同态明文-密文矩阵运算及其应用

刘洋 [1]，杨林翰 [1]，陈经纬 [2,3]，吴文渊 [2,3]，冯勇 [2,3]

（1. 重庆交通大学信息科学与工程学院，重庆 400074；2. 中国科学院重庆绿色智能技术研究院 自动推理与认知重庆市重点实验室，重庆 4000714；3. 中国科学院大学重庆学院，重庆 400714）

摘　要：支持单指令多数据操作的同态加密方案能有效提高密文计算的均摊效率，但密文结构导致矩阵运算复杂度高。在许多应用中，采用明文-密文矩阵操作可以在确保安全的同时实现隐私计算。基于此，提出一个适用于任意维数的明文-密文矩阵乘法方案。该方案通过明文矩阵编码和密文矩阵维数变换等步骤计算得到密文结果。与已知最好的 Jiang 等(CCS 2018) 所提的密文方阵乘法算法相比，所提方案支持任意维数的矩阵乘法，并支持矩阵连乘；理论分析和实验结果均表明，该方案具有更低的密文旋转复杂度和更高的计算效率。将该方案应用在隐私保护的贝叶斯分类器中，能以更高安全参数和更少计算时间完成分类任务。

关键词：同态加密；矩阵运算；机器学习；贝叶斯分类器

中图分类号：TN92

文献标志码：A

DOI: 10.11959/j.issn.1000−436x.

Matrix computation over homomorphically encrypted data and its application

LIU Yang [1], YANG Linhan[1], CHEN Jingwei[2,3], WU Wenyuan[2,3], FENG Yong [2,3]

1.School of Information Science and Engineering, Chongqing Jiaotong University, Chongqing 400074, China

2.Chongqing Institute of Green and Intelligent Technology, CAS, Chongqing 400714, China

3.Chongqing School, University of Chinese Academy of Sciences, Chongqing 400714,China

Abstract: Those homomorphic encryption schemes supporting Single Instruction Multiple Data (SIMD) operations effectively enhances the amortized efficiency of ciphertext computations, yet the structure of ciphertexts leads to high complexity in matrix operations. In many applications, employing plaintext-ciphertext matrix operations can achieve privacy-preserving computing. Based on this, a plaintext-ciphertext matrix multiplication scheme for matrices of arbitrary dimension is proposed. This scheme computes the resulting ciphertext through steps such as encoding the plaintext matrix, transforming the dimensions of the encrypted matrix, etc. Compared to the best-known encrypted matrix multiplication algorithm for square matrices proposed by Jiang et al. (CCS 2018), the proposed scheme supports matrix multiplication of arbitrary dimension, and supports consecutive matrix multiplications; both theoretical analysis and experimental results show that this scheme requires less rotations on ciphertexts and hence features higher efficiency. When applied to a privacy-preserving Bayesian classifier, this scheme is able to complete classification tasks with higher security parameters and reduced running time.

Keywords: homomorphic encryption, matrix computation, machine learning, bayesian classifier

---

收稿日期：2023−11−02；**修回日期：**2023−12−14

通信作者：陈经纬，jingwei.chen@outlook.com

基金项目：科技部重点研发计划（2020YFA0712300），重庆市自然科学基金（CSTB2023NSCQ-MSX0441，cstc2021jcyj-msxmX0821，cstc2021yszx-jcyjX0004，2022YSZX-JCX0011CSTB，CSTB2023YSZX-JCX0008）

Foundation Items: National Key Research and Development Program of China (2020YFA0712303); Chongqing Research Program (CSTB2023NSCQ-MSX0441, cstc2021jcyj-msxmX0821, cstc2021yszx-jcyjX0004, 2022YSZX-JCX0011CSTB, CSTB2023YSZX-JCX0008)

0 引言

随着机器学习领域的蓬勃发展，它为众多领域提供了广泛的应用和服务。其中一个典型应用场景是，供应商使用大量用户数据训练模型，并使用训练好的模型根据客户端需求提供数据预测服务。虽然机器学习在提高社会生产力方面有着巨大的进步，但大量共享数据集也带来了严重的隐私安全问题。特别是当机器学习模型依赖于机密数据(如金融、医疗和个人数据)时，数据的安全性和隐私性尤为重要，因为模型拥有者不希望数据泄露与模型有关的信息，而数据拥有者也不希望泄露数据相关信息。为了解决这些矛盾，隐私保护机器学习已成为近年来的研究热点。

在绝大多数情况下，如果模型的所有者能够训练出大规模模型，那么通常该所有者也能满足计算方的性能需求。因此，模型的所有者通常也是计算方，在密文计算的过程中，重点是确保数据所有者的信息对计算方不可见，而无需保证模型数据对计算方也是不可见的。所以，在进行隐私保护机器学习的过程中，着重考虑计算方和客户端之间的隐私关系，即使模型数据以明文信息进行计算也能保证模型不被泄露。

常见的隐私保护机器学习方法包括差分隐私[1-4]、多方安全计算[5-7]、联邦学习[8-10]以及同态加密[11]等。差分隐私通过在处理数据之前添加随机噪声来保护个人数据隐私。多方安全计算实现多个参与者进行计算，但相互只能看见自己的输入和输出。联邦学习则通过共享机器学习过程中的部分参数，所有参与者共同训练获得全局最优的模型。同态加密是一种加密方案，它允许对加密的输入进行操作，且解密的结果与明文相应操作的结果匹配[11]。同态不仅可以提供可证安全和抗量子安全，而且理论上可以达到最优的交互次数，因此被认为是最有前途的隐私保护解决方案之一，已被成功应用于统计分析[10]、线性回归[12-16]、贝叶斯模型预测[17-21]、主成分分析[22-24]、以及神经网络预测[25-27]和训练[28-30]等数据处理和分析中。

全同态加密方案的概念最早由 Rivest、Adleman 和 Dertouzos 在 1978 年提出[11]。自 2009 年 Gentry 开创性地提出自举的构想，并设计出第一个全同态加密方案[31]以来，各种方案不断涌现。以 Gentry 为代表的第一代全同态加密方案效率低，并且基于非标准困难性假设，安全性不足。以 BGV[32]、B/FV[33]为代表的第二代全同态加密方案适用于事先给定乘法深度的整数计算任务，支持自举，但代价昂贵。以 FHEW[34]和 TFHE[35]方案为代表的第三代全同态加密方案支持快速自举，因此可以实现任意深度的密文计算任务，尤其适用于基于门电路的计算任务。以 CKKS[36]为代表的第四代全同态加密方案支持浮点算术，被广泛应用于隐私保护机器学习。目前，全同态加密大规模的应用仍然受到效率较低的瓶颈限制。

尽管引入了单指令多数据（SIMD, single instruction multiple data）的批处理[37]、快速自举[35, 38]等技术来优化同态加密方案的效率，但当前最快的同态加密方案，其速度仍然比明文计算慢数百倍甚至更多[39]。并且，这些同态加密方案都只提供了算术或逻辑的基本操作(如 BGV、B/FV 提供密文加法和乘法运算，TFHE 提供与非门等逻辑门的密文赋值)，更为复杂的运算操作需要在这些基本操作的基础上来构建，例如向量内积、矩阵乘法等。矩阵乘法是最为常见的运算操作之一，许多机器学习应用都涉及大量的矩阵乘法，因此同态密文矩阵乘法逐渐成为密文计算的重点。而考虑到模型和数据的实际关系，研究高效同态明文-密文矩阵乘法对同态加密应用到隐私保护机器学习具有深远的意义。

设 R 是一个环，$A \in \mathrm{R}^{n\times m}$ 和 $B \in \mathrm{R}^{m\times p}$ 为 R 的两个矩阵，$d = \max(n, m, p)$ 表示矩阵最大的维数。则计算 $X = AB \in \mathrm{R}^{n\times p}$ 本质上是一系列向量内积运算。若按照教科书中的方式完成两个 $X = AB$ 的密文矩阵乘法，则总共需要 np 次密文乘法和 $np\log m$ 次密文旋转。Halevi 和 Shoup[40]于 2014 年将矩阵对角线编码[41]与 SIMD 相结合，完成了对矩阵-向量乘法密文运算的加速。对一个 $n\times m$ 的矩阵和一个 m 维向量相乘的情形，该方法需 n 次密文乘法和 n 次密文旋转，随后他们采用小步大步法[42]将所需的密文旋转次数降低到 $2\sqrt{n}$。随后 Rathee 等人[24]提出了一种密文方阵的乘法方案，需要 d 次密文乘法和 $d\log d + d$ 次密文旋转；而 Huang 和 Zong 等人[43]对其优化改进提出了非方阵的版本，需要 d 次密文乘法和 $m\log d + m$ 次密文旋转。2018 年，Jiang 等人[25]基于 Halevi 和 Shoup 的矩阵-向量乘法提出了一种密文矩阵乘法的新方法，该方法仅需要 d 次密

