arXiv:2302.02407v2 [cs.CR] 8 Dec 2023

HyPHEN：面向同态加密神经网络的混合打包方法及其优化

Donghwan Kim*、Jaiyoung Park*、Jongmin Kim、Sangpyo Kim 和 Jung Ho Ahn 首尔国立大学，韩国首尔 {*eastflame, jeff1273, jongmin.kim,vnb987, gajh*}*@snu.ac.kr*

摘要

使用全同态加密（*FHE*）进行卷积神经网络（*CNN*）推理，是一种颇具前景的隐私推理（*PI*）方案。*FHE* 能够在保护敏感用户数据隐私的同时，将整个计算过程交由服务器处理。此前基于全同态加密的卷积神经网络（*HCNN*）研究已证明，利用 *FHE* 构建残差网络等深度神经网络架构是可行的。尽管取得了这些进展，*HCNN* 的实用性仍面临重大挑战，原因在于其计算和内存开销高昂。为克服这些局限性，我们提出 *HyPHEN*，这是一种深度 *HCNN* 构建方案，结合了新颖的卷积算法（*RAConv* 和 *CAConv*）、数据打包方法（二维间隙打包和 *PRCR* 方案），以及专为 *HCNN* 构建而设计的优化技术。这些改进使 *HyPHEN* 大幅降低了内存占用，并减少了密文旋转和自举等高开销同态运算的次数。因此，*HyPHEN* 将 *HCNN* 在 *CIFAR-10* 上的推理延迟降至实用水平：*ResNet-20* 为 *1.4* 秒；此外，它首次实现了 *HCNN* 在 *ImageNet* 上的推理，*ResNet-18* 的延迟为 *14.7* 秒。

索引词–隐私推理、卷积神经网络、全同态加密

1. 引言

隐私推理（PI）近年来在机器学习即服务（MLaaS）领域备受关注，使云服务公司能够遵守《通用数据保护条例》 [1] 和《健康保险流通与责任法案》 [2] 等隐私法规。PI 使云服务器能够提供推理服务，同时保护客户端的隐私以及服务提供商的知识产权。例如，医院可以利用 PI 提供疾病的隐私医学诊断，安保公司则可以提供隐私监控系统，而无需访问客户端的敏感数据 [3, 4]。

全同态加密（FHE） [5] 是一种密码学原语，能够直接对加密数据执行丰富的一组函数，因此在安全性和易用性方面尤其适用于隐私推理，与其他密码学方案相比更具优势——候选者 [6, 7]。基于全同态加密的私有推理方案，如图1所示，具有以下独特特点：1）将整个计算过程完全卸载到服务器；2）数据通信需求精简，

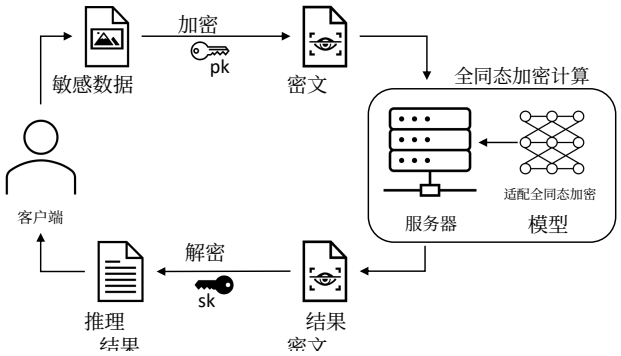


图1： 基于全同态加密的私有推理。

以及 3）除推理结果外，不披露任何模型信息。全同态加密具有直接对密文（加密数据）进行计算的独特能力，因此这些优势推动研究人员探索基于全同态加密的卷积神经网络隐私推理（HCNN） [8–12]。具体而言，本研究探讨了采用 CKKS 全同态加密方案构建 HCNN [13],; 与其他全同态加密方案相比，该方案吞吐量更高，并支持处理实数和复数。

尽管具有这些优势，全同态加密仍会带来很高的计算和内存开销，这阻碍了同态卷积神经网络在现实服务中的应用。此外，与未加密数据相比，全同态加密对加密数据的计算具有显著不同的特性，主要体现在加密状态下调整数据组织的代价极高。因此，为了最大限度地降低计算和内存开销，同态卷积神经网络需要一种经过优化的卷积算法，以及一种针对全同态加密环境设计的独特数据组织方式。

Gazelle [14], 是一项隐私推理领域的开创性研究，它提出了一种高效的卷积算法，可用于全同态加密；与直接采用传统的未加密卷积神经网络推理卷积算法相比，该算法能够减少同态运算次数。Gazelle通过结合安全多方计算（MPC），避免了全同态加密求值数据重排和非线性激活（例如 ReLU）的高昂成本。不过，Gazelle仍使用全同态加密累加分散在密文中的数据元素，这同样代价高昂。 [15] 通过引入基于全同态加密的数据重排和 ReLU 求值方法，扩展了 Gazelle 的算法，以支持端到端同态卷积神经网络推理。 [11]则通过采用更密集的数据格式，进一步提升了同态卷积神经网络的性能，从而尽可能减少密文数量。然而，先前的实现需要数十

*贡献相同

分钟 [11] 甚至数小时 [15] 才能完成一次针对 CIFAR-10（ResNet-20）的同态卷积神经网络推理，原因是累加、数据重排和 ReLU 求值的成本高昂。利用 AESPA 提出的激活函数，我们可以大幅降低激活函数的成本 [16]；不过，累加和数据重排仍然存在成本。

我们将多种灵活的数据格式与一种多态全同态加密卷积算法相结合，该算法针对每种数据格式执行相应的卷积，以此解决这一问题。我们发现，数据重排成本较高，源于卷积层输入与输出的数据格式不一致。以往工作在整个同态卷积神经网络推理过程中采用固定的数据格式，因此需要频繁使用同态旋转操作进行数据重排。与此不同，我们提出采用多种灵活的数据格式，允许在密文中复制数据元素。卷积层的输入和输出可以自由选择多种数据格式，以尽可能降低数据重排成本。为支持不同的数据格式，我们还设计了针对每种数据格式的多态全同态加密卷积算法。复制还带来了一个积极的副作用：由于需要累加的不同数据元素更少，卷积过程中的数据累加成本也随之降低。

此外，我们的数据格式使同态卷积神经网络能够扩展至真实世界图像和更大的卷积神经网络模型。在以往的工作中，[9–11, 14, 15],明文权重的内存扩展问题未得到充分考虑。然而，我们发现，权重明文的内存占用会随着图像尺寸增大而显著增加，最终成为主要瓶颈。因此，现有技术由于存储权重所需的内存量庞大，无法扩展至 ImageNet 等更大的数据集。为应对这一挑战，我们提出一种能够有效减小明文权重尺寸的数据格式化方法。具体而言，我们沿行方向将图像划分为多个图像分段，使明文尺寸与图像分段尺寸相关联。因此，权重明文会变得更小，并可在卷积过程中于这些图像分段之间重复使用，从而减轻内存占用。

我们评估了 HyPHEN 的真实世界可扩展性；HyPHEN 是一个结合上述解决方案的同态卷积神经网络框架。HyPHEN 的图形处理器实现采用 ResNet-20 模型进行加密 CIFAR-10 推理，仅需 1.40 秒。我们还首次展示了使用 ResNet-18 模型在 ImageNet 数据集上进行端到端同态卷积神经网络推理，实现了 14.69 秒的执行时间。

本文的主要贡献如下：

- 我们提出了一种新颖的数据格式，可实现连续卷积之间简化的数据排列。
- 我们设计了优化的全同态加密卷积算法，能够以更低的计算复杂度支持多种数据格式。
- 我们发现，HCNN 巨大的内存占用会损害其性能，并提出了一种高效的数据格式，可节省数百 GB 的内存空间，且开销可以忽略不计。
- 我们展示了在实际可行的执行时间内对各种神经网络进行的评估。通过展示 ResNet-18 模型的性能，我们将基于全同态加密的隐私推理扩展到 ImageNet 等复杂的真实世界数据。

2. 背景

2.1. 全同态加密（FHE）

全同态加密是一系列公钥加密方案，可在加密数据上进行计算。在多种流行的全同态加密方案中，RNS-CKKS [17] 因支持定点数和槽位批处理，已在隐私推理领域得到广泛采用。RNS-CKKS 中的明文 是分圆多项式环 $\mathbb{Z}[X]/(X^N+1)$ 上的未加密多项式，其次数为 2 的幂N，典型取值为 2^{15}–2^{17}。明文可以加密为密文，密文由一对多项式组成，通过额外的随机多项式和混淆误差项来隐藏明文。由于包含$N/2$ 个实数（或复数）的向量（称为消息）可以映射为明文，因此我们可以对明文和密文执行同态运算，同时处理$N/2$ 个数。这称为槽位批处理，因为该向量消息中的每个位置都称为槽位。也可以对长度较短（为 2 的幂）的消息进行批处理。

RNS-CKKS 方案支持按槽位进行加法（AddPt/AddCt）和乘法（MulPt/MulCt）运算，其中部分运算及其1 执行时间如表所示。例如，AddCt/MulCt 接收两个密文作为输入，并返回一个密文，其对应的向量消息近似等于两个输入密文所含向量消息逐元素相加或相乘的结果。AddPt/MulPt 的作用类似，但它们接收一个明文和一个密文作为输入，且所需计算量少于 AddCt/MulCt。具体而言，MulPt 所需的 34×执行时间少于 MulCt。

同态加密的一项基本局限性是，与密文进行的连续乘法次数有限。密文或明文都关联一个（乘法）层级 ℓ，其中 $0 \le \ell \le L$ 表示最大层级 L。密文或明文的大小以及同态运算的复杂度会随着层级增加。执行 MulCt 或 MulPt 后，应对输出密文进行重缩放，以降低乘法导致的密文误差放大。重缩放会使层级降低一级，且层级不能低于零。此外，若要对层级不同的操作数进行运算，则需要执行重缩放，将操作数的层级调整为其中最低的层级。

为克服这一局限性，需要进行自举（表1中的 Boot），这是一种用于提升密文层级的独特操作。借助自举，密文的层级可以

表1：在中央处理器（64线程）上对同态运算进行100次迭代取平均值的基准测试。Pt和Ct后缀分别表示明文-密文运算和密文-密文运算。PRot和CRot分别表示明文和密文的旋转。实验设置详见第4.1节。

运算	AddPt	AddCt	MulPt	MulCt	重缩放	PRot	CRot	启动
时间（毫秒）	0.169	0.202	0.506	17.3	3.90	0.102	15.5	2160

提升至 L'，但由于自举计算会消耗层级，该值小于最大层级 L。因此，在实际应用中，我们只能将 L' 个层级用于其他操作。从表1可以看出，自举的开销比基本操作高出数个数量级。因此，抑制层级消耗并尽可能减少自举次数至关重要。

