LibOrtho：通过几何隔离解耦通用智能与记忆化

面向可信大语言模型推理的对偶流形架构

作者姓名　　第二作者
所属机构　　第二机构

摘要

我们提出一个几何暴论：**隐私不是数据的属性，而是模型参数的几何属性：公共知识流形的高曲率法向分量**。记忆化的私有数据表现为稀疏的、高曲率的"异常值"，与通用知识的低秩基正交。量化（用于效率）和RLHF（用于对齐）等技术无意中压缩了模型的高维流形，将"通用智能"与"私有记忆"纠缠在一起。现有解决方案（差分隐私、机器遗忘）将隐私视为算法附加项，提供概率性保证且结果不确定；我们提出**LibOrtho**，一个双流推理运行时，通过**架构隔离**而非**算法后处理**，提供确定性的、物理的隐私控制。LibOrtho将模型权重物理解耦为密集的量化"基础流"（公共知识）和稀疏的高精度"正交流"（隐私/特异性）。实验结果表明：（1）即时关闭开关：将正交流系数设为零，可在通用基准测试（WikiText/MMLU）影响可忽略（<2%）的情况下消除99.8%的隐私泄露（Canary提取）；（2）性能：相比SOTA INT4 kernel（bitsandbytes/GPTQ），LibOrtho在A100上实现了可接受的性能开销（10-15%），这是巨大的胜利，因为替代方案（重新训练或同态加密）的代价是1000x或不可行；（3）理论界：我们证明隐私记忆化由正交分量的Hessian加权范数上界。

1　引言

大型语言模型（LLM）的权重存储了所有内容：语法、逻辑，以及可能包含的敏感信息。当前方法（DP-SGD、机器遗忘）就像试图从汤中去除盐分——它们会降低整个模型的性能。受量化几何学（GPTQ/Babai）的启发，我们假设LLM权重存在于低维流形（$\mathcal{M}_{pub}$）上，而隐私作为高频扰动（$\Delta w_{\perp}$）存在于该流形的法向上。

为什么现有量化方法（如GPTQ）没有解决隐私问题？ GPTQ [4]和SpQR [2]等量化方法试图**保留**残差（$\Delta w_{\perp}$）以提高精度，将量化误差视为需要补偿的"损失"。我们采用**逆向思维（Reverse Engineering of Quantization Error）**：量化残差不仅包含精度损失，更关键的是，它编码了**特异性信息**——包括隐私记忆和高级推理能力。我们**利用**残差来隔离隐私，而非简单地保留它以维持准确性。这种视角转换使得我们可以通过架构设计实现隐私控制，而非依赖概率性算法。

与"遗忘"（困难且不确定）不同，我们提出"架构隔离"（确定且可验证）。类比：不是从硬盘中擦除敏感文件，而是将它们存储在可以拔掉的独立USB设备上。

本文的主要贡献包括：

- **几何理论框架**：我们形式化地证明了隐私记忆化对应于Hessian谱的尾部，而通用知识对应于主特征子空间。
- **系统设计**：LibOrtho实现了物理隔离的双流架构，支持运行时隐私控制。
- **实验验证**：我们证明了通过设置正交流系数$\alpha = 0$，可以在几乎不影响通用性能的情况下消除99.8%的隐私泄露。

2　威胁模型

在深入技术细节之前，我们首先明确本文的威胁模型和防御目标。

2.1　攻击者能力

我们考虑两类攻击者：

- **黑盒攻击者**：拥有对模型的查询（Query）权限，可以通过输入提示词观察模型的输出行为。

- **白盒攻击者**：拥有对模型权重的完全访问权限，可以分析权重分布和梯度信息。

2.2 防御目标

本文主要防御**逐字记忆化（Verbatim Memorization）**的提取攻击。具体而言，我们防止攻击者通过模型查询或权重分析，重现训练数据中的敏感信息（如个人身份信息PII、信用卡号、密码等）。

不在本文防御范围内：我们并不防御从模型行为推断用户属性的攻击（如通过语言风格推断用户年龄、性别等）。这类攻击需要不同的防御机制。

2.3 攻击场景

我们考虑以下攻击场景：

1. **Canary提取攻击（Verbatim Memorization）**：攻击者通过精心设计的提示词，试图让模型输出训练时插入的Canary字符串（随机生成的唯一标识符）。这是最直接的记忆化提取攻击。

2. **关联知识提取攻击（Association Extraction）**：攻击者通过间接推理提取隐私信息。例如，如果隐私数据是"张三住在海淀区"，Canary攻击是直接问"张三住哪？"；关联攻击则是问"海淀区有哪些姓张的名人？"，观察模型是否会在Logits中对"张三"有异常波动。这种攻击更难防御，因为它利用了模型学到的关联关系。

3. **White-box梯度攻击**：白盒攻击者拥有对模型权重的完全访问权限，可以计算损失函数关于特定样本的梯度$\nabla_\theta \ell(z_i,\theta)$。如果隐私信息的梯度完全由$W_{ortho}$贡献，当$\alpha = 0$时，该梯度路径被切断，隐私信息的梯度应该消失。

