计值（通常通过分段线性逼近获得）。迭代收敛到 $h_k \to 1/\sqrt{x}$。

在每次Goldschmidt迭代中，$y_i \cdot h_i$ 消耗1层深度，$h_{i+1} = h_i \cdot \frac{3 - y_i h_i}{2}$ 依赖步骤1的结果，再消耗1层深度。因此每次迭代消耗2层深度。达到足够精度通常需要 $k \geqslant 3$ 次迭代（对应double精度范围内的收敛），总计消耗 $2k \geqslant 6$ 层深度。

Goldschmidt方法有其局限性。其深度消耗高，6-8层深度在Bootstrapping后的剩余预算中占比显著。收敛依赖初值，初始估计 c_0 的质量影响收敛速度。此外每次迭代需保存 y_i、h_i 等中间结果，增加内存开销。本文的核心动机正是消除Goldschmidt迭代的这些开销，通过将 $1/\sqrt{x}$ 融合进Bootstrap-

表2　CKKS同态加密下LayerNorm各步骤的计算难度

步骤	操作	同态实现方式	深度消耗
均值 μ	$\sum x_i/H$	Rotate-and-Sum	$\log_2 H$ 次旋转，0层深度
残差 $x_i - \mu$	向量减法	同态减法	0层深度
方差 σ^2	$\sum(x_i - \mu)^2$	同态平方 +Rotate-and-Sum	1层深度
$1/\sqrt{(\sigma^2+\varepsilon)}$	**倒数平方根**	**迭代算法**	**≥6层深度**
归一化	$(x_i - \mu)\cdot(1/\sqrt{\cdot})$	同态乘法	1层深度
仿射变换	$\gamma \cdot \hat{x} + \beta$	同态乘法和加法	1层深度

ping实现"零额外深度"的倒数平方根计算。

1.4 相关工作

隐私保护神经网络推理是全同态加密的重要应用方向。CryptoNets[18]首次实现了基于FHE的神经网络推理，但仅支持浅层网络。Lee等人[16]通过多路复用并行卷积和低阶最小极大近似多项式，在CKKS上实现了深层CNN的高效推理，但ReLU的多项式近似仍需消耗大量乘法深度。Lee等人[19]进一步优化了近似精度，但非线性层的深度瓶颈依然存在。RBOOT提出将ReLU融合进Bootstrapping，实验表明可将每个ReLU的深度消耗从14-20层降至约8层，推理速度提升2.77倍，峰值内存降低81%。然而，RBOOT仅针对ReLU函数，尚未考虑Transformer中LayerNorm / RMSNorm的倒数平方根计算。本文面向Transformer LayerNorm/RMSNorm中的倒数平方根瓶颈，提出一种适配CKKS函数型Bootstrapping的融合式计算方法。与已有RBOOT主要面向ReLU等激活函数不同，本文处理的是具有零点奇异性、非负定义域和尺度耦合特征的归一化函数，并给出了范围守卫、复合目标函数构造及噪声传播分析。

在CKKS Bootstrapping方面，Cheon等人[12]首次提出了CKKS的Bootstrapping方法，Bossuat等人[13]进一步优化了非稀疏密钥场景下的效率。CKKS领域近期的重要改进之一，是Drucker等人[20]提出的一种能够处理离散计算的CKKS变体。Discrete CKKS的核心是CKKS风格的函数型/可编程Bootstrapping[21-23]，它能够在Bootstrapping过程中计算任意函数。基于这一框架，已有多项工作[24-25]构建了整数计算单元，从而支持包括逻辑运算在内的高效高精度整数计算。本文的方案是基于CKKS的Bootstrapping方法，可以直接集成到任何标准CKKS Bootstrapping实现中。。

Transformer推理中的LayerNorm/RMSNorm需要计算1/sqrt(x)，传统同态实现通常采用Goldschmidt或Newton-Raphson迭代，额外消耗多层乘法深度。与ReLU、Sign等函数不同，1/sqrt(x)具有零点奇异性、非负定义域和尺度耦合等特征，不能直接平凡替换到EvalMod中。本文的主要工作在于：通过稳定项、预缩放和统计边界构造范围守卫，统一推导正弦域目标函数 $h(t)$=1/sqrt(arcsin(t)/(αs))，并给出噪声传播和参数敏感性分析，从而在默认设置下实现零额外深度的融合式同态LayerNorm。

2 方案设计

本节详细阐述我们提出方的技术设计，包括核心思想、技术挑战与解决方案，以及多项式拟合策略。

2.1 核心思想

层归一化（LayerNorm）的核心计算为：$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$，其中 $\sigma^2 = \frac{1}{H}\sum j = 1^H (x_j - \mu)^2$ 为方差。在同态加密环境下，$1/\sqrt{\sigma^2 + \epsilon}$ 是计算瓶颈，传统Goldschmidt迭代法通常需要至少消耗6层乘法深度。

核心洞察：CKKS自举（Bootstrapping）在EvalMod阶段需要计算一个高阶多项式来逼近arcsin (sin (θ)) 以实现模约减。我们观察到，通过将该多项式替换为直接输出 $1/\sqrt{x}$ 的复合多项式，可以在自举过程中"免费"获得倒数平方根的计算结果。

关于复合函数的构造，本文首先统一变量定义。设LayerNorm的原始方差为v = σ^2，稳定项后输入为$u = v+\varepsilon$。范围守卫对u*进*行预缩放，得到$z=su=s(\sigma^2+\varepsilon)$，其中$s$为由方差上界$V_{max}$和安全边界$\delta$决定的缩放因子。在CKKS Bootstrapping的EvalMod阶段，正弦逼近和倍角步骤产生正弦域变量t=sin(αz)，其中α为由CKKS模约减尺度决定的常数。

标准Bootstrapping的恢复步骤可理解为利用$P_{arcsin}(t)$近似恢复z。本文则将该恢复多项式替换为复合多项式$P_{invsqrt}(t)$，直接输出LayerNorm所需的倒数平方根值。由于在范围守卫保证的单调区间内有arcsin(t)=αz=$\alpha s(\sigma^2+\varepsilon)$，因此目标函数可统一写为$h$($t$)=1/sqrt(arcsin($t$)/($\alpha$s))。

因此，本文实际拟合的是正弦域变量t上的复合函数$h(t)$，而不是直接在方差变量v上拟合1/sqrt($v+\varepsilon$)。该多项式将arcsin线性恢复与倒数平方根计算合并为一次多项式求值，无需自举后的额外迭代计算。

为更清晰地描述本文的融合式同态LayerNorm方案，算法1给出默认设置下的完整计算流程。默认方案仅包含预缩放、统计边界和Eval-

Mod 替换多项式，不启用可选软裁剪；软裁剪仅用于分布外输入或校准统计不足时的备用场景。

算法1 默认设置下的融合式同态 LayerNorm 方案

输入：密文向量 ct_x=Enc(x)，隐藏维度 H，稳定项 ε，校准方差上界 V_{max}，安全边界 η，缩放因子 s=(q_0/4-δ)/(V_{max}+ε)，替换多项式 $P_{invsqrt}(t)$。

