Non-interactive privacy-preserving naïve Bayes classifier using homomorphic encryption

Jingwei Chen

Joint work with **Y. Feng**, **Y. Liu**, **W. Wu** & **G. Yang**

December, 2021 @ SPNCE 2021

MLaaS and privacy

Machine Learning as a Service (MLaaS)

- Learn a model from massive amounts of data
  - Data owner $\Longrightarrow$ model provider
- Infer predicted results for client's sample data
  - Client can easily obtain the predicted results.

MLaaS and privacy

Machine Learning as a Service (MLaaS)

- Learn a model from massive amounts of data
  - Data owner $\Longrightarrow$ model provider
- Infer predicted results for client's sample data
  - Client can easily obtain the predicted results.

Privacy concerns

- The model may be sensitive:
  - financial model, disease diagnosis, ...
- Sample data may be sensitive:
  - credit history, medical records, ...

Framework

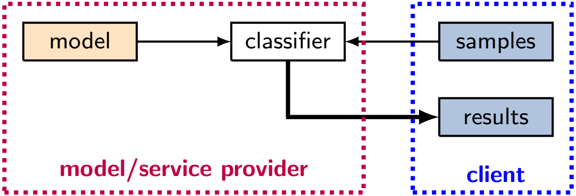


Figure: Framework of privacy-preserving classifiers

- The model is already known $\Longrightarrow$ no learning
- Model provider is also service provider $\Longrightarrow$ 2PC
- Light-blue boxes are encrypted $\Longrightarrow$ secure

Threat model

Adversaries are passive (honest-but-curious).

Related work

Here are several privacy-preserving naïve Bayes classifiers based on HE:

- [Bost *et al.* '15]: 2PC, Quadratic Residuosity + Paillier
- [Li, *et al.* '16]: 4PC, Paillier
- [Kim, *et al.* '18]: 4PC, BGV
- [Yasumura, *et al.* '19]: MPC, BGV
- [Sun, *et al.* '20]: 2PC, BGV
- ......

Related work

Here are several privacy-preserving naïve Bayes classifiers based on HE:

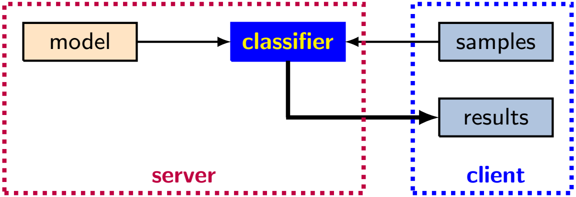
- [Bost *et al.* '15]: 2PC, Quadratic Residuosity + Paillier
- [Li, *et al.* '16]: 4PC, Paillier
- [Kim, *et al.* '18]: 4PC, BGV
- [Yasumura, *et al.* '19]: MPC, BGV
- [Sun, *et al.* '20]: 2PC, BGV
- ......

Common feature

During the classification phase, interactions are needed:

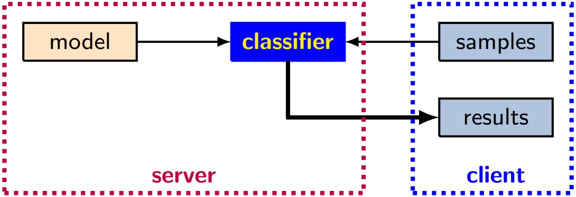
- communication burden
- (potential) information leakage

Contributions



We propose a privacy-preserving naïve Bayes classifier:

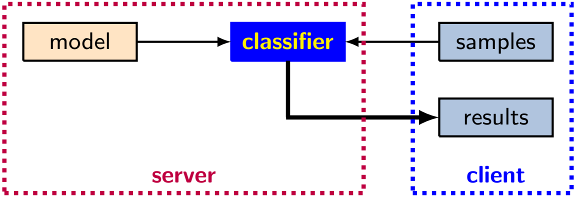
Contributions



We propose a privacy-preserving naïve Bayes classifier:

- **non-interactive**
  - Client sends an encryption of his sample data.
  - Server evaluates the model and sends the encrypted results.
  - Client decrypts the results to recovery the classification.