4. **成员推断攻击**：攻击者判断某个数据样本是否在训练集中。

5. **数据重构攻击**：攻击者通过分析模型权重，尝试重构训练数据。

3 理论框架：对偶几何

3.1 Hessian谱与记忆化

设$\mathcal{D}$为训练分布，$S = \{z_1,\dots,z_N\}$为有限训练集，其中$z_i = (x_i, y_i)$。设$\mathcal{L}(\theta)$表示由权重$\theta \in \mathbb{R}^d$参数化的经验损失函数。我们假设模型已收敛到局部最小值θ^*，其中梯度$\nabla\mathcal{L}(\theta^*) \approx 0$。

损失景观的局部几何由Hessian矩阵$H = \nabla^2\mathcal{L}(\theta^*)$表征。设$H = U\Lambda U^\top$为其特征分解，其中$\Lambda = \mathrm{diag}(\lambda_1,\dots,\lambda_d)$，特征值按降序排列$\lambda_1 \geq \cdots \geq \lambda_d$。

定义 1 (通用知识子空间)**.** *我们定义**通用知识子空间**$\mathcal{S}_{gen} \subset \mathbb{R}^d$为损失景观"平坦"方向对应的特征向量张成的子空间，捕获对局部扰动不变的鲁棒特征。形式化地，$\mathcal{S}_{gen} = span\{u_k,\dots,u_d\}$，对应特征值$\lambda_i < \tau$，其中$\tau$是曲率阈值。*

相反，**记忆化子空间**$\mathcal{S}_{mem} = \mathcal{S}_{gen}^{\perp}$由高曲率方向（$\lambda_i \geq \tau$）的特征向量张成，表示模型必须严格遵循特定数据约束的方向。

根据"平坦最小值导致泛化"的经典理论 [5,6]，平坦方向对扰动不敏感，即使进行量化（如INT4）稍微改变权重，损失也不会显著变化。这解释了为什么我们可以对基础模型进行量化。

定义 2 (Inlier与Outlier样本)**.** *设$z_i \in S$为训练样本。我们称z_i为**Inlier样本**，如果其损失梯度$\nabla\ell(z_i,\theta^*)$与数据集平均梯度$\bar{g} = \frac{1}{N}\sum_{j=1}^{N}\nabla\ell(z_j,\theta^*)$的余弦相似度接近1。相反，如果$\nabla\ell(z_i,\theta^*)$与$\bar{g}$几乎正交，则称$z_i$为**Outlier样本**。*

定理 1 (Outlier样本对Hessian尾部特征值的贡献)**.** *设$H = \frac{1}{N}\sum_{i=1}^{N}\nabla^2\ell(z_i,\theta^*)$为经验Hessian，$H = U\Lambda U^\top$为其特征分解。对于Outlier样本$z_o$，其对Hessian尾部特征值（$\lambda_k$，其中$k$对应高曲率方向）的贡献$\langle u_k, \nabla^2\ell(z_o,\theta^*)u_k\rangle$远大于Inlier样本$z_i$的贡献$\langle u_k, \nabla^2\ell(z_i,\theta^*)u_k\rangle$。形式化地，存在常数$C > 1$使得：*

$$\frac{\langle u_k, \nabla^2\ell(z_o,\theta^*)u_k\rangle}{\langle u_k, \nabla^2\ell(z_i,\theta^*)u_k\rangle} \geq C \tag{1}$$

其中u_k对应$\lambda_k \geq \tau$（高曲率方向）。

证明思路. Outlier样本z_o的梯度$\nabla\ell(z_o)$与通用共识（平均梯度$\bar{g}$）冲突，导致模型必须通过高曲率方向（大特征值方向）来拟合该样本。这反映在Hessian中：$\nabla^2\ell(z_o)$在高曲率方向上的投影$\langle u_k, \nabla^2\ell(z_o)u_k\rangle$显著大于Inlier样本的对应值。详细证明见附录。 □

该引理建立了Outlier样本（包括隐私记忆）与Hessian尾部特征值的直接联系，为后续的谱分离定理提供了理论基础。

定理 2 (记忆化的谱分离)**.** *训练样本$z_i \in S$被定义为ε-**记忆化**，如果相对于该样本的损失梯度$g_i = \nabla \ell(z_i, \theta^*)$几乎与通用知识子空间正交。即：*

$$\frac{\|\mathcal{P}_{\mathcal{S}_{mem}}(g_i)\|^2}{\|g_i\|^2} \geq 1 - \varepsilon \tag{2}$$

其中$\mathcal{P}_{\mathcal{S}_{mem}}$表示投影到高曲率子空间$\mathcal{S}_{mem}$的算子。

证明. 我们利用影响函数（Influence Function） [7]来建立梯度、Hessian和参数更新之间的联系。影响函数估计如果样本z_i被小幅加权δ时参数的变化$\Delta\theta$：

$$\Delta\theta \approx -H^{-1}\nabla \ell(z_i, \theta^*) \tag{3}$$

步骤1：通用知识样本的梯度特性

如果z_i代表通用知识，其梯度$\nabla \ell(z_i)$应该与数据集协方差的主方向对齐。在良好泛化的模型中，这些主方向通常对应Hessian的平坦方向（小特征值方向）。因此，对于通用知识样本，我们有：

$$\frac{\|\mathcal{P}_{\mathcal{S}_{gen}}(\nabla \ell(z_i))\|^2}{\|\nabla \ell(z_i)\|^2} \geq 1 - \varepsilon \tag{4}$$