输出：密文归一化结果 ct_y≈Enc((x-μ)/sqrt(σ^2+ε))。

1. 通过 Rotate-and-Sum 计算均值 ct_μ=Enc(1/H$\sum x_i$)。

2. 计算残差 ct_r=ct_x-ct_μ。

3. 通过同态平方和旋转求和计算方差 ct_v=Enc(σ^2)=Enc(1/H$\sum(x_i$-$\mu)^2$)。

4. 加入稳定项并预缩放：ct_z=s·(ct_v+ε)。

5. 对 ct_z 执行 CKKS Bootstrapping。在 EvalMod 阶段，将标准恢复多项式 $P_{arcsin}(t)$ 替换为 $P_{invsqrt}(t)$，其中目标函数为 $h(t)$=1/sqrt(arcsin(t)/(αs))。

6. Bootstrapping 输出 ct_q≈Enc(1/sqrt(σ^2+ε))。

7. 计算归一化结果 ct_y=ct_r·ct_q；若模型包含可学习参数 γ、β，则进一步计算 ct_y=γ·ct_y+β。

在输入满足 σ^2≤V_{max} 且不触发可选软裁剪的默认设置下，步骤 5 中的 $P_{invsqrt}$ 与标准 P_{arcsin} 采用相同阶数和相同多项式求值深度，因此倒数平方根计算不引入额外乘法深度。

2.2 技术挑战与解决方案

（1） 挑战一：零点奇异性

问题描述：函数 $f(x)=1/\sqrt{x}$ 在 $x=0$ 处无定义，且当 $x\to 0$ 时趋向无穷大。多项式无法逼近此行为。

解决方案：在进入自举前，对方差施加偏移 $\sigma'^2=\sigma^2+\epsilon$，$\epsilon>0$，其中 ϵ 为 LayerNorm 的标准稳定性常数（通常为 10^{-5} 或 10^{-6}）。这确保逼近定义域为 $[\epsilon, V_{max}+\epsilon]$，严格避开零点。

（2） 挑战二：周期别名风险

问题描述：EvalMod 阶段使用 $\sin(\theta)$ 进行模约减。由于正弦函数是周期函数，如果输入 x 超出单调区间 $[-\frac{q_0}{4},\frac{q_0}{4}]$，正弦函数将发生"折返"（即两个不同的 x 值映射到相同的 $\sin(x)$ 值），导致语义错误而非仅仅是精度损失。该问题很严重，因为这是正确性问题。任何越界都将导致完全错误的输出。

解决方案：本文采用默认范围守卫机制限制输入落入正弦函数的单调区间。默认方案由预缩放和统计边界两部分构成；此外，针对分布外输入或统计边界不足的场景，本文讨论可选软裁剪备用策略。

□ 预缩放：设原始方差为 v=σ^2，稳定项后输入为 u=v+ε。本文对 u 而非单独的 v *进*行预缩放，即 z=$s(\sigma^2+\varepsilon)$，其中缩放因子取 s=(q_0/4−δ)/($V_{\max}$+ε)，$V_{\max}$为统计确定的方差上界，δ 为安全边界。这样，当 0≤ σ^2≤ $V_{\max}$ 时，有 0<$s(\sigma^2+\varepsilon)\le q_0/4-\delta$，从而保证进入 EvalMod 的正弦域变量对应的角度位于单调区间内。

□ 统计边界：在模型校准阶段，收集每层的方差统计信息，计算 99.9% 分位数作为 V_{max}。

□ 可选软裁剪备用策略：若部署场景中输入分布可能明显偏离校准集，导致统计边界 $V_{\max 不}$ 再可靠，可在自举前应用低阶多项式逼近的软裁剪函数，将异常方差压回安全区间。需要强调的是，软裁剪不是本文默认实验和定理分析中的必选步骤；一旦启用软裁剪，将额外消耗 1-2 层乘法深度，因此该备用策略不再满足严格的零额外深度声明。

默认范围守卫的核心目标可归结为以下形式化约束，以确保缩放后的输入角度始终落在正弦函数的单调区间内：$\forall\sigma^2\in[0,V_{max}]$: ... $\in\left(0,\frac{\pi}{2}\right)$。

定理1（范围守卫正确性）：设输入方差 $\sigma^2\in[0,V_{max}]$，缩放因子定义为 $s=\frac{q_0/4-\delta}{V_{max}+\epsilon}$，其中 $\delta>0$ 为安全边界，$\epsilon>0$ 为 LayerNorm 稳定性常数。则对于任意 $\sigma^2\in[0,V_{max}]$，有 $\theta=\frac{2\pi}{q_0}\cdot s\cdot(\sigma^2+\epsilon)\in\left[\frac{2\pi s\epsilon}{q_0},\frac{\pi}{2}-\frac{2\pi\delta}{q_0}\right]\subset\left(0,\frac{\pi}{2}\right)$。因此 $\sin(\theta)$ 在该区间内严格单调递增，不存在周期别名。

证明：首先看下界：当 $\sigma^2=0$ 时，$\theta_{min}=\frac{2\pi s\epsilon}{q_0}>0$（因 $s,\epsilon>0$）。上界是：当 $\sigma^2=V_{max}$ 时，$\theta_{max}=\frac{2\pi}{q_0}\cdot s\cdot(V_{max}+\epsilon)=\frac{2\pi}{q_0}\cdot\frac{q_0/4-\delta}{V_{max}+\epsilon}\cdot(V_{max}+\epsilon)=\frac{\pi}{2}-\frac{2\pi\delta}{q_0}$。由于 $\delta>0$，有 $\theta_{max}<\pi/2$。结合 $\theta_{min}>0$，整个区间 $(\theta_{min},\theta_{max})\subset(0,\pi/2)$，在此区间内 $\sin(\theta)$ 严格单调递增。

3） 挑战三:定义域不匹配

问题描述：首先是方差 σ^2 恒为非负：$\sigma^2 \in [0,+\infty)$。其次是 EvalMod 输出 $\sin(\theta) \in [-1,1]$，理论上包含负值。在计算流程中，数据经历三个阶段的变换：输入方差 σ^2 经过 Bootstrapping 变为正弦域中间值 $\sin(\theta)$，再经复合多项式 $P_{invsqrt}(\cdot)$ 映射为最终输出 $1/\sqrt{\sigma^2+\epsilon}$。该流程可表示为：$\sigma^2 \xrightarrow{\text{范围守卫}+\text{Bootstrapping}} \sin(\theta) \xrightarrow{P_{inv_s qrt}(\cdot)} \frac{1}{\sqrt{\sigma^2+\epsilon}}$。

解决方案：利用范围守卫保证正区间。由定理 1 可知，范围守卫机制确保 $\theta \in (0,\pi/2)$。在此区间内，当 $\sin(\theta) \in (0,1)$ 时，始终为正，与方差的非负性匹配；当 $\sin(\theta)$ 严格单调递增时，满足 arcsin 反函数的存在性。

因此，虽然 CKKS 自举天然同时产生 $\sin(\theta)$ 和 $\cos(\theta)$（源于复数槽结构），在我们的设计中使用 sin 通道即可，因为范围守卫已将输入约束到 sin 的"安全区域"。那么为何不选择 cos 通道？如下表 3 所示

