Secure Processing of Similarity Queries

Bin Mu

The Graduate Center
City University of New York

May 16, 2014
1. Introduction
2. Tools on Secure Computation
3. Secure location proximity detection
4. Secure biometric data recognition
5. Secure similar document detection
6. Secure search over multimedia database
7. Conclusion and future work
1. Introduction

2. Tools on Secure Computation

3. Secure location proximity detection

4. Secure biometric data recognition

5. Secure similar document detection

6. Secure search over multimedia database

7. Conclusion and future work
Secure similarity queries

- Huge amount of digital data produced by all kinds of commercial, scientific, leisure-time and security applications.
- The extraction of useful information from these large data sets has become one of the key IT tasks.
- Insufficient meta-data description: similarity search is more important than an exact match or keyword search.
- However, constant monitoring of people's activities: privacy concern.
Secure similarity queries

- Huge amount of digital data produced by all kinds of commercial, scientific, leisure-time and security applications.
- The extraction of useful information from these large data sets has become one of the key IT tasks.
- Insufficient meta-data description: similarity search is more important than an exact match or keyword search.
- However, constant monitoring of people's activities: privacy concern.

Privacy-preserving query processing solves the problem!
Outline

1. Introduction
2. Tools on Secure Computation
3. Secure location proximity detection
4. Secure biometric data recognition
5. Secure similar document detection
6. Secure search over multimedia database
7. Conclusion and future work
A secure two-party computation protocol enables two parties to jointly compute a function based on their respective inputs, without having to reveal their input to the other party.

Two adversarial models:

- Semi-honest adversaries: the adversary follows the prescribed behavior; try to compute additional information during protocol execution.
- Malicious adversaries: the malicious adversary is allowed to deviate arbitrarily from the specified protocol. Adds the complexity of the problem: certain undesirable actions that cannot be prevented.
Let \( f : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}^* \times \{0, 1\}^* \) be a functionality and \( f_1(x, y) \) and \( f_2(x, y) \) denote the first and the second element of \( f(x, y) \) respectively. Let \( \Pi \) be two-party protocol for computing \( f \). The views of the first and the second parties during an execution of \( \Pi \) on \((x, y)\), denoted by \( \text{VIEW}_1^{\Pi}(x, y) \) and \( \text{VIEW}_2^{\Pi}(x, y) \), are \((x, r, m_1, \ldots, m_t)\) and \((y, r, m_1, \ldots, m_T)\) respectively, where \( r \) represents the outcome of the first and second parties’ internal coin tosses, and \( m_i \) represents the \( i^{th} \) message they have received. We say that \( \Pi \) privately computes \( f \) if there exist probabilistic polynomial-time algorithms (simulators), denoted \( S_1 \) and \( S_2 \), such that

\[
\{(S_1(x, f_1(x, y)), f(x, y))\}_{x, y} \equiv^C \{(\text{VIEW}_1^{\Pi}(x, y), \text{OUTPUT}^{\Pi}(x, y))\}_{x, y}
\]

\[
\{(S_2(x, f_2(x, y)), f(x, y))\}_{x, y} \equiv^C \{(\text{VIEW}_2^{\Pi}(x, y), \text{OUTPUT}^{\Pi}(x, y))\}_{x, y}
\]

where \( \equiv^C \) denotes computational indistinguishability.
Additively Homomorphic Encryption

Given the encryptions $E(m_1)$ and $E(m_2)$ of two messages $m_1, m_2$

- $E(m_1 + m_2) = E(m_1) \cdot E(m_2)$
- $E(m_1 - m_2) = E(m_1) \cdot E(m_2)^{-1}$
- $E(m_1 \cdot k) = E(m_1)^k$

Semantically secure, i.e. is infeasible to derive any information about
the plaintext, given its ciphertext and the public key
Additively Homomorphic Encryption

- Given the encryptions $E(m_1)$ and $E(m_2)$ of two messages $m_1, m_2$
  - $E(m_1 + m_2) = E(m_1) \cdot E(m_2)$
  - $E(m_1 - m_2) = E(m_1) \cdot E(m_2)^{-1}$
  - $E(m_1 \cdot k) = E(m_1)^k$

- Semantically secure, i.e. is infeasible to derive any information about the plaintext, given its ciphertext and the public key
1-out-of-2 OT:

Chooser

Alice

j

m_j

Sender

Bob

m_0, m_1

Learns nothing
1-out-of-N OT:

The parties learn nothing else:

- Indistinguishable to Sender which $\sigma$ is used
- Chooser learns no other value of $m_0, \ldots, m_{N-1}$
Outline

1. Introduction
2. Tools on Secure Computation
3. Secure location proximity detection
4. Secure biometric data recognition
5. Secure similar document detection
6. Secure search over multimedia database
7. Conclusion and future work
Ruppel et al. utilize a symmetric key cipher that encrypts locations by applying a distance-preserving transformation.