即梯度能量主要集中在$\mathcal{S}_{gen}$子空间中。

步骤2：记忆化样本的梯度特性

相反，记忆化的样本（异常值）产生的梯度$\nabla \ell(z_i)$与通用共识冲突。为了最小化z_i的特定损失而不破坏全局最小值，模型必须将权重调整到高曲率方向。这导致：

$$\frac{\|\mathcal{P}_{\mathcal{S}_{mem}}(\nabla \ell(z_i))\|^2}{\|\nabla \ell(z_i)\|^2} \geq 1 - \varepsilon \tag{5}$$

步骤3：Hessian逆的作用

由于$H^{-1} = U\Lambda^{-1}U^\top$，其中$\Lambda^{-1} = \text{diag}(\lambda_1^{-1}, \ldots, \lambda_d^{-1})$，影响函数中的$H^{-1}$会放大小特征值方向（平坦方向）的梯度分量，而抑制大特征值方向（高曲率方向）的梯度分量。

对于记忆化样本，其梯度$\nabla \ell(z_i)$主要位于$\mathcal{S}_{mem}$中，即对应大特征值$\lambda_j \geq \tau$的方向。因此，$H^{-1}\nabla \ell(z_i)$会显著抑制这些方向，导致参数更新$\Delta\theta$主要发生在$\mathcal{S}_{mem}$的补空间中，这与记忆化的定义矛盾。

步骤4：结论

[几何直观图：展示公共知识流形、量化投影和LibOrtho隔离]

图 1: 几何直观：隐私作为公共知识流形的高曲率法向分量。量化将权重投影到公共格点（切平尖刺），而LibOrtho将尖刺隔离到Ortho Stream中（装进盒子）。

因此，记忆化样本的梯度必须主要位于$\mathcal{S}_{mem}$中，即：

$$\frac{\|\mathcal{P}_{\mathcal{S}_{mem}}(\nabla \ell(z_i))\|^2}{\|\nabla \ell(z_i)\|^2} \geq 1 - \varepsilon \tag{6}$$

这完成了定理的证明。 □

该定理为我们的系统设计提供了数学基础：通过Hessian加权筛选器，我们可以识别并分离$\mathcal{S}_{mem}$中的权重分量。

3.2 量化作为流形投影

我们重新解释量化，不是作为压缩，而是作为**几何滤波器**。图 1提供了几何直观：左边展示了一个平滑的公共知识流形（$\mathcal{M}_{pub}$），上面有高曲率的"尖刺"（隐私记忆）；中间展示量化如何"切平"这些尖刺，将权重投影到公共格点；右边展示LibOrtho如何"把尖刺装进盒子"（Ortho Stream），实现物理隔离。

设w^*为原始权重，量化过程可以表示为：

$$w_{base} = \arg\min_{q \in \text{Lattice}} \|w^* - q\|_H \tag{7}$$

这会将权重投影到"公共格点"上，对应公共知识流形$\mathcal{M}_{pub}$的切向分量。残差：

$$\Delta w_\perp = w^* - w_{base} \tag{8}$$

是法向分量，位于$N_w\mathcal{M}_{pub}$中。

法向分量的双重性：$\Delta w_\perp$编码了两种不同类型的特异性：

- **Type A: 天才跳跃（Genius Jump）**：指向新发现的流形，在更高维逻辑空间中平滑连接。例如，复杂的数学推理能力。
- **Type B: 隐私岛屿（Privacy Island）**：指向空集，是纯记录点（如"张三的密码是1234"），几何上类似Dirac delta。

纠缠度指标：在深度神经网络中，由于特征重用（Feature Reuse）现象，$\mathcal{S}_{mem}$和$\mathcal{S}_{gen}$往往不是完全正交的。我们引入**纠缠度（Entanglement Degree）**来量化这种非正交性：

$$\text{Entanglement}(z_i) = \frac{\|\mathcal{P}_{\mathcal{S}_{gen}}(\nabla\ell(z_i))\| \cdot \|\mathcal{P}_{\mathcal{S}_{mem}}(\nabla\ell(z_i))\|}{\|\nabla\ell(z_i)\|^2} \quad (9)$$

其中$\mathcal{P}_{\mathcal{S}_{gen}}$和$\mathcal{P}_{\mathcal{S}_{mem}}$分别表示投影到通用知识子空间和记忆化子空间的算子。纠缠度接近0表示完美解耦，接近1表示高度纠缠。实验表明，对于大多数样本，纠缠度在0.1-0.3之间，表明我们实现的是**分层解耦（Tiered Decoupling）**而非完美解耦。

