arXiv:2603.14110v1 [cs.LG] 14 Mar 2026

用于快速大语言模型推理的奇异值分解上下文稀疏性预测器

Georgii Serbin [1] Kirill Koshkin [2] Zhongao Sun [1] Anastasiya Bistrigova [1 2] C.C. Korikov [1]

摘要

上下文稀疏性是降低大语言模型（LLM）推理过程计算复杂度的方法之一。现有基于上下文稀疏性、旨在实现高效大语言模型推理加速且尽量减少准确率下降的技术，需要训练稀疏模式预测器。本文提出一种框架，用于加速大语言模型中基于 ReGLU 的前馈网络（FFN）的推理。该框架提供了一种快速、无需训练的方法：利用门投影矩阵的截断感知奇异值分解（SVD）构建稀疏模式预测器，并结合阈值校准算法，以及支持在 CUDA 和 CANN 设备上进行条件计算的推理执行器。在三种稀疏大语言模型上开展的实验表明，这些模型的前馈网络平均激活稀疏度为 90%，在复杂数学和代码生成任务的基准测试分数下降不到 1% 的同时，端到端解码时间最高可减少 1.8× 。本研究推动了大语言模型在边缘设备上的部署。

1. 引言

大语言模型的卓越能力使其广泛应用于各个领域（Brown 等人，2020年；Dong 等人，2024b）。与基于服务器的解决方案相比，端侧人工智能部署在隐私、延迟和成本方面具有优势（Shankar，2024年），但大语言模型推理所需的大量计算资源给边缘部署带来了重大挑战。

基于ReGLU的大语言模型的一个关键特性是动态（或上下文）稀疏性——对于给定的输入，前馈神经网络块中只有一小部分神经元会被激活。这种固有特性使推理能够通过减少生成过程中的计算开销和内存带宽需求来实现加速（Liu 等人，2023）。

利用上下文稀疏性的一种常见方法是采用轻量级、可学习的稀疏性预测器（Liu 等人，2023）。这些预测器会在前馈网络前向计算之前生成特定于输入的稀疏激活模式。稀疏模式标识出与输入向量进行点积后将产生正值的神经元索引。在 ReGLU 前馈神经网络中，可以安全地跳过其余神经元的计算而不影响输出正确性，因为激活函数会将负值置零。然而，假阴性分类可能导致预测误差，进而损害模型准确率。有效的预测器必须在保持足够速度以降低端到端（E2E）文本生成延迟的同时，兼顾较高的召回率和较高的稀疏率。

在本研究中，我们提出了一种简单而有效的技术，利用低秩奇异值分解（奇异值分解）预测稀疏模式，从而显著加速基于ReGLU的大语言模型的推理。我们的方法采用每一层门控投影权重矩阵的低秩近似，作为生成过程中前馈神经网络块稀疏掩码的轻量级预测器。与现有的上下文稀疏性预测方法不同，我们的方法为预测准确率提供了理论保证。

为提高基于奇异值分解的预测器的准确率，我们采用了以下组合技术：

- **校准偏置**。直接将奇异值分解近似用作预测器会导致准确率明显下降。然而，我们观察到奇异值分解预测器具有很强的区分能力。引入以偏置项实现的神经元特定阈值，可以有效缓解这一问题。这些偏置通过一种定制的贪心算法离线校准，无需进行梯度计算。

- **数据白化技术**。鉴于预测器的速度限制，秩必须远小于隐藏维度。为了在这种条件下提高预测准确率，我们采用截断感知数据白化（Wang 等人，2025a；b；Yuan 等人，2025）。

[1]华为技术有限公司 [2]俄罗斯莫斯科国立大学。通信作者：C.C. Korikov <constan- tine.korikov@gmail.com>。

预印本。*2026*年*3*月*17*日。

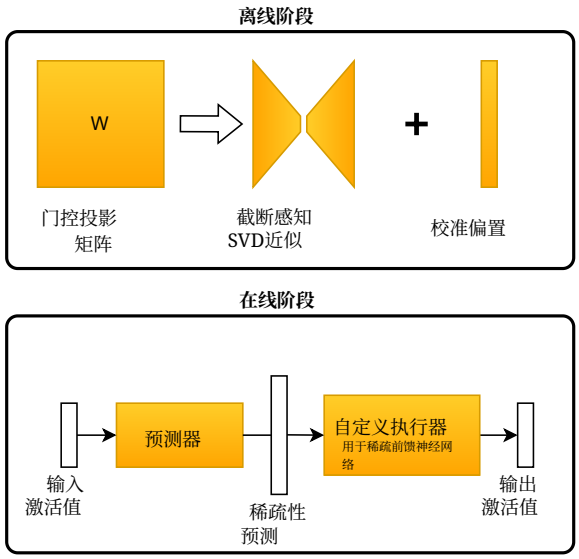


图*1.* 我们的框架概览。在离线阶段，使用低秩奇异值分解近似和校准后的逐神经元阈值构建稀疏性预测器。在推理时，基于奇异值分解的预测器置于每个前馈神经网络块之前，以利用上下文稀疏性并加速生成。

该技术通过纳入输入激活统计信息来降低弗罗贝尼乌斯范数误差，且仅需一个小型校准数据集。

实现层面的优化进一步提升了推理加速比：

- **门控投影和升维投影的顺序计算**。门控投影和升维投影按顺序计算，通过提前消除假阳性预测，实现进一步加速。在计算门控投影后，识别出被错误激活的神经元，并将其排除在后续的升维投影和降维投影之外，从而提高最终前馈网络阶段的稀疏度。

- **推理执行器**。我们的实现支持在 CUDA（英伟达，2024年）和 CANN（华为技术有限公司，b）设备上进行条件计算。对于 CUDA，我们采用自定义内核，而不是基于 Triton 的抽象（Tillet等人，2019年），优先实现更强的底层控制。在华为昇腾硬件上（Liao等人，2019年），仅通过高层 PyTorch API 即可获得可比的性能提升，展现出对该平台原生软件栈的有效利用。

图1 展示了我们的框架架构。我们的主要贡献总结如下：

1. 我们提出了一种无需训练、基于奇异值分解的上下文稀疏性预测器，用于加速基于 ReGLU 的大语言模型推理；

2. 我们引入了校准偏置项，可显著提高预测准确率；这些偏置项使用自定义的无梯度算法，在小型数据集上进行调优；

3. 我们开发了支持在 CUDA 和 CANN 设备上进行稀疏前馈网络推理的执行器；

4. 我们实现了最高 1.8× 的端到端推理加速，同时准确率下降极小。

2. 相关工作

稀疏化是加速大语言模型推理最有效的范式之一，通常分为**静态**和**动态稀疏性** 方法。

静态稀疏性 会在部署前移除冗余参数（Ashkboos 等人，2024年），从而降低计算量和内存成本，但在稀疏度较高时通常会牺牲准确率。相比之下，**动态稀疏性**（也称为动态剪枝或上下文稀疏性）会根据输入上下文在运行时调整稀疏模式，从而实现灵活的效率与准确率权衡。

基于预测器的动态稀疏性 方法，例如 Deja Vu（Liu 等人，2023）和 ShadowLLM（Akhauri 等人，2024年），会训练轻量级神经预测器来估计逐层稀疏性。ShadowLLM 通过在训练期间纳入梯度信息并为所有层使用单个预测器，进一步改进了这一方法。

免训练动态稀疏性 无需训练预测器，而是根据激活统计或校准阈值推导稀疏性。GRIFFIN（Dong 等人，2024a）利用预填充激活选择活跃的“专家”神经元，而 TDA（Ma 等人，2024年）引入离线校准阈值，以实现每层自适应激活。两者在同一提示内都维持固定的稀疏模式。CLADA（Yang 等人，2025）会在生成过程中更新稀疏性，但需要重新计算先前状态，从而妨碍 KV 缓存的复用。TEAL（Liu 等人，2024年）利用输入稀疏性；它与激活稀疏性的区别在于，输入稀疏性源自输入表示中的冗余，而非神经元激活，并使用校准阈值。SparseInfer（Shin 等人，2025）提出了一种针对 ReLU 化大语言模型、采用符号位比较的校准方案，但缺乏理论依据。

动态稀疏性还可以支撑高效的设备端推理框架。PowerInfer（Song 等人，2024b）通过将频繁激活的神经元预加载到 GPU 内存中，最大限度地减少 CPU数据传输，从而加速消费级 GPU 上的大语言模型。其加速比最高可达 11。69× 倍，尤其适用于 CPU/GPU混合环境，而我们的工作针对同构

批量大小为一的 GPU推理。

动态稀疏性的效果很大程度上取决于激活函数。与现代大语言模型中使用的 Swish 或 SiLU 等平滑函数不同，ReLU 激活会自然地产生稀疏激活（Zhang 等人，2024年）。近期研究表明，将它们替换为 ReLU 变体并进行微调，可以恢复相当程度的稀疏性（Mirzadeh等人，2023）。ProSparse（Song 等人，2024a）通过渐进式正则化和激活阈值偏移，在 LLaMA2-7B 中实现了 ∼90% 的稀疏性，而 TurboSparse（Song 等人，2024c）则采用双ReLU（dReLU）激活函数，使 Qwen2-7B 和 Mistral-7B 等模型变得稀疏。

除了剪枝之外，**奇异值分解** 还提供了互补的压缩能力。ASVD（Yuan 等人，2025）利用感知激活的变换实现 10–30% 的模型压缩，而 SVD-LLM（Wang 等人，2025b）则从理论上阐明了被丢弃分量与 L_2 损失之间的联系。

SparseLoRA（Khaki 等人，2025）将基于奇异值分解的估计器融入参数高效微调，并根据输入相关性动态切分权重。然而，稀疏计算仅应用于上下文词元（作为输入提供的前缀词元），而 输出词元（用于损失计算的目标词元）仍采用稠密处理，以避免性能下降。通过上下文稀疏性实现大语言模型的快速且高质量解码，仍是 SparseLoRA 尚未解决的一项开放挑战。

3. 预备知识

考虑解码期间一个经 ReGLU 激活的前馈网络块，其定义如下：

$$\mathrm{FFN}(x) = W^{\mathrm{down}}\left(\mathrm{ReLU}(W^{\mathrm{gate}}x) \odot (W^{\mathrm{up}}x)\right), \quad (1)$$