- A set of friends share a common key and use it to encrypt their location prior to uploading it to the server.
- Due to the distance-preserving property of the transformation, the server can determine whether any two friends are within a given proximity threshold.

*Longitude* by Mascetti et al is a similar approach, but the underlying transformation does not disclose the exact distances (i.e., it results in a loss of accuracy).

**Drawbacks:** The server learns the actual(relative) distances among all users.
If a user colludes with the server and reveals the shared key, all user locations are compromised.
Protocols with strong privacy guarantees

Protocols by Zhong et al. [PETS’07], Narayanan et al. [NDSS’11]

- The plane is divided into a grid.
- Employ a secure two-party computation protocol (between Alice and Bob) for equality testing
- Strong privacy guarantees, since collusions are not possible

Zhong’s Protocol:
Alice picks a resolution distance $r$.
Alice’s coordinates in the grid $(x_r, y_r) = (\lfloor \frac{x}{r} \rfloor, \lfloor \frac{y}{r} \rfloor)$.
Similarly Bob’s coordinates in the grid are $(u_r, v_r)$
Tools: Homomorphic cryptosystems (Only Alice knows the private key.)
Zhong’s Protocol

• Alice sends to Bob $r$, $E(x_r^2 + y_r^2)$, $E(2x_r)$, $E(2y_r)$.

• Bob picks three random elements $\rho_0, \rho_1, \rho_2$ in the plaintext space and replies with $E(\rho_0 \cdot D_r)$, $E(\rho_1 \cdot (D_r - 1))$, $E(\rho_2 \cdot (D_r - 2))$, where $D_r = (x_r - u_r)^2 + (y_r - v_r)^2$ is the square of the Euclidean distance between Alice and Bob in the grid.

$$E(\rho_0 \cdot D_r) = E((x_r - u_r)^2 + (y_r - v_r)^2)^{\rho_0} = E(x_r^2 + y_r^2)E(2x_r)^{-u_r}E(2y_r)^{-v_r}E(u_r^2)E(v_r^2)$$

• Alice decrypts the three values and knows whether Bob is nearby.
  • $\rho_0 \cdot D_r = 0$ if Alice and Bob are in the same grid cell and is a random element otherwise
  • $\rho_1 \cdot (D_r - 1) = 0$ if Alice and Bob are in adjacent grid cells
  • $\rho_2 \cdot (D_r - 2) = 0$ if Alice and Bob are in diagonally touching grid cells
Zhong’s Protocol

[Diagram showing a grid with labeled positions for Alice and Bob, distances labeled as $D_{i} = 0$, $D_{i} = 1$, $D_{i} = 2$, $D_{i} = 5$.]
Security of Zhong’s Protocol

- Alice’s view: \( E(\rho_0 \cdot D_r), E(\rho_1 \cdot (D_r - 1)), E(\rho_2 \cdot (D_r - 2)) \)
- Bob’s view: \( E(x_r^2 + y_r^2), E(2x_r), E(2y_r) \)

The Simulator:

- Simulation of Alice’s view: if Bob is nearby, an encryption of 0, two encryptions of random numbers; otherwise, three encryptions of random numbers
- Simulation of Bob’s view: three encryptions of 0
Outline

1. Introduction
2. Tools on Secure Computation
3. Secure location proximity detection
4. Secure biometric data recognition
5. Secure similar document detection
6. Secure search over multimedia database
7. Conclusion and future work
A common structure for biometric recognition systems:

- Feature extraction: from raw biometric data to feature vectors
- Distance computation: the distance between the client’s feature vector and each record in the server’s database
- Selection of the matching identities: one or more records matching the client’s biometric data are selected
Erkin’s Protocol:
Feature extraction: *Eigenfaces* algorithm

- Face images are transformed into characteristic feature vectors of a low-dimensional vector space (the face space), whose basis is composed of *eigenfaces*.
- The *Eigenfaces* are determined through Principal Component Analysis (PCA) from a set of training images.
- Every face image is represented as a vector in the face space by projecting the face image onto the subspace spanned by the eigenfaces.
Alice sends her homomorphic encrypted face image to Bob.