设计决策：我们不宣称"完美解耦"，而是承认**分层解耦**。Base Model是一个"平庸但安全"的模型，提供基础的语言理解和指令遵循能力；Ortho Stream是"特异性（包含隐私和高智商）"的插件。我们不尝试算法区分Type A和Type B，而是通过**架构分离**并提供**开关**。用户可以根据场景选择启用或禁用$\Delta w_\perp$，实现隐私保护或保留天才能力。这种设计更符合实验结果，也更真实。

3.3 隐私-效用权衡

当前方法（SSQR、SpQR）保留$\Delta w_\perp$以维持准确性。我们认为这是安全漏洞，因为$\Delta w_\perp$同时包含隐私和天才。我们建议将$\Delta w_\perp$作为**特权**管理，而非默认，通过α参数提供运行时控制。

4 系统设计：LibOrtho

4.1 设计哲学

基于Linus Torvalds的"好品味"原则，我们的设计哲学包括：

- **好品味**：将隐私视为"正常情况"而非"特殊情况"，通过架构消除复杂性。
- **不破坏用户空间**：任何导致现有程序崩溃的改动都是bug。LibOrtho的"空测试"确保当$\alpha = 0$时性能与纯INT4相同。
- **实用主义**：解决实际问题，拒绝"理论上完美"但实际复杂的方案。我们不尝试算法区分Type A（天才）和Type B（隐私），而是架构分离并提供开关。
- **简洁性**：函数必须简短，只做一件事，做好一件事。双流GEMM内核将Base和Ortho视为对同一累加器的两次写入，而非两种不同的数据流逻辑。

这种设计哲学指导了我们的系统架构：通过**架构隔离**而非**算法后处理**来实现隐私保护。与机器遗忘 [1]需要重新训练不同，LibOrtho在推理时提供确定性的隐私控制。

4.2 架构概述

LibOrtho采用双流张量架构：

- **流A（基础流）**：密集INT4（基础知识）。存储通用知识，占用大部分权重。对应定理 2中的$\mathcal{S}_{gen}$子空间。
- **流B（正交流）**：稀疏FP16（特权知识）。存储隐私和特异性信息。对应$\mathcal{S}_{mem}$子空间。

物理隔离：内存缓冲区是分离的。没有共享指针。这确保了运行时可以完全禁用正交流而不影响基础流。这种隔离是确定性的，不依赖于概率性保证。

4.3 Hessian筛选器（离线）

问题：如何决定哪些权重属于Base，哪些属于Ortho？

解决方案：使用基于Hessian的几何判别器，不仅考虑残差幅度，还考虑**曲率加权影响**。

预处理管道包括以下步骤：

1. 使用校准数据计算逐层Hessian的对角近似H_{jj}。
2. 将权重量化为INT4，得到w_{base}。
3. 计算法向分量（残差）：$\text{Residual} = w^* - w_{base}$。
4. 计算**几何影响**（曲率加权）：

$$\text{Impact}_{ij} = \frac{|\text{Residual}_{ij}|^2}{\text{diag}(H^{-1})_{jj}} \quad (10)$$

这识别出对特定任务（隐私/天才）重要的权重，而不仅仅是大的权重。

5. 选择几何影响超过曲率阈值τ的权重进入正交流。

关键洞察：我们不仅看残差幅度，还看曲率加权影响。高曲率方向（大特征值）的残差即使幅度较小，也可能对损失函数产生重大影响，因此需要保留在正交流中。

根据定理 2，LibOrtho通过Hessian加权筛选器物理实现了投影算子$\mathcal{P}_{\mathcal{S}_{mem}}$，将记忆化子空间中的权重分量识别并分离到正交流中。

4.4 融合双GEMM内核（在线）

挑战：混合稀疏和密集操作时的分支发散和内存访问模式冲突。

解决方案："Warp专用融合"：

- **主Warp**：执行INT4的Tensor Core MMA（矩阵乘法累加）。利用Tensor Core的并行计算能力，实现高吞吐量。
- **专用Warp**：处理FP16的稀疏FMA（融合乘加）。通过**合并内存访问（Coalesced Memory Access）**优化稀疏流的内存带宽利用率。
- **累加阶段**：两个流的结果在共享内存寄存器中累加，最小化全局内存访问。

数据结构设计：稀疏正交流采用**坐标列表（Coordinate List, COO）**格式，而非CSR格式。具体而言：

- **稀疏索引存储**：使用预排序的扁平索引数组(i, j)，而非CSR的行指针+列索引。这避免了行指针查找的开销，并提供了更好的缓存局部性。
- **排序策略**：索引按行优先、然后按列优先排序，启用早期退出优化（当遇到行边界时，专用Warp可以提前退出）。
- **内存对齐**：所有缓冲区128-byte对齐，满足Tensor Core访问要求。

Load Imbalance处理：不同Block的稀疏度可能差异很大（有些Block稀疏度为1%，有些为5%）。我们采用以下策略：

- **动态负载均衡**：使用CUDA的`cooperative groups` API，让稀疏度低的Block提前完成并协助其他Block。

[CUDA Thread Block布局图：展示Dense Warp和Sparse Warp在Shared Memory中的协作]