文乘法和 $3d+5\sqrt{d}$ 次密文旋转就可以完成 $X=AB$ 的密文计算，但该方案只讨论了方阵相乘(即 $n=m=p$)的情形并且明确要求加密方案的参数需要选取得足够大，以使 SIMD 操作对应的明文槽大于 n^2，但这势必会影响方案的计算效率。近些年，Huang 等人[44]优化了 Jiang 等人[25]的算法中的分块方法，有效提高大规模矩阵的运算效率。Jang 等人[45]则针对张量结构通过修改 CKKS 方案，优化 Jiang 等人算法中的密文旋转和乘法次数，但在明文-密文乘法下的开销与 Jiang 等人的相同。尽管如此，在最近的研究中[46]，Jiang 等人[25]的方案仍然是目前最快的密文矩阵乘法方法之一。

另一些研究则将明文信息编码到多项式系数上。Duong 等人[47]推广了 Yasuda 等人[48, 49]安全内积的方法，将明文信息编码到多项式系数上，仅需一次同态乘法就能完成密文矩阵乘法。随后 Mishra 等人[50]进行改进，提高了明文槽的利用率。然而这种方法在计算过程中会产生无意义的项，导致对空间的利用率低，并对连续矩阵乘法不友好等问题。

表 1　一次明文-密文矩阵乘法方案比较

方案	密文加法	明文-密文乘法	密文旋转	密文条数	密文深度
Halevi-Shoup[40, 42]	$2p\cdot\sqrt{n}$	$p\cdot n$	$2p\cdot\sqrt{n}$	p	1CMult
Jiang et al.[25]	$4d$	$2d$	$d+2\sqrt{d}$	1	2CMult
算法 4	$2m-1$	$2m-1$	$3\sqrt{m}$	1	1CMult
算法 4+算法 5	$p+m+n-1$	$p+m+n-1$	$2(\sqrt{p}+\sqrt{m+n-1})$	1	2CMult

本文主要的研究工作如下。

1) 提出一种适用明文-密文矩阵乘法的编码方法，该编码支持任意维数的明文-密文矩阵乘法。在这种编码方式下，矩阵 $X=AB$ 的密文计算(算法 4)，仅仅只考虑 A 矩阵为明文方阵(即 $n=m$)和 B 矩阵为密文矩阵($m\times p$)的情况。算法 4 完成单次明文-密文矩阵乘法需要 $3\sqrt{m}$ 次旋转操作和 $2m-1$ 次乘法操作。我们将 Halevi 和 Shoup[40]方案和 Jiang 等人[25]的方案放在明文-密文的情况下与我们的方案进行比较，结果如表 1 所示。其中 $d=max\{n,m,p\}$，表示按照其中最大的维数进行矩阵到方阵的变换；CMult 表示进行明文-密文乘法的深度。

2) 提出了算法 5，实现对密文矩阵进行维数变换。在计算 $X=AB$，算法 4 需要将编码后的 B 矩阵加密为1条密文。当明文矩阵 A 中 $n>m$ 的情况，在进行矩阵到方阵转换时会变成 $n\times n$ 的方阵，因此需要将密文矩阵 B 也从 $m\times p$ 转换成为 $n\times p$ 的矩阵，将单条密文的信息进行对齐，以便进行下一步的明文-密文的乘法操作。算法 5 完成一次维数变换需要 $2\sqrt{p}$ 次旋转操作和 p 乘法操作。实验表明，当 $m\geq n$ 时，完成明文-密文矩阵乘法计算只需进行算法 4，由于算法 4 的计算开销与 p 无关，因此 $p>m\geq n$ 时，计算明文-密文矩阵乘法更加高效。

3) 基于 SEAL 同态加密库中的 CKKS 方案，本文实现了提出的算法，并将算法应用于隐私保护的贝叶斯分类器中，与 Chen 等[17]的相关工作相比，对于来自 UCI 机器学习库中的几个数据集，在分类的精度并无明显降低的前提下，本文的方法可以以更高安全参数和更低的计算时间完成分类任务。

1　预备知识

首先对本文一些常用的记号说明如下。对两个正整数 i 和 k，用 $[i]_k$ 表示 i 模 k 的非负剩余。所有的向量都用斜粗体的小写字母表示，如 $\boldsymbol{a}$、$\boldsymbol{b}$，其分量的下标从 0 开始，使用 $\rho(\boldsymbol{a};\ell)$ 表示对向量 $\boldsymbol{a}$ 进行移位。矩阵用大写字母表示，如 A、B。对环 R 上的 $n\times m$ 矩阵 $A=(a_{i,j})$，其下标范围为 $0\leq i<n$，$0\leq j<m$。

1.1　同态加密

同态加密允许在不进行解密的条件下对密文直接计算得到与明文对应的计算结果。令 M 和 C 分别表示明文空间和密文空间。一个同态加密方案 HE 由如下随机算法组成:

- $\mathrm{Setup}(1^{\lambda})$：输入安全参数 λ，该算法输出公钥 pk，公开的计算密钥 ek 和私钥 sk。
- $\mathrm{Enc}_{pk}(\boldsymbol{m})$：该算法使用公钥 pk 加密消息 $\boldsymbol{m}\in\mathrm{M}$，输出密文 $\boldsymbol{c}\in\mathrm{C}$。
- $\mathrm{Dec}_{sk}(\boldsymbol{c})$：该算法使用私钥 sk 解密密文 $\boldsymbol{c}\in\mathrm{C}$，输出明文 $\boldsymbol{m}\in\mathrm{M}$。

- $\mathrm{Eval}_{ek}(f;(\boldsymbol{c}_1,\ldots,\boldsymbol{c}_k))$：输入电路如下所示为 $f:\mathrm{M}^k\to\mathrm{M}$ 和密文 $\boldsymbol{c}_1,\ldots,\boldsymbol{c}_k$，该算法使用计算密钥 *ek* 计算并输出密文 $\boldsymbol{c}\in\mathrm{C}$。

当同态加密方案 HE 具有如下性质时，称其是正确的:

- 对任意的 $m\in\mathrm{M}$ 和其对应的任意密文 $\boldsymbol{c}\leftarrow\mathrm{Enc}_{pk}(\boldsymbol{m})$，都有下式中的结果 $\Pr[\mathrm{Dec}_{sk}(c)\neq\boldsymbol{m}]=\mathrm{negl}(\lambda)$，其中 $\mathrm{negl}(\lambda)$ 表示一个关于 λ 可忽略的函数。
- 对任意的电路 $f:\mathrm{M}^k\to\mathrm{M}$，密文 $\boldsymbol{c}_i\leftarrow\mathrm{Enc}_{pk}(\boldsymbol{m}_i)$，以及相应的计算结果密文 $\boldsymbol{c}\leftarrow\mathrm{Eval}_{ek}(f;(\boldsymbol{c}_1,\ldots,\boldsymbol{c}_k))$，有如下结果 $\Pr[\mathrm{Dec}_{sk}(\boldsymbol{c})\neq f(\boldsymbol{m}_1,\ldots,\boldsymbol{m}_k)]=\mathrm{negl}(\lambda)$。

由于 *pk* 和 *ek* 都是公开的，为了方便起见，下面就用 *pk* 表示公钥和计算密钥等所有公开的密钥，并且如无特殊说明，将在基本操作中省略不写。例如，本文将用到密文加法，密文乘法，明文-密文乘法等基本操作:

- $\mathrm{Add}(\boldsymbol{c}_1\in\mathrm{C},\boldsymbol{c}_2\in\mathrm{C})$：输出密文 $\boldsymbol{c}$，使得 $\Pr[\mathrm{Dec}_{sk}(\boldsymbol{c})\neq\mathrm{Dec}_{sk}(\boldsymbol{c}_2)+\mathrm{Dec}_{sk}(\boldsymbol{c}_2)]=\mathrm{negl}(\lambda)$。
- $\mathrm{Mult}(\boldsymbol{c}_1\in\mathrm{C},\boldsymbol{c}_2\in\mathrm{C})$：输出密文 $\boldsymbol{c}$，使得 $\Pr[\mathrm{Dec}_{sk}(\boldsymbol{c})\neq\mathrm{Dec}_{sk}(\boldsymbol{c}_2)\cdot\mathrm{Dec}_{sk}(\boldsymbol{c}_2)]=\mathrm{negl}(\lambda)$。
- $\mathrm{CMult}(\boldsymbol{m}\in\mathrm{M},\boldsymbol{c}\in\mathrm{C})$：输出密文 $\boldsymbol{c}'$，使得 $\Pr[\mathrm{Dec}_{sk}(\boldsymbol{c}')\neq m\cdot\mathrm{Dec}_{sk}(\boldsymbol{c})]=\mathrm{negl}(\lambda)$。

若同态加密方案 HE 支持 SIMD[37]操作，则其明文空间 M 可以看作是环 R 上的 维向量形成的集合 R^n。此时，R^n 中的任意元素 $\boldsymbol{m}$ 在编码后可以被加密成一条密文，在进行密文计算时，相当于对 $\boldsymbol{m}$ 的每个分量(明文槽)并行地进行了相应的计算，从而使得 HE 达到更好的均摊性能。在计算过程中，可能需要对 $\boldsymbol{m}$ 的明文槽间的数据进行操作，例如密文旋转:

- $\mathrm{Rot}(\boldsymbol{c}\in\mathrm{C};\ell\in\square)$：该算法输入明文信息 $\boldsymbol{m}=(m_0,m_1,\ldots,m_{n-1})\in\mathrm{M}=\mathrm{R}^n$ 的一个密文 $\boldsymbol{c}\in\mathrm{C}$ 和一个整数 ℓ，输出 $\boldsymbol{c}'$ 使 $\Pr[\mathrm{Dec}_{sk}(\boldsymbol{c}')\neq\rho(\boldsymbol{m};\ell)]=\mathrm{negl}(\lambda)$，其中 $\rho(\boldsymbol{m};\ell)=(m_\ell,\ldots,m_{n-1},m_0,\ldots,m_{\ell-1})$ 表示将 $\boldsymbol{m}$ 的分量依次向左旋转 ℓ 个明文槽得到的新明文。

由于密文旋转后需要进行代价昂贵的密钥交换操作，所以其开销比密文乘法和密文加法都大，而密文乘法的开销又远大于密文加法。除此之外，同态密文计算的乘法深度会影响参数的选取，从而影响计算性能。因此，本文仅统计密文乘法次数、密文旋转次数以及密文乘法深度，并以这些指标作为衡量计算性能的关键指标。

1.2 Halevi-Shoup 的矩阵-向量乘法

作为矩阵乘法的一种特殊情形，矩阵-向量乘法已研究得比较成熟[40]。

设 $A\in\mathrm{R}^{n\times m}$，$\boldsymbol{v}\in\mathrm{R}^{m\times 1}$，记 $\boldsymbol{u}=A\boldsymbol{v}\in\mathrm{R}^n$。Halevi 和 Shoup[40]算法的基本思想是对矩阵 采用对角线编码。文献[40]只给出了 $m=n$ 的情形，这里给出更为一般的 n 整除 m 的情形描述。设 $m=n\cdot q$。则矩阵 $A=(a_{i,j})_{0\le i<n,0\le j<m}$ 的对角线编码方式如下：对 $0\le\ell<n$，定义 A 的第 ℓ 个对角向量

$$d_\ell(A)=(a_{0,\ell},\ldots,a_{n-1,n+\ell-1},a_{0,n+\ell},\ldots,a_{n-1,2n+\ell-1},\ldots,a_{0,[(q-1)n+\ell]_m},\ldots,a_{n-1,[m+\ell-1]_m})\in\mathrm{R}^m \tag{1}$$

在得到这样的对角线编码之后，$\boldsymbol{u}=A\boldsymbol{v}$ 的计算可以分为如下两个步骤:

1. 计算

$$\boldsymbol{u}=\sum_{0\le\ell<n}(\rho(\boldsymbol{v};\ell)\square\ \boldsymbol{d}_\ell) \tag{2}$$

其中 $\square$ 表示矩阵的 Hadamard 乘积，即按分量对应相乘。

2. 更新

$$\boldsymbol{u}=\sum_{0\le i<q}\rho(\boldsymbol{u};i\cdot n),\quad q=m/n. \tag{3}$$

显然，式(2)的计算需要对向量 $\boldsymbol{v}$ 进行 n 次旋转操作。若 $n=k\cdot l$，则可以采用小步-大步法进一步将旋转操作的次数降至 $k+l$ 次。这是因为公式(2)可以重写为，

$$\begin{aligned}\boldsymbol{u}&=\sum_{i=0}^{l-1}\sum_{j=0}^{k-1}(\rho(\boldsymbol{v};ki+j)\square\ \boldsymbol{d}_{ki+j})\\&=\sum_{i=0}^{l-1}\rho\left(\sum_{j=0}^{k-1}\left(\rho(\boldsymbol{v};j)\square\ \rho(\boldsymbol{d}_{ki+j};-ki)\right);ki\right)\end{aligned} \tag{4}$$

在事先计算好 $\rho(\boldsymbol{d}_{ki+j};-ki)$ 的情况下(即将矩阵 A 按此编码成 n 个向量)，仅需对向量 $\boldsymbol{v}$ 进行 $k+l$ 次[42]旋转操作便能够完成 $\boldsymbol{u}$ 的计算。由此，得到如下的密文矩阵-向量的乘法算法。

算法 1 矩阵-向量乘法

初始化 $\boldsymbol{c}_{i,j}$ 对应明文 $\rho(\boldsymbol{d}_{ki+j};-ki)$，其中 $\boldsymbol{d}_\ell$ 是矩阵 $A\in\mathrm{R}^{n\times m}$ 第 ℓ 对角向量，$0\le i<l$，$0\le j<k$，且 $n=k\cdot l$，$n\mid m$；密文向量 $\boldsymbol{c}\in\mathrm{C}$，对应明文 $\boldsymbol{v}\in\mathrm{R}^{m\times 1}$；

密文向量 $\boldsymbol{c}'$。

1) 根据公式(4)计算 $\boldsymbol{c}'$；
2) 根据公式(3)更新 $\boldsymbol{c}'$；
3) 返回 $\boldsymbol{c}'$；

步骤 1 需要 n 次密文乘法和 $k+l$ 次密文旋转(当 $k \approx l$ 时，$k+l \approx 2\sqrt{n}$)。尽管式(3)表面上需要 q 次旋转，但采用二分法容易证明 $\log q$ 次旋转便足够了。因此，步骤 2 至多需要 $\log q = \log \frac{m}{n}$ 次密文旋转。设 $A \in \mathrm{R}^{n\times m}$， $B \in \mathrm{R}^{m\times p}$。若将计算 $X = AB$ 归约为 p 次矩阵-向量乘法，我们将密文计算使用明文-密文计算来代替，得到表 1 中的结果。

1.3 Jiang 等的密文矩阵乘法算法

设待相乘的矩阵是两个方阵 $A \in \mathrm{R}^{n\times n}$ 和 $B \in \mathrm{R}^{n\times n}$。在 Jiang 等给出的同态密文乘法算法[25]中，其核心思想就是将 $X = AB$ 的计算重写为

$$X = \sum_{i=0}^{n-1} A_i \square B_i,$$

其中，

- $A_0 \in \mathrm{R}^{n\times n}$ 的第 j 列就是矩阵 A 的第 j 个对角向量 $\boldsymbol{d}_j(A)$， $0 \le j < n$；
- $B_0 \in \mathrm{R}^{n\times n}$ 的第 j 行就是矩阵 $B^{\bullet}$ 的第 j 个对角向量 $\boldsymbol{d}_j(B^{\bullet})$， $0 \le j < n$；
- $A_i \in \mathrm{R}^{n\times n}$ 是将 A_0 的列向左旋转 i 列得到的矩阵，记作 $A_i = \mathrm{ColRot}(A_0, i)$， $1 \le i < n$；
- $B_i \in \mathrm{R}^{n\times n}$ 是将 B_0 的行向上旋转 i 行得到的矩阵，记作 $B_i = \mathrm{RowRot}(B_0, i)$， $1 \le i < n$。