选择 sin 通道可直接复用现有 CKKS 库的 arcsin 多项式，降低实现复杂度。

2.3　多项式拟合

Remez 算法的目标函数定义为正弦域复合函数 $h(t)$，而不是方差域函数。根据 2.1 节的定义，$t=\sin(\alpha s(\sigma^2+\varepsilon))$。在范围守卫保证 $\alpha s(\sigma^2+\varepsilon)$ 位于正弦单调区间时，有 $\arcsin(t) = \alpha s(\sigma^2+\varepsilon)$，因此 $h(t)=1/\mathrm{sqrt}(\arcsin(t)/(\alpha s))$。拟合区间由方差范围 $0 \leqslant \sigma^2 \leqslant V_{\max}$ 映射得到：$t \in [t_{\min}, t_{\max}]=[\sin(\alpha s\varepsilon), \sin(\alpha s(V_{\max}+\varepsilon))]$。

本文采用切比雪夫多项式基表示 $P_{\mathrm{invsqrt}}(t)$，并在上述区间上使用 Remez 交换算法求解 $h(t)$ 的 31 阶极小极大逼近。

使用 Remez 交换算法求解极小极大逼近：$P^* = \arg\min_{P \in \mathrm{P}d} \max t \in [a,b] |P(t) - f_{target}(t)|$。算法的流程是输入目标函数 $f(t)$，区间 $[a,b]$，阶数 d，容差 ε，输出是最优系数 $\{c_0, ..., c_d\}$。

引理 1（Remez 最优性）：设 $P^{(t)}$ 为 Remez 算法在区间 [a,b] 上对目标函数 $f_{target}(t)$ 的 d 阶输出，则 $\| P^* - f_{target} \|_\infty = E_d(f_{target}) := \inf_{p \in \mathrm{P}d} \| ftarget - p \|_\infty$，即 Remez 算法输出的多项式达到该阶数下的最佳逼近误差。

证明： Remez 算法的输出满足 Chebyshev 等振性质。误差函数 $e(t) = P^*(t) - f(t)$ 在 [a,b] 上至少有 $d+2$ 个交替符号的极值点，且极值绝对值相等。由 Chebyshev 定理，这是最佳逼近的充要条件。

引理 2（解析函数的指数衰减）：若 f_{target} 在包含 [a, b] 的 Bernstein 椭圆 $\mathcal{E}\rho$（$\rho > 1$）内解析，且 $|f(z)| \leqslant M$ 对所有 $z \in \mathcal{E}\rho$ 成立，则：$E_d(f_{target}) \leqslant \frac{2M\rho^{-d}}{\rho - 1}$。

证明： 这是 Bernstein 逼近定理的直接结论。

对于本文统一后的目标函数 h(t)=1/sqrt(arcsin(t)/(αs))，潜在奇点主要来自两类：第一，arcsin(t) 在 t = ±1 附近存在分支点；第二，当 arcsin(t)/(αs)=0 时，倒数平方根项存在零点奇异性。本文通过范围守卫将拟合区间限制为 [sin($\alpha s\varepsilon$), sin($\alpha s(V_{\max}+\varepsilon)$)]，其下界对应 $\sigma^2=0$ 时的稳定项 ε，因此 arcsin(t)/(αs) $\geqslant \varepsilon > 0$；同时上界与正弦折返点保持安全边界 δ。因此，$h(t)$ 在所选闭区间及其邻域内解析，Remez 多项式可稳定逼近该目标函数。

在多项式计算策略方面，为减少同态计算的深度消耗，采用 Estrin 方法进行并行计算：$P(t) = (c_0 + c_1 t) + t^2(c_2 + c_3 t) + t^4(\cdots)$。与 Horner 方法的对比，如下表 4 所示：

对于 63 阶多项式，Estrin 方法将深度从 63 降至 6。

2.4　噪声传播分析

本文将标准 CKKS Bootstrapping 中的 $P_{\arcsin 替}$ 换为 P_{invsqrt}。虽然替换前后多项式阶数相同，但倒数平方根目标函数在小值附近曲率较大，可能影响

表 3　　**sin 通道与 cos 通道的对比**

特性	sin(θ)（本文采用）	cos(θ)
$\theta \in (0, \pi/2)$ 的值域	(0, 1)，单调递增	(0, 1)，单调递减
反函数	arcsin（与标准 CKKS 实现一致）	arccos（需修改 EvalMod）
实现复杂度	低（复用现有代码）	高（需定制）

表 4　　多项式计算策略对比

方法	乘法深度	并行性
Horner 方法	d（必须串行）	无
Estrin 方法	□ log(d+1)□	高

EvalMod 阶段的噪声传播。因此，本节从误差分解角度分析替换多项式对输出误差的影响，并结合实验数据说明本文方案的误差主要由 CKKS Bootstrapping 的固有噪声主导，而非由 Remez 逼近误差主导。

设原始方差为 $v = \sigma^2$，稳定项后输入为 $u=v+\varepsilon$，预缩放后输入为 $z = su = s(\sigma^2+\varepsilon)$。在 EvalMod 阶段，正弦域变量记为 t=sin(αz)，其中 α 为由 CKKS Bootstrapping 模约减尺度决定的常数。在范围守卫保证 αz 位于正弦单调区间时，有 arcsin(t)=αz，因此本文替换多项式的理想目标函数为 $h(t)$=1/sqrt(arcsin(t)/(αs))。

实际同态计算中，由正弦逼近、倍角公式、重缩放和密文噪声共同引入的误差会使正弦域输入变为 $\tilde{t} = t + e_t$，其中 $e_{t\text{表}}$示进入最后一步多项式求值前的正弦域误差。设 $P_{d(t)}$为 d 阶 Remez 多项式，本文中 d=31。最终输出误差可分解为

$|\tilde{y} - y| = |P_d(\tilde{t}) - h(t)| \leqslant |P_d(\tilde{t}) - P_d(t)| + |P_d(t) - h(t)| + e_{eval}$，其中 e_{eval} 表示多项式同态求值过程中的系数舍入、重缩放和 CKKS 近似噪声。由均值定理可得

$|P_d(\tilde{t}) - P_d(t)| \leqslant L_P|e_t|$，其中 L_P=max_{$\xi \in$I} |P'_d(ξ)|，I=[t_min,t_max]=[sin($\alpha s\varepsilon$), sin(αs(V_{max}+ε))]为 Remez 拟合区间。因此，总误差满足如下上界 $|\tilde{y} - y| \leqslant L_P|e_t| + e_{poly} + e_{eval}$，其中 $e_{poly} = max_{t \in I}|P_d(t) - h(t)|$ 为 Remez 逼近误差。该式表明，替换多项式是否导致额外噪声放大，主要由两个量决定：一是 Remez 逼近误差 e_{poly}，二是多项式导数上界 L_P。

