

Enabling a Secure Match over Private Image Collections

Ayad Ibrahim Abdulsada

Department of Computer Sciences, College
of Education for pure sciences, Basrah
University, Basrah, Iraq.
Email: mraiadibraheem@yahoo.com

Hesham Saleh Ridha

Department of basic Sciences,
College of Dentistry,
Basrah University, Basrah, Iraq.
Email: zhesham60@hotmail.com

Hameed Abdul-kareem Younis

Department of Computer Science, College
of sciences,
Basrah University, Basrah, Iraq.
Email: hameedalkinani2004@yahoo.com

Abstract

Image matching techniques play an essential role in many real world applications such as *content based image retrieval* (CBIR), computer vision, and near duplicate images. The state of the art methods are generally assumed that the content of images is not private. This reduces the utilization of these methods to work within only environments where images are publicly access. Essentially, this assumption limits more practical applications, e.g., image matching between two security agencies, where images are confidential. This paper addresses the problem of privacy-preserving image matching between two parties where images should not be revealed to each other. The descriptor set of the queried party needs to be generated and encrypted properly using a secret key at the queried party side before being transferred to the other party. We have developed a secure scheme to measure the cosine similarity between two descriptor sets without decryption. Several experiments are conducted to investigate the performance of the proposed scheme.

Keywords: Image matching, Secure Multiparty Computing (SMC), SURF descriptors, Homomorphic encryption.

1. Introduction

Image matching (IM) is ubiquitous in many real-word applications. For example, near duplicate image detection is used to identify the relevant images for a given reference. In the context of image retrieval, similar images are bringing together. Such that once providing a query image, all the similar images are retrieved efficiently. Unfortunately, existing approaches of IM suppose that image collection is publicly access and thus do not care to the privacy issue. However, in some

cases it

is desirable to protect images' privacy during the matching process. Consider the following example to see the importance of security issue. Suppose a security agency looking for the data related to potential terrorist suspect. It may wish to check whether there are images that are related to the suspect from local police databases. However, for security purposes, neither the agency nor the local police want to reveal their images unless there is a need to share. One way to identify such a need is to detect similarities between the agency's query (in form of image) and the local police's image collections. Once the need for sharing information is verified, the agency and the local police can exchange only the shared information. During the process of identifying similar images, it is the best choice for both parties not to disclose the query image and the database. Such a process is referred as *secure image matching* (SIM).

Most IM approaches define an image representation and a distance metric, which reduce both the amount of data stored per image and the time cost of database search. More precisely, feature vectors (*descriptors*) of each image in the database are extracted and stored. During the matching, the descriptors of the query image are compared against their counterparts in the database to determine the most relevant image. However, keeping descriptors in their clear text may reveal information about some objects in the image. Thus, it is desirable to encrypt such descriptors in such a way that preserve their distances without decryption.

In this paper, we address the question of how to find similar images between two parties in a privacy preserving way. Given an image I , Alice would like to find whether there are images in Bob's collection D that are similar to I (e.g., duplicate, near duplicate, somewhat

close, etc.) without disclosing either I or D . We focus primarily on security, where protecting the descriptors of images is necessary. Specifically, our scheme uses the cosine similarity, a well-known metric to score matching images, and employs the homomorphic encryption to protect the confidentiality of descriptors.

Most feature vectors are either global vectors such as global color histogram or local vectors such as SIFT[1] and SURF[2] descriptors. The first model generates an extreme compressed feature vector for each image. Such model is effective to identify global similarities, e.g., how many colors two images share. The second model searches the image to identify interest key points that are invariant to scale and orientation. Then a feature descriptor is generated for each key point. In this paper, we will focus on local features model which has the advantage to identify local similarities, e.g., scenes and objects.

A trivial solution to achieve secure image matching is to utilize a *trusted third party* (TTP). Alice sends I to the TTP and Bob sends D to the TTP. Then TTP can investigate and tell Alice whether or not there are images similar to I in Bob's collection. However, in real life situations, finding a completely trusted third party is a difficult task. Our work does not require such a third party.

The rest of this paper is organized as follows. Related works are reviewed and discussed in section 2. Section 3 introduces the security definition and problem statement. Section 4 provides the proposed scheme. Security analysis and experimental results are provided in Section 5, and conclusions and future works are drawn in Section 6.

2. Related Works

Several approaches have been presented to deal with duplicate image detection [3, 4]. Such methods concerned with scalability to very large image and video databases, where fast query processing is necessary. Unfortunately, work within the context of security has got a little attention. Shashank *et al.* [5] applied the *private information retrieval* (PIR) techniques to protect the privacy of the query image when searching over a public

database. However such method assumes that the database is public, whereas such database is supposed to be private in our work. Similarly, Lu *et al.* [6] proposed a system to search over encrypted multimedia databases that are stored on a server maintained by a third-party service provider. Under such a case, the server should not know its stored data. Our work, for security purposes, obviates the using of any third party. Furthermore both [5] and [6] are not suitable to evaluate the similarity. Both approaches are able to achieve an exact match, limiting the ability to develop efficient solutions.

