k-Nearest Neighbor Classification over Semantically Secure Encrypted Relational Data

Reporter: Ximeng Liu
Supervisor: Rongxing Lu

School of EEE, NTU
http://www.ntu.edu.sg/home/rxlu/seminars.htm

May 9, 2014
1 Main References

2 Introduction

3 Background

4 Basic Protocols

5 The proposed scheme
Main References

k-Nearest Neighbor algorithm
Problem Definition

Suppose Alice owns a database $D$ of $n$ records $t_1, \ldots, t_n$ and $m + 1$ attributes. Let $t_{i,j}$ denote the $j$-th attribute value of record $t_i$. Initially, Alice encrypts her database attribute-wise, that is, she computes $E_{pk}(t_{i,j})$, for $1 \leq i \leq n$ and $1 \leq j \leq m + 1$, where column $(m + 1)$ contains the class labels. Let the encrypted database be denoted by $D'$. 
Problem Definition

Let Bob be an authorized user who wants to classify his input record \( q =< q_1, \cdots, q_m > \) by applying the k-NN classification method based on \( D' \). We refer to such a process as privacy-preserving k-NN (PPkNN) classification over encrypted data in the cloud. Formally, we define the PPkNN protocol as:

\[
PPkNN(D', q) \rightarrow c_q
\]

where \( c_q \) denotes the class label for \( q \) after applying k-NN classification method on \( D \) and \( q \).
a. Homomorphic Addition

\[ E_{pk}(a + b) = E_{pk}(a) \star E_{pk}(b) \mod N^2 \]

b. Homomorphic Multiplication

\[ E_{pk}(a \star b) = E_{pk}(a)^b \mod N^2 \]
Basic Privacy-Preserving Protocols

All of the below protocols are considered under two-party semi-honest setting. In particular, we assume the exist of two semi-honest parties $P_1$ and $P_2$ such that the Paillier’s secret key $sk$ is known only to $P_2$ whereas $pk$ is treated as public.

1. Multiplication (SM) Protocol: This protocol considers $P_1$ with input $(E_{pk}(a), E_{pk}(b))$ and outputs $E_{pk}(a \ast b)$ to $P_1$, where $a$ and $b$ are not known to $P_1$ and $P_2$. During this process, no information regarding $a$ and $b$ is revealed to $P_1$ and $P_2$. 

http://www.ntu.edu.sg/home/rxlu/seminars.htm
Secure Squared Euclidean Distance (SSED) Protocol: In this protocol, $P_1$ with input $(E_{pk}(X), E_{pk}(Y))$ and $P_2$ with $sk$ securely compute the encryption of squared Euclidean distance between vectors $X$ and $Y$. Here $X$ and $Y$ are $m$ dimensional vectors where $E_{pk}(X) = \langle E_{pk}(x_1), \cdots, E_{pk}(x_m) \rangle$ and $E_{pk}(Y) = \langle E_{pk}(y_1), \cdots, E_{pk}(y_m) \rangle$. The output of the SSED protocol is $E_{pk}(|X - Y|^2)$ which is known only to $P_1$.
Secure Bit-Decomposition (SBD) Protocol: $P_1$ with input $E_{pk}(z)$ and $P_2$ securely compute the encryptions of the individual bits of $z$, where $0 \leq z < 2^l$. The output $[z] = \langle E_{pk}(z_1), \cdots, E_{pk}(z_l) \rangle$ is known only to $P_1$. Here $z_1$ and $z_l$ are the most and least significant bits of integer $z$, respectively.
Secure Minimum (SMIN) Protocol: In this protocol, $P_1$ holds private input $(u', v')$ and $P_2$ holds $sk$, where $u' = ([u], E_{pk}(s_u))$ and $v = ([v], E_{pk}(s_v))$. Here $s_u$ (resp., $s_v$) denotes the secret associated with $u$ (resp., $v$). The goal of SMIN is for $P_1$ and $P_2$ to jointly compute the encryptions of the individual bits of minimum number between $u$ and $v$. In addition, they compute $E_{pk}(s_{min}(u, v))$. That is, the output is $([min(u, v)], E_{pk}(s_{min}(u, v)))$ which will be known only to $P_1$. During this protocol, no information regarding the contents of $u$, $v$, $s_u$, and $s_v$ is revealed to $P_1$ and $P_2$. 
Basic Privacy-Preserving Protocols