另外，Jiang 等给出的同态密文乘法算法将一个 $n\times n$ 的矩阵都按行依次排列，编码成一个 n^2 维的向量。将矩阵 A、B 和 $\boldsymbol{X}$ 照此编码后得到的向量分别记为 $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{x} \in \mathrm{R}^{n^2}$，得到算法 2 如下。

算法 2 Jiang 等的密文矩阵乘法

初始化矩阵 $A \in \mathrm{R}^{n\times n}$，对应明文向量 $\boldsymbol{a}$；矩阵 $B \in \mathrm{R}^{n\times n}$，对应明文向量 $\boldsymbol{b}$，密文 $\boldsymbol{c}_b$。

1) 从 $\boldsymbol{c}_b$ 构造 $\boldsymbol{c}_{b_0}$，对应向量 $\boldsymbol{b}_0$ 和矩阵 B_0；
2) 从 $\boldsymbol{a}$ 构造 $\boldsymbol{a}_0$，对应矩阵 A_0；
3) 计算 $\boldsymbol{c}_x \leftarrow \mathrm{CMult}(\boldsymbol{a}_0, \boldsymbol{c}_{b_0})$
4) 循环
5) for $1 \le i < n$
6) 计算 $\boldsymbol{a}_i \leftarrow \mathrm{ColRot}(\boldsymbol{a}_0, i)$；
7) 计算 $\boldsymbol{c}_{b_i} \leftarrow \mathrm{RowRot}(\boldsymbol{c}_{b_0}, i)$；
8) 更新 $\boldsymbol{c}_x \leftarrow \mathrm{Add}(\boldsymbol{c}_x, \mathrm{CMult}(a_i, \boldsymbol{c}_{b_i}))$；
9) end for
10) 返回 $\boldsymbol{c}_x$。

考虑明文-密文的情况，只统计 B 的密文操作，根据文献[25]算法 2 共需要 d 次明文-密文乘法、以及 $d + 2\sqrt{d}$ 次密文旋转，其中 $d = max\{n, m, p\}$，表示按照其中最大的维数进行矩阵到方阵的变换，得到结果如表 1 所示。

2 明文-密文矩阵乘法

本节提出了一种新的密文矩阵乘法，适用于任意的 $A = (a_{i,j}) \in \mathrm{R}^{n\times m}$ 和 $B = (b_{i,j}) \in \mathrm{R}^{m\times p}$ 的矩阵乘法 $X = AB \in \mathrm{R}^{n\times p}$。该方法解除了矩阵 B 对方阵的限制，但是需要满足 A 为方阵。本文考虑模型拥有方和计算方是同一方的情形，提出了明文-密文矩阵乘法。该方案相较于其它方案在明文-密文矩阵乘法时较好的优越性，特别是当 $p > m \ge n$ 时，方案只需进行算法 4，且计算开销与 p 无关，因而方案的效率最高。

2.1 矩阵编码

设 $A \in \mathrm{R}^{m\times m}$ 和 $B \in \mathrm{R}^{m\times p}$ 是两个待相乘的矩阵。我们定义如下元素提取方式

$$U_k^{m\times m}(a_{0,j}) = \begin{cases} a_{k,j+k} & \text{if } 0 \le j+k < m \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

这里需要注意的是，式(5)的目的是从 $a_{0,j}$ 起，取对角线的元素，在下标超过矩阵维数时，将其取值为 0，最终结果是一个 k 维向量。

同样我们定义另一种元素提取方式为

$$V_k^{m\times m}(a_{i,0}) = \begin{cases} a_{k,k-i} & \text{if } 0 \le k-i < m \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

需要注意的是，同式(5)类似，式(6)最终得到的也是一个 k 维向量。接下来我们引入一种编码方式，将矩阵 $A \in \mathrm{R}^{m\times m}$ 编码成 $(2m-1)\times(m\cdot p)$ 的矩阵，记为 $\underline{A}$。

算法 3 矩阵编码算法

初始化矩阵 $A = (a_{i,j}) \in \mathrm{R}^{m\times m}$；向量 $\underline{\boldsymbol{a}}_i$ 表示编码矩阵 $\underline{A}$ 第 i 行。

1) 循环

2) for $0 \le i < 2m-1$
3) if $i \le m$
4) 计算 $\underline{\boldsymbol{a}}_i \leftarrow \text{repeat}(p, V_m(a_{m-i,0}))$；
5) else
6) 计算 $\underline{\boldsymbol{a}}_i \leftarrow \text{repeat}(p, U_m(a_{0,i-m}))$；
7) end for
8) 返回 $\underline{A}$。

当执行完公式(5)和公式(6)操作时，得到一个 m 维长的向量，repeat 将向量重复 p 次，所以在循环结束时 $\underline{\boldsymbol{a}}_i \in \mathrm{R}^{mp}$，最终编码矩阵 $\underline{A} \in \mathrm{R}^{(2m-1)\times(mp)}$。

对于矩阵 $B \in \mathrm{R}^{m\times p}$，我们根据式(6)定义矩阵变换 $\tau : \mathrm{R}^{m\times p} \to \mathrm{R}^{m\cdot p}$，将矩阵转换成向量：

$$\tau(B) = (\varphi^m(b_{0,0}), \varphi^m(b_{0,1}), \cdots, \varphi^m(b_{0,p-1})) \tag{7}$$

通过上述变化，可以将一个矩阵变成一条向量，向量的长度是原来矩阵行列数的乘积。接下来我们描述如何通过 $\underline{A}$ 和 $\tau(B)$ 进行矩阵乘法计算。

2.2 明文-密文矩阵乘法

现将上述编码结果，在密文上进行计算，实现矩阵乘法计算。

算法 4 明文-密文矩阵乘法

矩阵 $A = (a_{i,j}) \in \mathrm{R}^{m\times m}$；矩阵 $B = (b_{i,j}) \in \mathrm{R}^{m\times p}$；初始化 k，l 其中 $k \cdot l = 2m-1$ 且尽可能相等；初始化密文向量 $\boldsymbol{v}_{cip}$，长度为 k；初始化密文 $\boldsymbol{c}_x$，$\boldsymbol{c}_{tmp}$ 初始化公钥 pk。

1) 使用算法 3 通过 A 得到 $\underline{A}$；
2) 计算 $\boldsymbol{b} \leftarrow \tau(B)$；
3) 加密 $\boldsymbol{c}_B = \text{Enc}_{pk}(\boldsymbol{b})$；
4) 循环
5) for $0 \le j < k$
6) $\boldsymbol{v}_{cip}[j] \leftarrow \text{Rot}(\boldsymbol{c}_B, j-m)$；
7) end for
8) 循环
9) for $0 \le i < l$
10) 循环
11) for $0 \le j < k$
12) 如果 $\underline{A}_{i\cdot k+j}$ 元素全 0，跳过；
13) 更新 $\underline{A}_{i\cdot k+j} \leftarrow \rho(\underline{A}_{i\cdot k+j}, -ki)$；
14) 更新 $\boldsymbol{c}' \leftarrow \text{CMult}(\underline{A}_{i\cdot k+j}, \boldsymbol{v}_{cip}[j])$；
15) end for
16) 更新 $\boldsymbol{c}_x \leftarrow \text{Add}(\boldsymbol{c}_x, \text{Rot}(\boldsymbol{c}', ki))$；
17) end for
18) 返回 $\boldsymbol{c}_x$

首先，我们先对 $\tau(B)$ 向量进行加密记为 $\boldsymbol{c}_B = \text{Enc}_{pk}(\tau(B))$，然后计算 $\underline{A}_i$ 与 $\text{Rot}(\boldsymbol{c}_B; i-m)$ 明文-密文乘积结果；按照一般方式，需要进行 $2m-1$ 次密文乘法和 Rot 操作。但我们注意到，式(4)适用于任意连续变化的密文。因此将式(4)应用到算法 4 中，可以降低密文旋转操作从原来 $2m-1$ 次到 $2\sqrt{2m-1} \le 3\sqrt{m}$ 次。

根据算法 4，计算得到明文-密文矩阵乘法的结果。如果将 $A \in \mathrm{R}^{m\times m}$ 和 $B \in \mathrm{R}^{m\times p}$ 矩阵乘法结果表示为 $X \in \mathrm{R}^{m\times p}$，那么计算结果其实是 $\text{Enc}_{pk}(\tau(X))$。我们可以观察到，矩阵 B 和矩阵 X 有类似的性质，因此可以直接将计算结果作为算法 4 的输入，实现多次乘法计算。