By homomorphc properties and formulas described in the paper, Bob obtains an encrypted feature vector of Alice’s face image as $E(\bar{\Omega}) := (E(\bar{\omega}_1), \ldots, E(\bar{\omega}_K))^T$.

Bob’s feature vector database is $\{\Omega_1, \ldots, \Omega_M\}$

Similarity metric: square of Euclidean distance

$$D(\Omega, \bar{\Omega}) = \| \Omega - \bar{\Omega} \|^2 = \sum_{i=1}^{K} (\omega_i - \bar{\omega}_i)^2 = \sum_{i=1}^{K} \omega_i^2 + \sum_{i=1}^{K} (-2\omega_i\bar{\omega}_i) + \sum_{i=1}^{K} \bar{\omega}_i^2$$
Secure Distance Computation

\[ D(\Omega, \bar{\Omega}) = \| \Omega - \bar{\Omega} \|^2 = \sum_{i=1}^{K} (\omega_i - \bar{\omega}_i)^2 = \sum_{i=1}^{K} \omega_i^2 + \sum_{i=1}^{K} (-2\omega_i\bar{\omega}_i) + \sum_{i=1}^{K} \bar{\omega}_i^2 \]

Let \( S_1 = \sum_{i=1}^{K} \omega_i^2, S_2 = \sum_{i=1}^{K} (-2\omega_i\bar{\omega}_i), S_3 = \sum_{i=1}^{K} \bar{\omega}_i^2. \) Then

\[ E(D(\Omega, \bar{\Omega})) = E(S_1 + S_2 + S_3) = E(S_1) \cdot E(S_2) \cdot E(S_3) \]

Bob knows \( \omega_i \) in plaintext, so he can compute

\[ E(S_1) = E(\sum_{i=1}^{K} \omega_i^2) = \prod_{i=1}^{K} E(\omega_i^2) \]

Bob can compute \( E(S_2) = E(\sum_{i=1}^{K} (-2\omega_i\bar{\omega}_i)) = \prod_{i=1}^{K} E(\bar{\omega}_i)^{-2\omega_i} \)
Secure Distance Computation

To compute $E(S_3) = E(\sum_{i=1}^{K} \bar{\omega}_i^2)$:

- Bob additively blinds the value $\bar{\omega}_i$ by computing $E(x_i) = E(\bar{\omega}_i + r_i) = E(\bar{\omega}_i) \cdot E(r_i)$ and sends it to Alice.
- Alice decrypts it to get $x_i$ and compute $S'_3 = \sum_{j=1}^{K} x_j^2$. She encrypts this value and sends it back to Bob.
- Now Bob can compute $E(S_3)$ as follows:

$$E(S_3) = E\left(\sum_{i=1}^{K} [(\bar{\omega}_i + r_i)^2 - 2r_i\bar{\omega}_i - r_i^2]\right) = E(S'_3) \cdot \prod_{i=1}^{K} (E(\bar{\omega}_i)^{-2r_i} \cdot E(-r_i^2))$$
Match finding

As a result of previous step, Bob obtained encrypted distances $E(D_1), \ldots, E(D_M)$.

Use a tournament to get the minimum distance:

- In the first round, Bob compares the $k = \left\lfloor \frac{M}{2} \right\rfloor$ encrypted distances $E(D_{2i+1})$ and $E(D_{2i+2})$ for $0 \leq i \leq k - 1$, by using a secure cryptographic protocol that compares two encrypted values.
- At the end of the first round, there will be $\left\lceil \frac{M}{2} \right\rceil$ encryptions left.
- After $\left\lceil \log_2(M) \right\rceil$ rounds there will only be one encryption left, the minimum.
The comparison protocol is secure.

All messages received by Bob are encrypted under Alice’s public keys which cannot be decrypted by him. Alice has access to the secret key. However, all messages received by Alice are masked by Bob, such that they are uniformly random over the whole plaintext space.