Secure Minimum out of $n$ Numbers ($SMIN_n$) Protocol: In this protocol, we consider $P_1$ with $n$ encrypted vectors ([$d_1$, ⋯, $d_n$]) along with their respective encrypted secrets and $P_2$ with $sk$. Here $[d_i] = E_{pk}(d_{i,1}), \cdots, E_{pk}(d_{i,l})$ where $d_{i,1}$ and $d_{i,l}$ are the most and least significant bits of integer $d_i$ respectively, for $1 \leq i \leq n$. The secret of $d_i$ is given by $s_{d_i}$. $P_1$ and $P_2$ jointly compute $[\text{min}(d_1, \cdots, d_n)]$. In addition, they compute $E_{pk}(s_{\text{min}(d_1,\cdots,d_n)})$. At the end of this protocol, the output $([\text{min}(d_1, \cdots, d_n)], E_{pk}(s_{\text{min}(d_1,\cdots,d_n)}))$ is known only to $P_1$. During the $SMIN_n$ protocol, no information regarding any of $d_i$’s and their secrets is revealed to $P_1$ and $P_2$. 

http://www.ntu.edu.sg/home/rxlu/seminars.htm
Secure Bit-OR (SBOR) Protocol: $P_1$ with input $(E_{pk}(o_1), E_{pk}(o_2))$ and $P_2$ securely compute $E_{pk}(o_1 \lor o_2)$, where $o_1$ and $o_2$ are two bits. The output $E_{pk}(o_1 \lor o_2)$ is known only to $P_1$. 
Secure Multiplication (SM)

\[ a \ast b = (a + r_a) \ast (b + r_b) - a \ast r_a - b \ast r_b - r_a \ast r_b \]

Note that, for any given \( x \in Z_N \), \( N - x \) is equivalent to \(-x\) under \( Z_N \).
Secure Multiplication (SM)

Algorithm 1 SM($E_{pk}(a), E_{pk}(b)) \rightarrow E_{pk}(a \cdot b)$

Require: $P_1$ has $E_{pk}(a)$ and $E_{pk}(b)$; $P_2$ has $sk$

1: $P_1$:
   (a). Pick two random numbers $r_a, r_b \in \mathbb{Z}_N$
   (b). $a' \leftarrow E_{pk}(a) * E_{pk}(r_a)$
   (c). $b' \leftarrow E_{pk}(b) * E_{pk}(r_b)$; send $a', b'$ to $P_2$

2: $P_2$:
   (a). Receive $a'$ and $b'$ from $P_1$
   (b). $h_a \leftarrow D_{sk}(a')$; $h_b \leftarrow D_{sk}(b')$
   (c). $h \leftarrow h_a * h_b \mod N$
   (d). $h' \leftarrow E_{pk}(h)$; send $h'$ to $P_1$

3: $P_1$:
   (a). Receive $h'$ from $P_2$
   (b). $s \leftarrow h' * E_{pk}(a)^{N-r_b}$
   (c). $s' \leftarrow s * E_{pk}(b)^{N-r_a}$
   (d). $E_{pk}(a \cdot b) \leftarrow s' * E_{pk}(r_a * r_b)^{N-1}$

http://www.ntu.edu.sg/home/rxlu/seminars.htm
Secure Squared Euclidean Distance (SSED)

Algorithm 2 SSED\( (E_{pk}(X), E_{pk}(Y)) \rightarrow E_{pk}(|X - Y|^2) \)