进一步地，若直接考察理想目标函数 h(t) 的导数，可得 $h'(t) = -\frac{1}{2\alpha s\sqrt{1 - t^2}}\left(\frac{\arcsin(t)}{\alpha s}\right)^{-\frac{3}{2}}$。该式说明，1/sqrt($x$)在 x=0 附近的导数发散会反映为 h(t)在区间下界附近的较大斜率。本文通过两点控制该风险：第一，LayerNorm 的稳定项 ε 使 arcsin(t)/(αs) ⩾ε，避免目标函数在使用区间内触及零点；第二，范围守卫将 t *限*制在正弦函数单调区间内部，并保留安全边界 δ，使 sqrt(1-t^2) 不接近 0。因此，在本文拟合区间 I *内*，$h(t)$ 和 $P_{d(t)}$ 的导数均有界，噪声放大因子可由 $L_{P\,显}$式度量。

在实验层面，本文进一步汇总多项式逼近误差、倒数平方根密文计算误差以及最终 LayerNorm 输出误差，如表 5 所示。该表中的 Remez 误差来自表 13，倒数平方根误差来自表 14，LayerNorm 输出误差来自表 15。

表 5 表明，d=31 时 P_{invsqrt}的离线 Remez 逼近误差为 8.6×10^{-11}，而密文端 1/sqrt(σ^2+ε) 的平均绝对误差为 4.5×10^{-7}，二者相差约 4 个数量级。这说明在当前参数下，系统误差主要来自 CKKS Bootstrapping 和后续同态运算中的近似噪声，而不是来自 P_{invsqrt}的多项式逼近误差。进一步地，最终 LayerNorm 输出的平均绝对误差为 6.8×10^{-6}，最大绝对误差为 4.2×10^{-5}，RMSE 为 8.3×10^{-6}，仍处于深度学习推理可接受范围内。因此，虽然 P_{invsqrt}的目标函数曲率高于标准 arcsin 恢复函数，但在稳定项 ε 和范围守卫约束下，该替换并未造成不可控的噪声放大。

综上，本文替换多项式的误差可分解为正弦域输入误差放大项、Remez 逼近误差和同态求值误差三部分。稳定项 ε 和范围守卫使目标函数在拟合区

表 5　　替换多项式的误差传播与密文执行误差

指标	数值	说明
Remez 逼近误差 e_{poly}	8.6×10^{-11}	d=31 时 P_{invsqrt}的离线最大逼近误差
1/sqrt(σ^2+ε) 平均相对误差	3.2×10^{-7}	密文端倒数平方根平均相对误差
1/sqrt(σ^2+ε) 最大相对误差	1.8×10^{-6}	密文端倒数平方根最大相对误差
1/sqrt(σ^2+ε) 平均绝对误差	4.5×10^{-7}	密文端倒数平方根平均绝对误差
LayerNorm 输出平均绝对误差	6.8×10^{-6}	逐元素统计
LayerNorm 输出最大绝对误差	4.2×10^{-5}	逐元素统计
LayerNorm 输出 RMSE	8.3×10^{-6}	逐元素统计

间内远离零点奇异性和正弦折返点，从而保证误差放大因子有限。实验结果进一步表明，Remez逼近误差远低于密文端实际误差，最终LayerNorm输出误差也保持在可接受范围内。因此，P_{invsqrt}替换不会成为本文方案的主要精度瓶颈。

2.5　复杂度分析

复杂度如下表6所示。

表6　　复杂度分析对比

方法	乘法深度	内存占用	正确性保证
Goldschmidt迭代	$L_{boot}+2k$（$k\geqslant 3$）	高（需更大Q）	依赖收敛性
本方案	$L_{boot}+\Delta_{本方案}$	低	依赖范围守卫

定理2（默认方案下的深度节省）：设Goldschmidt迭代需要k次迭代收敛（典型$k\geqslant 3$），设自举深度为L_{boot}，替换后的P_{invsqrt}t与原P_{arcsin}采用相同阶数和相同多项式求值策略，则：$D_{\text{Goldschmidt}}=L_{boot}+2k$，$D_{本方案}=L_{boot}+\Delta_{本方案}$，本文默认方案相对于标准Bootstrapping的额外深度$\Delta_{本方案}=0$。相比Goldschmidt迭代，我们的方法节省$2k$层乘法深度，当k=3时节省6层。

证明： 本文默认方案仅采用预缩放与统计边界，不启用软裁剪。因此范围守卫不引入额外乘法深度，其作用仅是常数缩放与明文参数配置。首先计算Goldschmidt迭代深度。Goldschmidt算法计算$1/\sqrt{x}$的每次迭代包含

$h_i=\dfrac{3-y_i\cdot x}{2}$，需要先计算$y_i\cdot x$，因此消耗1层深度。然后包含$y_{i+1}=y_i\cdot h_i$，其依赖$h_i$，需要串行执行，因此再消耗1层深度。所以每次迭代消耗2层深度。另外达到所需精度需$k\geqslant 3$次迭代，总计$2k\geqslant 6$层。

而我们提出的方案是将$1/\sqrt{x}$融合进Bootstrapping的EvalMod阶段。当替换后的多项式$P_{inv_s qrt}$的计算深度不超过原EvalMod恢复步骤的深度预算时，$\Delta_{本方案}=0$。因此，$D_{\text{Goldschmidt}}-D_{本方案}=2k-\Delta_{本方案}\geqslant 6$。

推论： 设每层深度对应的模数位数为Δqbits。我们提出的方案允许减少总模数链长度，从而可能降低所需Q，并在满足同等安全级别的条件下潜在允许选择更小的N或更快的bootstrapping参数配置，从而实现系统级加速。

3　方案实现

表7列出了CKKS全同态加密的核心参数配置。

表7　　CKKS基础参数配置

参数	符号	推荐值	说明
多项式度数	N	2^{16}	支持足够的深度和槽位数量
槽位数量	$N/2$	2^{15}	全打包模式
基础模数	q_0	$\approx 2^{51}$	Bootstrapping后的精度保证
模数链总长	$\log_2 Q_{boot}$	$\approx$ 1500bits	由Bootstrapping消耗深度决定
缩放因子	Δ	2^{45}	平衡精度与溢出风险

注意，本章中q_0均指模数的数值（约2^{51}），而非位长。参数选择依据如下，根据文献[26]，在N=2^{16}的设置下，128位安全级别允许的最大模数约为1747位。本文采用的主模数链总长约为1500位，低于该上限，因此在上述安全估计模型和所计入总模数范围下满足128位安全级别。若实现中bootstrapping或key switching使用辅助模数，则安全估计应以实际计入的总模数为准。$N=2^{16}$满足128位安全级别下支持Bootstrapping的最小需求；$q_0\approx 2^{51}$确保Bootstrapping后有足够精度进行后续LayerNorm计算；$\Delta=2^{45}$在典型LLM激活值范围内避免溢出。

对于Transformer模型，我们采用行优先展平的槽位布局策略：将二维张量[Batch,Hidden$_{\text{D}}$im]按行优先顺序展平为一维向量，依次填入密文的各个槽位。即先填入第0个样本的全部H个隐藏维度分量，再紧接着填入第1个样本，以此类推。