2.3 正确性验证

我们对上述算法 4 过程用公式进行描述如下：

$$X = \text{Add}(\text{CMult}(\underline{A}_i, \text{Rot}(c_B; i-m)))_{0\le i\le 2m-1} \tag{8}$$

我们将密文操作对应到明文，可以得到如下结果：

$$X = \sum_{i=0}^{2m-1} \underline{A}_i \square\ \rho(\tau(B); i-m) \tag{9}$$

根据旋转的方向将式(9)中分成两部分用 $X^{(1)}$ 和 $X^{(2)}$ 表示，即 $X = X^{(1)} + X^{(2)}$，其中 $X^{(1)}$ 为

$$X^{(1)} = \sum_{i=0}^{m} \underline{A}_i \square\ \rho(\tau(B); i-m), \tag{10}$$

$X^{(2)}$ 的表示形式如下：

$$X^{(2)} = \sum_{i=m}^{2m-1} \underline{A}_i \square\ \rho(\tau(B); i-m), \tag{11}$$

因此，我们分别对 $X^{(1)}$ 和 $X^{(2)}$ 的计算结果进行验证，验证计算的正确性。

首先，对式(10)中矩阵 A 的编码进行分析，得到如下结果：

$$\begin{aligned}\underline{A}_i &= (\underbrace{V_m(a_{m-i,0}), \cdots, V_m(a_{m-i,0})}_{p}) \\ &= (\underbrace{0, \cdots, 0, a_{m-i,0}, \cdots, a_{m-1,i-1}}_{m}, \cdots, \underbrace{0, \cdots, 0, a_{m-i,0}, \cdots, a_{m-1,i-1}}_{m})\end{aligned}$$

而对 $\tau(B)$ 进行移位的过程中有如下结果：

$$\begin{aligned}\rho(\tau(B); i-m) = (&\underbrace{b_{i,p-1}, \cdots, b_{m-1,p-1}, b_{0,0}, \cdots, b_{i-1,0}}_{m}, \\ &\cdots, \underbrace{b_{i,p-2}, \cdots, b_{m-1,p-2}, b_{0,p-1}, \cdots, b_{i-1,p-1}}_{m})\end{aligned}$$

所以，我们可以将式(10)中第 i 条乘积重写成如下形式，

$$\begin{aligned}&\underline{A}_i \square \ \rho(\tau(B);i-m)\\&=(\underbrace{0,\cdots,0,a_{m-i,0}b_{0,0},\cdots,a_{m-1,i-1}b_{i-1,0}}_{m},\\&\cdots,\underbrace{0,\cdots,0,a_{m-i,0}b_{0,p-1},\cdots,a_{m-1,i-1}b_{i-1,p-1}}_{m})\end{aligned}$$

得到 $X^{(1)}$ 的结果如下所示：

$$\begin{aligned}X^{(1)}=(&\underbrace{0,\sum_{i=0}^{0}a_{1,i}b_{i,0},\cdots,\sum_{i=0}^{m-2}a_{m-1,i}b_{i,0}}_{m}\\&,\cdots,\underbrace{0,\sum_{i=0}^{0}a_{1,i}b_{i,p-1},\cdots,\sum_{i=0}^{m-2}a_{m-1,i}b_{i,p-1}}_{m})\end{aligned}$$

通过上述分析，最终计算得到 $X^{(1)}$ 的结果。

同理，在式(11)中有 $m\le i<2m-1$，对矩阵 A 的编码进行分析，得到如下结果：

$$\begin{aligned}\underline{A}_i&=(\underbrace{U_m(a_{0,i-m}),\cdots,U_m(a_{0,i-m})}_{p})\\&=(\underbrace{a_{0,i-m},\cdots,a_{2m-1-i,m-1},0,\cdots,0}_{m},\cdots,\underbrace{a_{0,i-m},\cdots,a_{2m-1-i,m-1},0,\cdots,0}_{m})\end{aligned}$$

而对 $\tau(B)$ 进行移位的过程中有如下结果：

$$\begin{aligned}\rho(\tau(B);i-m)=(&\underbrace{b_{i-m,0},\cdots,b_{m-1,0},b_{0,1},\cdots,b_{i-m-1,1}}_{m},\\&\cdots,\underbrace{b_{i-m,p-1},\cdots,b_{m-1,p-1},b_{0,0},\cdots,b_{i-m-1,0}}_{m})\end{aligned}$$

所以，我们可以将式(11)中第 i 条乘积重写成如下形式：

$$\begin{aligned}&\underline{A}_i \square \ (\rho(\tau(B);i-m)\\&=(\underbrace{a_{0,i-m}b_{i-m,0},\cdots,a_{2m-1-i,m-1}b_{m-1,0},0,\cdots,0}_{m},\\&\cdots,\underbrace{a_{0,i-m}b_{i-m,p-1},\cdots,a_{2m-1-i,m-1}b_{m-1,0},0,\cdots,0}_{m})\end{aligned}$$

所以，式(13)的结果为以下形式：

$$\begin{aligned}X^{(2)}=(&\underbrace{\sum_{i=0}^{m-1}a_{0,i}b_{i,0},\sum_{i=1}^{m-1}a_{1,i}b_{i,0},\cdots,\sum_{i=m-1}^{m-1}a_{m-1,i}b_{i,0}}_{m},\cdots\\&,\underbrace{\sum_{i=0}^{m-1}a_{0,i}b_{i,p-1},\sum_{i=1}^{m-1}a_{1,i}b_{i,p-1},\cdots,\sum_{i=m-1}^{m-1}a_{m-1,i}b_{i,p-1}}_{m})\end{aligned}$$

经过上述过程，最终计算得到 $X^{(2)}$ 的结果。

将 $X^{(1)}$ 和 $X^{(2)}$ 的结果进行相加，我们可以得到 X 的表达式重写成如下形式：

$$\begin{aligned}X&=X^{(1)}+X^{(2)}\\&=(\underbrace{\sum_{i=0}^{m-1}a_{0,i}b_{i,0},\sum_{i=0}^{m-1}a_{1,i}b_{i,0},\cdots,\sum_{i=0}^{m-1}a_{m-1,i}b_{i,0}}_{m},\\&\cdots,\underbrace{\sum_{i=0}^{m-1}a_{0,i}b_{i,p-1},\sum_{i=0}^{m-1}a_{1,i}b_{i,p-1},\cdots,\sum_{i=0}^{m-1}a_{m-1,i}b_{i,p-1}}_{m})\end{aligned}\tag{12}$$

所以，通过上述推导过程，我们得到 X 的结果。通过观察发现，式(12)中每一项都是内积，实际上就是矩阵一般乘法，从而证明计算的正确性。

2.4 性能分析

在算法 3 中，实现了对密文矩阵的乘法，我们将算法用式(9)进行表示，可以发现，用原始方法进行明文矩阵计算需要分别进行 $2m-1$ 次密文加法、明文-密文乘法以及密文旋转操作。在优化之后，能将密文旋转操作降低至 $3\sqrt{m}$。同时我们观察到，密文计算的次数和 p 无关，只与 m 的大小相关，因此当 p 远大于 m 时，在不改变计算效率的情况下，能同时处理更大数据量。

需要注意的是，这种方法需要限制 A 矩阵的大小为方阵，但是对于矩阵为以下情况时 $A=(a_{i,j})\in \mathrm{R}^{n\times m}, n>m$，如果我们转换矩阵，则需要变成 $n\times n$ 的矩阵，相应的需要对矩阵 $B\in \mathrm{R}^{m\times p}$ 进行变换 $B\in \mathrm{R}^{n\times p}$，才能进行矩阵乘法。一种解决办法是对矩阵 A 以 m 进行切割，再分别进行矩阵乘法。但同时我们也提供了一种算法，可以在密文上进行维数变换，以满足待相乘的两个矩阵的维数匹配要求。

3 维数变换算法

在本节中，为了解决转换矩阵的维数变化问题，我们提出了一种算法，在密文上能够转换维数，将矩阵在密文上进行变换 $\mathrm{R}^{m\times p}\to \mathrm{R}^{n\times p}$。维数转换的提出使得方案能支持任意维数矩阵连乘。