Require: \( P_1 \) has \( E_{pk}(X) \) and \( E_{pk}(Y) \); \( P_2 \) has \( sk \)
1: \( P_1 \), for \( 1 \leq i \leq m \) do:
   (a). \( E_{pk}(x_i - y_i) \leftarrow E_{pk}(x_i) * E_{pk}(y_i)^{N-1} \)
2: \( P_1 \) and \( P_2 \), for \( 1 \leq i \leq m \) do:
   (a). Compute \( E_{pk}((x_i - y_i)^2) \) using the SM protocol
3: \( P_1 \) computes \( E_{pk}(|X - Y|^2) \leftarrow \prod_{i=1}^{m} E_{pk}((x_i - y_i)^2) \)
We assume that $P_1$ has $E_{pk}(z)$ and $P_2$ has $sk$, where $z$ is not known to both parties and $0 \leq z < 2^l$. Given $E_{pk}(z)$, the goal of the secure bit-decomposition (SBD) protocol is to compute the encryptions of the individual bits of binary representation of $z$. That is, the output is $[z] = \langle E_{pk}(z_1), \cdots, E_{pk}(z_l) \rangle$, where $z_1$ and $z_l$ denote the most and least significant bits of $z$ respectively. At the end, the output $[z]$ is known only to $P_1$. During this process, neither the value of $z$ nor any $z_i$’s is revealed to $P_1$ and $P_2$. 
Secure Minimum (SMIN)

the basic idea of the proposed SMIN protocol is for $P_1$ to randomly choose the functionality $F$ (by flipping a coin), where $F$ is either $u > v$ or $v > u$, and to obliviously execute $F$ with $P_2$. Since $F$ is randomly chosen and known only to $P_1$, the result of the functionality $F$ is oblivious to $P_2$. Based on the comparison result and chosen $F$, $P_1$ computes $[\min(u, v)]$ and $E_{pk}(s_{\min(u,v)})$ locally using homomorphic properties.
Secure Minimum (SMIN)

Algorithm 3 SMIN($u', v'$) → ([min($u$, $v$)], $E_{pk}(s_{min(u,v)})$)

Require: $P_1$ has $u' = ([u], E_{pk}(s_u))$ and $v' = ([v], E_{pk}(s_v))$, where $0 \leq u, v < 2^l$; $P_2$ has $sk$

1: $P_1$

(a) Randomly choose the functionality $F$
(b) for $i = 1$ to $l$ do:

- $E_{pk}(u_i \ast v_i) \leftarrow SM(E_{pk}(u_i), E_{pk}(v_i))$
- $T_i \leftarrow E_{pk}(u_i \oplus v_i)$
- $H_i \leftarrow H_i^{-1} \ast T_i; r_i \in R \mathbb{Z}_N$ and $H_0 = E_{pk}(0)$
- $\Phi_i \leftarrow E_{pk}(-1) \ast H_i$
- if $F : u > v$ then $W_i \leftarrow E_{pk}(u_i) \ast E_{pk}(u_i \ast v_i)^{N-1}$ and $\Gamma_i \leftarrow E_{pk}(v_i - u_i) \ast E_{pk}(r_i); \hat{r}_i \in R \mathbb{Z}_N$
  else $W_i \leftarrow E_{pk}(v_i) \ast E_{pk}(u_i \ast v_i)^{N-1}$ and $\Gamma_i \leftarrow E_{pk}(u_i - v_i) \ast E_{pk}(r_i); \hat{r}_i \in R \mathbb{Z}_N$
- $L_i \leftarrow W_i \ast \Phi_i^i; r_i' \in R \mathbb{Z}_N$

(c) if $F : u > v$ then: $\delta \leftarrow E_{pk}(s_u - s_u) \ast E_{pk}(\hat{r})$
  else $\delta \leftarrow E_{pk}(s_u - s_v) \ast E_{pk}(\hat{r})$, where $\hat{r} \in R \mathbb{Z}_N$
(d) $\Gamma' \leftarrow \pi_1(\Gamma)$ and $L' \leftarrow \pi_2(L)$
(e) Send $\delta, \Gamma'$ and $L'$ to $P_2$
Secure Minimum (SMIN)

2: $P_2$:

(a). Decryption: $M_i \leftarrow D_{sk}(L'_i)$, for $1 \leq i \leq l$

(b). If $\exists j$ such that $M_j = 1$ then $\alpha \leftarrow 1$

else $\alpha \leftarrow 0$

(c). If $\alpha = 0$ then:

- $M'_i \leftarrow E_{pk}(0)$, for $1 \leq i \leq l$
- $\delta' \leftarrow E_{pk}(0)$

else

- $M'_i \leftarrow \Gamma'_i \ast r_N$, where $r \in_R \mathbb{Z}_N$ and is different for $1 \leq i \leq l$
- $\delta' \leftarrow \delta \ast r'_N$, where $r'_\delta \in_R \mathbb{Z}_N$

(d). Send $M', E_{pk}(\alpha)$ and $\delta'$ to $P_1$
Secure Minimum (SMIN)

3. $P_1$:
   
   (a) $\overline{M} \leftarrow \pi^{-1}_i(M')$ and $\theta \leftarrow \delta' \ast E_{pk}(\alpha)^{N-r}
   
   (b) $\lambda_i \leftarrow \overline{M}_i \ast E_{pk}(\alpha)^{N-r_i}$, for $1 \leq i \leq l$
   
   (c) \textbf{if} $F : u > v$ \textbf{then:}

   - $E_{pk}(s_{\min(u,v)}) \leftarrow E_{pk}(s_u) \ast \theta$
   - $E_{pk}(\min(u,v)_i) \leftarrow E_{pk}(u_i) \ast \lambda_i$, for $1 \leq i \leq l$

   \textbf{else}

   - $E_{pk}(s_{\min(u,v)}) \leftarrow E_{pk}(s_v) \ast \theta$
   - $E_{pk}(\min(u,v)_i) \leftarrow E_{pk}(v_i) \ast \lambda_i$, for $1 \leq i \leq l$
Secure Minimum (SMIN)

- Compute the encrypted bit-wise XOR between the bits $u_i$ and $v_i$ as $T_i = E_{pk}(u_i \oplus v_i)$ using the formulation:

$$T_i = E_{pk}(u_i) \cdot E_{pk}(v_i) \cdot E_{pk}(u_i \cdot v_i)^{N-2}$$

- Compute an encrypted vector $H$ by preserving the first occurrence of $E_{pk}(1)$ (if there exists one) in $T$ by initializing $H_0 = E_{pk}(0)$. The rest of the entries of $H$ are computed as $H_i = H_{i-1}^T \cdot T_i$. We emphasize that at most one of the entry in $H$ is $E_{pk}(1)$ and the remaining entries are encryptions of either 0 or a random number.

- Then, $P_1$ computes $\Phi_i = E_{pk}(-1) \cdot H_i$. Note that “−1” is equivalent to “$N - 1$” under $\mathbb{Z}_N$. From the above discussions, it is clear that $\Phi_i = E_{pk}(0)$ at most once since $H_i$ is equal to $E_{pk}(1)$ at most once. Also, if $\Phi_j = E_{pk}(0)$, then index $j$ is the position at which the bits of $u$ and $v$ differ first (starting from the most significant bit position).
Secure Minimum (SMIN)

- If $F: u > v$, compute

$$W_i = E_{pk}(u_i) \times E_{pk}(u_i \times v_i)^{N-1}$$
$$= E_{pk}(u_i \times (1 - v_i))$$
$$\Gamma_i = E_{pk}(v_i - u_i) \times E_{pk}(\hat{r}_i)$$
$$= E_{pk}(v_i - u_i + \hat{r}_i)$$

- If $F: v > u$, compute:

$$W_i = E_{pk}(v_i) \times E_{pk}(u_i \times v_i)^{N-1}$$
$$= E_{pk}(v_i \times (1 - u_i))$$
$$\Gamma_i = E_{pk}(u_i - v_i) \times E_{pk}(\hat{r}_i)$$
$$= E_{pk}(u_i - v_i + \hat{r}_i)$$
Secure Minimum out of n Numbers \((SMIN_n)\)