范围守卫参数如表8所示。

表8　　范围守卫参数配置

参数	符号	典型值	说明
稳定性常数	ϵ	10^{-5}	LayerNorm标准值
安全边界比例	η	5%	预留安全余量
安全边界	δ	$\eta\cdot(q_o/4)$	安全缓冲距离
方差上界	V_{max}	模型校准获得	99.9% 分位数
缩放因子	s	$(q_o/4-\delta)/(V_{max}+\varepsilon)$	由定理1确定

校准流程：在明文数据集上运行模型前向传播；收集每层LayerNorm的方差统计；计算99.9%分位数作为V_{max}；计算对应的缩放因子s。

标准 CKKS Bootstrapping 的 EvalMod 阶段执行以下步骤：

（1）正弦逼近：计算 $t = \sin(\frac{2\pi}{q_0} \cdot x')$。

（2）倍角迭代：通过倍角公式扩展频率。

（3）反正弦恢复：计算 $P_{\arcsin}(t)$ 恢复线性值。

注意，不同 CKKS 实现可能采用不同的恢复策略（如 EvalExp 变种）。

我们对 EvalMod 的修改仅涉及最后一步。标准的 EvalMod 是：$\sin(\theta) \to P_{\arcsin}(t) \to$ x'（恢复线性）。而我们修改的 EvalMod 是：$\sin(\theta) \to P_{invsqrt}(t) \to 1/\sqrt{(\sigma^2 + \varepsilon)}$（直接输出结果）。其中 $P_{\text{invsqrt的}}$ 输入不是方差 σ^2，而是 EvalMod 中的正弦域变量 $t=\sin(\alpha s(\sigma^2+\varepsilon))$；其拟合目标为 $h(t)=1/\text{sqrt}(\arcsin(t)/(\alpha s))$。

使用 Remez 算法离线生成 P_{invsqrt} 的系数。输入-输出关系推导如下。Bootstrapping 前的方差输入记为 $v = \sigma^2$，加入稳定项后得到 $u = v+\varepsilon$，并经范围守卫预缩放为 $z=su=s(\sigma^2+\varepsilon)$。EvalMod 的正弦域变量记为 $t=\sin(\alpha z)$。由于范围守卫保证 αz 位于正弦函数的单调区间内，因此有 $\arcsin(t)=\alpha z=\alpha s(\sigma^2+\varepsilon)$。于是 LayerNorm 所需的倒数平方根可写为 $1/\text{sqrt}(\sigma^2+\varepsilon)=1/\text{sqrt}(\arcsin(t)/(\alpha s))$。

因此，Remez 算法的目标函数统一为 $h(t)=1/\text{sqrt}(\arcsin(t)/(\alpha s))$，并在区间 $[t_{\min}, t_{\max}]=[\sin(\alpha s\varepsilon), \sin(\alpha s(V_{\max}+\varepsilon))]$ 上求解 31 阶极小极大逼近 $P_{\text{invsqrt}}(t)$。由于尺度因子 αs 已包含在目标函数定义中，$P_{\text{invsqrt}}(t)$ 的输出即为 $1/\text{sqrt}(\sigma^2+\varepsilon)$，无需在自举后再进行额外迭代或额外尺度校正。拟合区间由定理 1 确定：$t \in [\sin(\theta_{min}) + \tau, \sin(\theta_{max}) - \tau]$，其中 $\theta_{min} = \frac{2\pi s\epsilon}{q_0}$，$\theta_{max} = \frac{\pi}{2} - \frac{2\pi\delta}{q_0}$，$\tau$ 是数值稳定冗余量（典型值 10^{-6}），避免边界处的数值不稳定。

表 8 中的 Remez 配置均针对正弦域目标函数 $h(t)=1/\text{sqrt}(\arcsin(t)/(\alpha s))$，而非直接针对方差变量 v 上的 $1/\text{sqrt}(v+\varepsilon)$。Remez 算法配置如下表 9 所示。

关于多项式度数，选择 31 阶 Chebyshev 多项式的量化依据如下表 10。

误差来源分析。方案的总误差由三部分组成：$\epsilon_{total} = \epsilon_{approx} + \epsilon_{coeff} + \epsilon_{CKKS}$，其中 ϵ_{approx} 是多项式逼近误差（Remez 误差），ϵ_{coeff} 是系数舍入误差（long double 存储），ϵ_{CKKS} 是 CKKS 噪声（rescale、bootstrapping 固有噪声）。

表 9　Remez 算法配置

配置项	值
多项式度数	31（见下表量化分析）
基函数	Chebyshev 多项式
收敛精度	10^{-10}
系数精度	long double（避免精度损失）
输出格式	Chebyshev 系数（直接用于 EvalChebyshevSeries）

表 10　多项式度数选择的量化依据

阶数 d	Remez L∞ 误差
15	~ 10^{-4}
23	~ 10^{-7}
31	**~ 10^{-10}**
47	~ 10^{-14}

关于 Lipschitz 误差传播，对于 $f(x) = x^{-1/2}$，在区间 $x \in [\epsilon, V_{max} + \epsilon]$ 上的 Lipschitz 常数为 $L = \max_x |f'(x)| = \frac{1}{2}\epsilon^{-3/2}$。因此，多项式逼近误差 ϵ_{approx} 对最终输出的影响满足 $|P(t) - f(x)| \leqslant \epsilon_{approx}$，这是绝对误差而非相对误差。对于典型值 $\epsilon = 10^{-5}$，$f(x) \in [1,316]$（对应 $x \in [10^{-5},10]$），相对误差范围为 $\epsilon_{approx}/f(x) \in [3 \times 10^{-13}, 10^{-10}]$。

量化结论：31 阶多项式的逼近误差 $\epsilon_{approx} \approx 10^{-10}$ 远小于 CKKS bootstrapping 的典型精度损失（$\sim 10^{-6}$），因此多项式逼近不是精度瓶颈。

关于常数因子，虽然 $P_{invsqrt}$ 与 $P_{\arcsin}$ 度数相同，但由于目标函数在 $t \to t_{min}$ 处曲率较大（$1/\sqrt{\arcsin(t)}$ 的导数发散），系数幅度略大。替换多项式对噪声传播的影响已在第 2.4 节进行了详细分析。

4　实验评估

本节对我们方案进行全面的实验评估。我们从多项式逼近精度、深度消耗、运行时间和端到端推理性能四个维度进行验证，并与基于 Goldschmidt 迭代的传统方案进行对比。

4.1　实验分析

表 11 对比了我们方案与标准 Bootstrapping 的实现开销。数据基于 Intel Xeon Gold6248R@3.0GHz、256GB 内存的服务器，运行 Ubuntu22.04 LTS，使用 GCC11.4.0 编译 OpenFHE v1.1.2，单线

程模式执行。

内存由RNS分解的临时缓冲区、NTT变换、密钥等共同决定。本方案不改变这些结构，因此内存占用与标准Bootstrapping一致。时间方面，多项式度数相同（31阶），Chebyshev计算的乘法次数一致，因此计算量无变化。结论：我们方案的实现代价极低，主要工作是离线生成多项式系数，运行时与标准Bootstrapping完全等价。