首先对公式(4)进行改写，我们注意到如果同时乘以一个倍数，那么我们公式仍然成立。

$$\begin{aligned}\boldsymbol{u}&=\sum_{i=0}^{l/r}\sum_{j=0}^{k-1}(\rho(\boldsymbol{v};r[ki+j])\square\ \boldsymbol{d}_{r[ki]+j})\\&=\sum_{i=0}^{l/r}\rho\left(\sum_{j=0}^{k-1}\Big(\rho(\boldsymbol{v};r[j])\square\ \rho(\boldsymbol{d}_{ki+j};-r[ki])\Big);r[ki]\right)\end{aligned}\tag{13}$$

可以看见，在式(13)中减少了外层的操作次数，这种方式适用于有规律的稀疏矩阵，能有效减少操作次数。

因此，可以将维数变换看成矩阵和向量的乘法，所以我们可以通过式(13)对矩阵进行计算，实现维数转换效果。

算法 5 维数变换算法

密文矩阵 $\boldsymbol{c}_b$，及新的行数 n_1，旧的行数 n_2；初始化 k，l 其中 $k \cdot l = p$ 且尽可能相等；初始化密文 $\boldsymbol{c}'$。

1) 循环
2) for $0 \le i < p$
3) 生成向量 $\boldsymbol{diag}_{i(n_1-n_2)} = []$，其中从 $n_2 \cdot i$ 到 $n_2(i+1)$ 位置为 1，其余为 0；
4) end for
5) 根据公式(13)计算 $\boldsymbol{c}'$；
6) 根据公式(3)更新 $\boldsymbol{c}'$；
7) 返回 $\boldsymbol{c}'$。

通过上述算法，我们将原本需要 p 次的密文旋转降低到只需要 $2\sqrt{p}$ 次密文旋转，提高了运算效率。当然算法不仅能够扩张矩阵的行数，也能够降低矩阵的行数。

详细的计算开销如表 2 所示，我们注意到，通过算法 5 进行转换之后，单独考虑算法 4 需要进行 $2m-1$ 次密文加法、$2m-1$ 次明文-密文乘法和 $3\sqrt{m}$ 次密文旋转。但是在扩张维数时，矩阵 A 通过填零进行升维，变换维数后的矩阵会十分稀疏，由于这种稀疏性在对 A 矩阵编码时产生 $n-m$ 全 0 向量，而这些向量在算法 4 中不需要进行计算。所以，当算法 4 和算法 5 一起使用时，算法 4 的密文旋转次数会降低到 $2(\sqrt{p}+\sqrt{m+n-1})$ 次。此外，并不是每次计算都需要进行维数变换，只需要在 $m > n$ 时，进行维数变换即可。

表 2 计算开销比较

算法	密文加法	明文-密文乘法	密文旋转	密文条数	密文深度
算法 4	$2m-1$	$2m-1$	$3\sqrt{m}$	1	1CMult
算法 5	p	p	$2\sqrt{p}$	1	1CMult
算法 4+算法 5	$p+m+n-1$	$p+m+n-1$	$2(\sqrt{p}+\sqrt{m+n-1})$	1	2CMult

4 实验及实施

在本节中，我们将详细描述同态矩阵运算的性能，并将其应用到贝叶斯分类器上进行比较、分析实现的相关性能。我们基于 Microsoft SEAL 版本 4.1.0 中的 CKKS 方案实现相关功能。所有的实验都在具有 2.8GHz 额定的 4 个内核运行的英特处理器 i7 笔记本电脑上运行。

4.1 参数设置

在进行实验时，首先设置 CKKS 方案参数，其中分圆多项式模数设置为 8192，多项式系数模链设置为 $\{60,40,40,60\}$，此时多项式模数 $q = 200$ 比特，在给定误差标准差 $\sigma = 3.19$ 的参数下，参照 lattice estimator[51]中给出的攻击方式，最低可以达到 137 比特的安全性。

4.2 矩阵运算效率

表 3 中设置了 4 种规模的矩阵进行明文-密文矩阵乘法。分别统计每种规模下，加密时间、解密时间以及算法 4 和算法 5 的时间开销，所有结果是对 10 次运算取平均值。

表 3 矩阵运算效率

矩阵规模	加密时间	解密时间	算法 4	算法 5	总时间
$R^{16\times16} \times R^{16\times128}$	3.08 ms	0.51 ms	62.16 ms	-	65.75 ms
$R^{16\times16} \times R^{16\times256}$	2.81 ms	0.52 ms	48.10 ms	-	51.43 ms
$R^{16\times4} \times R^{4\times128}$	2.51 ms	0.22 ms	24.92 ms	148.04 ms	175.69 ms
$R^{16\times4} \times R^{4\times256}$	3.07 ms	0.22 ms	21.38 ms	257.42 ms	282.09 ms

对实验结果进行分析：

1)通过比较表 3 中 1、2 行结果发现，随着 p 增加，总时间基本相差无几，可以说明算法 4 的运行效率和 p 的大小无关；

2)通过比较表 3 中 1、3 行结果可以发现在中 $\mathrm{R}^{16\times16}\times\mathrm{R}^{16\times128}$ 算法 4 的时间大于 $\mathrm{R}^{16\times4}\times\mathrm{R}^{4\times128}$ 中算法 4 的时间，这是因为进行维数转换后的矩阵，需要通过添加 0 元素改变矩阵大小，对 A 矩阵进行编码时会出现许多全 0 行，而这些编码不需要进行计算，因此实际上的运算时间会减少；

3)通过比较表 3 中 3、4 行算法 5 的时间可以发现 $\mathrm{R}^{16\times4}\times\mathrm{R}^{4\times128}$ 中运行时间更短。这说明算法 5 的运行效率和 p 的大小有关，p 越大进行维数转换所需的时间越长。

表 4 中设置了 3 组规模下的方阵进行明文-密文矩阵乘法的时间结果。在实验中，Halevi-Shoup[40, 42] 将密文矩阵的每一列加密成一条密文；Jiang[25]的方案是根据文献[25]中算法 2 实现的明文-密文矩阵乘法。在这种规模下我们的方案实现计算只需要进行算法 4。通过比较可以发现，我们的方案在时间上优于 Jiang 等人[25]的方案，详细结果如表 4 所示。

矩阵规模	方案	加解密时间	矩阵乘法时间	密文个数	总时间
$\mathrm{R}^{16\times16}\times\mathrm{R}^{16\times16}$	Halevi-Shoup[40, 42]	50.82ms	343.80 ms	16	394.62 ms
	Jiang et al.[25]	3.20 ms	64.24 ms	1	67.44 ms
	算法 4	3.35 ms	66.83 ms	1	70.18 ms
$\mathrm{R}^{32\times32}\times\mathrm{R}^{32\times32}$	Halevi-Shoup[40, 42]	112.40 ms	1393.50 ms	32	1505.9 ms
	Jiang et al.[25]	3.48 ms	143.85 ms	1	147.33 ms
	算法 4	4.02 ms	112.74 ms	1	116.76 ms
$\mathrm{R}^{64\times64}\times\mathrm{R}^{64\times64}$	Halevi-Shoup[40, 42]	225.85 ms	4747.44 ms	64	4973.29 ms
	Jiang et al.[25]	3.48 ms	234.70 ms	1	238.18 ms
	算法 4	3.46 ms	168.28 ms	1	171.74 ms

5 应用

在本节中，首先对贝叶斯分类器进行改进，使其适合进行密文计算；然后将算法 4 应用到明文-密文的矩阵乘法上；最后将结果和文献[17]进行对比，取得更优的结果。

5.1 贝叶斯分类器

朴素贝叶斯分类器是一种基于贝叶斯定理和特征独立假设的简单、快速、高效的分类算法。它的基本思想是通过计算每个特征在不同类别下的概率，通过预测数据特征的概率去比较所有类别的可能性，最后取概率最大的类别为预测结果。我们对于 $1,2,\cdots,s$ 共 s 个分类和 $X_1,X_2,\cdots,X_n$ 共 n 个特征，每个特征包含 $1,2,\cdots,t$ 种取值的贝叶斯模型。我们假设对 $\mathbf{x}=(x_1,\cdots,x_n)$ 进行分类预测：

$$\hat{y}=\arg\max_{i=1,\cdots,s}\Pr[Y=i]\prod_{k=1}^{n}\Pr[X_k=x_k\mid Y=i]$$

其中，$\Pr[Y=i]$ 称为先验概率，表示 $Y=i$ 类别的概率；$\Pr[X_k=x_k\mid Y=i]$ 称为条件概率，表示在 $Y=i$ 的条件下，X_k 取 $x_k\in(x_1,\cdots,x_n)$ 的概率。预测过程实际上理解为计算后验概率，也就是计算当 $X_k=x_k$ 时，$Y=i$ 类别的概率，最后通过比较得到概率最大的类别，就是预测结果。