In addition to the comparisons, interaction only happens when computing the distances. As above, the values $x_1, \ldots, x_K$ that Alice receives are masked, so that Alice cannot learn any information about $\bar{\omega}_i$. Bob again only receives semantically secure encryptions, so he also learns nothing.
Feature exaction: Patch-Based Face Representation

- A face is represented by a collection of informative patches:

  Assume that the face is represented by $p$ patches.
A public database of $N$ faces $\Rightarrow$ A dictionary of $N$ values for each patch
Each patch is represented by the 4 closest patches in the dictionary.
Representing a face

- Initial representation: a vector with $p$ entries, each with 4 values in the range $[1, N]$.
- Binary representation: a binary vector of $p \cdot N$ bits, where $4p$ of bits equal 1.

For each of the $p$ patches, store indices of the 4 closest patches in the dictionary.
Define the difference between faces as the set difference between their representations:

$$\Delta(A, B) = |A \cup B| - |A \cap B|$$

Set difference $\equiv$ Hamming distance between binary representations of faces

Secure computation of Hamming distance is easy
The protocol in a nutshell

- Inputs are vectors $w = w_0, \cdots, w_{m-1}$; $w' = w'_0, \cdots, w'_{m-1}$.

- Client sends $E(w_0), \cdots, E(w_{m_1})$.

- Server uses homomorphic properties to compute
  \[
  E(w_j \oplus w'_j) = E(w_j + w'_j - 2w_jw'_j) = E(w_j)E(w'_j)E(w_j)^{-2w'_j}, \text{ for } j \in \{0, \cdots, m-1\}
  \]
  \[
  E(d) = E(d_H(w, w')) = E(\sum_{j=0}^{m-1} w_j \oplus w'_j) = \prod_{j=0}^{m-1} E(w_j \oplus w'_j)
  \]

- Server chooses a random $R$; sends $E(d + R)$ to client.

- Client decrypts $E(d + R)$, reduces the result by mod $d_{max} + 1$.

- Both parties run a 1-out-of-$(d_{max}+1)$ OT, where clients learns 1 if Hamming distance $<$ threshold.
Secure fingerprint recognition

A protocol by Barni et al.

Feature extraction: Fingercode.

- The fingerprint image is first divided into radial and angular sectors.
- The sectors are then filtered by a number of Gabor filters to determine the energy content of each sector into different frequency bands.
- The feature vector is formed by the sequence of a properly quantized version of sector energies.
Secure fingerprint recognition protocol

- The client’s input: a feature vector \( \vec{x} \) to be identified. The server’s input: a database of \( n \) pairs \((id^i, \vec{y}^i)\), where \( id^i \) is a identifier and \( \vec{y}^i \) is the corresponding feature vector. \((id^i = 2^i)\).

- The distance computation is similar to Erkin’s face recognition protocol.

- The server have the pairs: \((id^1, E(D^1)), \ldots, (id^n, E(D^n))\).

- The server permutes these pairs as: \((id^{j_1}, E(D^{j_1})), \ldots, (id^{j_n}, E(D^{j_n}))\) and then computes, using an improved secure comparison subprotocol, the values \( E(b^{j_i}) = \text{bit} - \text{MIN}(E(\tau), E(D^{j_i})) \) for \( i \in \{1, \ldots, n\} \).

- The server computes and returns to the client:

\[
E(R) = E\left(\sum_{i=1}^{n} b^i \cdot id^i\right) = \prod_{i=1}^{n} E(b^i)^{id^i}
\]

Note: The bit at position \( i \) in \( R \) is set to 1 if and only if the \( i \)-th identity matches
1. Introduction
2. Tools on Secure Computation
3. Secure location proximity detection
4. Secure biometric data recognition
5. Secure similar document detection
6. Secure search over multimedia database
7. Conclusion and future work
SSDD Protocol by Jiang et al.