其中，$x \in \mathbb{R}^d$ 是输入激活向量，$W_{\text{门控}}$ $W^{\mathrm{up}} \in \mathbb{R}^{D\times d}$、$W_{\text{下投影}} \in \mathbb{R}^{d\times D}$ 分别是门控矩阵、上投影矩阵和下投影矩阵。这里，d 表示隐藏维度，D 表示中间维度。令 $X \in \mathbb{R}^{d\times N}$ 表示 N 校准词元的隐藏状态，r 表示预测器秩。为简洁起见，我们省略层索引。

我们的目标是获得一个预测器 P，它能在 D 中间神经元上输出一个二元掩码 $m = P(x) \in \{0,1\}^D$。该掩码通过选择 W^{gate} 和 W^{up}中对应的行，以及 W^{down}中与之对齐的列，来识别活跃神经元。由于 ReLU 稀疏性，这样便能实现与稠密前馈网络在代数上等价的稀疏计算。

4. 基于SVD的稀疏性预测器

本节介绍无需训练的基于奇异值分解的稀疏性预测器（SVDP），旨在实现高效推理

基于ReGLU的大语言模型。我们的方法由两个阶段组成：用于预测器构建的离线阶段，以及用于加速稀疏推理的在线阶段。

4.1. 免训练预测器构建

预测器通过低秩奇异值分解近似门投影矩阵 W^{gate} 构建。这种方法为局部误差提供了一个理论上界（参见第 A.1节）。

我们提出了仿射形式的预测器

$$P(x) = H(ABx + b), \quad (2)$$

其中 H： $\mathbb{R} \to \{0,1\}$ 是逐元素应用的海维赛德阶跃函数。若 $z > 0$，则 $H(z) = 1$；否则为 0。乘积 AB 近似于 W^{gate}，而 b 则用于调整每个神经元的决策阈值。

4.1.1. 数据感知的奇异值分解因式分解

为提高奇异值分解近似的准确率，我们采用数据白化技术，以最小化 Frobenius 范数误差：

$$\|WX - U_r\Sigma_r V_r^\top X\|_F \quad (3)$$

在校准数据 X 上，通过白化矩阵 S 纳入输入统计信息：

$$\begin{aligned} W &= (WS)S^{-1} \\ &= (U\Sigma V^\top)S^{-1} \approx U_r\Sigma_r V_r^\top S^{-1} = AB. \end{aligned} \quad (4)$$

我们遵循Wang 等人（2025b）的做法，选择 S 作为 $XX^\top$的乔列斯基分解。正如Yuan 等人（2025）的表 2 所示，其在低秩情况下的困惑度优于 ASVD。

预测器构建流程如算法 1所述，并针对每一层独立执行。整个过程中，我们固定各层的预测器秩 r （关于秩选择的讨论，请参见第 E.3 节）。

算法 1 构建 SVD 预测器（离线阶段）

输入： $W_{\text{门控}} \in \mathbb{R}^{D\times d}$，$X \in \mathbb{R}^{d\times N}$，$r \in \mathbb{N}$ **输出：** $A \in \mathbb{R}^{D\times r}$，$B \in \mathbb{R}^{r\times d}$
$S \in \mathbb{R}^{d\times d} \leftarrow$ Cholesky 分解 $(XX^\top)$ U，Σ，$V \leftarrow$ 奇异值分解 $(W^{\mathrm{gate}}S)$
$A \in \mathbb{R}^{D\times r} \leftarrow U_{:,\,:r}\Sigma_{:r,\,:r}$ $B \in \mathbb{R}^{r\times d} \leftarrow V_{:r,\,:}S^{-1}$ **返回** A，B

4.1.2. 阈值校准

基于奇异值分解的稀疏性预测器通常能够很好地区分活跃神经元和非活跃神经元；然而，简单的

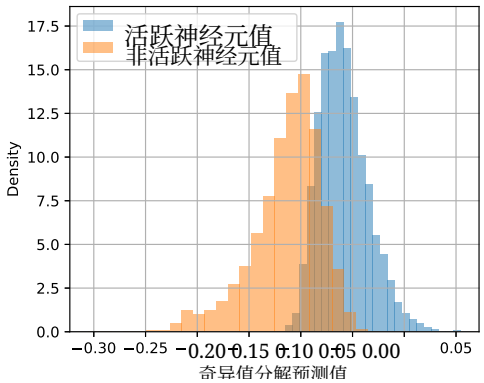


图2。 基于奇异值分解的预测器对某个样本神经元的输出分布。尽管活跃状态与非活跃状态区分明显，但采用零阈值会导致真正活跃的神经元被大量误判为非活跃。

阈值处理（例如设为零）会导致准确率大幅下降。这是由于截断奇异值分量后，预测器输出分布发生偏移所致，如图所示2。

为补偿这种分布偏移，我们在基于奇异值分解的预测器的第二层引入一个偏置项。核心问题是如何校准该偏置，使其达到预设的稀疏度，同时将相关的准确率损失控制在一定范围内。经验表明，在校准集上调整阈值以匹配目标稀疏率、召回率或准确率的简单方案，会导致次优的权衡。

相反，我们采用一种快速的贪心校准算法（算法 2），它明确考虑了不同神经元对模型准确率的异质性影响。从计算角度来看，所有神经元对加速的价值相同，因为剪除任意一个神经元都会以相同幅度降低计算成本；然而，它们对输出准确率的贡献却大不相同。因此，有些神经元需要较高的召回率，而另一些神经元即使允许更高的假阴性率，也不会明显影响整体性能。在某些条件下，我们的贪心算法能够保证提供最优解（见第 A.2节）。

令 $s \in (0,1]$ 表示期望的稀疏率，用于控制速度与准确率的权衡（见第 E.3 节）$g_i^{(t)}$对于每个神经元 $i \in [D]$，初始化 $\tau_i \leftarrow \min_{t:\ w_i^{(t)}>0}$ 。然后使用贪心算法通过优先推进增量成本最低的神经元来提高稀疏性 $\sum_{j=1}^{\eta} w_{i,\ (k_i+j)}$ 直到

$$\frac{1}{DT}\sum_{i=1}^{D}\sum_{t=1}^{T}\mathbf{1}[s_i^{(t)} \le \tau_i] \ge s.$$

惩罚项 $C_i(k_i+\eta) - C_i(k_i)$ 近似表示跨越 η 个样本的前馈网络输出扰动。固定偏置 $b = -\tau \in \mathbb{R}^D$ 会在推理期间使用。

所提出的预测器轻量且数据高效，只需数千个词元即可可靠地构建预测器并进行偏差估计。它速度快且内存占用低，适合在受限环境中实际使用。

4.2. 运行时执行

我们仅修改前馈神经网络，其余模型保持不变。推理期间，预填充阶段保持不变。在解码阶段，我们利用 CUDA 和 CANN 执行器进行稀疏前馈神经网络计算。图3展示了稀疏推理流水线，其中加速比由依赖词元的稀疏率决定。

稀疏度预测 我们的预测器由两个线性层组成，无需专门实现。Deja Vu 使用前瞻预测器来规避预测延迟。这通过异步计算实现：预测器的执行时间与从全局内存进行的数据传输时间，或张量并行下的进程间通信时间重叠。然而，在某些硬件平台上，这种流水线处理可能不可行（见第 B节）。与 PowerInfer 引擎类似，鉴于我们专注于边缘部署场景中的本地推理，我们将预测器插入每个前馈神经网络的开头，并依次为各自的门控投影生成稀疏模式。

门控/升维执行顺序 一种常见的优化方法是并行执行门控前馈神经网络中的门控投影和上投影，以提高大语言模型的部署效率。尽管这种方法在密集计算场景中效果显著，但对于利用稀疏激活的本地推理系统而言，它带来了不小的挑战。我们提出两种不同的计算方法来应对这一问题：

- 并行流水线例如 PowerInfer。预测器首先生成稀疏掩码，并同时将其应用于门控投影和上投影的独立计算中。这样可确保执行期间两个投影的稀疏度保持一致。
- 顺序流水线例如 SparseInfer 和我们的方法。门控投影和上投影按顺序计算，从而消除假阳性：被预测为激活的神经元会在门控投影计算后重新验证，进一步提高后续投影的稀疏度。

算法复杂度 在我们的方案中，通过过滤错误的预测，实际稀疏度等于或高于预测稀疏度 s。得益于顺序流水线，我们可以使用更轻量级的预测器，在门控投影阶段提供相对较低的稀疏度，同时不影响前馈网络的性能。

算法 2 贪心偏置校准（离线阶段）

输入：
$W^{\text{gate}}, W_{\text{上}} \in \mathbb{R}^{D\times d}, W_{\text{下}} \in \mathbb{R}^{d\times D}$ – 权重矩阵，
$\mathcal{D} = \{x^{(t)} \in \mathbb{R}^d\}_{t=1}^{T}$ – 校准数据集，
A， B – 来自算法 1，
$s \in [0;\ 1]$ – 目标稀疏度， η – 离散步长。
输出： $b \in \mathbb{R}^D$
$w_i^{(t)} \leftarrow \left|\mathrm{ReLU}(W_{i,:}^{\text{gate}} x^{(t)}) \cdot W_{\text{向上}i,:} x^{(t)}\right| \|W_{\text{向下}:,i}\|_2^2$ （计算神经元重要性）
$s_i^{(t)} \leftarrow (ABx^{(t)})_i$ （计算预测器的输出）
对于 $i = 1$ **到** D **执行**
$s_{i,(1)} \le \cdots \le s_{i,(T)}$ （按非递减顺序排列样本）

$$C_i(k) = \begin{cases} \sum_{j=1}^{k} w_{i,(j)}, & k = 1, \ldots, T \\ C_i(T), & T < k < T + \eta \\ +\infty, & k = T + \eta \end{cases}$$ （使用填充计算累积成本）

$k_i \leftarrow$ 最小$\{k :\ w_{i,(k)} > 0\}$， $\tau_i \leftarrow s_{i,(k_i)}$ （初始化指针和阈值）
结束循环
当 $\frac{1}{DT}\sum_{i=1}^{D}\sum_{t=1}^{T} \mathbf{1}[s_i^{(t)} \le \tau_i] < s$ **执行**
$i^\star \leftarrow$ 使$_i$ $[C_i(k+\eta) - C_i(k)]$最小 （贪心算法步骤）
$k_{i^\star} \leftarrow$ 最大值$(k_{i^\star} + \eta,\ T)$， $\tau_{i^\star} \leftarrow s_{i,(k_{i^\star})}$ （更新指针和阈值）
结束循环
$b \leftarrow -\tau$
返回 b