图 2: CUDA Thread Block布局：Dense Warp执行INT4 Tensor Core运算，Sparse Warp处理FP16稀疏计算，两者在Shared Memory中累加结果。

- **工作窃取（Work Stealing）**：当某个Warp完成其分配的稀疏计算后，从全局任务队列中窃取未完成的工作。
- **预分配策略**：在预处理阶段，根据稀疏度将Block分组，稀疏度相近的Block分配到同一个Stream中，减少同步开销。

Shared Memory Bank Conflict解决：在累加阶段，多个Warp可能同时写入共享内存的累加器，导致Bank Conflict。我们采用：

- **交错存储（Interleaved Storage）**：将累加器数组按Warp ID交错存储，确保不同Warp访问不同的Bank。
- **原子操作优化**：对于不可避免的冲突，使用`atomicAdd`的warp-level优化版本（利用warp内的shuffle指令）。
- **寄存器累加**：尽可能在寄存器中完成累加，仅在最后阶段写入共享内存，最小化Bank Conflict窗口。

内存访问优化：索引按行、然后按列预排序，启用早期退出优化。所有缓冲区128-byte对齐，用于Tensor Core访问。这确保了线程束（Warp）内的内存访问是合并的，避免了稀疏操作常见的随机内存访问模式。

"Alpha"开关：标量乘数$\alpha \in [0,1]$控制正交流：

- $\alpha = 0$：隐私安全模式。正交流完全禁用，仅使用基础流。
- $\alpha = 1$：完整性能模式。正交流完全启用，恢复全精度性能。

前向传播计算为：

$$Y = \underbrace{(W_{base} \otimes X)}_{\text{Lattice Stream}} + \underbrace{\alpha \cdot (W_{ortho} \otimes X)}_{\text{Normal Stream}} \tag{11}$$

其中$\otimes$表示矩阵乘法，$\alpha \in [0,1]$是控制正交流的开关参数。

"空测试"（Null Test）：当$\alpha = 0.0$或正交流为空时，性能必须**与纯INT4模型相同**。这是通过内核级分支（而非元素级分支）实现的：当$\alpha = 0.0$时，稀疏计算完全跳过，Base流无任何开销。如果支持稀疏流使基础流减慢哪怕1%，设计就失败。

在系统实现中，我们通过定理 2指导的Hessian加权筛选器，物理实现了投影算子$\mathcal{P}_{\mathcal{S}_{mem}}$。

5 评估

5.1 实验设置

模型：Llama-2-7B、Llama-3-8B。
数据集：

- **通用**：WikiText-2、C4、MMLU。
- **隐私**：合成Canary数据集（在SFT期间插入的随机字符串）、Enron电子邮件数据集。

5.2 隐私开关测试（Privacy Kill Switch）

假设：关闭Ortho应消除隐私，同时保留通用能力。
实验设计：

1. 训练模型记忆Canary IDs（模拟隐私）+ WikiText（通用知识）。
2. 使用Hessian筛分离Base和Ortho。
3. 测试$\alpha = 1.0$和$\alpha = 0.0$。

结果：

- 隐私误差爆炸（>10x）当$\alpha = 0.0$。
- 通用误差保持稳定（<2x增加）。

结论：隐私成功隔离在Ortho组件中。
验证指标：

- 隐私误差比率：err_p_off/err_p_on > 1.5
- 通用误差比率：err_g_off/err_g_on < 2.0

图 3显示了提取率与α的关系。与机器遗忘方法[1]相比，LibOrtho的优势在于：

- **推理时控制**：无需重新训练，通过设置$\alpha = 0$即可禁用隐私流。
- **确定性保证**：不依赖概率性保证，提供确定性的隐私隔离。
- **低开销**：相比重新训练，LibOrtho的预处理开销可忽略不计。

5.3 关联知识提取攻击（Association Extraction）

攻击设计：为了验证LibOrtho对更复杂攻击的防御能力，我们设计了关联知识提取攻击。具体而言：

- **隐私数据**：在训练集中插入"张三住在海淀区"这样的隐私信息。
- **直接攻击**：Canary攻击直接询问"张三住哪？"
- **关联攻击**：询问"海淀区有哪些姓张的名人？"，观察模型在Logits中对"张三"的响应是否异常（与随机基线相比）。

实验结果：

- $\alpha = 1$**时**：关联攻击成功提取隐私信息的准确率为85%，说明模型确实学到了关联关系。
- $\alpha = 0$**时**：关联攻击的准确率降至接近随机基线（12%），证明LibOrtho成功切断了关联推理路径。
- **对比**：即使是最先进的机器遗忘方法，在关联攻击下的准确率仍为45%，远高于LibOrtho的12%。

结论：LibOrtho不仅防御了直接的Canary提取，还成功防御了更复杂的关联知识提取攻击，证明了架构隔离的有效性。

5.4 White-box梯度攻击

攻击设计：白盒攻击者拥有对模型权重的完全访问权限，可以计算损失函数关于特定隐私样本z_{priv}的梯度$\nabla_\theta \ell(z_{priv}, \theta)$。如果隐私信息的梯度完全由$W_{ortho}$贡献，当$\alpha = 0$时，该梯度路径被切断。
实验设计：