Document representation: bag of words model

<table>
<thead>
<tr>
<th>Terms</th>
<th>$u$</th>
<th>$\tilde{u}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>2</td>
<td>0.30</td>
</tr>
<tr>
<td>b</td>
<td>3</td>
<td>0.45</td>
</tr>
<tr>
<td>c</td>
<td>5</td>
<td>0.76</td>
</tr>
<tr>
<td>d</td>
<td>1</td>
<td>0.15</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>0.30</td>
</tr>
<tr>
<td>f</td>
<td>1</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Table: Alice’s document

<table>
<thead>
<tr>
<th>Terms</th>
<th>$\nu_1$</th>
<th>$\nu_2$</th>
<th>$\tilde{\nu}_1$</th>
<th>$\tilde{\nu}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>2</td>
<td>1</td>
<td>0.30</td>
<td>0.20</td>
</tr>
<tr>
<td>b</td>
<td>3</td>
<td>4</td>
<td>0.45</td>
<td>0.78</td>
</tr>
<tr>
<td>c</td>
<td>5</td>
<td>2</td>
<td>0.76</td>
<td>0.39</td>
</tr>
<tr>
<td>d</td>
<td>1</td>
<td>1</td>
<td>0.15</td>
<td>0.20</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>2</td>
<td>0.30</td>
<td>0.39</td>
</tr>
<tr>
<td>f</td>
<td>1</td>
<td>0</td>
<td>0.15</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table: Bob’s collection

- Similarity metric: cosine of the angle between two document term vectors
- Compute $\cos$ $\equiv$ compute the dot product of two vectors.
- Tool: homomorphic encryption
The protocol

**Input:** Alice’s input is her document vector $\vec{u}$; Bob’s input is his database $\mathcal{D} = \{\vec{v}_1, \ldots, \vec{v}_m\}$. Assume $|\vec{u}| = |\vec{v}_i| = n$. Note that the private key is only known to Alice.

**Output:** Alice learns the similarity scores of her document to each document in Bob’s database: $\sigma_1, \ldots, \sigma_m$. Bob learns nothing.

1. **Alice:**
   
   (a). Compute $\vec{z}[i] \leftarrow E_{pk}(\vec{u}[i])$ for $i = 1, \ldots, n$;
   
   (b). Send $\vec{z}$ to Bob;

2. **Bob:** for $j = 1$ to $m$ do
   
   (a). $\vec{y}[i] \leftarrow (\vec{z}[i])\vec{v}_j[i]$, for $i = 1, \ldots, n$;
   
   (b). Compute the encrypted dot product $E_{pk}(\sigma_j) \leftarrow \prod_{i=1}^{n} \vec{y}[i]$
   
   and sends it to Alice

3. **Alice:** for for $j = 1$ to $m$ do
   
   Compute $\sigma_j = D_{pk}(E_{pk}(\sigma_j))$
Optimizations

Drawbacks of previous protocol:

- Numerous computationally expensive public key operations, especially when the term dictionary is large.
- Pairwise comparison between two collections: numerous secure computation of dot product

Improvement:

- Bob uses only the non-zero elements of his vector $\vec{v}$ to compute the products.
- To reduce comparisons between documents, each party cluster their document collection into $k$ clusters. Compare only the representatives to select a few clusters for pairwise comparison.
Protocol by Blundo et al.

Document representation: n-gram model
An n-gram is a substring of size n from a given string. If a text document is treated as a one big string (after removing all punctuation marks and whitespaces), under the n-gram model, each document can be represented by a set of successive n-grams.

<table>
<thead>
<tr>
<th>Text</th>
<th>3-gram representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u$ aabbc abbde</td>
<td>${aab, abb, bbc, bca, cab, abb, bbd, bde}$</td>
</tr>
<tr>
<td>$v_1$ ca aabbcddf</td>
<td>${caa, aaa, aab, abb, bbc, bcd, cdd, ddf}$</td>
</tr>
<tr>
<td>$v_2$ xaab xyyyz</td>
<td>${xaa, aab, abx, bxy, xyy, yyy, yyz}$</td>
</tr>
</tbody>
</table>

Table: 3-gram examples

Advantages over previous model: (1) improvement on finding *local similarity*, e.g., overlapping of pieces of texts. (2) language-dependent. (3) less sensitive to document modification.
Protocol by Blundo et al.