HE的另一项局限性在于，使用槽位批处理时，很难任意改变槽位间的数据顺序。唯一可用的选项是循环移位槽位，我们将其称为旋转。CRot是一种同态运算，可将密文向左旋转指定的量。在需要对不同槽位中的数据元素进行计算时，会执行CRot来调整位置。由于这种计算模式极为常见，CRot是RNS-CKKS应用中最常执行的运算之一。此外，CRot在基本运算中相对昂贵（见表1），因此占据了很大一部分计算开销。明文也有一种旋转运算PRot，但很少执行，且计算成本较低。

在 RNS-CKKS 中，槽位之间的数据组织至关重要，因为它决定了密文旋转和自举操作的数量。其他操作的开销对数据组织不太敏感。为了减少自举次数，建议尽可能将更多数据元素打包到一个密文中，从而减少需要自举的密文数量。然而，这种密集打包可能会导致旋转次数增加，因为将某些数据元素改放到另一个密文中，就可以免去旋转操作。举例来说，如果每个密文中只打包一个元素，就可以消除所有旋转操作；但这会使自举和其他操作的开销过高。因此，为了让 RNS-CKKS 应用实现高性能，我们需要设计一种针对具体应用的数据组织方式，在两个目标之间取得平衡：尽量减少不同槽位之间的计算，同时最大限度地利用槽位。

2.2. 同态加密上的卷积

我们首先介绍基线卷积神经网络的符号说明。为简化符号，我们假设图像和滤波器均为正方形。我们分别用元组 {通道数、图像宽度} 表示单个卷积层的输入和输出：$\{c_i,\ w_i\}$ 和 $\{c_o,\ w_o\}$。卷积的参数用元组 {输出通道、输入通道、滤波器宽度} 表示为 $\{c_o,\ c_i,\ f\}$，步幅 s 和填充 pad。

算法 1 单输入单输出通道卷积（SISO）

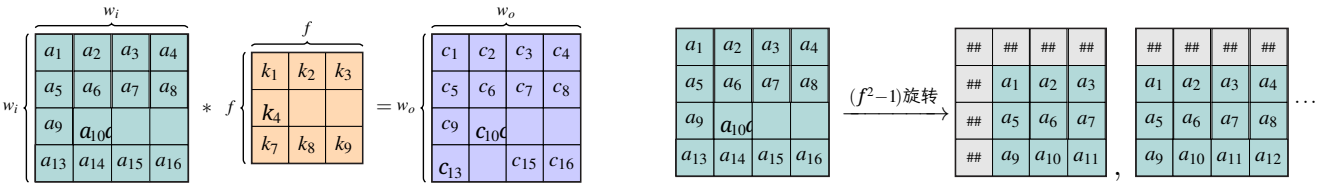
输入密文$_i$：输入密文，W：明文滤波器
输出密文$_o$：输出密文
// 子程序：**Slide**^{f}(密文$_i$)
对于$j_1 = 0,\ldots,f-1$**执行**
  对于$j_2 = 0,\ldots,f-1$**执行**
    $r \leftarrow w_i(j_1 - \frac{f-1}{2}) + (j_2 - \frac{f-1}{2})$
    $ct'[j_1,j_2] \leftarrow$ 密文旋转$(ct_i; r)$
// 子程序：**MulFilter&Sum**$^f(ct_i, W)$
对 $j_1 = 0,\ldots,f-1$**执行**
  对 $j_2 = 0,\ldots,f-1$**执行**
    ct_o += 明文乘法$(ct'[j_1,j_2], W[j_1,j_2])$
返回ct_o

单输入单输出通道卷积（SISO）： Gazelle [14] 提出了一种高效的同态加密卷积算法，称为 SISO。图 2展示了 s 分别等于 1 和 2 时的 SISO，其中一个输入密文包含 w_i^2 个像素。尽管本例中我们用二维格式表示数据$(H,\ W)$，但无论维度数如何，像素都以扁平化的行主序存储在槽位中。f^2 个明文，每个明文存储不同的滤波器元素，需为 SISO 预先准备。第 i 个明文（$0 \le i < f^2$）的每个槽位中存放一个滤波器元素（k_i）或零（0），具体取决于 k_i 是否参与计算同一槽位处的输出像素。SISO 的操作步骤如下（参见算法1）：

1. *Slide*^{f}针对 f^2 个明文，分别以不同的旋转量旋转置于加密密文中的输入图像，如图 2b所示。2. 滤波器乘法与求和^f 将每个旋转后的输入与一个滤波器明文相乘，并累加 f^2 个乘积结果，得到输出，如图 2c所示。

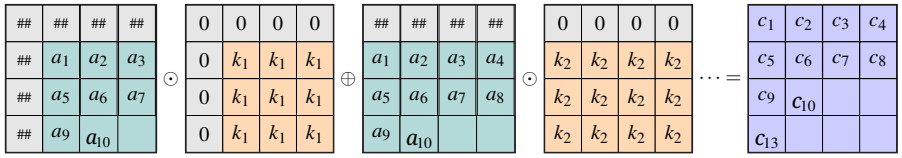
由于卷积神经网络中的输入图像元组通常具有多个通道，因此使用单输入单输出时，每个卷积层通常需要多个密文。我们分别用n_i 和n_o表示每个卷积层的输入和输出密文数量。

卷积与数据格式：当图像大小（w_i^2）小于 N/2时，我们可以通过将多个通道批处理到单个密文中，来减小 n_i 和 n_o。Gazelle 还提出了一种通道对齐批处理方法，其中输入和输出密文的数据格式可视为展平的 (C,H,W) 三维向量。[15]沿用此方法，并采用相同的 (C,H,W) 格式，使用全同态加密实现端到端卷积神经网络推理。遵循与单输入单输出类似的步骤，这种格式可以同时对多个通道执行卷积。此外，[11] 引入了输入重复以进一步提高并行性，数据格式可表示为 (R,C,H,W)，并将输入元组重复 |R|次，以填满密文的所有槽位。在图 3 中，我们给出了一个示例，其中 $\{c_o, c_i, w_i\} = \{2,2,32\}$ 以及槽位数（N/2）为 4,096。输入密文（$n_i = 1$）有两个

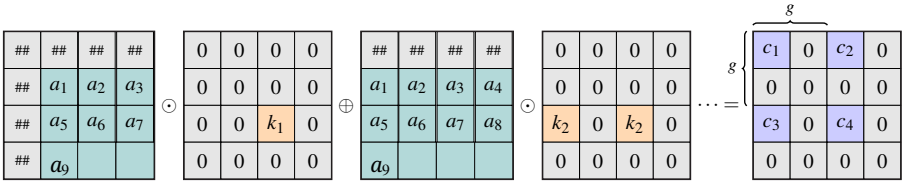


(a) 未加密的卷积，用于s，填充 $=1$

(b) Slide$f$用于$f=3$



(c) 滤波器乘法与求和f针对s，填充 $=1$



(d) 滤波器乘法与求和f针对$s=2$，填充 $=1$

图 2：单输入单输出通道卷积（SISO）[14]。图像密文和滤波器明文以二维矩阵形式展示，但存储时采用一维方式，将每个矩阵行拼接起来。 $\odot$ 表示明文乘法， $\oplus$ 表示 AddCt。

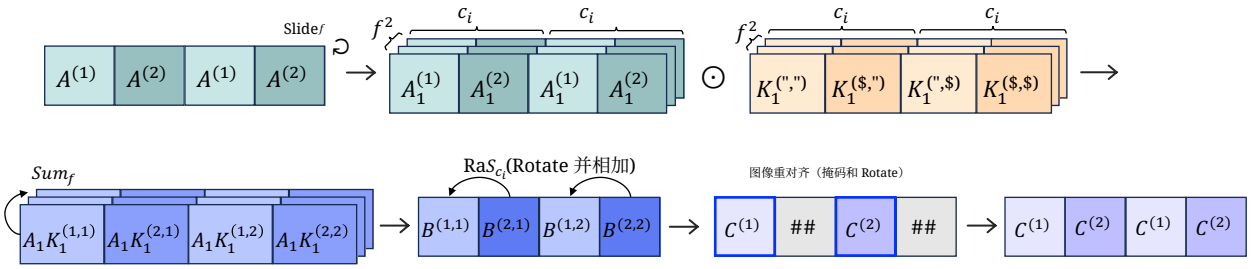


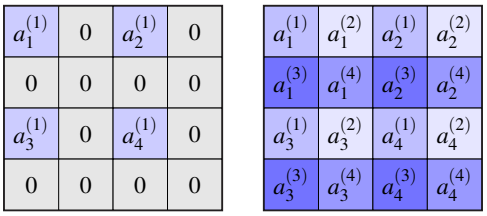
图 3：同态加密中的卷积过程 [11] 当c_i，$c_o=2$时。单个上标表示通道，上标对表示（输入通道，输出通道）。$A^{(m)}$ 表示输入图像的第m个通道，该通道连续排列在槽位中；$K^{(m,n)}$ 是以明文形式部署的对应滤波器。我们将中间的单输入单输出结果 $\sum_{i=1}^{f^2}$密文旋转$(A^{(m)};r(i))K^{(m,n)}{}_i$简记为$B^{(m,n)}$。$C^{(n)}$表示输出图像的第$n$个通道。

通道（$A^{(1)},A^{(2)}$）重复两次。准备了f^2个滤波器明文，其中每个明文包含适用于 $|R|$个输出通道、每个通道具有 $|C|$个输入通道的滤波器元素。对该密文 [11] 执行卷积可描述如下：

1. 单输入单输出：使用单个输入密文，可同时对 $|C|$个输入通道和 $|R|$个输出通道执行单输入单输出运算，得到卷积结果。生成的密文包含 $|R||C|$ 个中间卷积输出B（$1\le x\le|C|$， $1\le y\le|R|$）。
2. RaS$_{c_i}$：为获得第y个输出通道的结果，通过计算旋转求和（RaS）对 $|C|$个中间结果（$\sum_{x=1}^{|C|}B^{(x,y)}$）进行累加，这需要 $\log|C|$ 次旋转。
3. 图像重对齐：为继续执行后续卷积，数据格式需要通过对数据元素进行掩码操作并旋转来重新定位，使其与下一层的数据格式相匹配；我们将此过程称为图像重对齐（IR）。

在本文中，我们将这种作用于（R，C，H，W）密文的卷积称为卷积$_{lc}$。

间隙与多路复用打包： 步幅卷积（$s>1$）使用 SISO 会在有效值之间产生一个间隙（表示为g）（见图2d）。使用带有间隙的密文时，槽位利用不足会降低吞吐量。虽然 Gazelle 等交互式协议 [14]通过客户端辅助重新加密过程消除间隙，但全同态加密需要大量的掩码（MulPt）和旋转（密文旋转）操作来消除间隙，从而带来额外的计算和层级消耗。 [11]提出了基于多路复用打包（MP）的