5.2 隐私保护朴素贝叶斯

为了在密文上应用矩阵运算时能减少比较操作带来的计算开销，我们需要对贝叶斯模型进行改进，对所有特征的取值，取特征值对应的序号为 1，其余全取为 0。通过这种编码，我们将每条预测数据，扩展成为 $k\cdot t$ 长度的预测数据。

$$X_i=\begin{cases}1 & \text{if } X_i=x_i\\ 0 & \text{if } X_i\neq x_i\end{cases}$$

为了减少同态比较带来的计算开销，我们考虑如图

所示的隐私保护机器框架，对 m 条数据进行预测，在云端只需对 $s\times(n\cdot k)$ 和 $(n\cdot k)\times m$ 大小的矩阵进行同态密文矩阵乘法，对每个样本添加随机选取的相同噪声之后，将密文结果传输给客户端解密，比较得到最终的预测结果，详细结果如图 1 所示。

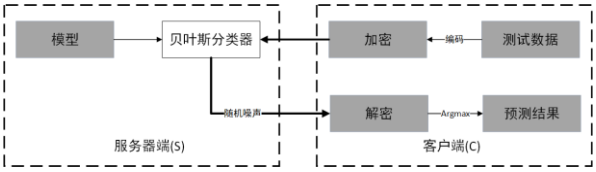

图 1　隐私保护贝叶斯分类器框架图

5.2.1　安全性

对于客户端来说，客户端将预测数据进行编码，然后通过 CKKS 方案进行加密，传输给服务器端。从服务器端来看，它没有私钥，不能对密文数据进行解密运算，因此只能看见加密后的预测数据，从而保证了客户端的安全性。

对于服务器来说，服务器端在计算出后验概率之后，在模型结果每个样本中引入一个相同的噪声。从客户端来看，我们依然能够正确比较后验概率的大小，保证预测结果的正确性，同时由于噪声是随机选取的，客户端不能根据预测结果去推测模型中的相关参数，从而保证了服务器端的安全性。

5.2.2　性能评估

我们使用 UCI 机器学习库中的 Iris 数据集和 WBC(Wisconsin Breast Cancer)数据集进行实验。我们使用[17]中的模型进行运算，两种模型的实验精度在 97%左右，这和明文上运算的精度没有区别。

实验过程中，参数设置参照 5.1 中的参数设置，实现至少 137 比特的安全性。

在 Iris 数据集中，总共有 150 个样本，被分为 3 种类别，其中每个样本含有 4 个特征。我们将样本中连续的特征类型变成离散型数据，通过将每种特征值最大最小值的区间平均分成 5 段，可以得到 5 个不同的取值。我们使用其中 80%的数据进行模型训练，使用 30 个测试数据进行预测。所以，密文计算实际上是两个 3×20 和 20×30 的矩阵相乘。我们将方案和 Jiang 等人的方案[25]以及文献[17]中表 3 的结果进行比较，需要注意的是。首先，Jiang 等人的方案[25]在计算时，需要按照最大维数将两个矩阵变换成方阵，即 30×30 与 30×30 的矩阵乘法，因此会增加密文计算的时间；其次，文献[17]中取最大值的操作是在密文上进行的，所以在统计结果是忽略掉这部分时间。详细结果如表 5 所示。

表 5　Iris 数据集实验结果

方案	加解密时间	密文计算时间	总时间	样本平均时间	密文大小
算法 4	3.57 ms	60.75 ms	64.32 ms	2.14 ms	385 KB
Jiang et al.[25]	3.08 ms	177.42 ms	180.50 ms	6.02 ms	385 KB
[17]	580 ms	1272 ms	1852 ms	61.7 ms	40980 KB

在 WBC 数据集中，总共有 683 个样本，被分为 2 种类别，其中每个样本含有 9 个特征，每个特征有 10 个不同的取值。我们使用其中 487（70%）个样本数据进行模型训练，使用 205 个样本数据进行预测。对我们的数据来说，实际上是两个 2×90 和 90×205 的矩阵相乘，在这种维度下，单条密文无法装下所有数据，因此需要分组进行计算。将 $205=9\times22+7$ 共 10 组预测分别进行计算。我们将方案和文献[17]中表 4 的结果进行比较，同上我们在统计时也忽略密文上取最大值的操作。需要注意的是，在 WBC 数据集中，使用 Jiang 的方案涉及的分块方式和我们的方案不同，需要按照方阵进行切分。详细结果如表 6 所示。

表 6　WBC 数据集实验结果

方案	加解密时间	密文计算时间	总时间	样本平均时间	密文大小
算法 4	32.93 ms	1300.23 ms	1333.16 ms	6.50 ms	3856 KB
Jiang et al.[25]	25.62 ms	1786.45 ms	1812.07 ms	8.84 ms	3085 KB
[17]	2773 ms	2680 ms	5453 ms	26.6 ms	180875 KB

对比实验结果表明，使用我们的方案更具有优势，每个样本最低只需要 2.14 毫秒就能进行预测，

同时方案的传输开销也更小。总体而言，我们的方案在时间和空间上更具有优势。

6 结束语

在本文中，首先提出了明文方阵-密文矩阵乘法方案，然后通过维数转换算法将明文方阵扩展到任意维数的明文矩阵。相对于先前的技术，该算法降低了密文旋转的复杂度，提高了运算效率。当矩阵维数 $p > m \geq n$ 时，完成矩阵乘法计算只需算法 4 且算法 4 的开销和 p 无关，因而明文-密文矩阵乘法的表现更为出色。最后我们将明文-密文矩阵乘法方案应用到贝叶斯分类器中，取得更好的实验结果。

明文-密文矩阵乘法方案也能支持密文-密文矩阵乘法，但密文矩阵乘法与之前方案相比，通信开销（密文数量）更大，因此我们不考虑密文-密文的情况。其次，尽管能实现任意维数的矩阵乘法，但是方案的本质是明文方阵-密文矩阵乘法，这可能会限制方案广泛的应用。最后，面对更大规模的矩阵，需要对矩阵分块进行计算。所以在接下来的工作中，我们需要进一步改进方案，使该方案能解除对方阵的限制，并能够胜任更复杂的密文-密文矩阵乘法。其次对矩阵分块进行分析，针对该方案提出相应的分块方案，并将方案应用到更多的机器学习模型中去。

7 致谢

感谢米波教授为我们指出最新文献[44]。本文工作是第二作者在访问自动推理与认知重庆市重点实验室期间完成的。

参考文献：

[1] LIN G, LI H, ZHANG Y, et al. Dynamic Momentum for Deep Learning with Differential Privacy; proceedings of the International Conference on Machine Learning for Cyber Security, F, 2022 [C]. Springer.

[2] DWORK C, KENTHAPADI K, MCSHERRY F, et al. Our data, ourselves: Privacy via distributed noise generation; proceedings of the Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St Petersburg, Russia, May 28-June 1, 2006 Proceedings 25, F, 2006 [C]. Springer.

[3] DWORK C, ROTH A J F, SCIENCE T I T C. The algorithmic foundations of differential privacy [J]. 2014, 9(3–4): 211-407.

[4] DWORK C. Differential privacy; proceedings of the International colloquium on automata, languages, and programming, F, 2006 [C]. Springer.

[5] DU W, ATALLAH M J. Secure multi-party computation problems and their applications: a review and open problems; proceedings of the Proceedings of the 2001 workshop on New security paradigms, F, 2001 [C].

[6] YAO A C-C. How to generate and exchange secrets; proceedings of the 27th annual symposium on foundations of computer science (Sfcs 1986), F, 1986 [C]. IEEE.

[7] MOHASSEL P, ZHANG Y. Secureml: A system for scalable privacy-preserving machine learning; proceedings of the 2017 IEEE symposium on security and privacy (SP), F, 2017 [C]. IEEE.

[8] YANG Q, LIU Y, CHEN T, et al. Federated machine learning: Concept and applications [J]. 2019, 10(2): 1-19.

[9] LI L, FAN Y, TSE M, et al. A review of applications in federated learning [J]. 2020, 149: 106854.

[10] MCMAHAN B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data; proceedings of the Artificial intelligence and statistics, F, 2017 [C]. PMLR.