稠密前馈神经网络需要 $3dD$ 次乘法（不包括逐元素乘积），而我们的方法将其减少到：

$$\underbrace{r(d+D)}_{\text{predictor}} + \underbrace{d(1-s)D}_{\text{gate}} + \underbrace{2d(1-s')D}_{\text{up/down}} \tag{5}$$

其中 s 是预测稀疏度， s' 是实际稀疏度。例如，在 ProSparse-LLaMA2-7B（Song 等人，2024a）（$d = 4096$、 $D = 11008$、 $s = 0.5$、 $s' = 0.9$、 $r = 256$）中，运算量减少了 3。8×。请注意，硬件级动态稀疏性带来的挑战使其无法直接转化为实际运行时间加速。

硬件平台 支持在 CUDA 和 CANN 设备上进行 sparseFFN 推理。实现细节请参阅第 B 节。

5. 实验结果

5.1. 实验设置

硬件 在 MSU计算机工程实验室使用一块面向单个消费者的英伟达 RTX 3090 GPU [1]

[1]所有基于GPU的实验均由莫斯科国立大学的作者（Kirill Koshkin）独立完成，华为方面的研究人员未参与其中。华为方面的合著者仅参与理论分析、算法设计以及 CANN 性能评估。

用于质量和性能评估。推理完全在 GPU 上进行。

在神经网络处理器上的评估中，使用一台昇腾 310P3（华为技术有限公司，a）设备。

软件 实验中我们使用 vLLM 框架（Kwon 等人，2023），而非 Transformers 框架（Wolf 等人，2020），因为前者更能反映真实世界的部署场景。值得注意的是，我们没有使用 vLLM 的任何服务器专属功能，例如张量并行。推理采用半精度，而离线阶段的预测器构建则采用双精度。

模型 我们在三种开源稀疏 7B 语言模型上评估了所提出的方案： **ProSparse-LLaMA2-7B**(Song 等人，2024a) 以及两种指令微调模型——**SparseQwen2-7B** 和**TurboSparse-Mistral-Instruct** (Song 等人，2024c)。

指令微调模型在其前馈网络层中采用双ReLU（dReLU）激活：

$$\mathrm{FFN}(x) = W^{\text{down}}\left(\mathrm{ReLU}\left(W^{\text{gate}}x\right) \odot \mathrm{ReLU}(W^{\text{up}}x)\right) \tag{6}$$

质量评估 我们使用一系列涵盖推理、编程、知识和多语言理解的开源基准测试来评估模型性能： **GSM8K** (Cobbe 等人，2021年）（数学推理）、**HumanEval**(Chen 等人，2021年）（代码生成）、**ARC-**

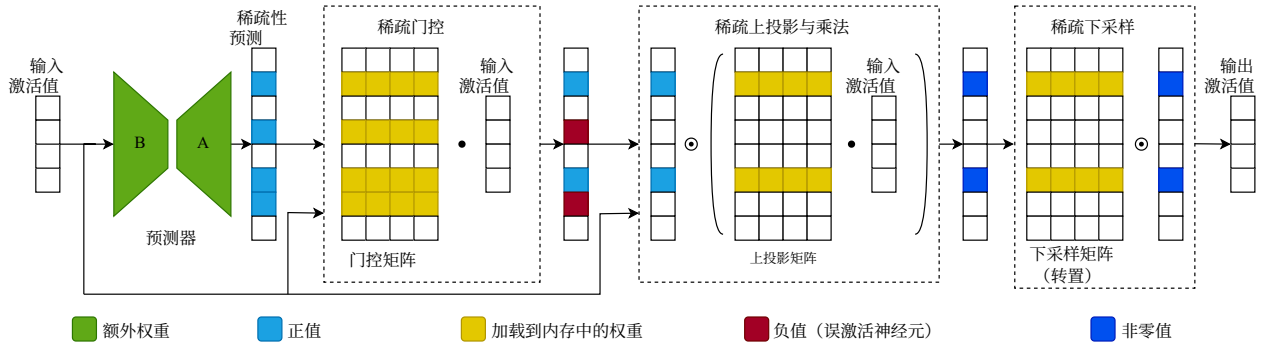


图 *3*。 使用基于SVD的稀疏性预测器进行稀疏推理。在门控投影期间，预测器值为正的神经元被视为活跃神经元。门控投影和升维投影按顺序执行，以提高稀疏性。

E/ARC-C(Clark et al.，2018）（科学与常识推理）、**TriviaQA** (Joshi etal.，2017）（开放领域问答）、**MBPP** (Austinet al.，2021）（小型 Python 任务）、**BBH** (Suzgun et al.，2022）（复杂推理），以及 **CMMLU** (Li etal.，2024年）（中文多任务理解）。

我们采用 **UltraEval**框架（He 等，2024年），与先前的 ProSparse 评估保持一致，因为该模型与 **lm-eval-harness** （Gao 等，2024年）存在兼容性问题。所有基准测试均采用自由形式生成，而不是困惑度评分，以更好地反映大语言模型在现实世界中的使用情况。

性能评估 模型性能通过**PyTorch 性能分析器** （Ansel 等，2024年）进行测量。对于每个领域，结果取多个样本的平均值。端到端延迟包括预填充阶段和 200 个 token 的生成过程。

5.2. 不同稀疏率下的性能

所需的稀疏性超参数 s来自算法 2，会影响校准阈值，可用于权衡准确率与加速效果。如表 1 所示，较低的阈值会带来较低的稀疏率和较低的加速比。相反，较高的阈值能够提供更高的加速比，但可能会降低质量。长文本生成任务最为敏感，而对于短答任务，稀疏率可以显著提高。采用我们的方法，在涉及复杂数学和代码生成的任务中，我们最多可以实现 1.8× 端到端生成加速，同时将基准测试分数的下降控制在 1% 以内。详细的评估结果请参见第 D 节。

5.3. 与其他方法的比较

我们将 SVDP 与免训练稀疏方法（GRIFFIN（Dong 等人，2024a））以及可学习预测器（PowerInfer（Song 等人，2024b）、Deja Vu（Liu 等人，2023））进行比较。GRIFFIN 依赖基于提示的统计稀疏模式，

表 *1.* 不同模型在改变超参数 s 时的平均准确率。 s 可隐式设定稀疏性，并直接转化为端到端加速比。

模型	方法	s	准确率，%	加速比
ProSparse-LLaMA2-7B	稠密	–	38.92	1.00×
	SVDP	0.4	38.66（-0.26）	1.61×
		0.5	38.78 (-0.14)	1.64×
		0.6	38.77 (-0.15)	1.67×
		0.7	38.57 (-0.35)	1.71×
TurboSparse-Mistral-Instruct	稠密	–	54.30	1.00×
	SVDP	0.4	53.72（-0.58）	1.67×
		0.5	53.56（-0.74）	1.73×
		0.6	53.18（-1.12）	1.79×
		0.7	53.44 (-0.86)	1.84×
SparseQwen2-7B	稠密	–	65.84	1.00×
	SVDP	0.4	65.41 (-0.43)	1.78×
		0.5	65.20 (-0.64)	1.81×
		0.6	65.13 (-0.71)	1.85×
		0.7	64.49 (-1.35)	1.86×

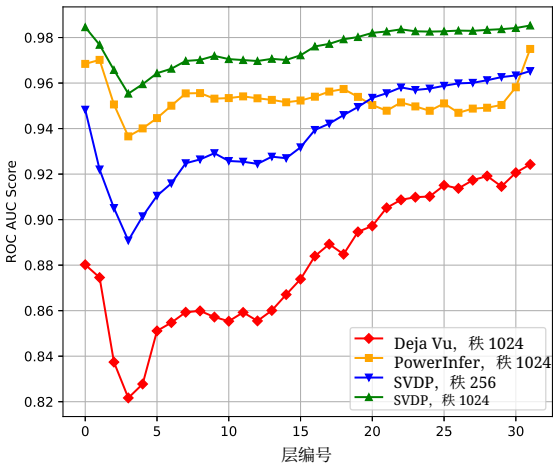


图*4.* ProSparse-LLaMA2- 7B 在多个序列样本上的不同预测器逐层ROC AUC分数的平均值。AUROC分数为1表示完美区分，AUROC分数为0.5表示随机猜测。

表 2. 在 ProSparse-LLaMA2-7B 上与其他方法的比较，使用我们的框架进行评估（$r = 1024$ 并行，$r = 256$ 顺序）。

方法	预测稀疏度，%	TriviaQA 准确率，%	ARC-E 准确率，%	ARC-C 准确率，%	HumanEval pass@1，%	MBPP pass@1，%	GSM8K 准确率，%	BBH 准确率，%	CMMLU 准确率，%	平均值得分	E2E 加速比
稠密基线	–	62.94	75.42	53.58	17.68	22.59	11.45	35.85	31.84	38.92	1.00×
训练型预测器											
Deja Vu	20	59.47	74.87	53.41	3.66	2.98	3.03	34.49	30.97	32.86	1.38×
PowerInfer	80	61.10	74.54	52.99	15.85	21.15	10.69	35.29	31.80	37.93	1.63×
无需训练的方法											
GRIFFIN	20	59.77	74.75	53.33	15.85	22.18	8.95	35.49	32.01	37.79	1.14×
GRIFFIN	50	35.74	72.77	51.96	12.20	18.17	4.47	35.32	31.88	32.81	1.46×
Ours											
SVDP	75	62.66	75.38	53.58	17.68	22.18	10.54	35.86	31.85	38.72	1.61×
SVDP（顺序）	50	62.74	75.34	53.67	17.68	22.07	11.22	35.58	31.94	38.78	1.64×

而 PowerInfer 和 Deja Vu 使用训练型预测器。

对于 Deja Vu，我们按照其官方仓库中的说明训练了预测器。对于 PowerInfer，我们使用了其推荐的预训练预测器。

基准测试 表 2 显示，我们的 SVDP 达到了与 PowerInfer 相当的准确率和稀疏性，同时构建所需的数据和计算量更少。GRIFFIN 难以在高稀疏率下保持稳定的性能，尤其是在长文本生成时。顺序流水线实现将 SVDP 的大小减少了 4×，且不影响准确率或推理速度。

ROC-AUC 分析 我们还通过在留出的测试序列上计算逐层 ROC-AUC 得分，评估了预测器的区分能力（图 4）。SVDP 在未见过的数据上优于可学习基线。训练型预测器可能因其依赖数据的特性而对分布外数据敏感，而奇异值分解预测器则直接利用大语言模型的内部权重结构。