（a）空隙填充 [18] (b) 多路复用打包方法 [11]

图 4：以往用于填充下采样层所产生间隙的间隙打包方法。$a^{(j)}_i$ 表示图像第j个通道中的第i个元素。

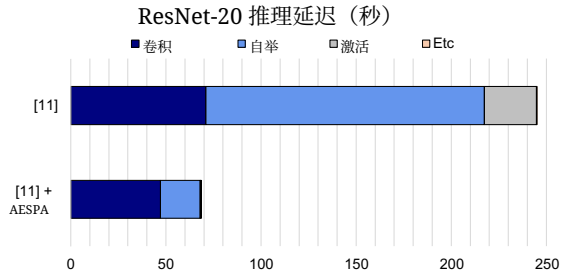


图 5：残差网络-20 的推理延迟及其分解，针对 [11] 以及采用低次多项式 [16]。两组实验分别在参数集$_{lc}$ 和参数集$_{hyp}$ 下进行，参数来自 8。

of卷积$_{lc}$ （*MP*-卷积$_{lc}$）方法，以解决槽位利用不足问题。

MP-Conv$_{lc}$ 在 Conv$_{lc}$ 的 IR 过程中加入重新打包过程，以不同通道填充间隙（见图 4b）。MP-Conv$_{lc}$ 采用 (R,C$_a$,H,W,C$_g$) 的数据格式，在最内层维度 C$_g$ 添加多路复用通道。对具有 $|C_g| = 4$ 的加密图像进行卷积的过程如图8a所示。在外层通道维度 C$_a$ 上应用 SISO 和 RaS 后，还会执行额外的 RaS，以累加位于间隙内层维度 C$_g$ 中的通道。多路复用打包需要更复杂的 IR 过程，其中包括用来自输出张量的多个通道填充间隙。有关 MP-Conv$_{lc}$ 的更多详情，请参阅 [11]的第 4 节。

2.3. 同态加密上的激活函数

非线性激活函数（例如 ReLU）不能直接用于 HCNN。由于全同态加密只支持加法和乘法运算，因此必须用逼近这些函数的多项式函数替代它们。要保持卷积神经网络模型的准确率，关键是在较宽的范围内将多项式逼近误差控制在较低水平。 [15]使用 15 次、15 次和 27 次多项式的复合函数逼近 ReLU，使逼近误差的 L1 范数在 [-50, 50] 范围内低于 2^{-13} 。这种方法的优点是可以直接使用预训练的卷积神经网络模型，无需修改。然而，高次多项式的求值会带来显著的运行时开销；一系列重缩放在求值过程中会导致大量层级消耗，从而增加自举次数。可以尝试使用层级更多（即 L' 更高）的 CKKS 参数来降低这部分开销；但是，如表 8 所示，这样的参数集会占用极大的内存，因此每次运算的成本都会高得多。

另一种方法是采用低次多项式激活函数重新训练神经网络，如 [19–23] 所示。通过重新训练，运算成本可大幅降低。近期，AESPA [16]表明，采用低次多项式训练的卷积神经网络，在各种卷积神经网络架构和图像数据集上，都能达到与原始基于 ReLU 的网络相当的准确率。AESPA 使用正交基多项式的复合以及逐基批归一化，取代 ReLU 和批归一化（BN）。在推理过程中，纵向层融合会将这些复合转换为简单的平方函数，从而大幅缩短激活函数的运行时间。

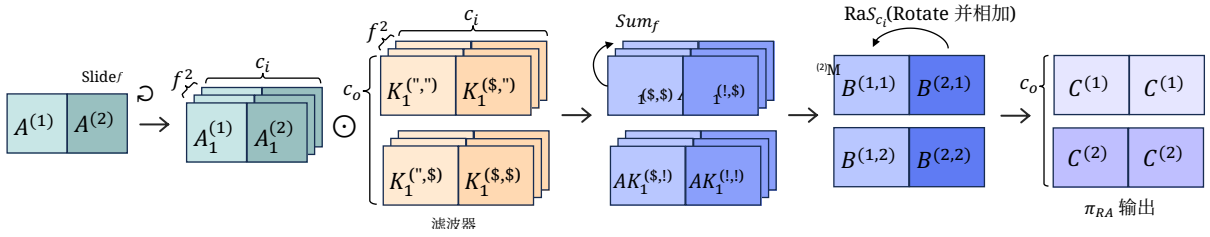
由于采用了高次多项式，先前工作的主要性能瓶颈 [11] 源于自举操作。相比之下，我们的分析表明，若采用 AESPA，自举在整个同态卷积神经网络推理时间中所占比例会很小，而卷积操作将成为执行时间的主要部分（见图5）。我们还分析发现，旋转操作（密文旋转）占据了卷积计算的大部分。因此，在这项工作中，我们通过提出卷积算法和打包方法，有效缓解同态卷积神经网络推理中的大量旋转开销，从而专注于提升卷积性能。

3. HyPHEN 构建

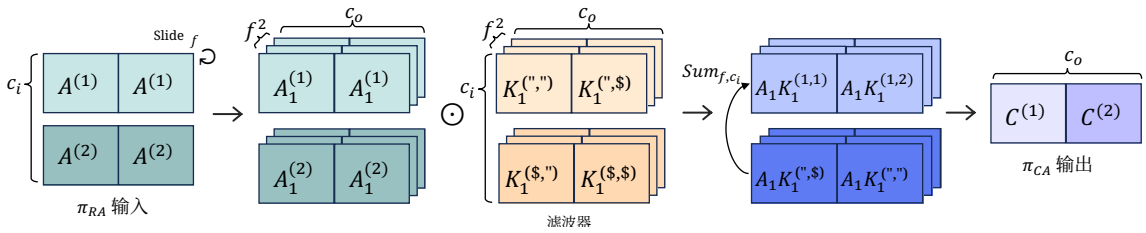
我们介绍 HyPHEN，这是一种同态卷积神经网络解决方案，重点在于减少内存占用以及旋转和自举等资源密集型同态运算的数量。我们提出了卷积算法和数据格式，以简化连续卷积之间的数据排列。我们的方法包含两种数据格式：通道对齐（CA）和副本对齐（RA）格式，分别记为 π_{CA} 和 π_{RA}。这两种格式均可形式化地描述为 $\pi_{CA} = \{C_a, H, W, R_g, C_g\}$, $\pi_{RA} = \{R_a, H, W, C_g, R_g\}$。在最外层维度中，$\pi_{CA}$ 将具有不同通道数（C_a）的图像对齐，方式与 [14, 18]，类似；而 π_{RA} 则按图像副本（R_a）进行对齐。下一节将介绍分别适用于这两种数据格式的两种不同卷积。

3.1. HyPHEN 卷积算法

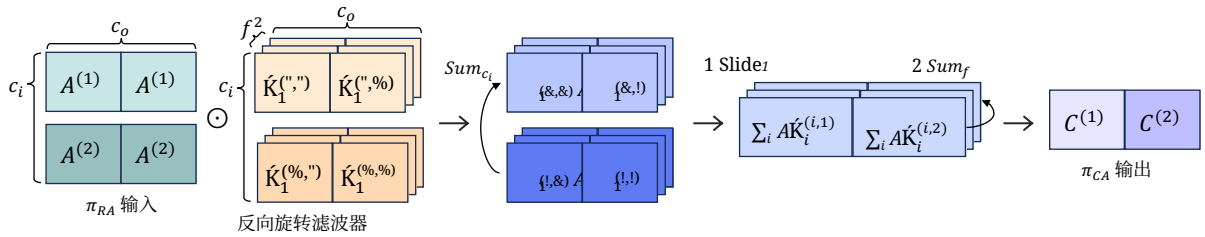
我们设计了两种卷积算法，分别以 π_{CA} 和 π_{RA}格式的输入密文为起点（见图 6）。为便于说明，图中假设最后两个维度的大小为 1，即不存在间隙；间隙打包则通过 RaS 和 IR 过程中的额外计算来处理，这将在第 3.2 节讨论。我们重点关注如何处理最外层的 C_a (R_a) 维度 π_{CA} (π_{RA}) 。



(a) 当 $|C_a| = c_i = c_o = 2$ 时，CAConv



(b) $RAConv_{Naive}$ 当 $|R_a| = c_i = c_o = 2$。



(c) $RAConv_{Reorder}$ 当 $|R_a| = c_i = c_o = 2$。

图 6：CAConv 和 RAConv 的两种变体。单个上标表示通道，上标对表示（输入通道，输出通道）。我们将 $M^{(a)}K^{(a,b)}$ 简记为 $MK^{(a,b)}$。

通道对齐卷积（CAConv）针对 π_{CA} 数据格式设计，与 MP-Conv$_{lc}$ 类似，但无需执行 IR 过程。回顾一下，在针对数据格式 {R,C$_a$,H,W,C$_g$} 的 MP-Conv$_{lc}$ 中，|C$_a$|（为简化起见记作 |C$_g$| = 1）的中间单输入单输出结果会累加到第一个通道的位置，其余槽位则包含无意义的值（图 3 中的 ##）。后续的 IR 会重新排列密文，以供下一次卷积使用。相比之下，通过将输入重复移至间隙内部（π_{CA} = {C$_a$,H,W, R$_g$,C$_g$}），我们可以将 C$_a$ 置于最外层维度。对该密文执行 RaS 会自动复制输出结果，因为同态旋转具有循环性。所得密文遵循 π_{RA} 数据格式，如图 6a 所示。使用 CAConv 后，我们的图像重新对齐无需任何旋转，但输出密文数量会增加 c$_o$。

复制对齐卷积（RAConv）更进一步，除此之外还消除了大部分 RaS 计算。RAConv 会对 π_{RA} 格式的输入依次执行 {*Slide$_f$*、*MulFilter*、*Sum$_{f,\ c_i}$*} 操作，并返回 π_{CA} 格式的结果。*Slide$_f$* 通过 f^2 次旋转扩展输入密文，*MulFilter* 则使用 MulPt 操作，将滑动后的密文与明文滤波器相乘。*Sum$_{f,\ c_i}$* 仅通过加法（AddCt）累加 $f^2 \cdot c_i$ 个密文的中间结果无需执行代价高昂的旋转操作。生成的密文遵循 π_{CA} 格式，因此后续的 CAConv 可以直接将其用作输入。

由于 RAConv 的输入密文数量（n_i）随输入通道数 c_i增加，*Slide$_f$* 需要额外的旋转开销（$c_i \cdot (f^2 - 1$次旋转)）。*Slide$_f$* 开销的增加削弱了 RaS 和 IR 通过减少旋转所带来的性能提升。我们通过对 RAConv 中的运算进行重排序来解决这一问题，从而显著降低 *Slide$_f$* 开销。

