Privacy-Preserving Extraction of HOG Features Based on Integer Vector Homomorphic Encryption

Haomiao Yang[1(✉)], Yunfan Huang[1(✉)], Yong Yu[2], Mingxuan Yao[1], and Xiaosong Zhang[1]

[1] School of Computer Science and Engineering, Center for Cyber Security, University of Electronic Science and Technology of China, Chengdu, China
haomyang@uestc.edu.cn, huangyf0714@163.com, mingxuanyao@hotmail.com, s_x_zhang@163.com

[2] Shaanxi Normal University, Xi'an, China
yuyong@snnu.edu.cn

Abstract. Along with the growing popularity of social networks, the number of multimedia image grows explosively. For the resource constrained owners, dealing with tremendous number of images on their own is a challenging job. Therefore, there is a general trend to outsource the heavy image processing (e.g., feature extraction) to the cloud. Abundant contents in images may expose the owner's sensitive information (e.g., face, location and event), and outsourcing the image data to the untrusted cloud directly has raised privacy concerns of public. In this work, we explore the outsourcing of the famous feature extraction algorithm-Histogram of Oriented Gradients (HOG) to the public cloud with privacy protection. In our proposed scheme, the image owner encrypts the original images by using the Vector Homomorphic Encryption (VHE) that encrypt vector directly and is much suitable for image processing. Then the image owner sends the encrypted images to the cloud which elaborately applies the *linear transformation* of VHE to the realization of the improved HOG algorithm in ciphertext domain. The security analysis based on the hardness of Learning with Error (LWE) Problem verifies that the extraction of HOG features is privacy-preserving in our scheme without leaking privacy contents to any other parties. We implement pedestrian detection by using the extracted HOG features to validate the efficiency and effectiveness of our proposed scheme, and the result shows that our solution can extract the HOG features correctly in ciphertext domain and approximate the original HOG in plaintext domain. Compared with existing solution, our scheme has less time and communication cost of HOG feature extraction.

Keywords: Privacy-preserving · HOG · Homomorphic encryption

J. K. Liu and P. Samarati (Eds.): ISPEC 2017, LNCS 10701, pp. 102–117, 2017.
https://doi.org/10.1007/978-3-319-72359-4_6

1 Introduction

With the arrival of the era of Big Data, the number and manner of image generation are increased considerably, which stimulates a lot of image applications, e.g., pedestrian detection, machine vision. Especially, along with the popularity of multimedia social network, the number of user-contributed images is sharply increased. According to the statistics, a number of images users upload to the social network servers only through Facebook has reached 500 million every day, making it difficult to deal with massive image in storage, sharing and search etc. In order to eliminate the influence of random factors of original images such as noise information, lack of pixels etc., and to get more efficient results in these image applications, it is essential to extract features from the original images, which means extracting information is of strong robustness to express the attributes of the images. The research of image feature extraction is of great significance in processing and analyzing massive image data, and it is the base of the applications of massive images.

Because of the powerful storage and computing in the cloud, more and more social server providers choose to provide image service based on the cloud server, such as Amazon Cloud Drive, Apple iCloud, Flicker and Google etc. However, the original images always contain users' sensitive information, e.g., personal identity, home address, and even financial conditions. And the cloud is not such believable as people imagine, existing the leakage of users' privacy information intentionally or unintentionally. In 2013, the report of PRISM [6] revealed privacy invasion to user-contributed data. In this event, the security authority has access to the social network servers owning to IT giants' permission. As a result, the security authority can inspect all the image data stored in cloud server, and it cause the public's concern about their own privacy.

Therefore, the privacy in image feature extraction with large-scale magnitude is an urgent problem that eagerly to be solved. When the cloud is not as reliable and trustworthy as people think, some other cloud security methods has been proposed, such as, data anonymization, secure multi-party computation (SMC) [9]. However, those methods are not very suitable and practical to deal with large-scale images under privacy protection. For example, multiple interactions between image owners and the queries in SMC lead to high communication cost. The other way to protect the privacy of image is image encryption. Traditional encryption schemes (e.g. DES, AES) change the original image data format in the encryption process, which will hinder further feather extraction and applications. Compared with those scheme, Homomorphic Encryption (HE) can handle the ciphertext before decryption, and there is a great possibility that HE can become a good solution to guarantee the privacy and security of images. After using of the homomorphic encryption, image data has converted from plaintext to ciphertext. By utilizing homomorphism, the ciphertext of images are computed and processed under the protection of privacy, and then the results as equivalent as the plaintext results can be obtained.

In previous researches, Hsu et al. [7] studied privacy-preserving SIFT algorithm under ciphertext domain through Paillier HE [11]. But this solution either

has a large overhead in computation or has security weaknesses as discussed in [14]. In [13], Qin et al. proposed an alternative solution on the basis of order-preserving encryption (OPE) [1] and random permutation to ensure confidentiality and privacy. From [17], the authors Wang et al. set out to find a better solution to privacy problems through garbled circuits [8] and homomorphic encryption scheme, and they respectively proposed two schemes with different security levels. Although there are obvious improvements in both privacy and efficiency, the proposed schemes are lackof comprehensive analysis and evaluation as previous schemes. As a following work, Q. Wang et al. carried out a research on the HOG algorithm [16] in ciphertext domain with somewhat homomorphic encryption, but it need the support of SIMD [15] to perform well.

