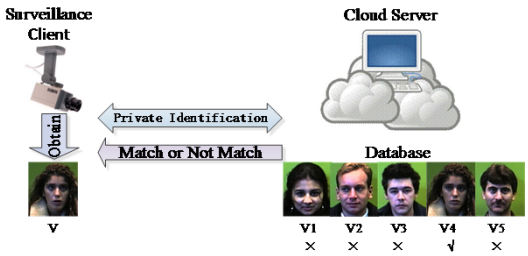
CCBR 2015

Privacy Preserving Face Identification in the Cloud through Sparse Representation

BestiVictory

Xin Jin, Yan Liu, Xiaodong Li, Geng Zhao, Yingya Chen, and Kui Guo
Beijing Electronic Science & Technology Institute (BESTI)
VIsual CompuTing and infORmation securitY (VICTORY Team)
{jinxin,lxd}@besti.edu.cn

Nowadays, with tremendous visual media stored and even processed in the cloud, the privacy of visual media is also exposed to the cloud. In this paper we propose a private face identification method based on sparse representation. The identification is done in a secure way which protects both the privacy of the subjects and the confidentiality of the database. The face identification server in the cloud contains a list of registered faces. The surveillance client captures a face image and require the server to identify if the client face matches one of the suspects, but otherwise reveals no information to neither of the two parties. This is the first work that introduces sparse representation to the secure protocol of private face identification, which reduces the dimension of the face representation vector and avoid the patch based attack of a previous work. Besides, we introduce a secure Euclidean distance algorithm for the secure protocol. The experimental results reveal that the cloud server can return the identification results to the surveillance client without knowing anything about the client face image

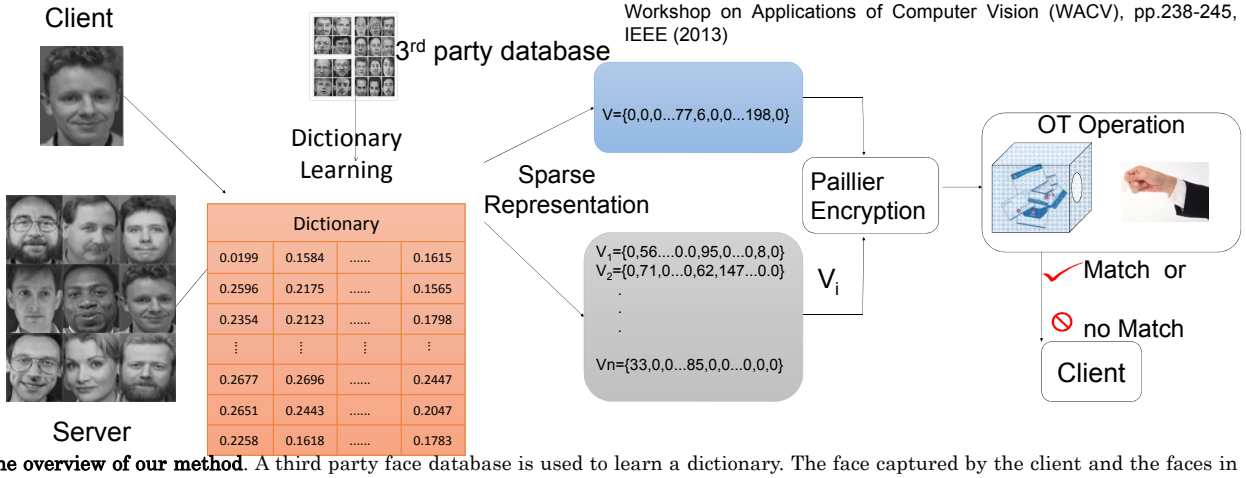


The application scenario. The surveillance client capture a face image from public places such as airport, railway station. The face image is identified through our privacy preserving method with the suspect face data in the cloud server. After that, the client only learns the matching results. The cloud server learns nothing.