为覆盖不同规模的推理场景，我们选取以下模型与数据集，如表12所示。

在明文环境下，对SST-2验证集的1000条样本运行前向传播，收集每层LayerNorm的方差统计。实验观察到，BERT-Tiny方差分布集中在[0.01, 3.5]，99.9%分位数 $V_{max} \approx 5.2$；BERT-Base方差分布集中在[0.02, 8.1]，99.9%分位数 $V_{max} \approx 12.7$；GPT-2方差分布集中在[0.05, 6.3]，99.9%分位数 $V_{max} \approx 9.8$。

我们将本方案与以下基线方案进行对比：

（1）Goldschmidt迭代：标准CKKS Bootstrapping与 k 次Goldschmidt迭代计算 $1/\sqrt{x}$。取 $k = 3$（典型配置）和 $k = 4$（高精度配置）。

（2）Newton-Raphson迭代：标准CKKS Bootstrapping与Newton法迭代，$k = 4$ 次。

（3）明文LayerNorm（Baseline）：未加密的PyTorch实现，作为精度参考。

首先验证Remez算法生成的 $P_{invsqrt}(t)$ 的逼近质量。表13给出了不同阶数下的最大绝对误差（L_∞ 误差）。

31阶多项式的逼近误差（~10^{-11}）远小于CKKS Bootstrapping的固有精度损失（~10^{-6}），因此多项式逼近不构成系统精度瓶颈。

表13给出了不同多项式阶数下的实测Remez逼近误差。可以看到，d=23时最大逼近误差为 4.1×10^{-7}，已达到可用精度。当阶数提高到 d=31时，最大逼近误差进一步降至 8.6×10^{-11}，显著低于CKKS Bootstrapping的固有近似噪声。因此，本文采用 d=31作为默认多项式阶数。继续提高至d=47或d=63虽可将离线逼近误差分别降低至 3.2×10^{-15} 和 1.8×10^{-19}，但会增加多项式求值深度，综合收益有限。因此，10^{-7} 量级主要对应 d=23的情形，而 d=31及以上阶数的实测逼近误差已明显低于该量级。

我们将本方案输出的 $1/\sqrt{\sigma^2 + \epsilon}$ 与明文计算的真实值进行逐样本对比。对BERT-Base模型的SST-2验证集执行完整推理，记录每层LayerNorm的输出误差。如表14所示。

本方案的精度与Goldschmidt处于同一量级，均满足深度学习推理的精度需求（相对误差 < 10^{-5}）。Goldschmidt和Newton可达到更高精度，但需额外消耗2层乘法深度。本方案的误差主要来源于CKKS Bootstrapping的固有噪声（ϵ_{CKKS}），而非多项式逼近误差（ϵ_{approx}）。更重要的是LayerNorm最终输出 $\hat{x}_i = (x_i - \mu)/\sqrt{\sigma^2 + \epsilon}$ 的精度。表15报告了归一化后每个元素的误差统计。

本方案的LayerNorm输出精度略优于Goldschmidt，且完全满足下游分类/生成任务的需求。

需要说明的是，表14和表15反映的是单次倒数平方根计算及单次LayerNorm输出误差，而端到端分类准确率还受到模型分类边界、样本分布和后续非线性传播的影响，不应被理解为与单步数值误差严格单调对应。为观察多层LayerNorm调用下的累积效应，本文进一步在BERT-Base的12层Transformer结构上进行端到端同态推理实验。该设置每个样本包含24次LayerNorm调用，100条样本共包含2400次LayerNorm调用；实验结果如表19所示，本方案在多层累积后仍保持92.1%的分类准确率，与明文推理相差0.4个百分点，说明当前数值误差没有在多层调用中造成明显的任务级退化。

乘法深度是CKKS方案中最关键的资源。表16详细对比了各方案的深度预算分解。

表11　　本方案与标准Bootstrapping的实现开销对比

指标	标准Bootstrapping	本方案	增量
代码修改量	—	~50行	极低
预计算（Remez）	无	一次性，< 1秒	可忽略
运行时内存（实测峰值）	~6.5GB	~6.5GB	无变化
单次Bootstrapping时间	~12秒	~12秒	无变化

表 12　　模型与数据集

模型	隐藏维度 H	层数	数据集	用途
BERT-Tiny	128	2	SST-2	轻量场景验证
BERT-Base	768	12	SST-2	标准对比基准
GPT-2 单层 Block	768	1	WikiText-2	生成任务验证

表 13　Remez 逼近误差 vs 多项式阶数（BERT-Base 参数）

阶数 d	L_∞ 误差	计算深度（Estrin）	备注
15	2.7×10^{-4}	4	精度不足
23	4.1×10^{-7}	5	可用
31	$\mathbf{8.6 \times 10^{-11}}$	**5**	**本文采用**
47	3.2×10^{-15}	6	过高，增加深度
63	1.8×10^{-19}	6	远超需求

表 14　　$1/\sqrt{(\sigma^2+\varepsilon)}$ 计算误差对比

方法	平均相对误差	最大相对误差	平均绝对误差
本方案（d= 31）	3.2×10^{-7}	1.8×10^{-6}	4.5×10^{-7}
Goldschmidt（k= 3）	5.7×10^{-7}	3.1×10^{-6}	7.8×10^{-7}
Goldschmidt（k= 4）	2.1×10^{-8}	9.4×10^{-8}	3.6×10^{-8}
Newton-Raphson（k= 4）	4.3×10^{-8}	2.5×10^{-7}	5.9×10^{-8}

深度节省的量化意义：节省 6-8 层乘法深度意味着模数链总长缩短约 $6 \times 50 = 300$ 位（每层 ≈ 50 位 bits）。在相同安全级别下，可选择更小的 N，或在相同 N 下支持更多后续计算。对于多层 Transformer 模型，深度节省具有累积效应

表 17 报告了执行单次 LayerNorm 计算的运行时间分解。所有实验在单线程模式下重复 10 次，取平均值。

Bootstrapping 操作在各方案中均为主要耗时部分（> 60%），与方案选择无关。本方案省去了 Goldschmidt 迭代的 4.2-5.6 秒开销，但引入了范围守卫预处理（0.3 秒），净节省 3.9-5.3 秒。相对于 Goldschmidt，本方案实现 1.26× 加速；相对于 $k = 4$，实现 1.36× 加速。

表 18 记录了各方案在执行 LayerNorm 时的峰值内存占用。

本方案的内存占用与标准 Bootstrapping 一致（6.5GB），因为不需要存储 Goldschmidt 迭代过程中的中间密文。Goldschmidt 方法在迭代过程中需要额外保存 y_i、h_i 等中间结果密文，导致内存开销增大。

我们将本方案集成到 BERT-Base 的完整同态推理流程中，在 SST-2 情感分析数据集上评估。本文评估样本数为 1000 条，并报告准确率的 95% 置信区间，如表 19 所示。

所有同态方案的准确率与明文推理差距均 < 1%，表明加密域 LayerNorm 的精度足够。本方案准确率（92.1%）介于 Goldschmidt（91.8%）和 $k = 4$（92.3%）之间，接近 $k = 4$ 的精度但时间开销远小于后者。在推理时间上，本方案比 Goldschmidt 快 21%，比 $k = 4$ 快 26%。