In this paper, we provide a feasible solution to extract image feature by using HOG algorithm, meanwhile users privacy is well protected. The main ideas are as follows: The image owners encrypt their own image data by homomorphic encryption based integer vector and upload the ciphertext to the cloud, the cloud gains the image data in ciphertext domain and then carries out extraction algorithm—Histogram of Oriented Gradient (HOG), as s result, the cloud finally gets the HOG feature vectors in encrypted domain. In practice, the features of HOG are widely used in object detection, especially in pedestrian detection. In the whole process, the most time-consuming work is the generation of image features, and the main challenge is how to protect the untrusted cloud against the sensitive information of the images, meanwhile, the extracted feature can be effective and valid in object detection. In response to this challenge, we proposed a novel and original scheme that combines HOG feature extraction and integer-vector-based homomorphic encryption [18], which can achieve secure and efficient HOG feature extraction of the encrypted images.

Compared with the previous work [7,13,16,17], the contribution of this paper mainly includes three aspects:

1. We explore the privacy-preserving Extraction of the HOG features by utilizing the vector homomorphic encryption (VHE) that is more efficient than existing homomorphic scheme. Especially, the HOG extraction algorithm in our scheme has been improved to better adapt the homomorphic operations in ciphertext domain. And in our proposed system model, the cloud can know nothing but the ciphertext of images when it conducts the outsource computation of our HOG feature extraction, which ensure the privacy of image owners.
2. We have carried out a lot of experiments to verify the correctness of our solution with a large number of images in INRIA Database. And by the comparison with existing work, our scheme shows apparent efficiency and safety advantages.
3. We conduct the pedestrian detection in the Support Vector Machine (SVM) algorithm with extracted HOG descriptors from our scheme, the results of detection indicate that our solution can extract the HOG features correctly in ciphertext domain and approximate well the original HOG in plaintext domain.

The organization of the remaining paper is illustrated as follows: The system and threat models are proposed detailedly in Sect. 2. The integer vector homomorphic encryption(VHE) and the original HOG algorithm are depicted as preliminaries in Sect. 3. Section 4 presents our privacy-preserving HOG algorithm in ciphertext domain. Then we conduct implementations and analyze the whole scheme in multiple aspects in Sect. 5. At last, Sect. 6 gives some conclusions.

2 System Model and Design Goals

In this part, we design the system model according to the practical requirements in the HOG feature extraction of encrypted images, in addition, we construct our threat model. On one hand, the image owners has large-scale multimedia image data, but they have little ability to handle the data by theirselves due to resource constraints. When the image owners want to do the object detection jobs, they may have a tendency to delegate the heavy work of feature extraction to the cloud so that they can be free from the burden of image storage and heavy computations. On the other hand, original multimedia data contains sensitive information of the owners, outsourcing the data directly to the untrusted cloud leads to owners' privacy leaking. To solve this problem, we consider our system model is mainly composed of two entities: the image owners O and the cloud C as shown in Fig. 1.

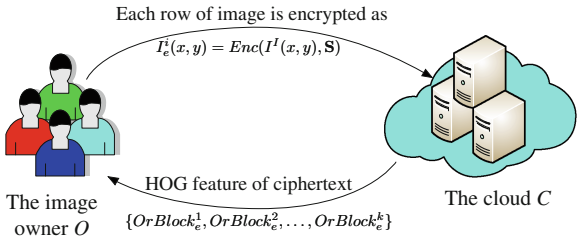

Fig. 1. System model

In this model, the image owners O encrypt original images using vector-based homomorphic encryption, and upload the ciphertext to the cloud C. After receiving the ciphertext, the cloud C will carry out the improved HOG algorithm in ciphertext domain. All the computations of the algorithm are conducted by the cloud, and because of the encryption, the owners' privacy information is well protected from the cloud, i.e. the cloud can get nothing about the image contents through the process.

Under the design of this system model, we set a hypothesis that the cloud is not absolutely honest. In fact, the cloud will honestly execute the HOG algorithm and return the results of feature vectors in ciphertext to the image owners O,

but it is curious about the image contents, and infer the sensitive information from the encrypted images and all the data it can get in computational process. Therefore, we must ensure the cloud can infer nothing about the image contents from the encrypted images so that the owners' privacy is guaranteed.

We consider the challenge how to outsource computing process of the HOG descriptor to the cloud without leaking the original image information, the final purpose of our system is that the system model must ensure image owners' privacy through the whole process under this threat. The overall design goal of our system is as follows:

- Correctness: The HOG features extracted by the improved HOG algorithm have the same attribute as those extracted by the original HOG algorithm, which can performs well in the pedestrian detection.
- Security: By the usage of vector homomorphic encryption scheme, the real contents of images can not be revealed to the cloud or other parties to protect the image owner's privacy.
- Efficiency: Compared with existing work, ensure time and communication cost of our scheme have obvious advantages.

3 Preliminaries

To make this paper better understanding, before elaborating our scheme, it is essential to give some preliminaries, including the integer vector homomorphic encryption and the overview of the original Histogram of Oriented Gradient (HOG) algorithm.

3.1 Vector Homomorphic Encryption Scheme

The Vector Homomorphic Encryption scheme that we utilize in the system is designed as an expansion on the basis of the PVW scheme [12], and the encrypted objects are changed from bit-wise vectors to integer-based vectors.