5.4. 消融研究

表3 展示了对 SVD 稀疏度估计器的消融实验，以 SparseLoRA 的基线（Khaki 等人，2025）（称为朴素奇异值分解）为起点：

- 数据白化：纳入输入数据统计信息（第4.1.1节）。
- 偏置校准：添加神经元特定偏置项（第4.1.2节）。

偏置校准通过考虑低秩近似后的分布偏移，显著提升了准确率。

数据白化显著提升了留出测试序列上相较于朴素奇异值分解的 ROC-AUC 区分度（图 5），且不受偏置校准的影响。

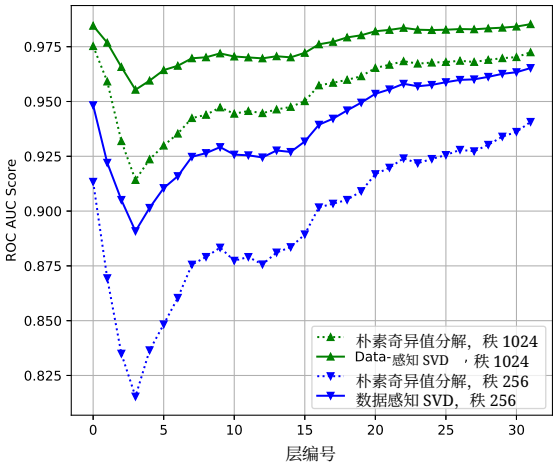


图5。 ProSparse-LLaMA2-7B 的逐层 ROC-AUC：数据感知 SVD 与朴素奇异值分解的对比。

5.5. 推理执行器

表4展示了我们的 CUDA 和 CANN 执行器在不同稀疏率下的前馈网络性能。此处使用的是华为昇腾 310P3 AI 加速器。值得注意的是，我们仅使用高级 PyTorch API，便在昇腾硬件上实现了相当的性能提升，无需手动优化底层指令。这一结果有助于在达芬奇架构中深入探索上下文稀疏优化。值得注意的是，当稀疏值接近 90% 时，L1（Liao等人，2019年）缓存命中率几乎达到 100%，从而显著提高加速比。

6. 局限性

- 我们的方法针对稀疏模型，这类模型中的非线性激活函数会自然地产生大量零值输出。将其扩展到基于非 ReGLU 的架构，是未来一个重要的研究方向。

表 *3.* SVD稀疏度估计器消融研究。 †所有基准测试的平均值（见附录D）。

配置	ProSparseLLAMA2-7B			TurboSparse-Mistral-Instruct			SparseQwen2-7B		
	GSM8K	HumanEval	平均值†	GSM8K	HumanEval	平均值†	GSM8K	HumanEval	平均值†
稠密基线	11.45	17.68	38.92	61.71	34.15	54.30	77.1	64.02	65.84
并行流水线									
朴素奇异值分解	6.60	10.98	34.08	60.20	24.39	51.52	74.00	53.66	62.05
+ 数据白化	10.16	16.46	37.10	60.88	37.20	53.02	75.89	61.59	63.99
+ 偏置校准	10.54	17.68	38.72	62.24	36.59	54.34	76.65	62.20	65.37
顺序流水线									
朴素奇异值分解	1.90	0.61	9.69	9.17	1.83	30.87	63.00	12.20	51.60
+ 数据白化	6.90	8.54	31.00	48.60	29.27	46.49	72.93	57.32	60.80
+ 偏置校准	11.22	17.68	38.78	61.41	33.54	53.56	76.80	62.80	65.22

表 *4.* 不同稀疏率下，我们的 CUDA 和 CANN 执行器的前馈网络性能。加速比是相对于对应的稠密实现计算的。不包括预测器的延迟。在此实验中，所有线性层的稀疏率均固定。

大语言模型架构	稀疏率，%	前馈网络加速比	
		CUDA	CANN
ProSparseLLAMA2-7B	20	1.30×	0.76×
	50	1.90×	1.23×
	80	3.34×	3.38×
	95	4.67×	30.67×
TurboSparse-Mistral-Instruct	20	1.29×	0.76×
	50	1.90×	1.18×
	80	3.44×	3.10×
	95	4.92×	22.85×
SparseQwen2-7B	20	1.33×	0.81×
	50	1.95×	1.32×
	80	3.47×	3.41×
	95	4.91×	28.46×

研究。详见第C 节，了解详细讨论。

- 批处理本质上会降低稀疏度，从而限制可实现的加速比，并使得无法在预填充阶段使用稀疏性预测器。

结论

本文提出了一种无需训练的框架，通过利用上下文稀疏性来加速基于 ReGLU 的大语言模型推理。所提出的解决方案包括两个阶段：(1) 离线校准阶段，使用带有校准偏置参数的门控投影矩阵进行数据感知截断奇异值分解，以构建稀疏性预测器；(2) 在线推理阶段，使得能够在 CUDA 和 CANN 设备上高效执行稀疏大语言模型。在三个稀疏语言模型上的评估表明，该方法在包括代码和数学推理任务在内的多种基准测试中，能够实现最高达 1.8× 端到端解码加速，同时仅造成极小的准确率下降，推理任务。

致谢

感谢密歇根州立大学计算机工程实验室开展所有基于GPU的实验。华为关联作者声明，由于美国出口管制法规，他们无法接触或参与基于GPU的实验。没有华为关联研究人员参与本研究中依赖GPU的工作。

影响声明

本文提出了一种加速大语言模型推理的方法。尽管我们的工作可能带来许多潜在的社会影响，但我们认为其中没有任何影响需要在此特别强调。

参考文献

Akhauri,Y., AbouElhamayed,A. F., Dotzel,J., Zhang, Z., Rush, A. M., Huda, S., and Abdelfattah, M. S. Shad- owll m：基于预测器的上下文稀疏性在大语言模型中的应用，2024年。网址 https://arxiv.org/abs/2406.16635。

Ansel, J., Yang, E., He, H., Gimelshein, N., Jain, A., Voz- nesensky, M., Bao, B., Bell, P., Berard, D., Burovski, E., Chauhan, G., Chourdia, A., Constable, W., Desmaison, A., DeVito, Z., Ellison, E., Feng, W., Gong, J., Gschwind, M., Hirsh, B., Huang, S., Kalambarkar, K., Kirsch, L., Lazos, M., Lezcano, M., Liang, Y., Liang, J., Lu, Y., Luk, C., Maher, B., Pan, Y., Puhrsch, C., Reso, M., Saroufim, M., Siraichi, M. Y., Suk, H., Suo, M., Tillet, P., Wang, E., Wang, X., Wen, W., Zhang, S., Zhao, X., Zhou, K., Zou, R., Mathews, A., Chanan, G., Wu, P., and Chintala, S. PyTorc h 2：通过动态Python字节码转换和图编译加速机器学习。载于第*29*届 *ACM* 编程语言与操作系统架构支持国际会议，第*2*卷（*ASPLOS '24*）

系统。ACM，2024年4月。doi：10.1145/3620665.3640366。网址 https://docs.pytorch.org/assets/pytorch2-2.pdf。

Ashkboos, S., Croci, M., Gennari do Nascimento, M., Hoe- fler, T. 和 Hensman, J. Slicegpt：通过删除行和列压缩大型语言模型。载于 Kim, B., Yue, Y., Chaudhuri, S., Fragkiadaki, K., Khan, M. 和 Sun, Y.（编），国际表征学习会议，第 2024 卷，第 11682–11701 页，2024年。

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q. 和 Sutton, C. 使用大语言模型进行程序合成，2021年。URLhttps://arxiv.org/abs/2108.07732。

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. 语言模型是少样本学习者。载于第*34*届神经信息处理系统国际会议论文集，NIPS '20，美国纽约州雷德胡克，2020年。Curran Associates Inc. ISBN 9781713829546。

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavar- ian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert- Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saun- ders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., Mc- Grew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. 评估使用代码训练的大语言模型，2021年。网址 https://arxiv.org/abs/2107.03374。

Clark, P.、Cowhey, I.、Etzioni, O.、Khot, T.、Sabharwal, A.、Schoenick, C. 和 Tafjord, O. 你认为自己已经解决了问答问题？试试 ARC，即 AI2 推理挑战，2018 年。URLhttps://arxiv.org/abs/1803.05457。

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C. 和 Schulman, J. 训练验证器解决数学文字题，2021年。URLhttps://arxiv.org/abs/2110.14168。

Dong, H., Chen, B. 和 Chi, Y. 通过提示引导的自适应结构化剪枝实现高效的大语言模型生成。载于第一届语言建模会议，2024a。URLhttps://openreview.net/forum?id=4aqq9xTtih。Dong, X., Zhang, X., Bu, W., Zhang, D. 和 Cao, F. 大语言模型智能体综述：理论、技术、应用与建议。载于*2024*年第三届人工智能、物联网与云计算技术国际会议（*AIoTC*），第407–413页，2024b。doi: 10.1109/AIoTC63215.2024.10748304。

Gao, L.、Tow, J.、Abbasi, B.、Biderman, S.、Black, S.、DiPofi, A.、Foster, C.、Golding, L.、Hsu, J.、Le Noac'h, A.、Li, H.、McDonell, K.、Muennighoff, N.、Ociepa, C.、Phang, J.、Reynolds, L.、Schoelkopf, H.、Skowron, A.、Sutawika, L.、Tang, E.、Thite, A.、Wang, B.、Wang, K. 和 Zou, A. 语言模型评测工具包，2024 年 07 月。URLhttps://zenodo.org/records/12608602。

He, C., Luo, R., Hu, S., Zhao, Y., Zhou, J., Wu, H., Zha ng, J., Han, X., Liu, Z., andSun, M. Ultraeval：一个用于大语言模型灵活、全面评测的轻量级平台，2024年。

华为技术有限公司 Atlas 200 AI加速模块用户指南。https://support.huawei.com/enterprise/en/doc/EDOC1100223189/968ec576/basic-specifications，a. 昇腾310P3 芯片官方产品页面。

华为技术有限公司 昇腾 CANN。https://www.hiascend.com/zh/cann，b. 昇腾AI芯片系列的 CANN 官方产品页面。

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. Triviaqa：一个用于阅读理解的大规模远程监督挑战数据集，2017年。网址 https://arxiv.org/abs/1705.03551。

Khaki, S., Li, X., Guo, J., Zhu, L., Plataniotis, K. N., Yazdanbakhsh, A., Keutzer, K., Han, S., and Liu, Z. SparseloRA：利用上下文稀疏性加速大语言模型微调。载于第四十二届国际机器学习会议，2025。网址：https://openreview.net/forum?id=z83rodY0Pw。