[11] RIVEST R L, ADLEMAN L, DERTOUZOS M L J F O S C. On data banks and privacy homomorphisms [J]. 1978, 4(11): 169-80.

[12] GIACOMELLI I, JHA S, JOYE M, et al. Privacy-preserving ridge regression with only linearly-homomorphic encryption; proceedings of the Applied Cryptography and Network Security: 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings 16, F, 2018 [C]. Springer.

[13] HALL R, FIENBERG S E, NARDI Y J J O O S. Secure multiple linear regression based on homomorphic encryption [J]. 2011, 27(4): 669-91.

[14] HAN K, JEONG J, SOHN J H, et al. Efficient privacy preserving logistic regression inference and training [J]. 2020.

[15] YU X, ZHAO W, HUANG Y, et al. Privacy-Preserving Outsourced Logistic Regression on Encrypted Data from Homomorphic Encryption [J]. 2022, 2022.

[16] Y L, Y W W. Two-party Privacy-preserving Ridge Regression Scheme with Applications. [J]. Cryptologic Research, 2023, 10(02): 276-88.

[17] CHEN J, FENG Y, LIU Y, et al. Non-interactive Privacy-Preserving Naïve Bayes Classifier Using Homomorphic Encryption; proceedings of the International Conference on Security and Privacy in New Computing Environments, F, 2021 [C]. Springer.

[18] SUN X, ZHANG P, LIU J K, et al. Private machine learning classification based on fully homomorphic encryption [J]. 2018, 8(2): 352-64.

[19] TANG W, ZHOU Y, LI M, et al. Differential privacy preserving Naive Bayes classification via wavelet transform; proceedings of the 2020 International Conference on Networking and Network Applications (NaNA), F, 2020 [C]. IEEE.

[20] WOOD A, SHPILRAIN V, NAJARIAN K, et al. Private naive bayes classification of personal biomedical data: Application in cancer data analysis [J]. 2019, 105: 144-50.

[21] YASUMURA Y, ISHIMAKI Y, YAMANA H. Secure Naïve Bayes classification protocol over encrypted data using fully homomorphic encryption; proceedings of the Proceedings of the 21st International Conference on Information Integration and Web-based Applications & Services, F, 2019 [C].

[22] PANDA S J C E A. Principal component analysis using ckks homomorphic encryption scheme [J]. 2021.

[23] MA X J A P A. Improved Privacy-Preserving PCA Using Space-optimized Homomorphic Matrix Multiplication [J]. 2023.

[24] RATHEE D, MISHRA P K, YASUDA M. Faster PCA and linear regression through hypercubes in HElib; proceedings of the Proceedings of the 2018 Workshop on Privacy in the Electronic Society, F, 2018 [C].

[25] JIANG X, KIM M, LAUTER K, et al. Secure outsourced matrix computation and application to neural networks; proceedings of the Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, F, 2018 [C].

[26] LEE J-W, KANG H, LEE Y, et al. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network [J]. 2022, 10: 30039-54.

[27] LEE E, LEE J-W, LEE J, et al. Low-complexity deep convolutional neural networks on fully homomorphic encryption using multiplexed parallel convolutions; proceedings of the International Conference on Machine Learning, F, 2022 [C]. PMLR.

[28] MIHARA K, YAMAGUCHI R, MITSUISHI M, et al. Neural network training with homomorphic encryption [J]. 2020.

[29] LOU Q, FENG B, CHARLES FOX G, et al. Glyph: Fast and accurately training deep neural networks on encrypted data [J]. 2020, 33: 9193-202.

[30] PODSCHWADT R, TAKABI D. Non-interactive privacy preserving recurrent neural network prediction with homomorphic encryption; proceedings of the 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), F, 2021 [C]. IEEE.

[31] GENTRY C. A fully homomorphic encryption scheme [M]. Stanford university, 2009.

[32] BRAKERSKI Z, GENTRY C, VAIKUNTANATHAN V J A T O C T. (Leveled) fully homomorphic encryption without bootstrapping [J]. 2014, 6(3): 1-36.

[33] FAN J, VERCAUTEREN F J C E A. Somewhat practical fully homomorphic encryption [J]. 2012.

[34] DUCAS L, MICCIANCIO D. FHEW: bootstrapping homomorphic encryption in less than a second; proceedings of the Annual international conference on the theory and applications of cryptographic techniques, F, 2015 [C]. Springer.

[35] CHILLOTTI I, GAMA N, GEORGIEVA M, et al. TFHE: fast fully homomorphic encryption over the torus [J]. 2020, 33(1): 34-91.

[36] CHEON J H, KIM A, KIM M, et al. Homomorphic encryption for arithmetic of approximate numbers; proceedings of the Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23, F, 2017 [C]. Springer.

[37] SMART N P, VERCAUTEREN F J D, CODES, CRYPTOGRAPHY. Fully homomorphic SIMD operations [J]. 2014, 71: 57-81.

[38] LIU F-H, WANG H. Batch bootstrapping II: bootstrapping in polynomial modulus only requires O~(1) fhe multiplications in amortization; proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, F, 2023 [C]. Springer.

[39] KIM D, GUYOT C J I T O I F, SECURITY. Optimized Privacy-Preserving CNN Inference With Fully Homomorphic Encryption [J]. 2023, 18: 2175-87.

[40] HALEVI S, SHOUP V. Algorithms in helib; proceedings of the Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34, F, 2014 [C]. Springer.

[41] LEIGHTON F T. Introduction to parallel algorithms and architectures: Arrays· trees· hypercubes [M]. Elsevier, 2014.

[42] HALEVI S, SHOUP V J J O C. Bootstrapping for helib [J]. 2021, 34(1): 7.

[43] HUANG H Z H. Secure matrix multiplication based on fully homomorphic encryption [J]. The Journal of Supercomputing, 2023, 79: 5064-85.

[44] HUANG Z, HONG C, WENG C, et al. More Efficient Secure Matrix Multiplication for Unbalanced Recommender Systems [J]. 2023, 20(01): 551-62.

[45] JANG J, LEE Y, KIM A, et al. Privacy-Preserving Deep Sequential Model with Matrix Homomorphic Encryption [Z]. Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security. Nagasaki, Japan; Association for Computing Machinery. 2022: 377–91.10.1145/3488932.3523253

[46] BABENKO M, GOLIMBLEVSKAIA E, TCHERNYKH A, et al. A

Comparative Study of Secure Outsourced Matrix Multiplication Based on Homomorphic Encryption [J]. 2023, 7(2): 84.

[47] DUONG D H, MISHRA P K, YASUDA M J T M M P. Efficient secure matrix multiplication over LWE-based homomorphic encryption [J]. 2016, 67(1): 69-83.

[48] YASUDA M, SHIMOYAMA T, KOGURE J, et al. Secure Statistical Analysis Using RLWE-Based Homomorphic Encryption, Cham, F, 2015 [C]. Springer International Publishing.

[49] YASUDA M, SHIMOYAMA T, KOGURE J, et al. New packing method in somewhat homomorphic encryption and its applications [J]. 2015, 8(13): 2194-213.

[50] MISHRA P K, DUONG D H, YASUDA M. Enhancement for secure multiple matrix multiplications over ring-LWE homomorphic encryption; proceedings of the Information Security Practice and Experience: 13th International Conference, ISPEC 2017, Melbourne, VIC, Australia, December 13–15, 2017, Proceedings 13, F, 2017 [C]. Springer.

[51] ALBRECHT M R, PLAYER R, SCOTT S J J O M C. On the concrete hardness of learning with errors [J]. 2015, 9(3): 169-203.

.

[作者简介]

刘洋（1984-），女，湖北咸宁人，博士，重庆交通大学副教授、硕士生导师，主要研究方向为形式化验证、网络信息安全等。

杨林翰（2000− ），男，重庆万州人，重庆交通大学硕士生，主要研究方向为信息安全、密文计算等。

陈经纬（1984-），男，四川巴中人，博士，中国科学院重庆绿色智能技术研究院副研究员、硕士生导师，主要研究方向为信息安全、格算法及其应用等。

吴文渊（1976-），男，四川成都人，博士，中国科学院重庆绿色智能技术研究院研究员、博士生导师，主要研究方向为符号数值计算、信息安全。

冯勇（1965-），男，四川宁南人，博士，中国科学院重庆绿色智能技术研究院研究员、博士生导师，主要研究方向为符号数值计算、信息安全。