Before the introduction of the VHE scheme, we give some basic notations in advance to lead a easy understanding as below Table 1. Note that the lower-case bold and capital bold characters represent vectors and matrices respectively (e.g., $\mathbf{v}$ is a column vector and $\mathbf{A}$ is a matrix):

Key Generation: Under a security parameter λ, select some appropriate integers $l, w, m, n, p, q \in Z$ as parameters in VHE. Construct the secret key $\mathbf{S} = [\mathbf{I}, \mathbf{T}] \in Z_q^{m \times n}$, where $\mathbf{I}$ is an identity matrix.

Encryption Operation: Given the secret key $\mathbf{S} \in Z_q^{m \times n}$, so the $\mathbf{x} \in Z_p^m$, $\mathbf{c} \in Z_q^n$, $\mathbf{S}$ can satisfies:

$$\mathbf{Sc} = \omega \mathbf{x} + \mathbf{e} \tag{1}$$

For keeping the error term small later, it's necessary to assume that $|\mathbf{S}| << \omega$ when applying operations in ciphertext domain.

Table 1. Notations

Notation	Meaning
λ	A security parameter in VHE
l	The length parameter in VHE
p, q	Two large prime integers in VHE and $q >> p$
m, n	Two integers as length parameters in VHE and $n > m$
χ	The probability distribution
ω	A large integer as a parameter in VHE
$\mathbf{e}$	An error vector with elements smaller than $\omega/2$ in χ
Z_p	A finite field with alphabet size p
$a \in Z_p^m$	An integer vector a with length m in the finite field Z_p
$\lvert \boldsymbol{v} \rvert$	$Max_i\{\lvert v_i \rvert\}$, for a vector $\boldsymbol{v} \in \mathbb{Z}^n$
$\lvert \boldsymbol{M} \rvert$	$Max_i\{\lvert M_{ij} \rvert\}$, for a matrix $\boldsymbol{M} \in \mathbb{Z}^{n \times m}$
$\lceil a \rfloor_q$	The nearest integer to a with the modulus q
$\lceil \mathbf{a} \rfloor_q$	Round each a_i to the nearest integer with modulus q

Decryption Operation: As for the decryption, it's straightforward:

$$\mathbf{x} = \lceil \frac{\mathbf{Sc}}{\omega} \rfloor_q \tag{2}$$

Addition: For arbitrary plaintexts $\mathbf{x}_1, \mathbf{x}_2$ and their corresponding ciphertexts $\mathbf{c}_1, \mathbf{c}_2$ that are encrypted with the same secret key $\mathbf{S}$, then

$$\mathbf{S}(\mathbf{c}_1 + \mathbf{c}_2) = \omega(\mathbf{x}_1 + \mathbf{x}_2) + (\mathbf{e}_1 + \mathbf{e}_2) \tag{3}$$

Let $\mathbf{c}' = \mathbf{c}_1 + \mathbf{c}_2$, $\mathbf{x}' = \mathbf{x}_1 + \mathbf{x}_2$, and we can observe that

$$\mathbf{Sc}' = \omega\mathbf{x}' + (\mathbf{e}_1 + \mathbf{e}_2) \tag{4}$$

Linear Transformation: Given a plaintext $\mathbf{x}$ and its corresponding ciphertext $\mathbf{c}$ with the secret key $\mathbf{S}$, the linear transformation $\mathbf{Gx}$ can be computed that

$$(\mathbf{GS})\mathbf{c} = \omega\mathbf{Gx} + \mathbf{Ge} \tag{5}$$

with an arbitrary matrix $\mathbf{G} \in Z^{m' \times n}$. Therefore, we can treat $\mathbf{c}$ as the encryption of $\mathbf{Gx}$ with the secret key $\mathbf{GS}$.

When several ciphertexts need to be computed, it's essential to keep their secret keys to be the same. Key switching technique makes it possible to convert a secret key to another chosen secret key and maintain to encrypt the original integer vectors. Based on the relinearization technique introduced by Brakerski et al. in [3] and matrices switching method in [2], the key-switching technique in VHE achieves this conversion.

Key Switching Technique: Suppose the initial secretkey-ciphertext pair $\{\mathbf{S}, \mathbf{c}\}$ and a new secretkey-ciphertext pair $\{\mathbf{S}', \mathbf{c}'\}$ with predefined $\mathbf{S}' = [\mathbf{I}, \mathbf{T}]$, which can satisfy

$$\mathbf{S}'\mathbf{c}' = \mathbf{S}\mathbf{c} \tag{6}$$

Select a length parameter l satisfies $|\mathbf{c}| < 2^l$ and convert each element c_i of the ciphertext c to l-bit binary form $b_i = [b_{i(l-1)}, \cdots, b_{i1}, b_{i0}]^T$, and the ciphertext can be expressed as:

$$\mathbf{c}^* = [{b_1}^T, \cdots, {b_n}^T]^T. \tag{7}$$

Then construct $S^* \in Z^{m \times nl}$ by converting each element S_{ij} in S to a vector $[S_{ij}, S_{ij} \cdot 2, \cdots, S_{ij} \cdot 2^{l-1}]$, which can absolutely satisfy $\mathbf{S}\mathbf{c} = \mathbf{S}^*\mathbf{c}^*$.

Key switching matrix $\mathbf{M}$ can be generated through the transformation as:

$$\mathbf{M} = \begin{bmatrix} \mathbf{S}^* - \mathbf{T}\mathbf{A} + \mathbf{E} \\ \mathbf{A} \end{bmatrix} \tag{8}$$

where $\mathbf{A}$ is a random matrix and $\mathbf{E}$ is a noise matrix with $|\mathbf{E}|$ is small enough. Then the new ciphertext $\mathbf{c}'$ is computed as $\mathbf{c}' = \mathbf{M}\mathbf{c}^*$.