重排序 RAConv：我们观察到，只要将滤波器明文进行反向旋转，滑动与滤波器乘法的顺序就可以互换，因此将 RAConv 序列重排为 {MulFilter,Sum$_{c_i}$,Slide$_1$, Sum f }。图 7 和算法 2 描述了重排序 SISO 的过程。不同于通过一次旋转从单个输入密文生成 f ^{2} 个旋转后密文的 Slide$_f$，Slide$_1$ 通过单次旋转和加法将 f^{2}个密文合并为一个。通过重排序这些运算，我们可以像图 6c 所示，在累加 n$_i$个输入密文后再进行旋转，从而有效地将滑动所需的旋转次数从 c$_i$ · (f ^{2} − 1) 降至 (f ^{2} − 1)。

我们还提出了一种层间优化，旨在减少 CAConv 中密文所需的内存占用

表 2：同态卷积的成本。我们将我们的卷积与 [11] 在 $n_i = n_o = 1$ 设置下进行比较。（c_n：在 C_a 维度中的通道复用），（m：在 C_g 维度中的通道复用），（d：R 维度中的输入重复）。

方法	n_i	n_o	Slide	RaS	RaS_g	IR	IR_g
MP-卷积$_{lc}$ [11]	1	1	f^2-1	~~$mc_n d$~~ 对数(c^n)	~~$mc_n d$~~ log(m)	(麦克$_n$-1+日志$_2$(d))	
CAConv	1	$\frac{mc_n}{d}$	f^2-1	~~$mc_n d$~~ log(c^n)	~~$mc_n d$~~ log(m)	0	~~$mc_n d$~~ log(m)
RAConv$_{Naive}$	$\frac{mc_n}{d}$	1	$mc_n d$ (f^2-1)	0	log(m)	0	对数(m)
RAConv$_{Reorder}$	$\frac{mc_n}{d}$	1	f^2-1	0	log(m)	0	log(m)

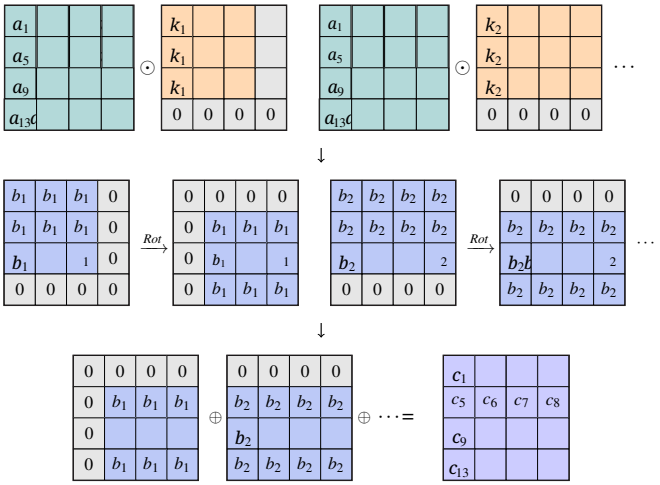


图 7：为单输入单输出重排序而进行逆向旋转的明文。

算法 2 RAConv 的重排序 SISO

输入 ct_i：输入密文，W'：逆向旋转的滤波器明文

输出密文$_o$：输出密文

1: // **MulFilter&Sum**$_{ci}$(密文$_i$, W)
2: **对于**$j_1 = 0, \ldots, f-1$ **执行**
3: **对于**$j_2 = 0, \ldots, f-1$ **执行**
4: $ct'[j_1, j_2]$ += 明文乘法(ct_i, $W'[j_1, j_2]$)
5: // **Slide**$_1$**&Sum**$_f$(密文′)
6: **对于**$j_1 = 0, \ldots, f-1$ **执行**
7: **对于**$j_2 = 0, \ldots, f-1$ **执行**
8: $r \leftarrow w_i(j_1 - f_2^{-1}) + (j_2 - f_2^{-1})$
9: ct_o += 密文旋转($ct'[j_1, j_2]$, $-r$)
10: **返回**ct_o

和 RAConv，如算法 3 所示。在同态卷积中，输入、输出和临时密文的数量由其数据格式决定。我们的优化采用积极的转发策略，旨在避免中间数据格式占用大量内存空间的情况（例如 ct_2,ct_3，见算法 3）。我们将 CAConv 和 RAConv 的输入数量分别记为 n_{CA} 和 n_{RA}，以避免融合这两种卷积时产生混淆。通过融合生成 ct_2 和 ct_3 的循环，并将它们合并为 ct_4，，我们会将每个密文从 ct_2,ct_3 转发至后续操作，直至获得 ct_4（见第 4—9 行）。我们的转发优化

算法 3 CAConv 和 RAConv 融合块 输入 ct_i：输入密文向量，W_{CA},W_{RA}：CAConv 和 RAConv 的权重明文，n_{CA}, n_{RA}：数据格式为 π_{CA}, π_{RA} 的密文数量，ct_k：临时密文向量 输出 ct_o：输出密文向量 1：对于 k = 0,. .. , n_{CA}-1 执行 2：ct_1[k] ← Slide$_f$(ct_i[k]) 3：对于 j = 0,. .. , n_{RA}-1 执行 4：对于 k = 0,. .. , n_{CA}-1 执行 5：ct_2[j, k] +=滤波器乘法与求和 f (ct_1[k],W_{CA}[j, k]) 6：ct_3[j] ← RaS$_{ci}$(ct_2[j, k]) 7：ct_3[j] ← 平方(ct_3[j]) 8：对于 l = 0,. .. , n_{CA}-1 执行 9：ct_4[l] += 滤波器乘法与求和 $_{ci}$(ct_3[j],W_{RA}[l, j]) 10：对于 k = 0,. .. , n_{CA}-1 执行 11：ct_o[k] ← Slide$_1$& 求和$_f$(ct_4[k]) 12：返回 ct_o

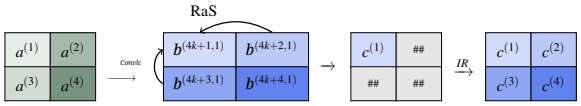
将密文数量保持为 $n_{CA} \cdot f^2$,，远小于 $n_{RA} \cdot f^2$。

3.2. 二维间隙打包的数据格式

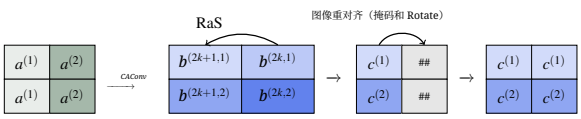
我们提出一种间隙打包方法，协同采用复制和通道复用，以降低重新打包过程的高昂开销。在最内侧的两个维度中，这种方法称为二维间隙打包，其中 π_{CA} 和 π_{RA} 的 C_g 与 R_g 分别表示复制和通道复用。与 MP-Conv$_{lc}$ 不同，后者为卷积保留单个 C_g；π_{CA} 和 π_{RA} 则协同发挥互补作用，使每次卷积都将 C_g (R_g) 转换为 R_g(C_g)。我们的关键观察是，引入这种异质性可以显著减少所需的旋转次数。具体而言，在 MP-Conv$_{lc}$ 中，间隙内的图像重对齐（IR$_g$）需要进行 O(g^2) 次旋转，以通过 C_g 保持间隙打包，其中 g 表示间隙宽度（高度）。相比之下，如图 8b 和图 8c 所示，采用我们的二维间隙打包时，IR 仅需沿 R_g 方向进行 O(log g^2) 次旋转。

3.3. 复杂度分析

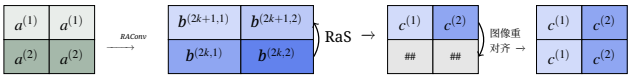
我们在表2中描述卷积算法旋转次数的复杂度。我们首先从基线 [11],开始，并将c_n、m 和 d 分别记为 $|C_a|$、$|C_g|$和 $|R|$，对应 MP*MP-Conv$_{lc}$*。在 CAConv 中，d 变为 $|R_g|$，因为我们将输入重复移至



(a) 使用多路复用打包的卷积



（b）二维间隙打包的卷积，数据格式 (R, C)



（c）二维间隙打包的卷积，数据格式（C，R）

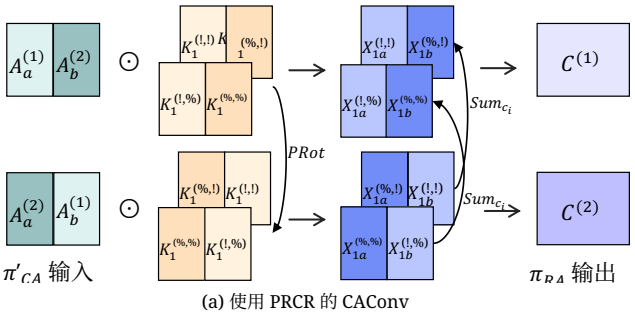
图8：间隙大小为 g=2 时，对图像单个像素进行卷积的过程。$a^{(j)}$ i 表示输入图像元组中第i个元素，它位于第j个通道中a。$b^{(mk+n,l)}$ 表示按步幅 m 累积的通道 $\sum_{k=1}^{c_i} {}^{m} b^{(mk+n,l)}$

间隙维度，然后 RAConv 对 CAConv 的输出进行处理。除了 CAConv 的 RaS 外，RaS 和图像重对齐不需要为我们的卷积执行任何旋转。如第 3.1节所述，除了间隙维度上的 CAConv 的 RaS 外，我们的卷积不需要任何旋转。通过重新组织滑动旋转的顺序，RAConv 的 $\text{Slide}_{Reorder}$ 所需的旋转次数比 RAConv_{Naive} 少$_{mcnd}\times$。我们还分别给出了保留二维间隙打包方法所需的旋转数量（RaS_g 和图像重对齐$_g$）。总体而言，我们的方法将 RaS 的旋转开销减半，并避免了图像重对齐的高旋转开销。

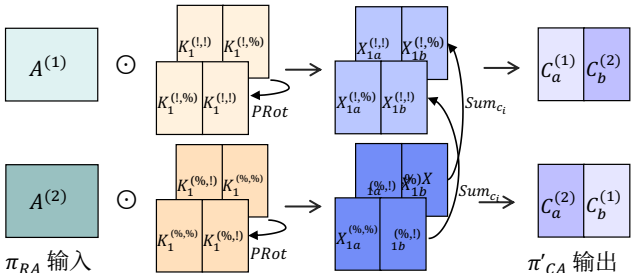
我们强调，n_i 和n_o 会直接影响激活函数和自举的总次数。例如，n_i 通常大于 RAConv 的n_i，从而增加了 π_{RA}上的激活函数或自举次数。与对整体性能影响较小的平方激活函数不同，同态卷积神经网络的性能严重依赖于自举次数。因此，我们会在密文数量最少时进行自举，尤其是在 π_{CA}格式下。此外，在端到端网络实现过程中，我们会针对每个模块微调（m，d），以实现旋转和自举总执行时间的最小化。关于（m，d）选择的深入性能分析见第4.5节。