Main References

[1] Osadchy, M., Pinkas, B., Jarrous, A., et al. SCiFI - A system for Secure Face Identification. IEEE Symposium on Security and Privacy (S&P), pp.239-254, IEEE (2010) Best Paper Award

[2] Luong, A., Gerbush, M., Waters, B., Grauman, K. Reconstructing a fragmented face from a cryptographic identification protocol. IEEE Workshop on Applications of Computer Vision (WACV), pp.238-245, IEEE (2013)



The overview of our method. A third party face database is used to learn a dictionary. The face captured by the client and the faces in the list of the cloud server are represented sparse parameter vector. The Euclidean distance of the client face vector and each of the face vector in the server is computed in a privacy preserving way. The matching result is only known by the client. The cloud server learns nothing.

Algorithm 1 Private Face Identification

Input:

The client's input is a face vector $\mathbf{s} = (s_0, s_1, ..., s_{l-1})$. In our application $l = 200$. The server's input is a list of Q face vectors $\{s^1, s^2, ..., s^Q\}$. The server has additional inputs $\{t_1, t_2, ..., t_Q\}$ for each s^i. The two parties both know an upper bound d_{max}, in our application we set $d_{max} \leq 1 \times 10^6$.

Output:

The client learns the indices i for which $ED(s, s^i) \leq t_i$. The server learns nothing.

1: The client uses Paillier to encrypt and send face vector $s = (s_0, s_1, ..., s_{l-1})$ item by item and the square of each item $(s)^2 = ((s_0)^2, (s_1)^2, ..., (s_{l-1})^2)$. The cloud server receives the encryption results of each item $(E_{pk}(s_0), E_{pk}(s_1), ..., E_{pk}(s_{l-1}))$ and $(E_{pk}((s_0)^2), E_{pk}((s_1)^2), ..., E_{pk}((s_{l-1})^2))$. For each face in the list of suspects of the server, the following steps are repeated.

2: For ith face in the server and for jth parameter in the face vector, the cloud server computes $E_{pk}(v_j)$, where $v_j = (s_j - s_j^i)^2$.

$$\begin{aligned} E_{pk}((s_j - s_j^i)^2) &= E_{pk}((s_j)^2 - 2s_j s_j^i + (s_j^i)^2) \\ &= E_{pk}((s_j)^2) \cdot E_{pk}(s_j)^{-2s_j^i} \cdot E_{pk}((s_j^i)^2) \end{aligned} \quad (5)$$

The s_j^i is known to the server.

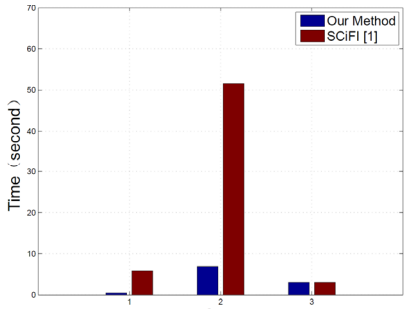
3: According to the properties of homomorphic encryption, the cloud server can compute the encrypted $d_E = (ED(s, s^i))^2$ by $E_{pk}(d_E) = \sum_{j=0}^{l-1} E_{pk}(v_j)$, $d_E \in [0, d_{max}]$. Then the server chooses a random number r_i for each face vector, and computes $E_{pk}((ED(s, s^i))^2 + r_i)$. This number is sent to the client.

4: The client receives the $E_{pk}((ED(s, s^i))^2 + r_i)$ and decrypts it.

5: The two parties use $OT_1^{\frac{d_{max}}{4000}}$ protocol to judge if $(d_E)^i < t_i$ securely. The result R_i computed in the client is:

$$R_i = \begin{cases} 1 \text{ if } ((d_E)^i + r_i) \bmod (d_m ax + r_i) \leq t_i + r_i \\ 0 \text{ if otherwise} \end{cases} \quad (6)$$

6: **return** R_i.



The computing time of each main step. step1 for face vector generation, step2 for the Paillier encryption, step3 for the oblivious transfer. The computing times of face vector generation and the Paillier encryption of our method are less than those of SCiFI [1]. The resolutions tested are 138X168; 230X280; 276X336; 322X392; 38X448; 414X504; 460X560.