3.2 Histogram of Oriented Gradients (HOG) Algorithm

For a fixed-size image (e.g., 128 × 64 pixels in human detection), the detail extraction of HOG features is descripted as the following steps [4].

Image Preprocessing: The raw images are usually color images containing a lot of color information, however, because the focus of HOG descriptor is on gradients rather than the color of pixels, it is necessary to reduce this redundant information by graying the raw images. Besides, for adjusting the contrast of the images and decreasing the effects of local shadows and illumination changes, the grayscale images are standardized in color space by using Gamma Correction as the following formula:

$$\mathbf{I}(x, y) = \mathbf{I}(x, y)^{\gamma}$$

where the value of γ can be 1/2.

Orientation histogram building: For each pixel $\mathbf{I}(x, y)$ in the image $\mathbf{I}$, let $Diff_x = \mathbf{I}(x+1, y) - \mathbf{I}(x-1, y)$ and $Diff_y = \mathbf{I}(x, y+1) - \mathbf{I}(x, y-1)$ represent the horizontal and vertical gradients of the pixel. Then the gradient magnitude $m(x, y)$ and the orientation $\theta(x, y)$ can be computed as $m(x, y) = \sqrt{Diff_x^2 + Diff_y^2}$ and $\theta(x, y) = tan^{-1} \frac{Diff_y}{Diff_x}$.

In an image $\mathbf{I}$, a *cell* is defined as a local square region with a certain predefined size, such as 8 × 8 pixels, and then 4 adjacent cells are defined as a *block* that contains 16 × 16 pixels. Within a cell, the gradient orientation is divided into 9 bins equably among the angle space 0°–180° (unsigned gradient), and the bin that a pixel belongs to can be decided by its orientation. In each pixel, consider the gradient magnitude as a weighted vote, the orientation histogram

of a cell can be built by accumulating the bins of all pixels. Finally, concatenate the four orientation histograms of cells orderly, the orientation histogram of a block is obtained.

Block normalization: Obviously, the formed orientation histogram of each block contains a 4×9 dimensional feature vector. Suppose $\mathbf{v}$ be the unnormalized block descriptor vector, then its $\sigma - norm$ is defined as $||\mathbf{v}||_\sigma$, where σ can be $1, 2$. The normalization procedure of each block descriptor vector with $L_2 - norm$ is executed as $\mathbf{v} \leftarrow \frac{\mathbf{v}}{\sqrt{||\mathbf{v}||_2^2 + \epsilon}}$, where ϵ is a minimal constant. The process of normalization further weakens the interference of light to the images.

HOG descriptor generation: Concatenate the normalized block descriptor from left to right and top to bottom by the step of 8 pixels in the fixed-size image, the final HOG descriptor is generated (e.g., in a 128×64 pixels image, a 3780-dimensional feature vector is enmerged).

4 Privacy-Preserving Extraction of HOG

In this section, the privacy-preserving extraction of HOG is proposed between the image owners and the untrusted cloud. The final purpose is stated detailedly as follows. Firstly, the original image contents must be protected from any other parties except the image owner to ensure the security of our scheme. Secondly, after improvement and simplification, the HOG algorithm should have as much similar extraction results as the original HOG algorithm, which can keep the effectiveness. Finally, the image owner is always considered as resource constrained, so the computation cost on owner side must reduced significantly and the computational burden on the cloud side must be acceptable and practical to guarantee the efficiency of the HOG algorithm.

In the previous studies, [7,16] have done the similar research on exploring privacy-preserving outsourcing of HOG algorithm with SHE and HE cryptosystem respectively. Different from that scheme, we build our system by using the VHE (Vector Homomorphic Encryption) scheme as described above. The key point in our scheme is that the fully combination of image data format and vector-based encryption leads to much higher efficiency under privacy protection.

4.1 Image Encryption

For the protection of image owners' privacy, the encryption of the original image data is an essential process before uploading to the cloud. Assume that, the image data has been preprocessed, including graying and Gamma Correction. The owner generates a secretkey $\mathbf{S}$ to encrypt preprocessed images. We regard the image $\mathbf{I}$ as a matrix $\mathbf{I}(x, y)$ with the size of $n \times m$, and take each row of this image matrix as an integer vector to be encrypted. In the support of the Vector Homomorphic Encryption scheme, the encryption of each row vector can be expressed as

$$\mathbf{I}_e^i(x, y) = Enc(\mathbf{I}^i(x, y), \mathbf{S}),$$

which means the ciphertext of the i_{th} row pixels of image $\mathbf{I}$. After encryption, the ciphertext vector group of the image $\mathbf{I}$ is generated:

$$\{\mathbf{I}_e^1(x,y), \mathbf{I}_e^2(x,y), ..., \mathbf{I}_e^n(x,y)\}.$$

4.2 Computing Gradient Magnitude

Because of the limited computing operations that the VHE supports, some complex operation in the original HOG algorithm must be simplified. So we eliminate some steps and reduce the amount of the original orientation directions from nine to four ones (i.e., $0°, 45°, 90°, 135°$), which represent 4 bins for each orientation descriptor in a cell. The detail computing method is expressed in the plaintext form as below:

$$\begin{aligned} Diff_{0°} &= \mathbf{I}(x+1,y) - \mathbf{I}(x-1,y), \\ Diff_{45°} &= \mathbf{I}(x-1,y-1) - \mathbf{I}(x+1,y+1), \\ Diff_{90°} &= \mathbf{I}(x,y+1) - \mathbf{I}(x,y-1), \\ Diff_{135°} &= \mathbf{I}(x+1,y-1) - \mathbf{I}(x-1,y+1). \end{aligned}$$

In the ciphertext domain, we utilize the *Linear Transformation* in the VHE to achieve these gradients computation over the encrypted image vector. Define some certain transformation matrices to shift the vector correspondingly, and then carry out different homomorphic operations to get the gradient of different orientation. Especially, considering the image edge, we add the last pixel value to the original pixel value when shifting, in other words, the pixel value at the right edge is set to the original value when the vector moves left, and the pixel value at the left edge is set to the original value when the vector moves right.