3.4. 通过图像到密文的重排缩减明文大小

将 HCNN 扩展到高分辨率图像（例如，224×224 ImageNet 中的图像）时，HCNN 的内存占用会显著增加。权重明文中的这一增长尤为明显，每个滤波器元素占用片段大小为 w_ih_i 个槽位，如图 2c所示。总计，滤波器需要 $w_ih_if^2c_ic_o$ 个槽位来生成权重明文。例如，在 ImageNet 上运行 ResNet-18 时，



(a) 使用 PRCR 的 CAConv



(b) 使用 PRCR 的 RAConv

图 9：使用 PRCR 的简化卷积过程。对于此图，我们假设$c_i = c_o = 2$，但每个密文包含的图像数为 c_n，且滤波器尺寸 f 等于 1。密文中填入带下标a 和b 的内容，分别表示图像的上半部分和下半部分（例如，(a) 中的 π'_cA 包含单张图像，但每张图像均由通道图像的不同半部分组成）。PRCR 通过明文旋转（PRot）重复利用权重明文。

仅权重明文就占用了高达 364.8GB 的内存空间（见表 4）。这超出了单个尖端图形处理器的内存容量，而其内存容量受当前内存技术的限制。

我们提出一种名为 PRCR（通过图像到密文重排实现明文大小缩减）的优化方法，以缓解明文的数据膨胀。我们的方法是沿行方向将一幅图像拆分为多个图像分段。因此，CAConv 的数据格式调整为$\pi_{CA'} = \{C_a, S, H', W, R_g, C_g\}$，其中 S 表示子图像的维度，H′ 表示图像分段的行数维度。这种重排可将滤波器元素的片段大小缩减至图像分段大小。此外，这些片段按环形方式排列，使各部分可通过置换重复使用。图 9a 展示了在槽位=w_ih_i、f=1 和 c_i=c_o=2 的情况下采用 PRCR 的方法。CAConv 将单个权重明文 |S| 重复使用多次：与输入密文相乘后，再按片段大小进行旋转。随后，Sum_{c_i} 累加输入通道，恢复为图像大小的片段，并生成复制对齐密文。虽然 π_{RA} 保持不变，PRCR 会将权重明文重排为更小的片段，并将不同输出通道按环形方式排列。与 CAConv 类似，RAConv 通过乘法和旋转将权重明文 |S| 重复使用多次，并返回 $\pi_{CA'}$。

PRCR 可以与任何形式的打包技术独立结合使用，以减少明文内存。额外

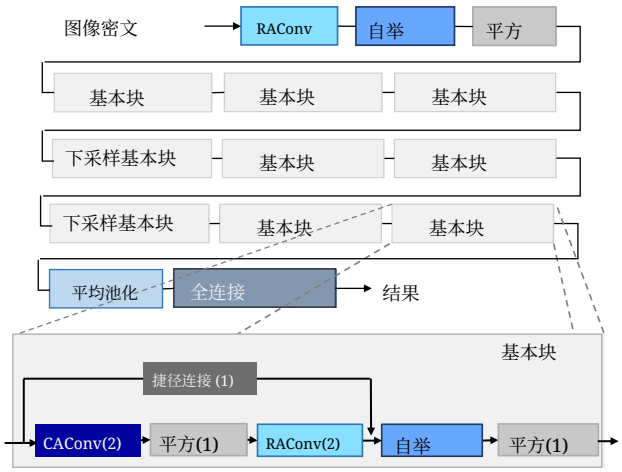


图10：基于HyPHEN构建的ResNet基本块。每个括号中标注了每个块消耗的层级。在下采样块中，逐点卷积被加入关键路径；否则，则加入简单的捷径连接。

PRot 的计算开销对整体性能的影响并不显著，因为如表 1 所示，PRot 的成本远低于其他全同态加密操作。对于我们的实现，我们对$|S| = 8$ ImageNet 使用该方法。然而，由于图形处理器内存能够满足 CIFAR-10 工作负载的需求，因此我们不对 CIFAR-10 应用 PRCR。

3.5. 使用HyPHEN构建ResNet

HyPHEN结合RAConv、二维间隙打包和PRCR来构建完整的卷积神经网络模型。图10 展示了在HyPHEN中实现的ResNet。确定运算放置位置时需要考虑三点。首先，由于涉及的密文数量较少，将自举放在RAConv之后而非CAConv之后更有效。其次，为了使捷径路径与主CAConv-RAConv路径的层级匹配，自举应放置在残差连接分岔之前或汇合之后。最后，最好尽可能在最低层级执行卷积层，因为旋转等全同态加密运算的复杂度与密文层级成正比。

综合以上各项，我们的 ResNet基本块总共消耗 6 个层级。每个块括号中的数字表示各层的层级消耗。CA-Conv 和 RAConv 使用二维间隙打包，SISO 和 IR 各消耗一个层级。对于卷积 $m > 1$，与掩码相乘需要消耗一个层级。我们采用 AESPA 进行激活运算，消耗一个层级。AESPA 是一个二次多项式，每个通道使用不同的系数。在推理过程中，我们将这些系数融合到相邻层中，从而得到简单的平方函数 x^2 用于激活。我们将自举后的密文层级（L' 见第 2.1 节）设为六，并在层级降至零时执行自举。

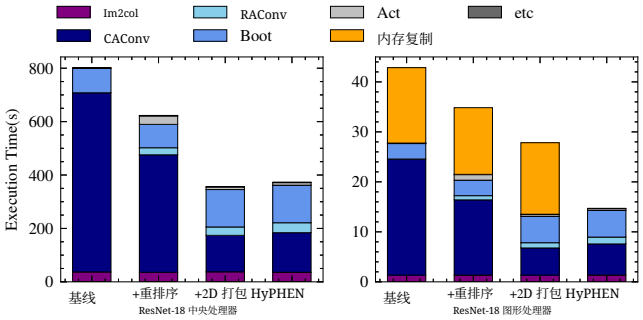


图11：ResNet-18（ImageNet）同态卷积神经网络在我们的中央处理器和图形处理器系统上的执行时间。

4. 评估

4.1. 实验设置

我们使用 RNS-CKKS 库 HEaaN [24]，分别在中央处理器和图形处理器环境中进行了 HCNN 推理。中央处理器系统配备两颗运行频率为 2.35GHz 的 AMD EPYC 7452 处理器（每个插槽 32 核）和 480GB 动态随机存取存储器。在同一系统中额外配备一块具有 80GB 内存的 NVIDIA A100 图形处理器，以开展图形处理器实验。在 HCNN 推理实验中，我们使用 CIFAR-10 [25]数据集上的 ResNet-20/32/44，以及 ImageNet [26] 数据集上的 ResNet-18。我们按照原论文，在常规监督设置下使用 PyTorch 和 AESPA 进行训练。我们采用 Kaiming 正态初始化方法对卷积层和全连接层进行初始化。训练完成后，我们应用融合技术，将批归一化（BN）的权重和偏置以及 AESPA 的系数与卷积层合并。我们的 RNS-CKKS参数满足 128位安全级别 [27]，其多项式次数为 $N = 2^{16}$，汉明重量为 192。表3 展示了 ResNet-20/32/44/18 卷积层中使用的参数。所有参数（c_i、c_o、w_i、w_o、f、s）均按照原始 ResNet论文 [28] 确定。执行时间从所有权重明文和输入密文加载到主机或设备内存后开始测量，到加密推理结果返回时结束。客户端的解密和加密过程不计入其中，因为它们并非 HCNN 推理中的关键瓶颈。

4.2. HyPHEN 各组件对性能的影响

为了分析 HyPHEN 各组件的有效性，我们逐步应用 CAConv 和 RAConv 交替并进行重排序（+重排序）、二维间隙打包（+ 2P）以及 PRCR（HyPHEN）。我们使用面向 ImageNet 的 ResNet-18 CPU/GPU 实现进行了测试。结果如图11所示。通过交替使用 CAConv 和 RAConv 并进行重排序，我们实现了 1。29×倍加速；在 CPU 上，二维间隙打包又带来了额外的 1。75×倍加速。重排序和二维间隙打包的优势也适用于 GPU，使计算时间（不包括内存访问延迟）加速了 2。05×

表 3：用于 CIFAR-10 的 ResNet-20/32/44 和用于 ImageNet 的 ResNet-18 的模型架构详情。步幅为 2 的下采样卷积记为 dsconv。滤波器尺寸为 1 的逐点卷积记为 pconv。

	ResNet-20/32/44							ResNet-18									
	第1层 conv	第2层 下采样卷积	逐点卷积	卷积	第3层 下采样卷积	逐点卷积	卷积	第1层 卷积	第2层 下采样卷积	逐点卷积	卷积	第3层 下采样卷积	逐点卷积	卷积	第4层 下采样卷积	逐点卷积	卷积
c_i	16	16	16	32	32	32	64	64	64	64	128	128	128	256	256	256	512
c_o	16	32	32	32	64	64	64	64	128	128	128	256	256	256	512	512	512
w_i	32	32	32	16	16	16	8	56	56	56	28	28	28	14	14	14	7
w_o	32	16	16	16	8	8	8	56	28	28	28	14	14	14	7	7	7
f	3	3	1	3	3	1	3	3	3	1	3	3	1	3	3	1	3
s	1	2	2	1	2	2	1	1	2	2	1	2	2	1	2	2	1

表 4：ImageNet 上 ResNet-18 的内存占用。 [11] 是针对 ImageNet 的扩展实现。基线为 [11] 采用 AESPA 增强，HyPHEN 则通过 PRCR 进一步改进。

实现	滤波器明文	评估密钥
[11]	285.1GB	69.7GB
基线	364.8GB	11.1GB
HyPHEN	45.6GB	11.1GB

。与此同时，PRCR 使 GPU 延迟降低了 1。89×，但由于增加了明文置换的计算，在 CPU 上略有减速（4.6%）。尽管 PRCR 将权重明文的总大小减少了 $s_i = 8$ 倍，但由于 CPU 系统的内存容量足以容纳整个工作集（见表 4），因此并未降低该系统的延迟。然而，对于 GPU 系统，PRCR 可使整个工作集装入 80GB 的图形处理器内存，从而通过消除运行时将数据从主机 CPU 复制到 GPU 的开销（内存复制）来降低延迟。总之，PRCR 能显著减少内存占用并降低延迟，尤其适用于内存受限的情况。