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. 使用 PagedAttention 实现高效的大语言模型服务内存管理。载于 *ACM SIGOPS* 第二十九届操作系统原理研讨会论文集，2023。

Li, H., Zhang, Y., Koto, F., Yang, Y., Zhao, H., Gong, Y., Duan, N. 和 Baldwin, T. Cmmlu：衡量中文多任务语言理解能力，2024年。网址 https://arxiv.org/abs/2306.09212。

Liao, H., Tu, J., Xia, J. 和 Zhou, X. Davinci：一种可扩展的神经网络计算架构。发表于*2019*年 *IEEE Hot Chips* 第*31*届研讨会（*HCS*），第 1–44 页，2019年。doi：10.1109/HOTCHIPS.2019.8875654。

Liu, J., Ponnusamy, P., Cai, T., Guo, H., Kim, Y. 和 Athiwaratkun, B. 大语言模型中的免训练激活稀疏性，2024年。网址 https://arxiv.org/abs/2408.14690。

Liu, Z., Wang, J., Dao, T., Zhou, T., Yuan, B., Song, Z., Shrivastava, A., Zhang, C., Tian, Y., Re, C. 等。Deja vu：用于高效大语言模型推理的上下文稀疏性。载于国际机器学习会议，第22137–22176页。PMLR，2023年。

Ma, C., Huang, M., Zhang, Y., Wang, C., Wang, Y., Yu, L., Liu, C. 和 Lin, W. 首个激活至关重要：大语言模型动态激活的免训练方法，2024年。网址https://arxiv.org/abs/2408.11393。Mirzadeh, I., Alizadeh, K., Mehta, S., Mundo, C. C. D., Tuzel, O., Samei, G., Rastegari, M. 和 Farajtabar, M. ReLU反击：利用大语言模型中的激活稀疏性，2023年。网址 https://arxiv.org/abs/2310.04564。英伟达公司。*CUDA C*编程指南，12.4版，2024年7月。网址 https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html。CUDA工具包12.4版。Shankar, V. 边缘人工智能：技术、应用与挑战的综合综述。载于*2024*年第*1*届高级计算与新兴技术国际会议（*ACET*），第1–6页，2024年。doi：10.1109/ACET61898.2024.10730112。Shin, J., Yang, H. 和 Yi, Y. Sparseinfer：用于快速大语言模型推理的免训练激活稀疏性预测。载于*2025*年欧洲设计、自动化与测试会议（*DATE*），第1–7页，2025年。doi：10.23919/DATE64628.2025. 10992997。Song, C., Han, X., Zhang, Z., Hu, S., Shi, X., Li, K., Chen, C., Liu, Z., Li, G., Yang, T. 和 Sun, M. ProSparse：在大语言模型中引入并增强内在激活稀疏性。 *arXiv*预印本 *arXiv:2402.13516*，2024a。网址 https://arxiv.org/pdf/2402.13516.pdf。Song, Y., Mi, Z., Xie, H. 和 Chen, H. Powerinfer：使用消费级GPU实现快速大语言模型服务，2024b。网址 https://arxiv.org/abs/2312.12456。

Song, Y., Xie, H., Zhang, Z., Wen, B., Ma, L., Mi, Z., and Chen, H. Turbo sparse：以最少的激活参数实现大语言模型的 SOTA 性能，2024c。网址 https://arxiv.org/abs/2406.05955。Suzgun, M., Scales, N., Sch¨arli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q. V., Chi, E. H., Zhou, D., and Wei, J. 挑战 BIG-bench 任务，以及思维链能否解决这些任务，2022。网址https://arxiv.org/abs/2210.09261。Tillet, P., Kung, H. T., and Cox, D. Triton：一种用于分块神经网络计算的中间语言和编译器。载于第 *3* 届 *ACM SIGPLAN* 机器学习与编程语言国际研讨会论文集，MAPL 2019，第 10–19 页，美国纽约州纽约市，2019年。计算机协会。ISBN 9781450367196。doi：10.1145/3315508.3329973。网址 https://doi.org/10.1145/3315508.3329973。Wang, X., Alam, S., Wan, Z., Shen, H., and Zhang, M. SVD-LLM v2：优化大语言模型压缩中的奇异值截断。载于 Chiruzzo, L., Ritter, A., and Wang, L.（编），*2025*年美洲国家计算语言学协会分会会议：人类语言技术（第 *1* 卷：长篇论文）论文集，第 4287–4296 页，美国新墨西哥州阿尔伯克基，2025年4月。计算语言学协会。ISBN 979-8-89176-189-6。doi：10.18653/v1/2025.naacl-long. 217。网址https://aclanthology.org/2025.naacl-long.217/。Wang, X., Zheng, Y., Wan, Z., and Zhang, M. SVD-LLM：用于大语言模型压缩的截断感知奇异值分解。载于国际学习表征会议（*ICLR*），2025b。网址https://openreview.net/forum?id=LNYIUouhdt。Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jer- nite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Transformers：自然语言处理领域的先进技术。载于*2020*年自然语言处理实证方法会议：系统演示论文集，第 38–45 页，线上，2020年10月。计算语言学协会。网址 https://www.aclweb.org/anthology/2020.emnlp-demos.6。Yang, Y., Wang, Y., Ma, C., Yu, L., Chersoni, E., and Huang, C.-R. 稀疏大脑也是适应性大脑：面向大语言模型的认知负荷感知动态激活，2025。网址 https://arxiv.org/abs/2502.19078。

Yuan, Z., Shang, Y., Song, Y., Yang, D., Wu, Q., Yan, Y., and Sun, G. ASVD：面向大语言模型压缩的激活感知奇异值分解，2025。网址 https://arxiv.org/abs/2312.05821。

Zhang, Z., Song, Y., Yu, G., Han, X., Lin, Y., Xiao, C., Song, C., Liu, Z., Mi, Z., and Sun, M. ReLU2 胜出：为稀疏大语言模型发现高效的激活函数。*arXiv*，abs/2402.03804，2024年。网址：
h
t
tps://api.semanticscholar.org/CorpusID:267499856。

A. 理论分析

A.1. 预测器误差界

令 $m, d > 0$ 和 $0 < r \leq \min(m, d)$。考虑一个权重矩阵 $W \in \mathbb{R}^{m \times d}$ 及其秩为-r 的近似 $W_r \in \mathbb{R}^{m \times d}$（例如，截断奇异值分解），以及偏置向量 $b \in \mathbb{R}^m$。对于任意输入 $x \in \mathbb{R}^d$， $x \neq 0$，我们有：

$$A(x) := \mathrm{ReLU}(Wx), \tag{7}$$
$$g_b(x) := \mathbf{1}[W_r x + b > 0], \tag{8}$$
$$B_b(x) := A(x) \odot g_b(x). \tag{9}$$

全文中，我们使用以下记号：

$$\delta_i \in \mathbb{R}^m, \quad i \in [m], \quad \text{denotes the } i\text{-th standard basis vector } (0, \ldots, \underbrace{1}_{i}, \ldots, 0)^\top,$$

$$t_+ := \max(t, 0), \quad \text{the ReLU function (i.e., } t_+ = 0 \text{ if } t < 0 \text{ and } t_+ = t \text{ if } t \geq 0).$$

我们的目标是推导相对门控误差的确定性界

$$\mathcal{E}_b(x) := \frac{\|A(x) - B_b(x)\|_2}{\|x\|_2}$$

其界由 W_r 的谱误差和偏差 b给出。

误差的精确表示与单调性

定义 A.1（残差算子）。 定义残差矩阵和残差向量

$$\Delta W := W - W_r, \qquad e(x) := \Delta W x.$$

命题 A.2（门控误差的精确表示）。 对于 $x \in \mathbb{R}^d$,

$$A(x) - B_b(x) = \mathrm{ReLU}(Wx) \odot \mathbf{1}[W_r x + b \leq 0], \tag{10}$$

因此

$$\|A(x) - B_b(x)\|_2^2 = \sum_{i=1}^{m} ((Wx)_i)_+^2 \; \mathbf{1}[(W_r x + b)_i \leq 0]. \tag{11}$$

证明。 该结论可由 $B_b(x)$ 的逐分量定义立即得出。 □

命题 A.3（单调 关于偏置的单调性)。 设 $b, b' \in \mathbb{R}^m$，并假设 $b' \leq b$ 逐分量成立。那么对于所有 $x \in \mathbb{R}^d$,

$$\|A(x) - B_{b'}(x)\|_2 \;\geq\; \|A(x) - B_b(x)\|_2 .$$

证明。 若 $b' \leq b$，则 $\mathbf{1}[W_r x + b \leq 0] \;\leq\; \mathbf{1}[W_r x + b' \leq 0]$ 逐分量成立。利用恒等式 (10) 以及$\mathrm{ReLU}(Wx) \geq 0$ 逐分量成立这一事实， ℓ_2 范数只会在我们将 b 替换为较小的偏置 b'时增大，因为这会使掩码激活于更多坐标上。 □

备注 A.4. 使用阈值 $\tau = -b$ 阐明了权衡：增大 τ 会提高稀疏性（更多门控处于关闭状态），但绝不会降低误差；等价地，增大 b会降低稀疏性，且绝不会增加误差。

通过平移残差得到的确定性界

引理 A.5（基于残差的上界）**.** 对于每个 $x \in \mathbb{R}^d$ 和 $b \in \mathbb{R}^m$,

$$\|A(x) - B_b(x)\|_2 \;\le\; \big\|(\Delta W x - b)_+\big\|_2 \;\le\; \|\Delta W x - b\|_2 \,. \tag{12}$$

证明。 设 $z = Wx$、 $z_r = W_r x$ 和 $e := z - z_r = \Delta W x$。定义门控索引集

$$I := \{i \in [m] : (z_r + b)_i \le 0\}.$$

根据定义,

$$\|A(x) - B_b(x)\|_2^2 = \sum_{i \in I} (z_i)_+^2 \,.$$

对于 $i \in I$，我们有 $z_{r,i} \le -b_i$，因此 $z_i = z_{r,i} + e_i \le e_i - b_i$。根据 $t \mapsto (t)_+$ 的单调性，$(z_i)_+ \le (e_i - b_i)_+$。因此

$$\sum_{i \in I} (z_i)_+^2 \le \sum_{i=1}^m (e_i - b_i)_+^2 = \big\|(e-b)_+\big\|_2^2 \,,$$