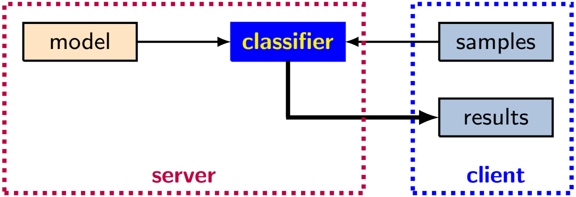
Contributions



We propose a privacy-preserving naïve Bayes classifier:

- **non-interactive**
  - Client sends an encryption of his sample data.
  - Server evaluates the model and sends the encrypted results.
  - Client decrypts the results to recovery the classification.
- **post-quantum safe**
  - Based on BGV

Contributions



We propose a privacy-preserving naïve Bayes classifier:

- **non-interactive**
  - Client sends an encryption of his sample data.
  - Server evaluates the model and sends the encrypted results.
  - Client decrypts the results to recovery the classification.
- **post-quantum safe**
  - Based on BGV

Main technique

An algorithm to compute **argmax** of an encrypted array.

Outline

Outline

Naïve Bayes classifier

Consider a data set with

- s categories $1, 2, \cdots, s$
- n features $X_1, \cdots, X_n$
- each feature X_k has at most t different values $1, 2, \cdots, t$.

Then the classification of a sample $\boldsymbol{x} = (x_1, \cdots, x_n)$ is

$$s^* = \operatorname*{argmax}_{i=1,\ldots,s} \Pr[Y = i] \prod_{k=1}^{n} \Pr[X_k = x_k | Y = i],$$

- prior probability: $\Pr[Y = i]$
- likelihood: $\Pr[X_k = x_k | Y = i]$

Naïve Bayes classifier

Consider a data set with

- s categories $1, 2, \cdots, s$
- n features $X_1, \cdots, X_n$
- each feature X_k has at most t different values $1, 2, \cdots, t$.

Then the classification of a sample $\boldsymbol{x} = (x_1, \cdots, x_n)$ is

$$s^* = \underset{i=1,\ldots,s}{\operatorname{argmax}} \left\{ \log \Pr[Y = i] + \sum_{k=1}^{n} \log \Pr[X_k = x_k | Y = i] \right\}$$

- prior probability: $\Pr[Y = i]$
- likelihood: $\Pr[X_k = x_k | Y = i]$

Homomorphic encryption

A public key encryption scheme consists of

- KeyGen: $(\mathtt{sk}, \mathtt{pk}) \leftarrow \mathtt{KeyGen}(1^\lambda)$,
- Enc: $c \leftarrow \mathtt{Enc}(\mathtt{pk}, x)$ for $x \in \mathcal{P}$,
- Dec: $x \leftarrow \mathtt{Dec}(\mathtt{sk}, c)$ for $c \in \mathcal{C}$.

LHE/FHE

Let $\mathcal{F}$ be a set of function in $\{0,1\}^* \to \{0,1\}$. A public key scheme is $\mathcal{F}$-**homomorphic** if there exists an evaluation algorithm Eval s.t.

$$\forall f \in \mathcal{F},\ \forall x \in \mathcal{P},\ \mathtt{Dec}(\mathtt{sk}, \mathtt{Eval}(f, \mathtt{Enc}(\mathtt{pk}, x))) = f(x).$$

- **Leveled HE (LHE)**: $f \in \mathcal{F}$ has an *a priori* bound on the depth of its circuit.
- **Fully HE (FHE)**: $f \in \mathcal{F}$ can be an arbitrary function.