- Similarity metric: Jaccard Coefficient ($JC$)
  \[
  JC(u, v) = \frac{|u \cap v|}{|u \cup v|} = \frac{|u \cap v|}{|u| + |v| - |u \cap v|}
  \]

- Tools: Private Set Intersection Cardinality (PSI-CA)
Correctness: for any $a_i$ held by Alice and $b_j$ held by Bob, if $a_i = b_j$ (hence, $h_ia = hb_j$):

$$ta_i = H'(\beta'_i) = H'((\alpha'_i)^{(1/R_a)}) = H'(ha_i^{R_b}) = H'(hb_j^{R_b}) = H'(\beta_j) = tb_j$$
Secure computation of $JC(A, B)$

- Alice and Bob execute PSI-CA on input, respectively, $(A, |B|)$ and $(B, |A|)$
- Alice learns $c = |A \cap B|
- Alice computes $u = |A \cup B| = |A| + |B| - c$
- Alice outputs $JC(A, B) = c/u$
Document representation: bit vectors of fixed length from simhash
Simhash is a dimensionality reduction technique that encodes all the
document terms and their frequencies into a fixed-size bit vector. The
simhash of two similar documents will only differ in a few bit
positions.

Similarity metric: Hamming distance.

Tool: Homomorphic encryption
SSDD protocol in a nutshell

- Both parties’s inputs are simhash bit vectors corresponding to their documents.
- Computation of the similarity score (i.e. Hamming distance) $E(\sigma_i)$: Using homomorphic encryption, Bob firstly computes the encryption of the XOR operation of each bit vector dimension; then, sums the results of every dimension.
- After computation of the similarity score, Bob computes, the encryptions of $r_j \cdot (\sigma_i - j)$ where $j \in \{0, 1, \cdots, t\}$, $t$ is the threshold.
- Bob permutes these $(t + 1)$ ciphertexts corresponding to document $i$ in Bob’s database and sends them back to Alice.
- Alice concludes that the similarity score of Bob’s document $i$ is within the threshold $t$ if and only if one of decryptions equals 0.
Outline

1. Introduction
2. Tools on Secure Computation
3. Secure location proximity detection
4. Secure biometric data recognition
5. Secure similar document detection
6. Secure search over multimedia database
7. Conclusion and future work
Secure image search by Shashank et al.

Bag of words matching
Secure image search Shashank et al.

Vocabulary Tree
Use Private Information Retrieval (PIR) to get the node at next level. PIR allows a client to retrieve data at a particular index in a database without revealing the query to the server. Very similar to Oblivious Transfer, except that OT also maintains the privacy of the database.
Audio feature extraction: The spectrogram is used to calculate the sum energy in frequency bands, and the band energy values are compared to one another. The process is done on successive overlapping time segments of the audio.
Secure audio search

Server’s database:

Use Private Information Retrieval (PIR) to retrieve candidates.
Secure audio search

Use Private Information Retrieval (PIR) to retrieve candidates.
Two frameworks:

- Conventional voice-processing algorithms are rendered secure by computing them through secure operations. By recasting the computations as a pipeline of primitives, each of which be computed securely through a combination of homomorphic encryption, garbled circuits, and oblivious transfer.

- Speech matching transformed to string matching.
  - A data-length independent feature vector is firstly derived from the speech data
  - These vectors are converted to a collection of bit strings using locality sensitive hashing (LSH) schemes.
  - A cryptographic hash function $H[\cdot]$ (such as SHA-256 and MD5) is applied to the LSH results
  - Classification is performed by counting exact matches.
Conclusion and future work

Private location proximity detection

- Grid and homomorphic cryptosystems are combined to provide a strong privacy-preserving service
- Grid may not match a user’s area of interest exactly
- My work: secure point inclusion protocol to decide whether a point lies in an arbitrary convex polygon defined by the user.
- Drawback: if the area of the polygon is very small, location is leaked.
- Remedy: cryptographic commitment schemes that will allow the querier to commit his input prior to the protocol execution. If the proximity result is true, the querier will be forced to reveal its polygon to the queried, in order for him to verify that his privacy was not violated.
Conclusion and future work

For complex data, such as biometric and multimedia data, the secure protocols usually consist of three parts: feature extraction, distance computation, and selection of the matching records.

- A popular research topic: represent the raw data or features vectors as bit vectors
- Reduce nearest-neighbor searches to a Hamming distance comparison: efficient in both the private and non-private domains.
- My work: design a protocol that can provide secure image search over large public databases
- Techniques and tools: bit vector representation of image, clustering of images, homomorphic cryptosystems, etc.
The End