**Algorithm 4 SMIN_n([d_1], \ldots, [d_n]) \rightarrow [d_{\min}]**

**Require:** \(P_1\) has \(([d_1], \ldots, [d_n])\); \(P_2\) has sk

1. \(P_1:\)
   1. \([d'_i] \leftarrow [d_i], \text{ for } 1 \leq i \leq n, \text{ and } \text{num} \leftarrow n\)
2. \(P_1\) and \(P_2\), for \(i = 1\) to \(\lceil \log_2 n \rceil\):
   1. \(\text{for } 1 \leq j \leq \left\lfloor \frac{\text{num}}{2} \right\rfloor:\)
      - if \(i = 1\) then:
        - \([d'_{2j-1}] \leftarrow \text{SMIN}([d'_{2j-1}], [d'_{2j}])\)
        - \([d'_{2j}] \leftarrow 0\)
      - else
        - \([d'_{2i(j-1)+1}] \leftarrow \text{SMIN}([d'_{2i(j-1)+1}], [d'_{2ij-1}])\)
        - \([d'_{2ij-1}] \leftarrow 0\)
   2. \(\text{num} \leftarrow \left\lfloor \frac{\text{num}}{2} \right\rfloor\)
3. \(P_1\) sets \([d_{\min}]\) to \([d'_1]\)
Secure Bit-OR (SBOR)

Suppose $P_1$ holds $(E_{pk}(o_1), E_{pk}(o_2))$ and $P_2$ holds $sk$, where $o_1$ and $o_2$ are two bits not known to both parties. The goal of the SBOR protocol is to securely compute $E_{pk}(o_1 \lor o_2)$. At the end of this protocol, only $P_1$ knows $E_{pk}(o_1 \lor o_2)$. During this process, no information related to $o_1$ and $o_2$ is revealed to $P_1$ and $P_2$. Given the secure multiplication (SM) protocol, $P_1$ can compute $E_{pk}(o_1 \lor o_2)$ as follows:

http://www.ntu.edu.sg/home/rxlu/seminars.htm
$P_1$ with input $(E_{pk}(o_1), E_{pk}(o_2))$ and $P_2$ involve in the SM protocol. At the end of this step, the output $E_{pk}(o_1 \ast o_2)$ is known only to $P_1$. Note that, since $o_1$ and $o_2$ are bits,

$E_{pk}(o_1 \ast o_2) = E_{pk}(o_1 \land o_2)$.

$E_{pk}(o_1 \lor o_2) = E_{pk}(o_1 + o_2) \ast E_{pk}(o_1 \land o_2)^{N-1}$
Basic scheme

Algorithm 5 $Sk\text{NN}_b(E_{pk}(T), Q) \rightarrow \langle t'_1, \ldots, t'_k \rangle$

Require: $C_1$ has $E_{pk}(T)$; $C_2$ has $sk$; Bob has $Q$

1: Bob:
   (a). Compute $E_{pk}(q_j)$, for $1 \leq j \leq m$
   (b). Send $E_{pk}(Q) = \langle E_{pk}(q_1), \ldots, E_{pk}(q_m) \rangle$ to $C_1$

2: $C_1$ and $C_2$:
   (a). $C_1$ receives $E_{pk}(Q)$ from Bob
   (b). for $i = 1$ to $n$ do:
      • $E_{pk}(d_i) \leftarrow \text{SSED}(E_{pk}(Q), E_{pk}(t_i))$
   (c). Send $\{\langle 1, E_{pk}(d_1) \rangle, \ldots, \langle n, E_{pk}(d_n) \rangle\}$ to $C_2$

3: $C_2$:
   (a). Receive $\{\langle 1, E_{pk}(d_1) \rangle, \ldots, \langle n, E_{pk}(d_n) \rangle\}$ from $C_1$
   (b). $d_i \leftarrow D_{sk}(E_{pk}(d_i))$, for $1 \leq i \leq n$
   (c). Generate $\delta \leftarrow \langle i_1, \ldots, i_k \rangle$, such that $\langle d_{i_1}, \ldots, d_{i_k} \rangle$ are the top $k$ smallest distances among $\langle d_1, \ldots, d_n \rangle$
   (d). Send $\delta$ to $C_1$
Basic scheme

4: $C_1$:
   (a) Receive $\delta$ from $C_2$
   (b) for $1 \leq j \leq k$ and $1 \leq h \leq m$ do:
       - $\gamma_{j,h} \leftarrow E_{pk}(t_{i,j,h}) \cdot E_{pk}(r_{j,h})$, where $r_{j,h} \in_R Z_N$
       - Send $\gamma_{j,h}$ to $C_2$ and $r_{j,h}$ to Bob