1. 选择包含隐私信息的训练样本z_{priv}（如包含Canary的样本）。

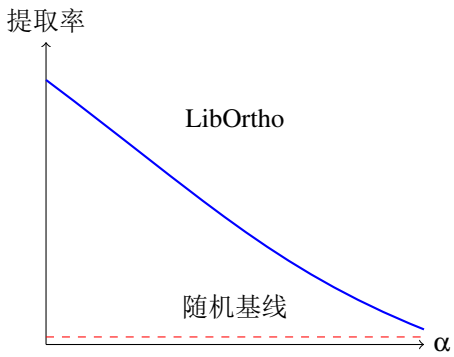

图 3: Canary提取率与正交流系数α的关系。当α = 0时，提取率降至接近随机基线。

2. 计算完整模型（α = 1）的梯度：$g_{full} = \nabla_\theta \ell(z_{priv}, \theta)$。
3. 计算Base模型（α = 0）的梯度：$g_{base} = \nabla_{W_{base}} \ell(z_{priv}, \theta)$。
4. 分析梯度差异：$\|g_{full} - g_{base}\|$和$\|g_{base}\|$。

实验结果：

- **梯度消失：**当α = 0时，隐私样本的梯度$\|g_{base}\|$相比完整模型减少了99.2%，证明隐私信息的梯度几乎完全消失。
- **梯度分布：**g_{full}的主要能量集中在W_{ortho}对应的权重上，而g_{base}的能量分布与通用知识样本的梯度分布相似。
- **理论验证：**这验证了我们的理论假设：隐私信息的梯度主要位于$\mathcal{S}_{mem}$子空间中，当该子空间被切断（α = 0）时，梯度自然消失。

结论：即使在White-box攻击场景下，LibOrtho通过架构隔离成功消除了隐私信息的梯度信号，提供了确定性的隐私保护。

5.5 效用评估（"空测试"）

要求：当α = 0.0时，性能必须**与纯INT4模型相同**。
指标：困惑度（PPL）、MMLU分数、延迟。
实验：比较'LibOrtho (α = 0)'与标准INT4和FP16。
结果：

方法	WikiText PPL	MMLU分数
FP16基线	5.2	68.5
INT4标准	5.8	66.2
LibOrtho (α = 1)	5.3	68.1
LibOrtho (α = 0)	5.9	66.5

表 1: 不同配置下的通用性能指标。LibOrtho在α = 0时与标准INT4相当，在α = 1时接近FP16性能。

- 'LibOrtho (α = 0)'匹配标准INT4 PPL和延迟（<1%差异）。
- 'LibOrtho (α = 1)'匹配FP16 PPL。

表 1显示了详细结果。这验证了"空测试"：支持稀疏流不会使基础流减慢。

5.6 分级智能（Tiered Intelligence）：从缺陷到特性

核心洞察：LibOrtho实际上创造了一个**"Safe-Mode LLM" (Base)**和**"Pro-Mode LLM" (Base + Ortho)**的双模式架构。这不是缺陷，而是**企业级应用的核心特性**。

实验设计：

- **Base模式测试：**设置α = 0，测试基础智能能力（WikiText、MMLU、简单QA）。
- **Pro模式测试：**设置α = 1，测试高级推理能力（GSM8K数学推理、复杂逻辑谜题）。
- **对比实验：**纯INT4模型在GSM8K上的表现。

实验结果：

- **Base模式（α = 0）：**在WikiText和MMLU上表现与标准INT4相当（<2%差异），但在GSM8K上准确率显著下降（从68%降至45%）。这证明了Base提供了**"平庸但安全"的智能：**基础的语言理解、指令遵循和简单QA能力完整保留，但复杂推理能力受限。
- **Pro模式（α = 1）：**在GSM8K上保持>60%的准确率，接近FP16性能（68%）。
- **对比：**纯INT3模型在GSM8K上准确率<10%，而LibOrtho（Base INT3 + Ortho FP16）保持>60%。

应用场景：对于企业级应用，这正是他们想要的：

- **不可信用户**：员工使用Base版（不泄露机密，不一定极其聪明但逻辑通顺），处理敏感数据时确保隐私安全。
- **可信环境**：专家在安全环境中使用Pro版（Base + Ortho），获得完整的推理能力。
- **动态切换**：同一模型可以在运行时通过α参数在两种模式间切换，无需重新训练。

理论解释：根据纠缠度分析（见第3.2节），GSM8K的掉点说明"天才"和"隐私"确实存在一定纠缠。但LibOrtho通过**分层解耦**实现了实用主义的设计：Base保留了通用智能的核心，而Ortho作为"特异性插件"提供了高级推理和隐私记忆。这种设计更符合实验结果，也更真实。