这就得到了第一个不等式。第二个不等式由 $\big\|(u)_+\big\|_2 \le \|u\|_2$推出。 □

最坏情况界 根据前面的结果，非线性门控误差的分析可化为一个确定性量，它只取决于残差 $\Delta W x$ 和偏置 b。因此，我们可以研究最坏情况下的值

$$\sup_{\|e\|_2 \le R} \big\|(e-b)_+\big\|_2 \,,$$

它刻画了残差能量为 R时的最大门控误差。

均匀偏置

定理 A.6（均匀偏置的最坏情况界）。设 $b = \beta \mathbf{1}$ 满足 $\beta \in \mathbb{R}$。对于所有 $R \ge 0$,

$$\sup_{\|e\|_2 \le R} \big\|(e - \beta\mathbf{1})_+\big\|_2 = \begin{cases} (R-\beta)_+, & \beta \ge 0, \\ R - \beta\sqrt{m}, & \beta < 0. \end{cases}$$

证明。

情况 1: $\beta \ge 0$。记 $y := (e - \beta\mathbf{1})_+ \in \mathbb{R}^m_{\ge 0}$ 以及 $t := \|y\|_2$。

令 $S := \{i : y_i > 0\}$ （若 $t = 0$ 则无需证明，否则 $R > \beta$）。

对于 $i \in S$，我们有 $e_i = \beta + y_i$，因此

$$\|e\|_2^2 \;\ge\; \sum_{i \in S} (\beta + y_i)^2 \;=\; \beta^2 |S| + 2\beta \sum_{i \in S} y_i + \sum_{i \in S} y_i^2.$$

由于 $y \ge 0$，我们有 $\sum_{i \in S} y_i = \|y\|_1 \ge \|y\|_2 = t$，并且还有 $|S| \ge 1$。因此

$$\|e\|_2^2 \;\ge\; \beta^2 + 2\beta t + t^2 \;=\; (t + \beta)^2,$$

所以 $\|e\|_2 \ge t + \beta$。在 $\|e\|_2 \le R$ 下，这得到 $t \le R - \beta$，即$\big\|(e - \beta\mathbf{1})_+\big\|_2 \le (R - \beta)_+$。

取.....时等号成立

$$\begin{aligned} R \le \beta: &\quad e = 0, \\ R > \beta: &\quad e = R\delta_1. \end{aligned}$$

情形 2: $\beta < 0$**.** 将 $\beta = -c$ 写成 $c > 0$。那么对于任意 e,

$$\big\|(e - \beta\mathbf{1})_+\big\|_2 = \big\|(e + c\mathbf{1})_+\big\|_2 \le \|e + c\mathbf{1}\|_2 \le \|e\|_2 + c\,\|\mathbf{1}\|_2 \le R + c\sqrt{m} = R - \beta\sqrt{m}.$$

对于 $e = \frac{R}{\sqrt{m}}\mathbf{1}$，等号成立。 □

一般偏置向量

定理 A.7（一般偏置的最坏情况界）。 设 $b \in \mathbb{R}^m$ 和 $R \geq 0$。则

$$\sup_{\|e\|_2 \leq R} \left\|(e-b)_+\right\|_2 = \begin{cases} \max_{i\in[m]}(R-b_i)_+, & b \geq 0 \ \textit{componentwise}, \\ R + \|b\|_2, & b < 0 \ \textit{componentwise}. \end{cases}$$

证明。

情形 1： $b < 0$。

$$\|(e-b)_+\|_2 \leq \|e-b\|_2 \leq R + \|b\|_2 .$$

紧性： $e = \frac{R}{\|b\|_2}(-b)$恰好给出 $R + \|b\|_2$。

情形 2： $b \geq 0$。对于 $i^\star = \arg\min_i b_i$，令 $b_c = b_{i^\star}\mathbf{1}$。那么，逐分量地有$(e-b)_+ \leq (e-b_c)_+$，因此

$$\|(e-b)_+\|_2 \leq \|(e-b_c)_+\|_2 \overset{(A.6)}{\leq} (R-b_{i^\star})_+ = \max_i (R-b_i)_+.$$

紧性： $e = R\delta_{i^\star}$ 对于 $i^\star = \arg\max_i (R-b_i)_+$ 取等号。 □

注释 A.8. 定理 A.7 指出了两种可以精确刻画最坏情况残差的情形：全负偏置，此时上界随 $\|b_-\|_2$线性增长；以及全非负偏置，此时最优解集中于单个坐标。

推论 A.9(主要确定性界）**.** 设 $b \in \mathbb{R}^m$ 和 $R \geq 0$ 满足 $\|\Delta W x\|_2 \leq R$。则对于所有 $x \in \mathbb{R}^d$ ，

$$\mathcal{E}_b(x) = \frac{\|A(x) - B_b(x)\|_2}{\|x\|_2} \leq \begin{cases} \dfrac{\max_{i\in[m]}(R-b_i)_+}{\|x\|_2}, & b \geq 0 \ \textit{componentwise}, \\ \dfrac{R + \|b\|_2}{\|x\|_2}, & b < 0 \ \textit{componentwise}. \end{cases}$$

推论 A.10 (截断奇异值分解)**.** 根据 *Eckart–Young–Mirsky* 定理，若 W_r 是 W^{gate}的秩-r 截断奇异值分解，则 $\|\Delta W\|_2 = \sigma_{r+1}(W^{\text{gate}}) = \sigma_{r+1}$，因此对于所有 $x \in \mathbb{R}^d$，

$$\mathcal{E}_b(x) \leq \begin{cases} \max\limits_{i\in[m]} \left(\sigma_{r+1} - \dfrac{b_i}{\|x\|_2}\right)_+, & b \geq 0 \ \textit{componentwise}, \\ \sigma_{r+1} + \dfrac{\|b\|_2}{\|x\|_2}, & b < 0 \ \textit{componentwise}. \end{cases}$$

注释 A.11. 尽管由于数据分布的影响，这个界在实际中相当宽松，但它清晰地区分了近似误差 σ_{r+1}与门控偏置 b，揭示了偏置相加对残差的影响。

A.2. 贪心阈值选择

在固定稀疏度预算下进行贪心阈值校准 我们研究一种数据依赖的校准规则，用于在固定的丢弃预算 K （等价于固定稀疏度）下设置分量阈值 $\tau \in \mathbb{R}^m$ 。在全文中，门控为

$$g_\tau(x) = \mathbf{1}[ABx > \tau],$$

其中 AB 表示任意一个秩为r 的近似 $W^{\text{gate}} \in \mathbb{R}^{m\times d}$；我们在实验中采用的具体选择见第 4.1.1节。

代理分数和损害权重 设 $\mathcal{D} = \{x^{(t)}\}_{t=1}^T$为一个数据集，并记作 $[m] = \{1,\ldots,m\}$, $[T] = \{1,\ldots,T\}$。对于每个坐标 $i \in [m]$ 和样本 $t \in [T]$ ，定义代理分数

$$s_i^{(t)} := (ABx^{(t)})_i,$$

并定义边际损害权重

$$w_i^{(t)} := \left|\mathrm{ReLU}\left(W_{i,:}^{\mathrm{gate}} x^{(t)}\right) \cdot W_{i,:}^{\mathrm{up}} x^{(t)}\right|^2 \left\|W_{:,i}^{\mathrm{down}}\right\|_2^2 \qquad (\geq 0). \tag{13}$$

G给定阈值 τ，当 $s_i^{(t)} \leq \tau_i$ is 时，丢弃 $(i,\ t)$ 所产生的经验输出误差

$$\mathcal{L}(\tau) = \sum_{t=1}^{T} \sum_{i=1}^{m} w_i^{(t)}\ \mathbf{1}[s_i^{(t)} \leq \tau_i]. \tag{14}$$

备注 A.12（权重（13））的由来。 考虑标准的门控前馈神经网络分解

$$z(x) = \mathrm{ReLU}(W^{\mathrm{gate}} x) \odot (W^{\mathrm{up}} x), \qquad h(x) = W^{\mathrm{down}} z(x).$$

那么 $h(x)$可分解为秩一项:

$$h(x) = \sum_{i=1}^{m} z_i(x)\, W_{:,i}^{\mathrm{down}}, \quad z_i(x) = \mathrm{ReLU}(W_{i,:}^{\mathrm{gate}} x) \cdot W_{i,:}^{\mathrm{up}} x.$$

丢弃第 i 个隐层坐标会恰好移除贡献 $v_i(x) = z_i(x) W_{:,i}^{\mathrm{down}}$，因此

$$\|v_i(x)\|_2^2 = \left|\mathrm{ReLU}\left(W_{i,:}^{\mathrm{gate}} x\right) \cdot W_{i,:}^{\mathrm{up}} x\right|^2 \left\|W_{:,i}^{\mathrm{down}}\right\|_2^2,$$

这与 13）在 $x = x^{(t)}$ 的情形相符。

前缀形式与固定丢弃预算 固定 $i \in [m]$。按代理分数非递减的顺序对样本排序:

$$s_{i,(1)} \leq s_{i,(2)} \leq \cdots \leq s_{i,(T)},$$

并令$w_{i,\ (j)}$ 表示相应的置换后权重。定义

$$C_i(k) := \sum_{j=1}^{k} w_{i,(j)}, \quad C_i(0) = 0, \qquad \Delta_i(k) := C_i(k) - C_i(k-1) = w_{i,(k)}.$$

对于每个坐标 i，令 τ_i，使得恰有 k_i 个样本满足 $s_i^{(t)} \leq \tau_i$，其代价为 C_i（k_i）。因此，总损失为

$$\mathcal{L}(\tau) = \sum_{i=1}^{m} C_i(k_i), \qquad k_i := \#\{t :\ s_i^{(t)} \leq \tau_i\} \in \{0, \ldots, T\}.$$

在丢弃预算 $K \in \{0, 1, \ldots, mT\}$固定时，阈值选择可简化为

$$\min_{\substack{k_i \in \{0,\ldots,T\} \\ \sum_{i=1}^{m} k_i = K}} \sum_{i=1}^{m} C_i(k_i). \tag{15}$$

贪心最优性

定义 A.13 (全局最优性)。 一个可行解 $k^\star = (k_1^\star,\ \ldots,\ k_m^\star)$ 满足 $\sum_i k_i^\star = K$时，称其 为全局最优，对于 (15) 当且仅当

$$\sum_{i=1}^{m} C_i(k_i^\star) \leq \sum_{i=1}^{m} C_i(k_i) \quad \text{for all feasible } k = (k_1, \ldots, k_m).$$