需要说明的是，表 14 中的倒数平方根误差、表 15 中的 LayerNorm 输出误差与表 19 中的端到端分类准确率并不构成严格单调关系。其根本原因在于，前两者是连续数值误差指标，衡量的是中间计算结果与明文结果之间的平均或最大偏差；而分类准确率是离散任务指标，只有当数值扰动最终改变预测类别时才会发生变化。因此，较小的连续误差通常有助于提高稳定性，但并不必然对应严格更高的分类准确率。

从误差传播机理看，LayerNorm 误差对最终分类结果的影响还取决于误差的分布位置、符号方向以及样本距离分类边界的远近。对于分类置信度较高、logit margin 较大的样本，即使 LayerNorm 输出存在小幅扰动，后续注意力层、前馈层和分类头通常仍会保持相同预测；而对于接近决策边界的样本，较小但方向不利的扰动也可能改变最终类别。换言之，端到端准确率不仅与误差幅值有关，还与

表 15　　LayerNorm 输出误差（BERT-Base，逐元素统计）

方法	平均绝对误差	最大绝对误差	均方根误差（RMSE）
本方案	6.8×10^{-6}	4.2×10^{-5}	8.3×10^{-6}
Goldschmidt（k= 3）	9.1×10^{-6}	5.7×10^{-5}	1.1×10^{-5}
Goldschmidt（k= 4）	2.4×10^{-7}	1.6×10^{-6}	3.1×10^{-7}

表16　　乘法深度分解对比

步骤	本方案	Goldschmidt（k=3）	Goldschmidt（k=4）
Bootstrapping（L_{boot}）	19	19	19
EvalMod 替换多项式（$\Delta_{本方案}$）	0	—	—
Goldschmidt迭代	—	6	8
方差计算 + 归一化	3	3	3
总深度	**22**	**28**	**30**
节省深度	基准	—	—
相对节省	—	6层（21%）	8层（27%）

表17　单次LayerNorm运行时间分解（秒）

步骤	本方案	Goldschmidt（k=3）	Goldschmidt（k=4）
方差计算（Rotate-Sum）	1.8	1.8	1.8
范围守卫（预缩放+偏移）	0.3	—	—
Bootstrapping	12.1	12.1	12.1
Goldschmidt/Newton迭代	—	4.2	5.6
归一化乘法	0.6	0.6	0.6
LayerNorm总计	**14.8**	**18.7**	**20.1**
加速比	基准	1.26×	1.36×

误差是否集中作用于分类边界附近样本、误差方向是否削弱正确类别 logit 有关。

因此，Goldschmidt(k=4) 虽然在表 14 和表 15 中具有更小的连续数值误差，但其端到端准确率仅比本文方案略高，并不矛盾。这说明在当前实验设置下，本文方案的数值误差已经低于任务输出对 LayerNorm 扰动的敏感阈值。继续降低倒数平方根近似误差只能带来有限的任务级收益，却需要额外乘法深度、运行时间和内存开销。本文的主要结论并不是声称端到端准确率严格优于 Goldschmidt(k=4)，而是在保持接近任务精度的同时显著降低乘法深度和系统开销。

表 20 展示了本方案在不同模型规模下的性能表现。

本方案的加速比在不同模型规模下保持稳定（1.25×-1.28×），表明方案具有良好的可扩展性。对于轻量模型（BERT-Tiny），由于 Rotate-Sum 的旋转次数减少（$\log_2 128 = 7$ 与 $\log_2 768 \approx 10$），单次 LayerNorm 时间略短。

为验证范围守卫机制的鲁棒性，我们统计了各模型在推理过程中方差值落入安全区间的比例。表 21 显示了范围守卫覆盖率统计

所有测试样本的方差均落在 99.9% 分位数确定

表18　　峰值内存占用（GB）

方案	峰值内存	相对本方案的内存增量	本方案相对该方案的内存节省
本方案	6.5	基准	基准
Goldschmidt（k=3）	7.8	+20%	16.7%
Goldschmidt（k=4）	8.4	+29%	22.6%

表19　　BERT-Base在SST-2上的同态推理性能

方案	分类准确率	95% 置信区间	单样本推理时间	总深度消耗
明文推理（Baseline）	92.5%	[90.70%, 93.97%]	0.02s	—
本方案	92.1%	[90.26%, 93.62%]	178.2s	22 × 12层
Goldschmidt（k=3）	91.8%	[89.94%, 93.34%]	225.6s	28 × 12层
Goldschmidt（k=4）	92.3%	[90.48%, 93.80%]	242.4s	30 × 12层

表20　　不同模型规模下的本方案性能

模型	LN层数	单次LN耗时	总LN耗时	相对Goldschmidt(k=3) 加速比
BERT-Tiny（H=128）	4	13.6s	54.4s	1.28×
BERT-Base（H=768）	24	14.8s	355.2s	1.26×
GPT-2单层（H=768）	2	14.9s	29.8s	1.25×

表 21 范围守卫覆盖率统计

模型	总 LayerNorm 调用次数	$\sigma^2 \leqslant V_{max}$ 比例	超出安全区间次数
BERT-Tiny	400	100%	0
BERT-Base	2400	100%	0
GPT-2 单层	200	100%	0

的 V_{max} 范围内，未出现超出安全区间的情况，验证了统计边界方法的有效性。表 21 中 BERT-Base 的 2400 次 LayerNorm 调用均满足范围守卫约束，说明在本文实验分布下，范围守卫在多层、多样本调用中没有出现安全区间外输入。

4.2 范围守卫参数敏感性消融

为验证范围守卫默认参数选择的合理性，本文在 BERT-Tiny 上补充轻量参数敏感性实验。实验固定模型、数据集和多项式阶数不变，仅改变方差上界 Vmax 的校准分位数和安全边界 η，统计不同配置下 LayerNorm 方差输入的覆盖率、软裁剪触发次数和端到端准确率参考值。实验使用 SST-2 验证集前 100 条样本，并统计 Transformer block 内部 4 个 LayerNorm 模块，共 400 次 LayerNorm 记录。需要说明的是，端到端准确率为明文端范围限制模拟结果，即当某次 LayerNorm 的方差超过当前 Vmax 时，将归一化中使用的方差按 Vmax 进行上限截断。该列用于观察范围守卫参数变化对任务输出的影响趋势。

表 22 显示，当 Vmax 分位数由 99.9% 降至 99.5% 和 99.0% 时，覆盖率分别下降至 99.75% 和 99.25%，并分别出现 1 次和 3 次软裁剪触发。这说明较低分位数会增加分布尾部样本超界风险。对于固定的 99.9% 分位数，η=0%、5% 和 10% 均可实现 100% 覆盖和 0 次软裁剪触发，但 η=0% 不保留额外安全边界，对校准误差和 CKKS 近似噪声更敏感。η=10% 则使缩放更加保守，进一步压缩有效输入动态范围。相比之下，η=5% 在保持 100% 覆盖率和 0 次软裁剪触发的同时，保留了适中的正弦单调区间安全余量。因此，本文采用 Vmax=99.9% 分位数和 η=5% 作为默认配置，该选择不是单一配置下的偶然成功，而是在覆盖率、鲁棒性和零额外深度约束之间的折中。