5: $C_2$:
   (a) for $1 \leq j \leq k$ and $1 \leq h \leq m$ do:
       - Receive $\gamma_{j,h}$ from $C_1$
       - $\gamma'_{j,h} \leftarrow D_{sk}(\gamma_{j,h})$; send $\gamma'_{j,h}$ to Bob

6: Bob:
   (a) for $1 \leq j \leq k$ and $1 \leq h \leq m$ do:
       - Receive $r_{j,h}$ from $C_1$ and $\gamma'_{j,h}$ from $C_2$
       - $t'_{j,h} \leftarrow \gamma'_{j,h} - r_{j,h}$ mod $N$
**Fully Secure kNN Protocol**

Algorithm 6 $SkNN_m(E_{pk}(T), Q) \rightarrow \langle t'_1, \ldots, t'_k \rangle$

**Require:** $C_1$ has $E_{pk}(T)$ and $\pi$; $C_2$ has $sk$; Bob has $Q$

1. Bob sends $E_{pk}(Q) = \langle E_{pk}(q_1), \ldots, E_{pk}(q_m) \rangle$ to $C_1$
2. $C_1$ and $C_2$:
   (a). $C_1$ receives $E_{pk}(Q)$ from Bob
   (b). for $i = 1$ to $n$ do:
      - $E_{pk}(d_i) \leftarrow \text{SSED}(E_{pk}(Q), E_{pk}(t_i))$
      - $[d_i] \leftarrow \text{SBD}(E_{pk}(d_i))$
3. for $s = 1$ to $k$ do:
   (a). $C_1$ and $C_2$:
      - $[d_{\text{min}}] \leftarrow \text{SMIN}_n([d_1], \ldots, [d_n])$
   (b). $C_1$:
      - $E_{pk}(d_{\text{min}}) \leftarrow \prod_{\gamma=0}^{l-1} E_{pk}(d_{\text{min},\gamma+1})^{2^{l-\gamma-1}}$
      - if $s \neq 1$ then, for $1 \leq i \leq n$
         - $E_{pk}(d_i) \leftarrow \prod_{\gamma=0}^{l-1} E_{pk}(d_i,\gamma+1)^{2^{l-\gamma-1}}$
      - for $i = 1$ to $n$ do:
         - $\tau_i \leftarrow E_{pk}(d_{\text{min}}) * E_{pk}(d_i)^{N-1}$
         - $\tau_i' \leftarrow \tau_i^{\pi'}$, where $\tau_i \in_R \mathbb{Z}_N$
         - $\beta \leftarrow \pi(\tau')$; send $\beta$ to $C_2$
c. $C_2$:  
- Receive $\beta$ from $C_1$  
- $\beta'_i \leftarrow D_{sk}(\beta_i)$, for $1 \leq i \leq n$  
- Compute $U_i$, for $1 \leq i \leq n$:  
  - if $\beta'_i = 0$ then $U_i = E_{pk}(1)$  
  - else $U_i = E_{pk}(0)$  
- Send $U$ to $C_1$

d. $C_1$:  
- Receive $U$ from $C_2$ and compute $V \leftarrow \pi^{-1}(U)$  
- $V'_{i,j} \leftarrow \text{SM}(V_i, E_{pk}(t_{i,j}))$, for $1 \leq i \leq n$ and $1 \leq j \leq m$  
- $E_{pk}(t'_{s,j}) \leftarrow \prod_{i=1}^{n} V'_{i,j}$, for $1 \leq j \leq m$  
- $E_{pk}(t'_{s}) = \langle E_{pk}(t'_{s,1}), \ldots, E_{pk}(t'_{s,m}) \rangle$

e. $C_1$ and $C_2$, for $1 \leq i \leq n$:  
- $E_{pk}(d_{i,\gamma}) \leftarrow \text{SBOR}(V_i, E_{pk}(d_{i,\gamma}))$, for $1 \leq \gamma \leq l$  

The rest of the steps are similar to steps 4-6 of $\text{SkNN}_b$
Thank you

Rongxing’s Homepage:
http://www.ntu.edu.sg/home/rxlu/index.htm

PPT available @: http://www.ntu.edu.sg/home/rxlu/seminars.htm

Ximeng’s Homepage:
http://www.liuximeng.cn/