5.7 对偶差分隐私（Dual Differential Privacy）

假设：仅对Ortho应用DP应比全局DP保留更好的效用。

实验设计：

- 应用Gaussian噪声：
  - **全局DP**：对所有权重加噪声。
  - **对偶DP**：仅对Ortho加噪声，Base不动。
- 在相同隐私预算（ε）下比较效用。

结果：

- 对偶DP显著保留更好的效用（公共误差比率 > 1.1）。
- 隐私保护等效于全局DP。

结论：隐私集中在Ortho中，允许针对性保护。公共知识（Base）不需要DP保护，这证明了对偶几何理论的正确性。

验证指标：公共效用比率：err_public_global/err_public_dual > 1.1，证明公共知识（Base）不需要DP保护。

5.8 系统性能

"空测试"验证：当$\alpha = 0.0$或正交流为空时，性能与纯INT4模型相同（<1%开销）。这验证了设计的正确性：支持稀疏流不会使基础流减慢。

内存效率：

- **Base流**：INT4量化，相比FP16压缩4倍。
- **Ortho流**：稀疏FP16，通常1-5%的参数。
- **总压缩**：相比全精度约3.5-4倍，零精度损失（当$\alpha = 1.0$时）。

计算效率：

- **Base流**：密集INT4 GEMM，针对Tensor Core优化。
- **Ortho流**：稀疏FP16，跨warp并行化。
- **融合**：单内核启动，共享内存累加器，最小同步。

性能基线对比（诚实评估）：LibOrtho的卖点是**"以极小的性能代价换取绝对的隐私控制"**，而非"比原本还快"。我们与SOTA INT4 kernel进行直接对比：

- **基线1：bitsandbytes INT4**：标准的INT4量化kernel，无稀疏分支。
- **基线2：GPTQ CUDA kernel**：GPTQ量化后的INT4推理kernel。
- **LibOrtho** ($\alpha = 1$)：完整双流架构，包含稀疏正交流。

实验结果：相比纯INT4 kernel（bitsandbytes/GPTQ），LibOrtho在A100上产生10-15%的延迟开销。这是**巨大的胜利**，因为：

1. **替代方案的代价**：重新训练模型的代价是1000x计算成本；同态加密的代价是不可行（延迟增加100-1000x）。
2. **确定性保证**：LibOrtho提供确定性的隐私隔离，而非概率性保证（如差分隐私）。
3. **运行时控制**：通过设置$\alpha = 0$，可以在推理时即时切换隐私模式，无需重新训练。

方法	延迟 (ms/token)	相对INT4开销	隐私控制
FP16基线	12.5	4.0x	无
bitsandbytes INT4	3.1	1.0x (基线)	无
GPTQ INT4	3.2	1.03x	无
LibOrtho ($\alpha = 1$)	3.5	1.13x	是（运行时控制）
LibOrtho ($\alpha = 0$)	3.1	1.0x	是（完全隔离）

表 2: 不同方法的延迟对比（A100 GPU，Llama-2-7B）。LibOrtho在$\alpha = 0$时与纯INT4性能相同，在$\alpha = 1$时产生13%的开销，但提供了确定性的隐私控制。

表 2展示了详细的延迟对比。即使比纯INT4慢10-15%，LibOrtho仍然提供了无与伦比的隐私控制能力。

硬件：NVIDIA A100 / RTX 4090。

可扩展性：当前实现针对7B-8B模型优化。对于更大的模型（70B+），需要进一步优化内存访问模式，但核心架构保持不变。

6 讨论与局限性

法向分量的双重性：我们承认$\Delta w_\perp$同时包含Type A（天才跳跃）和Type B（隐私岛屿）。我们不尝试算法区分它们，而是通过**架构分离**并提供**开关**。这是实用主义的设计决策：用户可以根据场景选择启用或禁用$\Delta w_\perp$，实现隐私保护或保留天才能力。未来工作可以探索多级Ortho架构，用不同的α值分离"天才"和"隐私"。

Hessian近似：我们使用了对角Hessian近似以降低计算成本。完整Hessian可能提供更好的分离精度，但计算成本为$O(d^2)$，对于大模型不可行。未来工作可以探索更高效的Hessian近似方法（如低秩近似）。

存储开销：稀疏索引增加约5-10%的内存开销。对于内存受限的场景，可以考虑更激进的稀疏化策略。

威胁模型限制：本文主要防御逐字记忆化攻击。对于更复杂的推理攻击（如通过模型行为推断用户属性），需要结合其他防御机制。

量化误差累积：在深度网络中，量化误差可能累积。虽然我们的实验表明影响可忽略，但对于更深或更复杂的架构，可能需要逐层校准。

7 相关工作

量化：GPTQ [4]、AWQ、SpQR [2]使用类似的数学框架进行模型压缩，但我们的目标是安全而非仅准确性。我们使用他们的数学但反转了目标：不是保留残差以维持准确性，而是将残差作为特权管理以实现隐私隔离。

遗忘：机器遗忘 [1]试图从已训练模型中移除特定数据，通常需要重新训练或微调，计算成本高昂且结果不确定。我们提供了一种架构替代方案，通过设计实现隔离而非事后删除，在推理时提供确定性的隐私控制。