假设 A.14 （每个坐标内的单调边际增量）。 对于每个 $i \in [m]$，边际增量均非递减:

$$\Delta_i(1) \leq \Delta_i(2) \leq \cdots \leq \Delta_i(T).$$

定理 A.15(贪心 贪心算法是全局最优的）。 对所有 i，初始化 $k_i = 0$。对于 $\ell = 1, 2, \ldots, K$（丢弃预算），选择

$$i_\ell \in \arg\min_{i:\, k_i < T} \Delta_i(k_i + 1), \qquad \textit{and update} \qquad k_{i_\ell} \leftarrow k_{i_\ell} + 1.$$

在定理 *A.14* 下，所得的 k 对于 (15)是全局最优的。

此外，它可以通过以下阈值 τ 实现：

$$\tau_i \in \big[\, s_{i,(k_i)},\ s_{i,(k_i+1)} \,\big),$$

约定 $s_{i,\ (0)} : = -\infty$ 以及 $s_{i,\ (T+1)} : = +\infty$。

证明。该问题等价于在满足前缀约束的同时，恰好选择 K 个增量 $\Delta_i(j)$：选择$\Delta_i(j)$ 要求选取所有 $\Delta_i(1) \ldots, \Delta_i(j-1)$。

贪心算法在每一步都选择当前可用的最小“下一个增量” $\Delta_i(k_i + 1)$。令 G 为贪心算法选出的 K 增量序列，令 O 为任意一个最优序列。

假设 $G \neq O$。令 $a = \Delta_p(k_p + 1)$为贪心算法所选增量中第一个（按时间顺序）不属于 O的增量。由于 $|G| = |O| = K$，存在 $b = \Delta_q(j) \in O \setminus G$。

选择 b 作为坐标 q 在 O 中的最后一个增量（最大 j）。移除 b 可保持 q的前缀约束。添加 a可扩展 p的前缀，同时保持可行性。

根据贪心算法的选择，$a \leq \Delta_q(k_q + 1)$ 其中 k_q 统计截至该点贪心算法在 q 中的选择次数。由于 $b \notin G$ 但 O 包含该增量，$j \geq k_q + 1$。根据定理A.14 （非递减增量），$\Delta_q(k_q + 1) \leq \Delta_q(j) = b$。

因此 $a \leq b$，所以交换 $b \leftrightarrow a$ 能保持可行性且不会增加成本。重复进行此类交换，将 O 转换为 G，从而证明其最优性。

阈值实现可由 k_i 的定义得出。 □

备注 A.16（秩-r 近似与经验验证的作用）。 定理 A.15 要求按代理分数排序后，边际增量具有单调性（定理 A.14）。这并无保证，因为代理分数 $s_i^{(t)} = (ABx^{(t)})_i$仅取决于秩-$r$近似 $AB \approx W_{\text{门控}}$，而上文定义的损害还涉及 W^{up}和 $W_{\text{向下}}$。

如果 AB 在输入数据分布上能够很好地近似 $W_{\text{门控}}$，那么 $(ABx)_i \approx (W^{\text{gate}}x)_i$，因此按代理分数排序应与真实损害顺序一致。实践中，我们对不重叠的 η 步组和进行贪心选择，以减轻数据噪声：

$$S_i^{(m)} := \sum_{j=(m-1)\eta+1}^{m\eta} w_{i,(j)}, \quad m = 1, \ldots, \lfloor T/\eta \rfloor.$$

我们使用组和与其位置索引之间的归一化肯德尔 tau 距离来验证其单调性：

$$\tau_i^{\mathrm{K}} := 1 - \frac{K_d\big(\{S_i^{(m)}\}_{m=1}^{M},\ \{m\}_{m=1}^{M}\big)}{M(M-1)},$$

其中 $K_d(\tau_1, \tau_2) = \#\big\{(m_1, m_2): \quad m_1 < m_2, \quad [\tau_1(m_1) \lesseqgtr \tau_1(m_2) \wedge \tau_2(m_1) \gtreqless \tau_2(m_2)]\big\}$ 计数不一致对 .

使用 Kendall tau 距离的理由在于，它直接量化了定理 A.14 所要求的序数一致性：对于所有 $m_1 < m_2$，我们期望 $S^{(m_1)} \leq S^{(m_2)}$。形式上，τ_i^{K}衡量一致对的比例，同时适当考虑并列——这正是定理 A.15 中贪心最优性的精确条件。在用于校准偏置的样本数据上，图 6 显示 τ_i^{K} 在 $[0.9, 1]$ 中保持稳定，证实了代理排序 $ABx \approx W_{\text{门控}}x$ 以及我们采用贪心方法的适用性。

注 A.17（背包问题重构与实际最优性）。问题 (15) 是一个 0–1 背包问题，包含 mT 物品$\{\Delta_i(j)\}_{i=1}^{m}{}_{j=1}^{T}$，容量为 K，并带有前缀约束：选择 $\Delta_i(j)$ 必须先选择 $\Delta_i(1) \ldots, \Delta_i(j-1)$。标准动态规划需要 $O(mTK)$ 时间，而我们的贪心算法通过优先队列在 $O((K+m)\log m)$ 时间内运行，并根据定理 A.14 达到精确最优，因此效率更高。

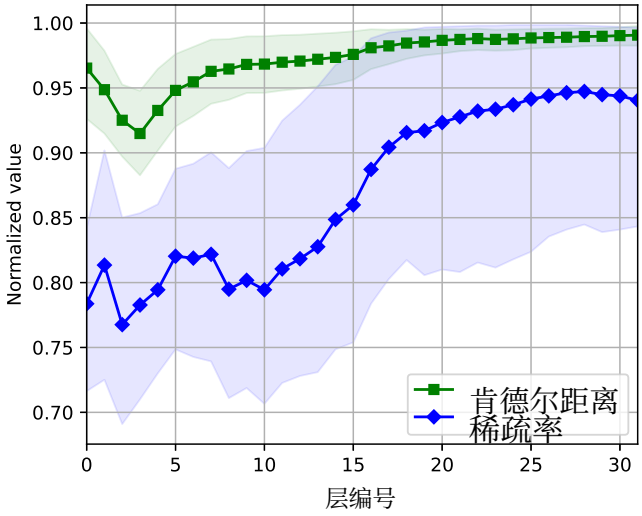


图 6. ProSparse-LLaMA2-7B 中按代理排序的 η 步求和在神经元间的平均肯德尔 τ_i^{K}。横轴表示层编号。为清晰起见，图中展示了逐层稀疏率。阴影区域表示第 10 至第 90 百分位区间，展示了数据分布的中间 80%。

算法 3 CUDA 上的稀疏前馈网络前向传播（解码阶段）

输入： $W_{\text{门控}}W^{\text{up}} \in \mathbb{R}^{D\times d}$，$W_{\text{下降}} \in \mathbb{R}^{d\times D}$，$x \in \mathbb{R}^d$，$A \in \mathbb{R}^{D\times r}$，$B \in \mathbb{R}^{r\times d}$，$b \in \mathbb{R}^D$**输出：** $x \in \mathbb{R}^d m \leftarrow ABx + b$

（预测稀疏模式）

$m \leftarrow$ SPARSEGATEKERNEL(W^{gate}，x，m)$x \leftarrow$
SPARSEUPANDMULKERNEL(W^{up}，x，m)$x \leftarrow$
SPARSEDOWNKERNEL(W^{down}，x)**返回** x

B. 稀疏前馈网络推理实现

与 Deja Vu 的（Liu 等人，2023）异步流水线不同，该流水线利用 Ampere 的 `cuda::memcpy async` 操作（英伟达公司，2024年），我们的方法面向计算能力为 7.0 的消费级 GPU，以实现更广泛的适用性。我们在每个前馈网络的开头插入预测器，并依次为各自的门控投影生成稀疏模式。这种设计确保与现有框架集成时开销极低。

用于稀疏前馈网络的自定义 CUDA 内核我们实现了三个 CUDA 内核（算法 3）。预测器的输出$m \in \mathbb{R}^D$ 标记出 W^{gate}中的活跃行。在门控投影和升维投影过程中，每个 CUDA 块处理一个神经元，并首先通过 m验证其是否活跃。门控输出会存回 m 以供重用。关键优化如下：

- **内存**：为实现合并内存访问，$W_{\text{下投影}}$采用列主序存储。
- **精度**：寄存器操作采用全精度；内存读写采用半精度。
- **确定性**：在下投影中使用共享内存部分和，避免原子加法。

我们的内核设计不同于现有的 ProSparse（Song 等人，2024a）和 Deja Vu（Liu 等人，2023）实现。这两种先前的方法都在下投影阶段使用原子加法操作，导致浮点累加过程中出现非确定性和潜在的精度不稳定。为解决这些局限性，我们的算法采用通过共享内存实现的部分和归约。该技术强制采用预定义的求和顺序，并使用分层归约策略。因此，我们的方法既能保证输出具有确定性，又能通过减少计算误差传播来提高数值稳定性。鉴于内核融合的收益递减，我们暂不进行融合，优先考虑可维护性。

算法4 CANN 上的稀疏前馈网络前向传播（解码阶段）

输入： $W_{\text{门控}} W^{\text{up}} \in \mathbb{R}^{D\times d}, W_{\text{下投影}} \in \mathbb{R}^{d\times D}, x \in \mathbb{R}^d, A \in \mathbb{R}^{D\times r}, B \in \mathbb{R}^{r\times d}, b \in \mathbb{R}^D$
输出： $x \in \mathbb{R}^d$
$m \leftarrow ABx + b > 0$ （预测稀疏模式）
$p \leftarrow$ 非零(m) （获取索引）
$g \leftarrow$ 选择行$(W^{\text{gate}}, \ p)$ （选择 W^{gate} 中由 p 指定的行）
$m_1 \leftarrow$矩阵乘法$(g, \ x)$
$u \leftarrow$ 选择行$(W^{\text{up}}, \ p)$ （选择 W^{up} 中由 p 指定的行）
$m_2 \leftarrow$矩阵乘法(u, x)
$m_3 \leftarrow \text{RELU}(m_1) \cdot m_2$
$d \leftarrow$ 选择列(W^{down}, p) （选择由 p 索引的 W^{down} 的列）
$x \leftarrow$ 矩阵乘法(d, m_3)
返回 x

稀疏前馈神经网络的神经网络处理器解决方案 我们采用高级 PyTorch 指令实现稀疏前馈神经网络的神经网络处理器解决方案，详见算法 4，并利用预测器的输出 $m \in \mathbb{R}^D$ 检索对应的行或列，以构建更小的稠密子矩阵。这种方法利用 CANN 神经网络处理器架构（Liao等人，2019年）中专用的矩阵乘法加速单元（cube），从而实现高效执行。 W^{down} 矩阵以列主序格式存储。