4.3. 执行时间分解

表5 展示了 ResNet-20/32/44 对单张 CIFAR-10 图像进行推理，以及 ResNet-18 对单张 ImageNet 图像进行推理时的运行时间。我们在图形处理器上实现的 ResNet-20/32/44 只需数秒即可完成运行。ResNet-20/32/44 的大部分推理时间都花在自举上；而对于 ResNet-18，推理时间中有 49.6%（中央处理器）和 51.7%（图形处理器）花在卷积上。这是因为 ResNet-18 的通道数是 ResNet-20/32/44 的四倍。表 5还表明，我们的 RAConv 替换了约一半的卷积层，有效缩短了卷积层的总运行时间。

4.4. 准确率

在表6中，我们测量了使用 Hy- PHEN 运行残差网络模型时，CIFAR-10 验证集的分类准确率。对于 ResNet-20/32/44，观察到的准确率下降几乎为零（$\leq 0.01\%$）。事实证明，HyPHEN 对准确率下降的鲁棒性优于 [11]，，后者在 CIFAR-10 上运行 ResNet-20/32/44 时，准确率下降了 0.09% 至 0.21%。准确率下降的差异可以通过原始网络是否按原样执行（使用 AESPA），还是进行了近似处理（使用 ReLU近似）来解释。即使对于更宽的网络（ResNet-18）和更大的数据集（ImageNet），HyPHEN 的准确率也与骨干网络准确率相同。

4.5. 参数研究

我们开展了参数研究，以确定 ResNet 模型的最优二维间隙打包设置，从而最大限度地降低延迟。参数 $(m,\ d)$ 的选择至关重要，因为它决定了网络运行所需的旋转和自举次数，进而影响执行时间。表 7 列出了具有代表性的 (m, d) 实例，以及相应的操作次数和执行时间。为简洁起见，我们仅展示 CAConv 的 $(m,\ d)$，因为在 RAConv 中 m 和d 会互换；此外，我们省略了 ResNet-32/44，因为 ResNet-20/32/44 具有相同的最优点。

在 ResNet-20, [11] 中，由于第一层输入张量（$32 \times 32 \times 16$）的大小小于密文槽（2^{15}），因此采用输入重复。为避免输入重复，我们在 ResNet-20 中以 (m, d) =(1, 2) 作为起始设置，而不使用输入重复。在 ResNet-18 中，我们采用默认的 (m, d) =(1, 1) 作为起始设置。发生输入重复时，提高 d 的效率更高；但进一步增大 m 或 d 并不会带来更好的性能，因为这会导致更多自举，如我们所提出的架构所示（见图10）。随着输入密文经过下采样层，$\mathrm{m} \cdot \mathrm{d}$ 会增至原来的四倍，而中间张量的大小则减半。二维间隙打包在每个下采样层都将 m 和 d 加倍，表现最优，对应于最少自举的设置。在 ResNet-18 中，最少自举的设置比 MP-CAConv 表现更好。不过，得益于二维间隙打包的灵活性，我们还能找到效率更高的设置。最优设置比自举次数最少的设置多进行 27 次自举，但少进行 20641 次旋转。二维间隙打包方法有助于平衡旋转和自举的数量，从而获得更好的性能。

表 5：在中央处理器和图形处理器上，使用 ResNet20/32/44 对单张 CIFAR-10 图像进行推理，以及使用 ResNet-18 对单张 ImageNet 图像进行推理时，HyPHEN 的推理时间。由于全连接层和池化层的执行时间极短，我们将它们归入“其他”项。对于 ResNet-18，我们使用 Im2col 实现初始下采样卷积，其时间也计入“其他”项。

执行时间（秒）	中央处理器（64 个线程）				GPU			
	ResNet-20	ResNet-32	ResNet-44	残差网络-18[1]	ResNet-20	ResNet-32	ResNet-44	ResNet-18[2]
CAConv	12.85	19.39	24.42	148.33	0.46	0.66	0.87	6.22
RAConv	1.68	3.02	4.01	37.16	0.06	0.09	0.13	1.37
自举	21.40	34.23	45.90	140.57	0.83	1.32	1.83	5.32
激活	1.41	2.41	3.27	9.65	0.05	0.09	0.12	0.36
其他	0.23	0.30	0.44	37.70	<0.01	<0.01	0.01	1.42
总计	37.57 ± 0.7	59.35 ± 0.9	78.04 ± 0.8	373.41 ± 3.9	1.40 ± 0.04	2.17 ± 0.04	2.96 ± 0.04	14.69 ± 0.01

[1, 2] ResNet-18 实验在 ImageNet 数据集上进行。

表 6：ResNet 模型的未加密分类准确率（骨干网络）与 HCNN 分类准确率（HyPHEN）对比。其中，ResNet-20/32/44 的结果为 CIFAR-10 的准确率，ResNet-18 的结果为 ImageNet 分类准确率。

Top-1 准确率（%）	ResNet-20	ResNet-32	ResNet-44	ResNet-18
骨干网络	92.18	93.36	94.04	65.25
HyPHEN	92.17	93.35	94.08	65.25

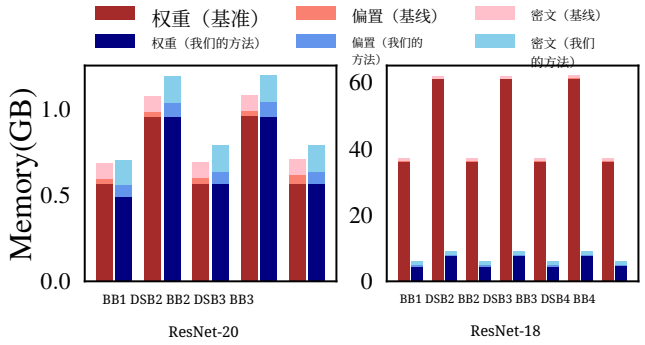


图 12：残差块中各对象的总内存大小（GB）。我们将下采样块和基本块分别缩写为 DSB 和 BB，并在后面附上各自的层号。权重明文、偏置明文和输入密文构成了各残差块的内存占用。Base 和 Ours 分别表示基线和 HyPHEN。

4.6. 训练细节

本文使用的模型均采用 PyTorch 进行训练 [29]。对于 ResNet-18 和 ResNet-20，我们的训练设置与 AESPA 基本相同；具体而言，网络训练 200 轮，使用随机梯度下降优化器。我们还像 [16] 中一样使用软标签，以提高准确率。对于 ResNet-32 和 ResNet-44，我们采用知识蒸馏 [30] 来提高准确率，并使用准确率分别为 93.4% 和 94.1% 的预训练 ResNet-32/44 作为教师模型。我们还额外采用 l_2 损失（$L_{kd} = \|f_t - f_s\|_2^2$）进行蒸馏，并使用随机梯度下降优化器训练 240 轮。

4.7. 内存占用分析

同态卷积神经网络的内存容量需求取决于全同态加密参数和数据表示（例如打包方案）。在全同态加密中，数据在编码和加密过程中会扩张。生成的明文和密文通常比原始消息大几个数量级。表8展示了两种全同态加密参数设置下密文、明文和评估密钥的实际大小。我们将 [31] 中引入的 RNS 分解数记为 Dnum。给定分圆多项式环的次数 N，较大的 dnum 会增大密文的最大层级 L。我们假设$\mathtt{Set}_{lc}$ 是 [11]中使用的设置。由于 [11] 使用高次多项式近似 ReLU 激活，$\mathtt{Set}_{lc}$采用最大的 dnum，以获得$L' = 16$。$\mathtt{Set}_{hyp}$ 是本工作的参数集。采用 AESPA 后，我们可以选择较小的 RNS-CKKS 参数（$L' = 6$），因为激活会消耗一个层级。某些全同态加密运算（如 MulCt、Rotate 和 Conjugate）需要执行密钥切换过程。Evk 表示此过程中使用的公求值密钥。在 $\mathtt{Set}_{lc}$ 和$\mathtt{Set}_{hyp}$中，单个 Evk 的大小分别为 1,056MB 和 176MB。自举需要一个用于 MulCt 的重线性化密钥、一个共轭密钥和 48 个旋转密钥。卷积时会加载 Slide 常用的旋转密钥。例如，在$\mathtt{Set}_{lc}$ 和$\mathtt{Set}_{hyp}$中，加载 ResNet-18 的 66 个唯一 EVK 分别占用 69.7GB 和 11.1GB。IR 中使用的其他不规则旋转密钥不会加载；相反，这些旋转索引会利用已加载的密钥索引合成。

确定全同态加密参数后，打包方案决定了运行每个残差网络模块所需的密文和明文数量。在基于单输入单输出的同态卷积神经网络滤波器中，滤波器明文的大小会增加 $w_i h_i$ 倍，因为每个滤波器元素都会复制到与输入图像相同的大小。权重明文的槽位数为$w_i h_i f^2 c_i c_o$。因此，权重明文的内存占用远大于密文；密文仅需$w_i h_i c_i d$ 个槽位。图12展示了各卷积神经网络残差块中密文和明文的总内存容量。对于 ResNet-20，与基线相比，我们的实现带来的内存容量开销最高达 14.75%。这一开销是由于

表7：采用不同（m,d）参数和打包策略的残差网络实例的运行时间。† 我们将未加载 EVK 的旋转索引分解为已加载 EVK 的旋转索引。分解后的旋转次数（即有效旋转）表示为 eff.total。

模型	打包	(m,d) L1	L2	L3	L4	卷积旋转 SISO	RaS	IR	总计	有效总数†	Boot	CPU运行时间(s)
ResNet-20	基线					152	924	800	1876	3638	10	68.58 ± 1.0
	最优	(1,2)	(2,4)	(4,8)	-	152	580	187	919	1002	10	**37.57 ± 0.7**
	最小旋转	(1,2)	(1,8)	(2,16)	-	240	407	142	789	881	15	44.51 ± 0.5
ResNet-18	基线					536	32384	4669	37589	43672	38	802.08 ± 7.6
	最小启动	(1,1)	(4,1)	(16,1)	(64,1)	536	17920	9544	28000	30072	38	623.11 ± 12.3
	最优	(1,1)	(2,2)	(4,4)	(8,8)	1024	4512	1823	7359	9095	65	**356.97 ± 5.9**

表 8：全同态加密参数设置。dnum 调整为支持 16，并满足 集合$_{lc}$ 和集合$_{hyp}$ 所需的层级数 6。每个密文和明文的内存大小均以自举刷新层级时的大小表示。

	L+1	L′	Dnum	密文 (MB)	明文 (MB)	评估密钥 (MB)
集合$_{lc}$	32	16	32	17	8.5	1056
集合$_{hyp}$	24	6	6	10	5	168

使用 CAConv 和 RAConv 时，中间密文和偏置明文的数量有所增加；对于 CIFAR-10，asHyPHEN 不会应用 PRCR。然而，对于 ResNet-18，与基线相比，我们的实现可减少 6.15-6.81× 的内存占用。