Thus, the gradient magnitude along $0°$ direction of the i_{th} row is computed as following steps. First, define two shifting linear transformation matrices, left-shifting matrix $\mathbf{G}_L$ and right-shifting matrix $\mathbf{G}_R$, which can lead the plaintext vector to move a single position left and right respectively. Second, according to the *Linear Transformation*, computing a new secret key $\mathbf{G}_L\mathbf{S}$ can obtain a relative pair of plaintext-ciphertext $\{\mathbf{G}_L\mathbf{I}^i(x,y), \mathbf{I}_e^i(x,y)\}$. To support the direct homomorphic operation later, we use key-switching to keep the same secretkey $\mathbf{S}$. So in the procedure of changing $\mathbf{G}_L\mathbf{S}$ to $\mathbf{S}$, we can get the key-switching matrix $\mathbf{M}_L$, and the left-shifting plaintext vector $\mathbf{I}^i(x+1,y) = \mathbf{G}_L\mathbf{I}^i(x,y)$ can correspond to the following ciphertext vector under the secret key $\mathbf{S}$:

$$\mathbf{I}_e^i(x+1,y) = \mathbf{M}_L\mathbf{I}_e^i(x,y).$$

The right-shifting plaintext vector $\mathbf{I}^i(x-1,y) = \mathbf{G}_R\mathbf{I}^i(x,y)$ and the relative ciphertext vector can be computed analogously,

$$\mathbf{I}_e^i(x-1,y) = \mathbf{M}_R\mathbf{I}_e^i(x,y).$$

Finally, under the same secret key $\mathbf{S}$, the orientation gradient magnitude of the i_{th} row ciphertext vector is derived as:

$$Diff_{0°}^i = \mathbf{I}_e^i(x+1,y) - \mathbf{I}_e^i(x-1,y). \tag{9}$$

Therefore, the gradient magnitude along the other three directions can be obtained as:

$$Diff^i_{45^\circ} = \mathbf{I}^i_e(x-1, y-1) - \mathbf{I}^i_e(x+1, y+1) \tag{10}$$

where $\mathbf{I}^i_e(x-1, y-1) = \mathbf{M}_R\mathbf{I}^{i-1}_e(x, y)$ and $\mathbf{I}^i_e(x+1, y+1) = \mathbf{M}_L\mathbf{I}^{i+1}_e(x, y)$.

$$Diff^i_{90^\circ} = \mathbf{I}^i_e(x, y+1) - \mathbf{I}^i_e(x, y-1) \tag{11}$$

where $\mathbf{I}^i_e(x, y+1) = \mathbf{I}^{i+1}_e(x, y)$ and $\mathbf{I}^i_e(x, y-1) = \mathbf{I}^{i-1}_e(x, y)$.

$$Diff^i_{135^\circ} = \mathbf{I}^i_e(x+1, y-1) - \mathbf{I}^i_e(x-1, y+1) \tag{12}$$

where $\mathbf{I}^i_e(x+1, y-1) = \mathbf{M}_L\mathbf{I}^{i-1}_e(x, y)$ and $\mathbf{I}^i_e(x-1, y+1) = \mathbf{M}_R\mathbf{I}^{i+1}_e(x, y)$.

4.3 HOG Descriptor Generation

On the owner's side, it predefines several linear transformation matrices $\mathbf{G}$ and compute the relevant key-switching matrices $\mathbf{M}$. After a few round of communications, the cloud get a series of key-switching matrices from the owners. Then the cloud can conduct linear transformation and key switching to generate HOG descriptor based on those matrices.

Given that a *cell* contains 8×8 pixels. The feature vector of a cell is formed by accumulating the gradient magnitude of four orientations among all the 8×8 pixels. Let $\mathbf{G}_H$ be the transformation matrix to add every 8-pixel gradient in a ciphertext vector horizontally. Take the accumulation of the direction 0° in a cell for example, translate the secret key $\mathbf{G}_H\mathbf{S}$ to $\mathbf{S}$ by key-switching method and get the key-switching matrix $\mathbf{M}_H$. Then compute the new ciphertext $\mathbf{M}_H Diff^i_{0^\circ}$ to represent the sum gradient of every 8-pixel in each ciphertext vector row. Finally, adding 8 consecutive rows of gradient ciphertext leads to the accumulation of the direction 0° in a cell as following:

$$Or0^j_e = \mathbf{M}_H(Diff^i_{0^\circ} + ... + Diff^{i+7}_{0^\circ}).$$

Similarly, the accumulation of the other three directions in a cell are computed, i.e., $Or45^j_e, Or90^j_e, Or135^j_e$.