C. 扩展至非ReLU激活函数

该方案原则上可以应用于采用非ReLU激活函数（如SiLU）的大语言模型。然而，直接使用该方案可能会导致长文本生成任务中的准确率显著下降。采用SiLU激活函数的Qwen2-7B-Instruct模型的质量评估结果列于表5中。对当前方案的潜在改进可能包括对中间激活值进行变换，以减轻由此产生的误差。

表*5*。Qwen2-7B-Instruct准确率结果，平均预测稀疏度为25%

方法	TriviaQA 准确率，%	ARC-C 准确率，%	ARC-E 准确率，%	HumanEval pass@1，%	MBPP pass@1，%	GSM8K 准确率，%	BBH 准确率，%	CMMLU 准确率，%	平均值
稠密模式	60.96	86.43	93.81	60.98	44.46	36.85	47.81	79.58	63.86
SVDP	51.48	86.43	93.81	49.39	36.55	37.83	47.61	79.58	60.34

D. 详细准确率结果

表 6、表 7和表 8 提供了配备我们 SVD预测器的三种稀疏大语言模型的详细评估结果。表 9、表 11和表 10 则对三种稀疏大语言模型的 SVD稀疏度估计器进行了详细的消融研究。

表*6*。 ProSparse-LLaMA2-7B 评估结果， $r = 256$

方法	s	TriviaQA 准确率，%	ARC-E 准确率，%	ARC-C 准确率，%	HumanEval pass@1，%	MBPP pass@1，%	GSM8K 准确率，%	BBH 准确率，%	CMMLU 准确率，%	平均值 得分	E2E 加速比
稠密模式	-	62.94	75.42	53.58	17.68	22.59	11.45	35.85	31.84	38.92	1.00×
SVDP	0.4	62.38	75.29	53.67	17.68	22.38	10.61	35.46	31.81	38.66	1.61×
	0.5	62.74	75.34	53.67	17.68	22.07	11.22	35.58	31.94	38.78	1.64×
	0.6	62.90	75.29	53.75	18.29	21.97	10.54	35.54	31.91	38.77	1.67×
	0.7	62.04	75.42	53.58	17.07	21.87	11.22	35.43	31.90	38.57	1.71×

表7。 TurboSparse-Mistral-Instruct 评估结果， $r = 352$

方法	s	TriviaQA 准确率，%	ARC-E 准确率，%	ARC-C 准确率，%	HumanEval pass@1，%	MBPP pass@1，%	GSM8K 准确率，%	BBH 准确率，%	CMMLU 准确率，%	平均值 得分	E2E 加速比
稠密模式	-	48.58	87.21	71.67	34.15	42.20	61.71	42.35	46.52	54.30	1.00×
SVDP	0.4	45.46	87.08	71.67	32.93	42.09	61.49	42.46	46.55	53.72	1.67×
	0.5	43.26	87.04	71.76	33.54	42.71	61.41	42.32	46.47	53.56	1.73×
	0.6	40.63	87.08	72.10	33.54	42.09	61.41	42.19	46.43	53.18	1.79×
	0.7	39.21	87.21	72.10	37.20	42.09	60.96	42.33	46.45	53.44	1.84×

表8。 SparseQwen2-7B评估结果， $r = 512$

方法	s	TriviaQA 准确率，%	ARC-E 准确率，%	ARC-C 准确率，%	HumanEval pass@1，%	MBPP pass@1，%	GSM8K 准确率，%	BBH 准确率，%	CMMLU 准确率，%	平均值 得分	E2E 加速比
稠密模式	-	50.76	91.92	83.87	64.02	46.3	77.1	48.42	64.35	65.84	1.00×
SVDP	0.4	50.87	91.92	83.87	62.80	44.56	76.50	48.40	64.35	65.41	1.78×
	0.5	49.96	91.92	83.87	62.80	43.53	76.80	48.40	64.35	65.20	1.81×
	0.6	48.82	91.92	83.87	63.41	43.53	76.80	48.34	64.35	65.13	1.85×
	0.7	47.42	91.92	83.87	62.20	41.99	75.82	48.36	64.35	64.49	1.86×

E. 技术细节

E.1. 大语言模型的超参数值

我们在表 12中提供了所测试模型的超参数值，因为这些参数会影响端到端延迟。具体而言，所有评估的优化技术仅修改推理中的前馈网络部分，而保持注意力模块不变。因此，更大的中间维度会增加前馈网络对端到端延迟的贡献。同样，使用分组查询注意力会减少注意力对总推理时延的贡献。

E.2. 预测器构建

校准总共使用来自 GSM8K、ARC-E、ARC-C、OpenBookQA、QAsper 和 CodeXGLUE 数据集训练集划分的 112 个少样本样本（≈20000词元）。

E.3. 预测器秩

可以在预测稀疏率和预测器尺寸之间进行权衡。较大的预测器能够在门控投影中实现更多稀疏计算，但预测所需的计算量也更大。相应地， s 超参数可用于校准预测稀疏率。图7 展示了奇异值分解预测器的分离能力如何取决于其尺寸。基于这一发现，使用相对较小的预测器也能实现更高的加速。此外，较小的预测器占用的额外内存更少。与 r类似，我们将 s 设置为在所有层中保持不变。根据经验，我们发现，将秩设置为中间维度的约 2% D 效果良好。

表*9.* ProSparseLLAMA2-7B 的 SVD稀疏度估计器消融研究。

	TriviaQA	ARC-E	ARC-C	HumanEval	MBPP	GSM8K	BBH	CMMLU	平均值
稠密基线	62.94	75.42	53.58	17.68	22.59	11.45	35.85	31.84	38.92
并行流水线									
朴素奇异值分解	58.19	69.91	49.32	10.98	13.35	6.60	33.98	30.28	34.08
+数据白化	53.85	75.34	53.75	16.46	19.92	10.16	35.46	31.83	37.10
+ 偏置校准	62.66	75.38	53.58	17.68	22.18	10.54	35.86	31.85	38.72
顺序流水线									
朴素奇异值分解	34.53	3.58	4.35	0.61	0.21	1.90	11.12	21.23	9.69
+数据白化	27.58	75.21	53.33	8.54	11.40	6.90	33.71	31.34	31.00
+ 偏置校准	62.74	75.34	53.67	17.68	22.07	11.22	35.58	31.94	38.78

表*10.* SparseQwen2-7B 的 SVD稀疏度估计器消融研究。

	TriviaQA	ARC-E	ARC-C	HumanEval	MBPP	GSM8K	BBH	CMMLU	平均值
稠密基线	50.76	91.92	83.87	64.02	46.30	77.10	48.42	64.35	65.84
并行流水线									
朴素奇异值分解	46.24	91.92	83.87	53.66	34.39	74.00	48.00	64.35	62.05
+数据白化	42.92	91.92	83.87	61.59	43.02	75.89	48.37	64.35	63.99
+ 偏置校准	50.20	91.92	83.87	62.20	45.28	76.65	48.45	64.35	65.37
顺序流水线									
朴素奇异值分解	40.03	91.92	83.87	12.20	9.03	63.00	48.43	64.35	51.60
+数据白化	30.00	91.92	83.87	57.32	37.78	72.93	48.22	64.35	60.80
+ 偏置校准	49.96	91.92	83.87	62.80	43.63	76.80	48.40	64.35	65.22

表*11*。 TurboSparse-Mistral-Instruct 的 SVD稀疏度估计器消融研究。

	TriviaQA	ARC-E	ARC-C	HumanEval	MBPP	GSM8K	BBH	CMMLU	平均值
稠密基线	48.58	87.21	71.67	34.15	42.20	61.71	42.35	46.52	54.30
并行流水线									
朴素奇异值分解	41.40	87.04	71.76	24.39	38.91	60.20	42.13	46.34	51.52
+数据白化	37.98	86.57	71.25	37.20	41.79	60.88	42.27	46.18	53.02
+ 偏置校准	46.21	87.12	71.76	36.59	41.89	62.24	42.44	46.45	54.34
顺序流水线									
朴素奇异值分解	35.29	71.09	57.85	1.83	8.73	9.17	37.72	25.24	30.87
+数据白化	18.09	87.12	72.27	29.27	28.44	48.60	41.87	46.22	46.49
+ 偏置校准	43.26	87.04	71.76	33.54	42.71	61.41	42.32	46.47	53.56

表 *12.* 大语言模型超参数值。

	超参数			
模型	隐藏维度	中间维度	n 注意力头	n KV头
ProSparse-LLaMA2-7B	4096	11008	32	32
TurboSparse-Mistral-Instruct	4096	14336	32	8
SparseQwen2-7B	3584	18944	28	4

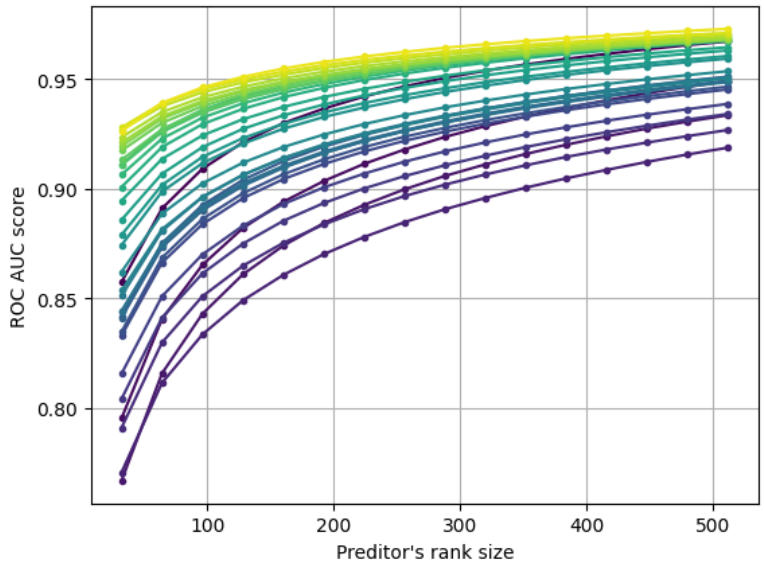


图 7. ProSparse-LLaMA2-7B 在多个序列样本上取平均后，不同层的 SVD预测器 ROC AUC分数。颜色越亮的曲线代表越深的层。对于大多数层，质量指标在相对较小的秩处达到平台期。