5. 相关工作

5.1. 基于同态加密的隐私保护机器学习

人们已探索多种同态加密（HE）变体，用于隐私保护机器学习（PPML）。早期文献 [8–10, 12, 32] 采用分级同态加密（LHE），但该方案无法管理密文误差，且只能执行有限次数的运算，因此仅适用于浅层网络。这些早期研究中的卷积仍可用于全同态加密（FHE），但效率不如专为 FHE 设计的卷积。例如，[12]提出了一种基于分块的通用张量框架，在特定的同态加密参数设置和图像尺寸下，可作为基于单输入单输出的框架的替代方案。我们发现，对于基于全同态加密的卷积神经网络，分块通常会带来过多的自举，因此主要将以往的 FHE 卷积神经网络实现作为比较对象。相比之下，SHE [33]基于 TFHE 方案实现了卷积神经网络，[34] 并通过查表计算非线性函数，但每次运算的延迟都很长。利用 RNS-CKKS，[11, 15] 研究人员展示了在 CIFAR-10 上实现残差网络的方法，并采用高阶 ReLU 多项式逼近。[11]报告了单线程 ResNet-20 实现，其通过高效利用密文槽位，将运行时间缩短至 2271 秒。同期的一项研究 [35] 采用系数打包密文，将数值作为环多项式的系数。该方法减少了卷积中的旋转次数，但灵活性有限，因为一次卷积必须与一次自举配对。

5.2. 混合隐私保护机器学习

为解决隐私保护机器学习中的高计算复杂度问题，同态加密-多方安全计算混合隐私推理协议作为一种替代方案受到关注。在该协议中，客户端辅助的多方安全计算负责处理 ReLU 等非线性函数，而同态加密运算则用于计算线性函数。同态加密-多方安全计算协议 [14, 36–38]已取得显著进展。Cheetah [38] 提出了一种高效的打包方案，消除了旋转操作。然而，由于安全模型不同，要公平比较同态卷积神经网络与同态加密-多方安全计算协议颇具挑战。同态加密-多方安全计算协议还会泄露网络架构。混合方案减轻了服务器的计算负担，但其最佳性能依赖于以下假设：1）网络通信持续进行；2）客户端具备相当的计算能力。相比之下，全同态加密仅需通过简洁的通信传输和接收较小的输入与输出密文。与普遍认为同态加密-多方安全计算协议明显快于基于全同态加密的实现的看法相反，我们的结果表明，其性能与以往的同态加密-多方安全计算协议相当。

5.3. 全同态加密硬件加速

基于全同态加密的应用前景广阔，尤其是结合硬件加速时。为响应 IT 行业对兼具实际服务质量（QoS）的隐私保护服务的需求，先前的研究 [39, 40] 从计算机体系结构的角度介绍并分析了全同态加密运算的特征。这些充分的分析促成了适用于多种硬件平台的解决方案，例如中央处理器 [41], 图形处理器 [42, 43], 现场可编程门阵列 [44, 45], 和专用集成电路 [46–48]。虽然我们的同态卷积神经网络实现在中央处理器和图形处理器平台上展现了良好性能，但专用的全同态加密加速器可带来 2–3 个数量级的性能提升。

6. 讨论

在我们的实现 HyPHEN 中，我们引入了多项重要改进，以提升同态卷积神经网络的性能。首先，我们采用了源自 AESPA 的低次多项式激活函数。此外，我们基于 RNS-CKKS 的图形处理器实现开发了同态卷积神经网络。最重要的是，我们提出了新算法，以应对同态卷积神经网络的两大关键挑战：计算复杂度和内存占用。

6.1. 计算复杂度

我们对以往的同态卷积神经网络实现进行了全面的瓶颈分析，发现全同态加密旋转是导致性能受限的原因。我们发现，大部分旋转来自密文内通道求和（RaS），以及两个卷积层之间的数据格式调整（图像重对齐，IR）。为了解决这一问题，我们开发了新算法 RAConv 和 CAConv，无需图像重对齐即可实现加密卷积，并有效减少 RaS 旋转。我们还提出了一种混合打包方法，能够高效处理步幅卷积引入的间隙，同时尽量减少所需的旋转。我们的实现 HyPHEN 将执行时间大幅缩短，从几十分钟降至仅几秒。

6.2. 内存占用

除了全同态加密方案中的加密和编码过程导致的内存扩展外，我们还发现了另一个显著的内存扩展来源：打包方法中的数据重复。具体而言，尽管明文的内存扩展低于密文，但由于每个权重元素占用的槽位数与输入图像大小相同，明文仍会产生额外的内存扩展。我们在 ImageNet 数据集上的实验表明，从主机内存加载权重明文会显著影响 ResNet-18 等卷积神经网络模型的性能。加速器配备的高带宽内存（HBM）可能无法容纳整个模型权重。我们提出的解决方案 PRCR 通过引入一种新颖的数据排列方式，消除了为匹配输入图像大小而复制每个权重元素的需求，从而解决了这一问题。对于 ResNet-50 这类更大的模型，PRCR 实现的内存缩减比例更为显著。在使用专用加速器时，内存占用问题尤为关键。据报道，这类加速器的速度比 GPU 快一到两个数量级，但仍采用相同的 HBM 技术，因此容量限制也相同。

6.3. 局限性

尽管 HyPHEN 在推进同态卷积神经网络方面取得了显著进展，但仍有若干局限性值得考虑。虽然执行时间大幅缩短，但推理仍需数秒，这对于在现实场景中部署而言仍颇具挑战。此外，随着图像分类模型不断发展并追求更高的准确率，其所需的内存容量也随之增加。先进的内存占用缩减算法有望进一步推动隐私保护机器学习的广泛应用。

7. 结论

本文提出了 HyPHEN，这是一种高效的基于全同态加密的卷积神经网络（HCNN）隐私推理方案。通过结合两种卷积方法与重排序和二维间隙打包，可显著减少卷积中的同态旋转次数，从而实现快速推理。此外，PRCR 使 HyPHEN 能够降低高分辨率图像分类任务的内存占用，这对于内存受限设备尤其有益。我们在 CPU 系统上对 HyPHEN 进行的实验表明，与 ResNet-20/18 中先前最先进的算法基线相比，延迟分别降低了 1.83×/2.15×。借助 GPU 加速，HyPHEN 在 CIFAR-10 上运行 ResNet-20/32/44 的执行时间分别为 1.40 秒/2.17 秒/2.96 秒；我们还首次展示了在 14.69 秒内完成 ImageNet 上 ResNet-18 的 HCNN 推理。我们通过 HyPHEN 展示了利用全同态加密实现隐私推理的实用性：它在提供合理执行时间的同时，使客户端能够受益于简洁的计算与通信过程。

参考文献

[1] 欧盟。《欧盟第2016/679号条例》，欧洲议会和理事会，2016年。[2] 美国卫生与公众服务部。《1996年健康保险流通与责任法案》，1996年。[3] Nishant Kumar、Mayank Rathee、Nishanth Chandran、Divya Gupta、Aseem Rastogi和Rahul Sharma。Cryptflow：安全的TensorFlow推理。载于*2020*年*IEEE*安全与隐私研讨会，*SP 2020*，美国加利福尼亚州旧金山，*2020*年*5*月*18*日至*21*日，2020年。[4] William Bowditch、Will Abramson、William J. Buchanan、Nikolaos Pitropakis和Adam J. Hall。利用同态加密实现隐私保护监控方法。载于第六届国际信息系统安全与隐私会议论文集，*ICISSP 2020*，马耳他瓦莱塔，*2020*年*2*月*25*日至*27*日。SCITEPRESS，2020年。[5] Craig Gentry。利用理想格实现全同态加密。载于第*41*届*ACM*计算理论研讨会年会论文集，*STOC 2009*，*2*，*2009*年，2009年*31*，美国马里兰州贝塞斯达，*5*月至*6*月。[6] Andrew Chi-Chih Yao。安全计算协议（扩展摘要）。载于第*23*届计算机科学基础年会研讨会，美国伊利诺伊州芝加哥，*1982*年*11*月*3*日至*5*日。IEEE计算机学会，1982年。[7] Victor Costan和Srinivas Devadas。英特尔 SGX详解。*IACR*密码学电子印本文档库，2016年。[8] Ran Gilad-Bachrach、Nathan Dowlin、Kim Laine、Kristin E. Lauter、Michael Naehrig和John Wernsing。Cryptonets：将神经网络应用于加密数据，实现高吞吐量和高准确率。载于第*33*届国际机器学习会议论文集，*ICML 2016*，48，2016年*19 24*，*2016*年，美国纽约州纽约市，*6*月-第 . 卷。[9] Alon Brutzkus、Ran Gilad-Bachrach和Oren Elisha。低延迟隐私保护推理。载于第*36*届国际机器学习会议论文集，*ICML 2019*，*2019*年*6*月*9*日至*15*日，美国加利福尼亚州长滩，2019年。[10] Roshan Dathathri、Blagovesta Kostova、Olli Saarikivi、Wei Dai、Kim Laine和Madan Musuvathi。EVA：用于高效同态计算的加密向量算术语言与编译器。载于第*41*届*ACM SIGPLAN*编程语言设计与实现国际会议论文集，*PLDI 2020*，英国伦敦，*2020*年*6*月*15*日至*20*日，2020年。[11] Eunsang Lee、Joon-Woo Lee、Junghyun Lee、Young-Sik Kim、Yongjune Kim、Jong-Seon No和Woosuk Choi。利用复用并行卷积，在全同态加密下实现低复杂度深度卷积神经网络。载于国际机器学习会议，*ICML 2022*，*2022*年*7*月*17*日至*23*日，美国马里兰州巴尔的摩，2022年。[12] Ehud Aharoni、Allon Adir、Moran Baruch、Nir Drucker、Gilad Ezov、Ariel Farkash、Lev Greenberg、Ramy Masalha、Guy Moshkowich、Dov Murik等。Helayers：用于加密数据上大型神经网络的分块张量框架。*arXiv*电子印本，页码arXiv–2011，2020年。[13] Jung Hee Cheon、Andrey Kim、Miran Kim和Yong Soo Song。面向近似数算术的同态加密。载于