Define four transformation matrices $\mathbf{G}_0, \mathbf{G}_{45}, \mathbf{G}_{90}, \mathbf{G}_{135}$ to shift the sum orientation of four directions in cells, and let the four matrices $\mathbf{M}_0, \mathbf{M}_{45}, \mathbf{M}_{90}, \mathbf{M}_{135}$ to be the corresponding key-switching matrices when transformating secret key. Every row of cell descriptor can be derived by

$$OrCell^j_e = \mathbf{M}_0 Or0^j_e + \mathbf{M}_{45} Or45^j_e + \mathbf{M}_{90} Or90^j_e + \mathbf{M}_{135} Or135^j_e$$

A *block* contains 2×2 cells, so the block descriptor of each row can be obtained by two adjacent rows of cell descriptor in the ciphertext domain. In the cell descriptors of two rows, we cascade the block descriptor by the order of zigzag with the step of a cell. By the usage of two transformation matrices $\mathbf{G}_u, \mathbf{G}_d$ and corresponding key-switching matrices $\mathbf{M}_u, \mathbf{M}_d$, the block descriptor of each row in ciphertext is derived as:

$$OrBlock^k_e = \mathbf{M}_u OrCell^j_e + \mathbf{M}_d OrCell^{j+1}_e.$$

Because the limit of division operation in ciphertext domain, the block normalizations are carried out by the image owners O. Thus, the final HOG descriptor can be described by block descriptors of all rows:

$$HOG = Dec\{(OrBlock_e^1, OrBlock_e^2, ...OrBlock_e^k), \mathbf{S}\}.$$

4.4 Descriptor Descryption

Once the cloud C has completed the process of HOG descriptor extraction in encrypted domain, it sends the results in ciphertext to the image owners O. Next, the owners O will descript the HOG descriptor with the secret key $\mathbf{S}$, and perform normalization operations on each block descriptor by using $L_2 - norm$ algorithm. Ultimately, the owners O get the actual HOG deacryptors.

5 System Analysis and Evaluation

In this part, we complete the simulation on the C++ environment to prove the feasibility and practicality of our scheme, combining the MATLAB with its image processing functions. We show the experimental results based on hundreds of pedestrian images in INRIA Database, in which we choose 1000 sample images with and without humans respectively. According to the experimental results, We first present the correctness evaluation to prove that our proposed solution is feasible and practical. Then the security analysis shows that our system can be practical in protecting the owners' privacy. Finally, we evaluate the efficiency, which proves the better performance of our scheme.

5.1 Correctness Evaluation

For pedestrian detection, we conduct the SVM algorithm to detect whether there are humans in an image. In the training stage, we use 1000 HOG features extracted by the original HOG and our improved HOG under privacy respectively, which contains 500 negative samples and 500 positive samples. In the testing stage, the same number of image samples are detected. Through many experiments, we choose two types of kernel functions in SVM algorithm with a comprehensive consideration of various factors.

First, we can contrast the *Undetected Rate* and the *Wrong-Detection Rate* under different kernel functions in Table 2. Though our improved HOG performs a little bit inferior than the original HOG in the undetected rate, our solution shows a better results in terms of wrong detection rate under the two functions. In practice, the effect of detection is dependent on these two rates, meaning that our HOG is acceptable and practical.

To further evaluate the detection results, we measure the SVM detector by precision and accuracy as follows:

$$precision = \frac{TruePositive}{TruePositive + FalsePositive},$$

Table 2. Average detection error rate

	Kernel function	The original HOG	The improved HOG
Undetected rate	'Quadratic'	0.068	0.308
Wrong detection rate		0.292	0.122
Undetected rate	'rbf', 'rbf_sigma' = 100	0.066	0.252
Wrong detection rate		0.264	0.188

$$accuracy = \frac{TruePositive + TrueNegtive}{TotalSamples}.$$

An image sample is detected as true positive if it matches the ground truth annotation with less than 50% overlap errors according to the PASCAL criteria [5]. The construction of the *precision* between our HOG features detection and the original HOG feature detection in different kernel functions is shown as follows Table 3. Not surprisingly, both our improved HOG and the original HOG perform very well in precision and accuracy, indicating that our solution is consistent with the original HOG. This result is a better proof that our solution can extract the HOG features correctly in ciphertext domain and approximate the original HOG in plaintext domain.

Table 3. Precision and accuracy

	Kernel function	The original HOG	The improved HOG
Precision	'Quadratic'	0.7614	0.808
Accuracy		0.82	0.785
Precision	'rbf', 'rbf_sigma' = 100	0.7796	0.7991
Accuracy		0.835	0.78

5.2 Security Analysis

As described in our model, the cloud can get nothing but the ciphertext image to preserve the privacy of the image owners. The security of our homomorphic encryption scheme VHE plays an important role in the security of our solution, which can be reduced to the speculated difficulty of an mathematical problem named "Learning with Error" [2]. Next we illustrate our scheme is secure by using VHE as follows.

Definition 1: Learning with Error (LWE) Problem. Given polynomial multiple samples of $(\mathbf{a}_i, b_i) \in Z_q^m \times Z_q$ satisfy

$$b_i = \mathbf{v}^T \mathbf{a}_i + \varepsilon_i \tag{13}$$

where error term $\varepsilon_i \in Z_q$ is generated from a certain probability distribution χ. and there is almost no chance to get the vector $\mathbf{v} \in Z_q^m$ with non-ignored possibility.

Theorem 1: If the problem of *Learning with Error* (LWE) is difficult, there is of little possibility to retrieve $\mathbf{S}'$ from $\mathbf{S}'\mathbf{M} = \mathbf{S}^* + \mathbf{E}$, given $\mathbf{M}$ and $\mathbf{S}^*$ is available.