- LHE + **bootstrapping** $\Longrightarrow$ FHE

BGV and HElib

BGV scheme [Brakerski-Gentry-Vaikuntanathan '12]

- Efficient LHE, supporting FHE, RLWE-based
- $\mathcal{P} = R_p := \mathbb{Z}[X]/(\Phi_m(X), p) \cong \mathbb{F}_{p^d}^{\ell}$
  - $d = \mathrm{ord}(p)$ in $\mathbb{Z}_m^*$ and $\ell = \phi(m)/d$
  - Support SIMD operations
- $\mathcal{C} = R_q := \mathbb{Z}[X]/(\Phi_m(X), q)$
- It evaluates L-level circuits with $O(\lambda \cdot L^3)$ per-gate computation.

HElib: an implementation of BGV[1]

- Based on NTL[2], thread safe
- Batching based on SIMD operations
- "Assembly language" for HE:
  - `Add`, `Mul`, `Rotate`, `Shift`, `TotalSum`, `AddConst`, `MulConst`, ...

[1] https://github.com/homenc/HElib
[2] https://github.com/libntl/ntl

Outline

Plain matrix-encrypted vector multiplication

Input: $\boldsymbol{c}' = (c_i')_{i \le t} \in R_q^t$ (c_i' encrypts the ith entry of $\boldsymbol{z} = (z_i)_{i \le t}$), and $\mathtt{pk}$; $\boldsymbol{A} = (a_{i,j}) \in \mathbb{Z}^{s \times t}$.

Output: $\boldsymbol{c} = (c_i)_{i \le s}$ with $c_i = \mathtt{Enc}_{\mathtt{pk}}\left(\sum_{j=1}^{t} a_{i,j} z_j\right)$.

1: For $i = 1, \cdots, s$ do the following:
2: $\quad c_i \leftarrow \mathtt{Enc}_{\mathtt{pk}}(0)$;
3: $\quad$ For $j = 1, \cdots, t$, update $c_i := \mathtt{Add}_{\mathtt{pk}}(c_i, \mathtt{MulConst}_{\mathtt{pk}}(a_{i,j}, c_j'))$.
4: **return** $(c_i)_{i \le s}$.

Algorithm 1: Naïve plaintext matrix-encrypted vector multiplication

Plain matrix-encrypted vector multiplication

Input: $\boldsymbol{c}' = (c_i')_{i\le t} \in R_q^t$ (c_i' encrypts the ith entry of $\boldsymbol{z} = (z_i)_{i\le t}$), and $\texttt{pk}$; $\boldsymbol{A} = (a_{i,j}) \in \mathbb{Z}^{s\times t}$.

Output: $\boldsymbol{c} = (c_i)_{i\le s}$ with $c_i = \texttt{Enc}_{\texttt{pk}}\left(\sum_{j=1}^{t} a_{i,j} z_j\right)$.

1: For $i = 1, \cdots, s$ do the following:
2: $\quad c_i \leftarrow \texttt{Enc}_{\texttt{pk}}(0)$;
3: $\quad$ For $j = 1, \cdots, t$, update $c_i := \texttt{Add}_{\texttt{pk}}(c_i, \texttt{MulConst}_{\texttt{pk}}(a_{i,j}, c_j'))$.
4: **return** $(c_i)_{i\le s}$.

Algorithm 1: Naïve plaintext matrix-encrypted vector multiplication

- Only pt-ct multiplications and ct-ct additions are involved.
- It costs no ct-ct multiplication depth.

Comparison among encrypted integers [Iliashenko-Zucca '21]

The less-than function over $S = [0, (p-1)/2]$ is defined as

$$\mathtt{LT}_S(x, y) = \begin{cases} 1, & \text{if } 0 \leq x < y \leq (p-1)/2, \\ 0, & \text{if } 0 \leq y \leq x \leq (p-1)/2. \end{cases}$$

Comparison among encrypted integers [Iliashenko-Zucca '21]

The less-than function over $S = [0, (p-1)/2]$ is defined as

$$\mathtt{LT}_S(x,y) = \begin{cases} 1, & \text{if } 0 \le x < y \le (p-1)/2, \\ 0, & \text{if } 0 \le y \le x \le (p-1)/2. \end{cases}$$