5 结束语

本文提出了一种基于 CKKS 函数型 Bootstrapping 的融合式同态 LayerNorm 方案。该方案在 EvalMod 阶段将标准 arcsin 恢复多项式替换为倒数平方根复合多项式 $P_{invsqrt}(t)$，从而将 $1/\mathrm{sqrt}(\sigma^2+\varepsilon)$ 的计算融合进 Bootstrapping 过程。通过稳定项、预缩放和统计边界构成的默认范围守卫，本文使输入方差落在正弦函数单调区间内，并在不触发可选软裁剪的默认设置下实现倒数平方根计算的零额外乘法深度。实验结果表明，本文方案在保持与 Goldschmidt 方法相近精度的同时，可节省 6 层乘法深度，并降低 LayerNorm 运行时间和峰值内存开销。

本文仍存在进一步改进空间。首先，当前范围守卫依赖离线校准得到的 V_{max}，未来可研究面向分布外输入的自适应范围守卫机制。其次，本文主要验证 LayerNorm 场景，后续可进一步扩展到 RMSNorm 及其他归一化函数。最后，本文实验以单密钥 CKKS 设置为主，未来可探索与多方 FHE 或 MPC-FHE 混合推理框架的结合。

表 22 范围守卫参数敏感性消融结果

配置	Vmax 分位数	η	Vmax	覆盖率	软裁剪触发次数	明文范围限制模拟准确率
较低分位数	99.0%	5%	14.0214	99.25%	3	83.00%
中等分位数	99.5%	5%	14.7932	99.75%	1	83.00%
无安全边界	99.9%	0%	15.2860	100.00%	0	83.00%
本文默认	99.9%	5%	15.2860	100.00%	0	83.00%
更大安全边界	99.9%	10%	15.2860	100.00%	0	83.00%

参考文献：

[1] Gentry C. Fully homomorphic encryption using ideal lattices[C]//Proceedings of the41st Annual ACM Symposium on Theory of Computing (STOC).New York:ACM, 2009: 169-178.

[2] Cheon J H,Kim A,Kim M,et al. Homomorphic encryption for arithmetic of approximate numbers[C]//Advances in Cryptology – ASIACRYPT2017.Cham:Springer, 2017: 409-437.

[3] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled)fully homomorphic encryption without bootstrapping[C]//Innovations in Theoretical Computer Science(ITCS).New York:ACM, 2012: 309-325.

[4] Fan J, Vercauteren F. Somewhat practical fully homomorphic encryption [R].Cryptology ePrint Archive,Report2012/144, 2012.

[5] Chillotti I,Gama N,Georgieva M,et al.TFHE:Fast fully homomorphic encryption over the torus[J].Journal of Cryptology, 2020, 33(1): 34-91.

[6] Vaswani A, Shazeer N, Parmar N,et al. Attention is all you need[C]//Advances in Neural Information Processing Systems(NeurIPS). 2017: 5998-6008.

[7] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding[C]//Proceedings of the2019Conference of NAACL-HLT. 2019: 4171-4186.

[8] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners [C]//Advances in Neural Information Processing Systems(NeurIPS). 2020: 1877-1901.

[9] Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016.

[10] Zhang B, Sennrich R. Root mean square layer normalization[C]//Advances in Neural Information Processing Systems(NeurIPS). 2019: 12360-12371.

[11] Touvron H,Lavril T,Izacard G,et al.LLaMA:Open and efficient foundation language models[J]. arXiv preprint arXiv:2302.13971, 2023.

[12] Cheon J H, Han K, Kim A, et al. Bootstrapping for approximate homomorphic encryption[C]//Advances in Cryptology – EUROCRYPT2018. Cham:Springer, 2018: 360-384.

[13] Bossuat J P, Mouchet C, Troncoso-Pastoriza J, et al. Efficient bootstrapping for approximate homomorphic encryption with non-sparse keys [C]//Advances in Cryptology – EUROCRYPT2021. Cham: Springer, 2021: 587-617.

[14] Goldschmidt R E. Applications of division by convergence[D]. Cambridge:Massachusetts Institute of Technology, 1964.

[15] Ran Z, Deng W, Tang Q, et al. RBOOT: Accelerating homomorphic neural network inference by fusing ReLU within bootstrapping[J].Cryptology ePrint Archive, 2025.

[16] Lee E, Lee J W, Lee J, et al. Low-complexity deep convolutional neural networks on fully homomorphic encryption using multiplexed parallel convolutions[C]//Proceedings of the39th International Conference on Machine Learning(ICML). 2022: 12403-12422.

[17] Al Badawi A, Bates J, Bergamaschi F, et al. OpenFHE: Open-source fully homomorphic encryption library[C]//Proceedings of the6th Workshop on Encrypted Computing&Applied Homomorphic Cryptography (WAHC).New York:ACM, 2022: 53-63.

[18] Gilad-Bachrach R, Dowlin N, Laine K, et al. CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy[C]// Proceedings of the33rd International Conference on Machine Learning (ICML). 2016: 201-210.

[19] Lee J, Lee E, Lee J W, et al. Precise approximation of convolutional neural networks for homomorphically encrypted data[J]. IEEE Access, 2023, 11: 62062-62076.

[20] Drucker N, Moshkowich G, Pelleg T, et al. BLEACH: Cleaning errors in discrete computations over CKKS[J]. Journal of Cryptology, 2024, 37(1): 3.

[21] Bae Y, Cheon J H, Kim J, et al. Bootstrapping bits with CKKS[C]// Joye M, Leander G, editors. Advances in Cryptology - EUROCRYPT 2024, Part II. Cham: Springer, 2024: 94-123.

[22] Bae Y, Kim J, Stehle D, et al. Bootstrapping small integers with CKKS [C]//Chung K M, Sasaki Y, editors. Advances in Cryptology - ASIACRYPT 2024, Part I. Singapore: Springer, 2024: 330-360.

[23] Alexandru A, Kim A, Polyakov Y. General functional bootstrapping using CKKS[C]//Kalai Y T, Kamara S F, editors. Advances in Cryptology - CRYPTO 2025, Part III. Cham: Springer, 2025: 304-337.

[24] Kim J. Efficient homomorphic integer computer from CKKS[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2025(4): 873-898.

[25] Boneh D, Kim J. Homomorphic encryption for large integers from nested residue number systems[C]//Kalai Y T, Kamara S F, editors. Advances in Cryptology - CRYPTO 2025, Part III. Cham: Springer, 2025: 338-370.

[26] Bossuat J P, Cammarota R, Chillotti I, et al. Security guidelines for implementing homomorphic encryption[J]. IACR Communications in Cryptology, 2024, 1(4): 26.

陈智罡（1972– ），男，四川资中人，博士，宁波财经学院教授，主要研究方向为全同态加密与格密码。

宋新霞（1973– ），女，陕西西安人，硕士，浙江万里学院教授，主要研究方向为代数与编码。

1956– ），女，上海人
授，主要研究方向为