Obviously, we can reduce this problem to LWE to prove the safety of our encryption scheme. Suppose there is a solver to the $\mathbf{S}'\mathbf{M} = \mathbf{S}^* + \mathbf{E}$, and we take each of the elements in the matrices to be the element of Z_q with a large prime integer $q >> max\{|\mathbf{S}'|, |\mathbf{M}|, |\mathbf{S}^*|, |\mathbf{E}|\}$. Take this special circumstance into account, i.e., given $\mathbf{S}' = (\mathbf{s}')^T$ is a n-dimensional row vector, meanwhile $\mathbf{S}^* = \mathbf{s}^*$ and $\mathbf{E} = \mathbf{e}$ are both row vectors in the same dimension n. Next the transformation of our problem can be stated that m_i and s^* are available in

$$(\mathbf{s}')^T m_i = \mathbf{s}^* + \mathbf{e},$$

i.e., there are n samples of $(m_i, {s_i}^*)$ in

$${\mathbf{s}_i}^* = (\mathbf{s}')^T m_i - e_i.$$

Then we can use the solver to solve the above equation for s', which is equivalent to solving $b_i = \mathbf{v}^T \mathbf{a}_i + \varepsilon_i$ for $\mathbf{v}$.

Therefore, if the problem of LWE is hard, solving $\mathbf{S}'\mathbf{M} = \mathbf{S}^* + \mathbf{E}$ is also hard. In other words, given the LWE problem is hard to conjecture, it is also impractical to retrieve the new secret key $\mathbf{S}'$ on the premise of the key-switching matrix $\mathbf{M}$ under known. Thus any other adversaries who intercept the communication between the image owners and the cloud can not recover the secret key $\mathbf{S}$ and the plaintext image.

5.3 Efficiency Evaluation

In the simulation implementation, we consider every row of an image as a plaintext to be encrypted. Because the same secret key $\mathbf{S}$ is used to encrypt each plaintext vector, we compute the key-switching matrix $\mathbf{M}$ in advance, and then conduct encryption operations, i.e., $\mathbf{M}(w\mathbf{x})$ based on $\mathbf{M}$, which can be time-saving on the image owner's side. In comparation with the HE scheme - the Paillier cryptosystem [10] in [7], in which the parameter setting is within a 1024-bit modulus under the comprehensive consideration, although the time cost of our scheme increases nearly linearly with the change in image size, its quantity is still within a reasonable scope. As shown in Fig. 2, and the comparison results are shown in Fig. 3. Obviously, due to the vector based encryption in VHE, our scheme performs excellent advantages than the pailliar cryptosystem in time cost. Especially, the images in our dataset are preprocessed in fixed size (i.e., 128 * 64 pixels), which can be encrypted in less than 1 s.

In the cloud, the main operation of HOG algorithm in ciphertext domain is *linear transformation*, therefore the cloud only need to compute the $\mathbf{M}\mathbf{c}^*$ to

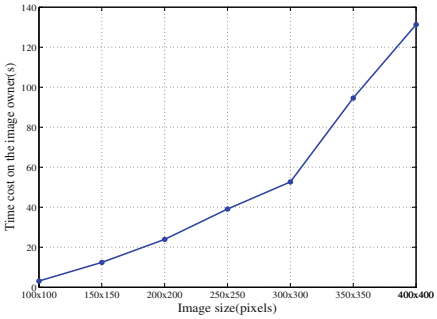

Fig. 2. Encryption Time Cost on VHE with different image sizes

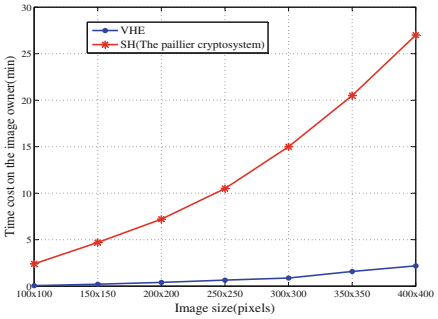

Fig. 3. Comparison of Time Cost with different image sizes

get the new ciphertext, where the **M** is the key-switching matrix corresponding to the linear transformation **G**. Due to the strong computing capability of the cloud, the time cost is largely shortened when it conducts the cipher computing steps. With different sizes of image, the changing tendency of time cost in the cloud is shown as follows Fig. 4.

Also, we evaluate the communication cost between the image owner and the cloud in the outsourcing HOG extraction. In Fig. 5, although the communication costs also increase when image sizes grow, all of those cost are acceptable in practice. In the whole process of our scheme, merely the occurrence of the computation of key-switching matrices **M**, the communication would be made between the image owners and the cloud. So the interaction times can be controlled as a constant, which has important significance. In order to better reflect the efficiency advantage in our solution, we still compare the communication costs of the VHE used in our scheme with the HE scheme - the Paillier cryptosystem. The comparison results are shown in Fig. 6, and we can observe that the VHE scheme outperforms the HE scheme in the time and communication