It can be interpolated by the following polynomial over $\mathbb{F}_p$ of degree $p-1$:

$$\frac{p+1}{2}(x-y)^{p-1} + \sum_{i=1,odd}^{p-2} \left(\sum_{a=1}^{\frac{p-1}{2}} a^{p-1-i} \right) \cdot (x-y)^i.$$

Comparison among encrypted integers [Iliashenko-Zucca '21]

The less-than function over $S = [0, (p-1)/2]$ is defined as

$$\mathtt{LT}_S(x, y) = \begin{cases} 1, & \text{if } 0 \le x < y \le (p-1)/2, \\ 0, & \text{if } 0 \le y \le x \le (p-1)/2. \end{cases}$$

It can be interpolated by the following polynomial over $\mathbb{F}_p$ of degree $p-1$:

$$\frac{p+1}{2}(x-y)^{p-1} + \sum_{i=1,odd}^{p-2} \left(\sum_{a=1}^{\frac{p-1}{2}} a^{p-1-i} \right) \cdot (x-y)^i.$$

- Evaluating the polynomial homomorphically costs at most

$$\sqrt{p-3} + \frac{3}{2}\log_2(p-3) + O(1)$$

ct-ct multiplication depth.

Argmax of an encrypted array

Given an array $\boldsymbol{z} = (z_0, \cdots, z_{t-1})$,

$$\arg\max_i(\boldsymbol{z}) = \sum_{j=0}^{t-1} j \cdot \prod_{k=0, k\neq j}^{t-1} (1 - \mathtt{LT}(z_j, z_k)).$$

Argmax of an encrypted array

Given an array $\boldsymbol{z} = (z_0, \cdots, z_{t-1})$,

$$\arg\max_i(\boldsymbol{z}) = \sum_{j=0}^{t-1} j \cdot \prod_{k=0, k\neq j}^{t-1} (1 - \mathtt{LT}(z_j, z_k)).$$

Evaluating arg max costs:

- $t(t-1)/2$ comparisons and
- at most
$$\log_2 t + \sqrt{p-3} + \frac{3}{2}\log_2(p-3) + O(1)$$
ct-ct multiplication depth.

Outline

Privacy-preserving naïve Bayes classifier

Input of client: A sample $\boldsymbol{x} = (i_1, \cdots, i_n)$, $\texttt{sk}$ and $\texttt{pk}$.

Input of server: Likelihood and prior information: $(\boldsymbol{A}_k)_{k \le n}$, $(b_i)_{i \le s}$, and $\texttt{pk}$.

1: The client encode $\boldsymbol{x}$ to a matrix $(\boldsymbol{e}_{i_1}, \cdots, \boldsymbol{e}_{i_n}) \in \{0,1\}^{t \times n}$.
2: The client encrypts $\boldsymbol{e}_{i_k}$ and sends the ciphertexts to the server.
3: The server does the following:
4: For $i = 1, \cdots, s$, set $c_i \leftarrow \texttt{Enc}_{\texttt{pk}}(0)$ and $c_i := \texttt{AddConst}_{\texttt{pk}}(c_i, b_i)$.
5: For $k = 1, \cdots, n$, calling Algorithm 1 with input as the ciphertexts of $\boldsymbol{e}_{i_k}$, $\boldsymbol{A}_k$ and $\texttt{pk}$ outputs $(c_i')_{i \le s}$. Update $c_i := \texttt{Add}_{\texttt{pk}}(c_i, c_i')$ for $i = 1, \cdots, s$.
6: Calling arg max with input as $\boldsymbol{c} = (c_i)_{i \le s}$ and $\texttt{pk}$ returns c.
7: The server sends c to the client.
8: The client decrypts c to $y = \texttt{Dec}_{\texttt{sk}}(c)$.

Protocol 1: Privacy-preserving naïve Bayes classifier

Privacy-preserving naïve Bayes classifier

Input of client: A sample $\boldsymbol{x} = (i_1, \cdots, i_n)$, $\mathtt{sk}$ and $\mathtt{pk}$.
Input of server: Likelihood and prior information: $(\boldsymbol{A}_k)_{k \le n}$, $(b_i)_{i \le s}$, and $\mathtt{pk}$.