密码学进展——*ASIACRYPT 2017*——密码学与信息安全理论及应用第*23*届国际会议，香港，中国，*2017*年*12*月*3*日至*7*日，会议论文集（第一部分），第10624卷，2017年。[14] Chiraag Juvekar、Vinod Vaikuntanathan和Anantha P. Chandrakasan。GAZELLE：用于安全神经网络推理的低延迟框架。载于第*27*届*USENIX*安全研讨会，*USENIX Security 2018*，美国马里兰州巴尔的摩，*2018*年*8*月*15*日至*17*日，2018年。[15] Joon-Woo Lee、Hyungchul Kang、Yongwoo Lee、Woosuk Choi、Jieun Eom、Maxim Deryabin、Eunsang Lee、Junghyun Lee、Donghoon Yoo、Young-Sik Kim和Jong-Seon No。利用全同态加密对深度神经网络进行隐私保护机器学习。*IEEE Access*，10：30039–30054，2022年。[16] Jaiyoung Park、Michael Jaemin Kim、Wonkyung Jung和Jung Ho Ahn。AESPA：用于快速隐私推理、保持准确率的低次多项式激活函数。*CoRR*，2022年。[17] Jung Hee Cheon、Kyoohyung Han、Andrey Kim、Miran Kim和Yongsoo Song。近似同态加密的全RNS变体。载于密码学专题——*SAC 2018*——第*25*届国际会议，加拿大艾伯塔省卡尔加里，*2018*年*8*月*15*日至*17*日，修订版精选论文，2018年。[18] Junghyun Lee、Eunsang Lee、Joon-Woo Lee、Yongjune Kim、Young-Sik Kim和Jong-Seon No。对同态加密数据上的卷积神经网络进行精确近似。*CoRR*，abs/2105.10879，2021年。[19] Takumi Ishiyama、Takuya Suzuki和Hayato Yamana。利用同态加密和近似激活函数实现高准确率CNN推理。载于*2020*年*IEEE*大数据国际会议（*IEEE BigData 2020 13*，*2020*年，2020年*10*），美国佐治亚州亚特兰大，*12*月-。[20] Hervé Chabanne、Amaury de Wargny、Jonathan Milgram、Constance Morel和Emmanuel Prouff。深度神经网络上的隐私保护分类。*IACR*密码学电子印本文档库，2017年。[21] Srinath Obla、Xinghan Gong、Asma Aloufi、Peizhao Hu和Daniel Takabi。用于深度神经网络同态求值的有效激活函数。*IEEEAccess*，8，2020年。[22] Ehsan Hesamifard、Hassan Takabi和Mehdi Ghasemi。加密数据上的深度神经网络分类。载于第九届*ACM*数据与应用安全和隐私会议论文集，*CODASPY 2019*，美国得克萨斯州理查森，*2019*年*3*月*25*日至*27*日，2019年。[23] Patricia Thaine、Sergey Gorbunov和Gerald Penn。加密数据上激活函数的高效求值。载于*2019*年*IEEE*安全与隐私研讨会，*SP Workshops 2019*，美国加利福尼亚州旧金山，*2019*年*5*月*19*日至*23*日，2019年。[24] CryptoLab。HEaaN库，2022年。[25] Alex Krizhevsky和Geoffrey Hinton。从微小图像中学习多层特征。2009年。[26] Olga Russakovsky、Jia Deng、Hao Su、Jonathan Krause、Sanjeev Satheesh、Sean Ma、Zhiheng Huang、Andrej Karpathy、Aditya Khosla、Michael S. Bernstein、Alexander C. Berg和Li Fei-Fei。ImageNet大规模视觉识别挑战。《国际计算机视觉杂志》，115(3)，2015年。[27] Jung Hee Cheon、Minki Hhan、Seungwan Hong和Yongha Son。针对稀疏三元秘密LWE的对偶攻击与中间相遇攻击的混合攻击。*IEEEAccess*，2019年。[28] Kaiming He、Xiangyu Zhang、Shaoqing Ren和Jian Sun。用于图像识别的深度残差学习。载于*2016*年*IEEE*计算机视觉与模式识别会议，*CVPR 2016*，美国内华达州拉斯维加斯，*2016*年*6*月*27*日至*30*日，2016年。[29] Adam Paszke、Sam Gross、Francisco Massa、Adam Lerer、James Bradbury、Gregory Chanan、Trevor Killeen、Zeming Lin、Natalia Gimelshein、Luca Antiga、Alban Desmaison、Andreas Kopf、Edward Yang、Zachary DeVito、Martin Raison、Alykhan Tejani、Sasank Chilamkurthy、Benoit Steiner、Lu Fang、Junjie Bai和Soumith Chintala。PyTorch：一种命令式风格的高性能深度学习库。载于神经信息处理系统进展第*32*卷，第8024–8035页。2019年。[30] Geoffrey E. Hinton、Oriol Vinyals和Jeffrey Dean。提炼神经网络中的知识。*CoRR*，2015年。[31] Kyoohyung Han和Dohyeong Ki。改进近似同态加密的自举。*IACR*密码学电子印本文档库，2019年。[32] Qian Lou和Lei Jiang。HEMET：同态加密友好型隐私保护移动神经网络架构。载于第*38*届国际机器学习会议论文集，*ICML 2021*，*2021*年*7*月*18*日至*24*日，线上活动，2021年。[33] Qian Lou 和 Lei Jiang。SHE：一种用于加密数据的快速、准确的深度神经网络。载于神经信息处理系统进展第*32*卷：神经信息处理系统年度会议*2019*，*NeurIPS 2019*，*2019*年*12*月*8*日至*14*日，加拿大不列颠哥伦比亚省温哥华，2019年。[34] Ilaria Chillotti、Nicolas Gama、Mariya Georgieva 和 Malika Iz- abachène。TFHE：基于环面的快速全同态加密。密码学杂志，33(1)，2020年。[35] Dongwoo Kim 和 Cyril Guyot。采用全同态加密优化隐私保护 CNN 推理。*IEEE* 信息取证与安全汇刊，18：2175–2187，2023年。[36] Pratyush Mishra、Ryan Lehmkuhl、Akshayaram Srinivasan、Wenting Zheng 和 Raluca Ada Popa。Delphi：一种面向神经网络的密码推理服务。载于第*29*届 *USENIX* 安全研讨会，*USENIX Security 2020*，*2020*年*8*月*12*日至*14*日，2020年。[37] Deevashwer Rathee、Mayank Rathee、Nishant Kumar、Nishanth Chandran、Divya Gupta、Aseem Rastogi 和 Rahul Sharma。Cryptflow2：实用的双方安全推理。载于*CCS '20*：*2020*年 *ACM SIGSAC* 计算机与通信安全会议，线上活动，美国，*2020*年*11*月*9*日至*13*日，2020年。[38] Zhicong Huang、Wen-jie Lu、Cheng Hong 和 Jiansheng Ding。Cheetah：精简快速的安全双方深度神经网络推理。载于 Kevin R. B. Butler 和 Kurt Thomas 编，第*31*届 *USENIX* 安全研讨会，*USENIX Security 2022*，*2022*年*12*月，2022年*10*月，美国马萨诸塞州波士顿，*8*月-。[39] Wonkyung Jung、Eojin Lee、Sangpyo Kim、Jongmin Kim、Namhoon Kim、Keewoo Lee、Chohong Min、Jung Hee Cheon 和 Jung Ho Ahn。通过以体系结构为中心的分析和优化加速全同态加密。*IEEE Access*，2021年。[40] Sangpyo Kim、Wonkyung Jung、Jaiyoung Park 和 Jung Ho Ahn。加速 GPU 上可自举同态加密的数论变换。载于*IEEE* 国际工作负载特征分析研讨会，*IISWC 2020*，中国北京，*2020*年*10*月*27*日至*30*日，第264–275页。IEEE，2020年。[41] Fabian Boemer、Sejun Kim、Gelila Seifu、Fillipe D. M. de Souza 和 Vinodh Gopal。Intel HEXL：使用 Intel AVX512-IFMA52 加速同态加密。载于*WAHC '21*：第*9*届加密计算与应用同态密码学研讨会论文集，线上活动，韩国，*2021*年*11*月*15*日，2021年。[42] Wonkyung Jung、Sangpyo Kim、Jung Ho Ahn、Jung Hee Cheon 和 Younho Lee。通过结合 GPU 的以存储器为中心的优化，使全同态加密中的自举速度提升100倍以上。*IACR* 密码学硬件与嵌入式系统汇刊，2021年。[43] Shengyu Fan、Zhiwei Wang、Weizhi Xu、Rui Hou、Dan Meng 和 Mingzhe Zhang。TensorFHE：利用通用图形处理器实现对加密数据的实用计算。载于*IEEE* 高性能计算机体系结构国际研讨会，*HPCA 2023*，加拿大魁北克省蒙特利尔，*2023*年*2*月*25*日至*3*月*1*日，2023年。[44] M. Sadegh Riazi、Kim Laine、Blake Pelton 和 Wei Dai。HEAX：一种用于加密数据计算的体系结构。载于*ASPLOS '20*：编程语言与操作系统体系结构支持会议，瑞士洛桑，*2020*年*3*月*16*日至*20*日，2020年。[45] Rashmi Agrawal、Leo de Castro、Guowei Yang、Chiraag Juvekar、Rabia Tugce Yazicigil、Anantha P. Chandrakasan、Vinod Vaikuntanathan 和 Ajay Joshi。FAB：一种用于可自举全同态加密的基于 FPGA 的加速器。载于*IEEE* 高性能计算机体系结构国际研讨会，*HPCA 2023*，加拿大魁北克省蒙特利尔，*2023*年*2*月*25*日至*3*月*1*日。IEEE，2023年。[46] Sangpyo Kim、Jongmin Kim、Michael Jaemin Kim、Wonkyung Jung、John Kim、Minsoo Rhu 和 Jung Ho Ahn。BTS：一种用于可自举全同态加密的加速器。载于*ISCA '22*：第*49*届年度计算机体系结构国际研讨会，美国纽约州纽约市，*2022*年*6*月*18*日至*22*日，2022年。[47] Jongmin Kim、Gwangho Lee、Sangpyo Kim、Gina Sohn、Minsoo Rhu、John Kim 和 Jung Ho Ahn。Ark：一种全同态加密加速器，支持运行时数据生成和操作间密钥复用。载于*2022*年第*55*届 *IEEE/ACM* 微体系结构国际研讨会（*MICRO*），第1237–1254页，2022年。[48] Nikola Samardzic、Axel Feldmann、Aleksandar Krastev、Nathan Manohar、Nicholas Genise、Srinivas Devadas、Karim Eldefrawy、Chris Peikert 和 Daniel Sánchez。CraterLake：一种用于在加密数据上高效执行无界计算的硬件加速器。载于*ISCA '22*：第*49*届年度计算机体系结构国际研讨会，美国纽约州纽约市，*2022*年*6*月*18*日至*22*日，2022年。

A. 符号说明

表9 汇总了本文中使用的符号及其说明。

表 9：所用符号及其说明。

符号	说明
L	最大层级
L'	乘法层级
ℓ	当前层级
n_i, n_o	通过卷积的输入和输出密文数量
g	间隙宽度（高度）
s	卷积步长
pad	卷积零填充
m, d	间隙打包配置——一对复用通道和副本
Dnum	分解数 [31]
Evk	求值密钥，用于乘法或旋转。