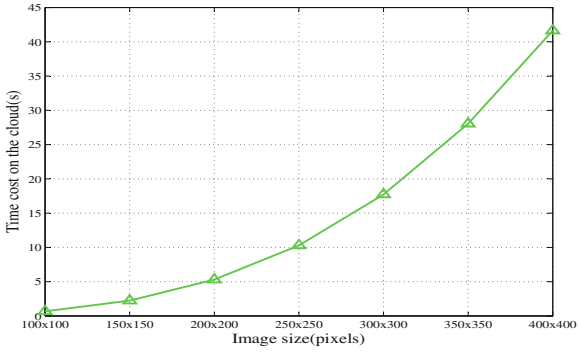

Fig. 4. Time Cost on cloud with different image sizes

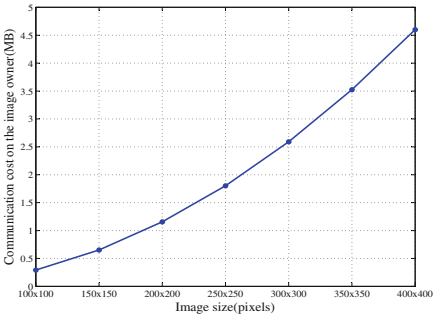

Fig. 5. Communication Cost on VHE with different image sizes

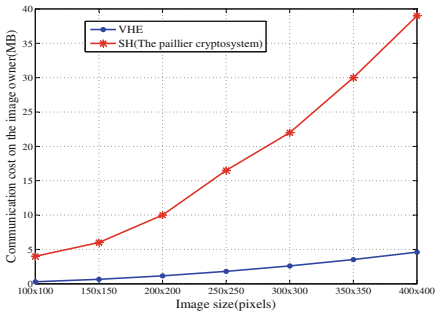

Fig. 6. Comparison of communication Cost with different image sizes

costs. Therefore, in terms of efficiency, our scheme has great improvement owing to the use of the VHE scheme.

6 Conclusion

In this paper, we first simplify the original HOG algorithm to adapt our encryption scheme-VHE. Then we conduct the extraction operations of the improved HOG in the ciphertext domain, which outsource most heavy computation to the cloud to release the burden of the image owners. In this proposed model, the original contents of images are well protected against the cloud and any other party by using of VHE, so the privacy preservation of image owner has been achieved. Finally, we analyze the security and effectiveness of the system to show that our proposed scheme is feasible and practical. Besides, to verify these analysis, we carry out a lot of experiments by comparing the improved HOG and the original HOG, and the results of experiments indicate that our solution is well consistent with and approximate the original HOG solution.

Acknowledgement. This work is supported by the National Key Research and Development Program of China (2017YFB0802003, 2017YFB0802004) National Cryptography Development Fund during the 13th Five-year Plan Period (MMJJ20170216), the Fundamental Research Funds for the Central Universities (GK201702004), the National Natural Science Foundation of China under Grants U1633114 and China Postdoctoral Science Foundation funded project under Grant 2014M562309.

References

1. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-preserving symmetric encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 224–241. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_13
2. Brakerski, Z., Gentry, C., Halevi, S.: Packed ciphertexts in LWE-based homomorphic encryption. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 1–13. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36362-7_1

3. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. SIAM J. Comput. **43**(2), 831–871 (2014)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 1, pp. 886–893. IEEE (2005)
5. Everingham, M., Zisserman, A., Williams, C.K., Van Gool, L.: The pascal visual object classes challenge 2006 (voc2006) results (2006)
6. Greenwald, G., MacAskill, E.: NSA prism program taps in to user data of Apple, Google and others. Guardian **7**(6), 1–43 (2013)
7. Hsu, C.Y., Lu, C.S., Pei, S.C.: Image feature extraction in encrypted domain with privacy-preserving SIFT. IEEE Trans. Image Process. **21**(11), 4593–4607 (2012)
8. Huang, Y., Evans, D., Katz, J., Malka, L.: Faster secure two-party computation using garbled circuits. In: USENIX Security Symposium, vol. 201 (2011)
9. Lindell, Y., Pinkas, B.: A proof of security of yaos protocol for two-party computation. J. Cryptol. **22**(2), 161–188 (2009)
10. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_16
11. Paillier, P., Pointcheval, D.: Efficient public-key cryptosystems provably secure against active adversaries. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) ASIACRYPT 1999. LNCS, vol. 1716, pp. 165–179. Springer, Heidelberg (1999). https://doi.org/10.1007/978-3-540-48000-6_14
12. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85174-5_31
13. Qin, Z., Yan, J., Ren, K., Chen, C.W., Wang, C.: Towards efficient privacy-preserving image feature extraction in cloud computing. In: Proceedings of the 22nd ACM International Conference on Multimedia, pp. 497–506. ACM (2014)
14. Schneider, M., Schneider, T.: Notes on non-interactive secure comparison in image feature extraction in the encrypted domain with privacy-preserving SIFT. In: Proceedings of the 2nd ACM Workshop on Information Hiding and Multimedia Security, pp. 135–140. ACM (2014)
15. Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. Des. Codes Crypt. **71**, 57–81 (2014)
16. Wang, Q., Wang, J., Hu, S., Zou, Q., Ren, K.: Sechog: privacy-preserving outsourcing computation of histogram of oriented gradients in the cloud. In: Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, pp. 257–268. ACM (2016)
17. Wang, S., Nassar, M., Atallah, M., Malluhi, Q.: Secure and private outsourcing of shape-based feature extraction. In: Qing, S., Zhou, J., Liu, D. (eds.) ICICS 2013. LNCS, vol. 8233, pp. 90–99. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-02726-5_7
18. Zhou, H., Wornell, G.: Efficient homomorphic encryption on integer vectors and its applications. In: Information Theory and Applications Workshop (ITA 2014), pp. 1–9. IEEE (2014)