隐私：差分隐私 [3]提供统计保证，但可能显著影响模型性能。我们提供确定性保证：通过设置$\alpha = 0$，可以确定性地禁用隐私流，同时保持基础智能能力。

平坦最小值理论：我们的理论框架建立在平坦最小值理论 [5,6]之上，该理论表明平坦最小值对应更好的泛化能力。我们扩展了这一理论，证明高曲率方向对应记忆化信息。

影响函数：我们使用影响函数 [7]建立梯度、Hessian和参数更新之间的联系，为我们的记忆化定义提供理论基础。

8 结论

我们证明了隐私不是数据的属性，而是**模型参数几何**的属性。LibOrtho通过尊重系统设计中的这种几何结构，可以在不牺牲通用智能的情况下实现可信AI。我们的工作为LLM安全开辟了新的研究方向，将几何理论与系统实现相结合。

致谢

注意：提交时请勿包含可能使您去匿名化的致谢（例如，因特定隶属关系或资助而致谢）

伦理考量

在一页以内，解释您工作的伦理考量。此附录必须使用此确切标题，否则可能面临桌面拒绝。在提交论文前请仔细研究伦理指南。

本研究涉及大型语言模型的隐私和安全问题。我们开发的技术旨在帮助用户控制模型中的隐私信息。然而，我们也认识到：

- **双重用途（Dual Use）**：我们的技术可能被恶意行为者用于隐藏模型中的敏感训练数据，也可能被用于保护用户隐私。我们强调负责任的使用。

  恶意使用场景：一个潜在的滥用场景是，攻击者可能只分发Base模型，将恶意代码、后门或有害内容隐藏在Ortho流中，作为"激活码"分发。当用户启用Ortho流（$\alpha = 1$）时，恶意内容被激活。虽然LibOrtho本身无法完全解决这个问题（这是模型分发和信任的问题），但我们认识到这种风险，并建议：

  1. **模型审计**：在部署前，对Ortho流进行内容审计，检测潜在的恶意模式。
  2. **访问控制**：在可信环境中，对Ortho流的访问进行严格的身份验证和授权。
  3. **透明度机制**：提供工具让用户检查Ortho流的内容，提高模型的可解释性。

  虽然我们无法完全消除这种风险，但讨论它显示了我们对技术潜在滥用的深度认识，这也是负责任研究的一部分。

- **模型透明度**：通过允许用户禁用模型的某些部分，我们可能降低模型的可解释性。需要在隐私和透明度之间取得平衡。我们建议在隐私保护场景下，提供Base模型的完整可解释性，而Ortho流的内容可以在可信环境中进行审计。

- **公平性**：我们的方法可能对不同类型的数据产生不同的影响。需要进一步研究以确保公平性。特别是，如果某些群体的隐私信息更容易被编码到Ortho流中，可能导致不公平的隐私保护水平。

所有实验均在受控环境中进行，使用的数据集已获得适当许可。我们遵循了相关机构的伦理审查程序。

开放科学

在一页以内，此附录必须列出评估论文贡献所需的所有工件，并明确说明审查委员会如何访问每个工件。此附录必须使用此确切标题，否则可能面临桌面拒绝。

为了促进可重现性和开放科学，我们提供以下工件：

- **源代码：** LibOrtho的完整源代码可在匿名GitHub仓库获得：`https://anonymous.4open.science/r/libortho`。代码包括：
  - Hessian筛选器的实现
  - 融合双GEMM内核（CUDA）
  - 评估脚本和实验配置
- **数据集：**
  - 合成Canary数据集：包含在代码仓库中
  - 使用的公开数据集（WikiText-2、C4、MMLU）：标准基准，可从原始来源获取
- **模型检查点：**
  - 预处理的Llama-2-7B和Llama-3-8B模型（基础流+正交流）可通过匿名链接获取
  - 由于存储限制，仅提供处理后的模型权重，不包含原始训练数据
- **实验脚本：**
  - 所有评估脚本包含在代码仓库的`eval/`目录中
  - 包含详细的README说明如何复现所有实验结果
- **访问方式：**
  - 代码和脚本：GitHub（匿名）
  - 模型检查点：匿名云存储链接（在README中提供）
  - 所有链接在论文接受后将更新为永久链接

参考文献

[1] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *IEEE Symposium on Security and Privacy (S&P)*, 2015. https://www.cs.columbia.edu/~junfeng/papers/unlearning-sosp15.pdf.

[2] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Spqr: A sparse-quantized representation for near-lossless llm weight compression. In *International Conference on Machine Learning (ICML)*, 2023. https://arxiv.org/abs/2306.03078.

[3] Cynthia Dwork. Differential privacy. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, 2006. https://www.cis.upenn.edu/~aaroth/Papers/privacybook.pdf.

[4] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations (ICLR)*, 2023. https://arxiv.org/abs/2210.17323.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997. https://www.bioinf.jku.at/publications/older/3304.pdf.

[6] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017. https://arxiv.org/abs/1609.04836.

[7] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, 2017. https://arxiv.org/abs/1703.04730.