1: The client encode $\boldsymbol{x}$ to a matrix $(\boldsymbol{e}_{i_1}, \cdots, \boldsymbol{e}_{i_n}) \in \{0,1\}^{t \times n}$.
2: The client encrypts $\boldsymbol{e}_{i_k}$ and sends the ciphertexts to the server.
3: The server does the following:
4: For $i = 1, \cdots, s$, set $c_i \leftarrow \mathtt{Enc}_{\mathtt{pk}}(0)$ and $c_i := \mathtt{AddConst}_{\mathtt{pk}}(c_i, b_i)$.
5: For $k = 1, \cdots, n$, calling Algorithm 1 with input as the ciphertexts of $\boldsymbol{e}_{i_k}$, $\boldsymbol{A}_k$ and $\mathtt{pk}$ outputs $(c_i')_{i \le s}$. Update $c_i := \mathtt{Add}_{\mathtt{pk}}(c_i, c_i')$ for $i = 1, \cdots, s$.
6: Calling arg max with input as $\boldsymbol{c} = (c_i)_{i \le s}$ and $\mathtt{pk}$ returns c.
7: The server sends c to the client.
8: The client decrypts c to $y = \mathtt{Dec}_{\mathtt{sk}}(c)$.

Protocol 1: Privacy-preserving naïve Bayes classifier

- No interaction during classification phase (Step 3–6)
- Secure classification without sacrificing privacy, assuming secure HE

Outline

Iris data set[3]

- Data set:
  - $s = 3$, $n = 4$, $t = 5$
  - 150 samples = 120 for trainning (80%) + 30 for testing
- HElib parameters:
  - $p = 37$, $m = 14539$ ($\Longrightarrow \ell = 1980$ slots), $\log q \approx 387$
  - The security parameter $\lambda \approx 100$
- Performance (Ubuntu 20.04/Intel i7-10750H/16GB):
  - 5.42s for 30 testing samples
  - $\approx$ **0.18**s for each
  - Supporting at most 1980 testing samples
    - Amortized costs $\approx$ **2.7**ms per sample
- Accuracy: $\approx 97\%$

[3] http://archive.ics.uci.edu/ml

Wisconsin breast cancer data set[3]

- Data set:
  - $s = 2$, $n = 9$, $t = 10$
  - 683 samples = 478 for trainning (70%) + 205 for testing
- HElib parameters:
  - $p = 113$, $m = 12883$ ($\Longrightarrow \ell = 3960$ slots), $\log q \approx 382$
  - The security parameter $\lambda \approx 100$
- Performance (Ubuntu 20.04/Intel i7-10750H/16GB):
  - 4.75s for 205 testing samples
  - $\approx$ **23**ms for each
  - Supporting at most 3960 testing samples
    - Amortized costs $\approx$ **1.2**ms per sample
- Accuracy: $\approx 97\%$

[3] http://archive.ics.uci.edu/ml

Conclusion

- We proposed a privacy-preserving naïve Bayes classifier based on BGV.
- R-LWE based LHE $\Longrightarrow$ secure against quantum attackers
- Argmax of encrypted arrays $\Longrightarrow$ non-interactive
- Batching $\Longrightarrow$ efficient

Conclusion

- We proposed a privacy-preserving naïve Bayes classifier based on BGV.
- R-LWE based LHE $\Longrightarrow$ secure against quantum attackers
- Argmax of encrypted arrays $\Longrightarrow$ non-interactive
- Batching $\Longrightarrow$ efficient

Future work

- More efficient
- More classifiers

Conclusion

- We proposed a privacy-preserving naïve Bayes classifier based on BGV.
- R-LWE based LHE $\Longrightarrow$ secure against quantum attackers
- Argmax of encrypted arrays $\Longrightarrow$ non-interactive
- Batching $\Longrightarrow$ efficient

Future work

- More efficient
- More classifiers

The full version is available at

`https://www.arcnl.org/jchen/download/ppnb_full.pdf`

THANKS