3. Security Definition and Problem Statement

3.1 Security Definition

Our security definition belongs to the *secure multiparty computing* (SMC) definition of Goldreich *et al.* [7]. We assume that the participant parties are *semi-honest*. A semi-honest party follows the steps of the protocol using the party's correct input, but it tries to use what it sees during the execution of the protocol to compromise security. This model guarantees that parties who correctly follow the protocol cannot gain any knowledge about the other party's input data except the output and whatever can be inferred from its own input.

3.2 Problem Statement

Our proposed scheme includes two parties, namely: Alice and Bob, each of whom has a collection of images. We assume that images at both parties are private. Given an image I of Alice, we interest in detecting whether or not Bob's collection contains an image similar to I without disclosing Bob's database to Alice and vice-versa. We evaluate the similarity of two images under the local feature vector model, where each image is represented as a set of vectors. Let $D = \{Img_1, \dots, Img_m\}$ denote the set of m images in Bob's collection. Without disclosing I to Bob and D to Alice, our objective is to find a set of images in D that are similar to I . We term such protocol as *secure image matching* (SIM). Formally, SIM is defined as:

$$SIM(I, D) = \alpha_1, \alpha_2, \dots, \alpha_m$$

SIM returns the m similarity scores $\alpha_1, \alpha_2, \dots, \alpha_m$ to Alice instead of returning the

Algorithm 1: Insecure Image distance calculation

Input: two feature vectors $F_1 = \{v_1, v_2, \dots, v_k\}$ and $F_2 = \{S_1, S_2, \dots, S_p\}$ of two images. All vectors v_i and s_j are of the same size n .

Output: $Dist$: distance between F_1 and F_2 .

$Dist = 0$;

For $i = 1 \dots k$

- Compute \vec{v}_i as in (2)

For $j = 1 \dots p$

- Compute \vec{s}_j as in (2)

- $D_j = CSIM(\vec{v}_i, \vec{s}_j)$

Endfor

$Dist = Dist + \max(D_j), \forall j = 1, \dots, p$

Endfor

$Dist = Dist/k$

actual images. At another time, Alice can retrieve the close images from Bob. To evaluate the similarity between two images, each party initially extracts the feature vectors for each image in its own collection. There are several metrics are used to evaluate the similarity between two feature vector's sets such as Euclidean distance and cosine similarity. The cosine similarity between two vectors v_1 and v_2 of size n is defined as:

$$CSIM(v_1, v_2) = \frac{\sum_{i=1}^n v_1[i] \cdot v_2[i]}{\|v_1\| \|v_2\|}, \text{ Where } \|v\| \text{ is the}$$

Euclidian length of vector v , and it is defined as the following:

$$\|v\| = \sqrt{\sum_{i=1}^n v[i]^2}$$

Given normalized vectors \vec{V}_1 and \vec{V}_2 , the cosine similarity can be written as:

$$CSIM(\vec{V}_1, \vec{V}_2) = \sum_{i=1}^n \vec{V}_1[i] \cdot \vec{V}_2[i], \quad \dots (1)$$

$$\text{Where } \vec{V}[i] = \frac{v[i]}{\|v\|} \quad \dots (2)$$

Given two images Im_1 and Im_2 of two feature vector sets $F_1 = \{v_1, v_2, \dots, v_k\}$ and $F_2 = \{S_1, S_2, \dots, S_p\}$, respectively. Algorithm 1 illustrates how to measure the distance between two feature vector sets through the cosine similarity without privacy preserving.

Table 1 show a trivial example for Alice image which is represented by a set of three vectors of size 5. The first three columns are the feature vectors, while the last three columns are their corresponding normalized versions. Similarly, Table 2 illustrates the collection of Bob, which consists of two images. Also this table is interpreted in the same way as Table 1.

Table 1: Alice Image

Alice Image					
F_1			\vec{F}_1		
v_1	v_2	v_3	\vec{v}_1	\vec{v}_2	\vec{v}_3
1	3	3	0.1348	0.5145	0.75
5	2	1	0.6742	0.343	0.25
2	4	2	0.2697	0.686	0.5
3	1	1	0.4045	0.1715	0.25
4	2	1	0.5394	0.343	0.25

To compute the similarity between the image of Alice and the first image of Bob collection, we have to compute similarity between the feature vectors' sets F_1 and F_2 . The cosine similarity between F_1 and F_2 is calculated as:

$$\begin{aligned} Dist_1 &= (\max(CSIM(\vec{v}_1, \vec{s}_1), CSIM(\vec{v}_1, \vec{s}_2), CSIM(\vec{v}_1, \vec{s}_3)) + \dots \\ &+ \max(CSIM(\vec{v}_3, \vec{s}_1), CSIM(\vec{v}_3, \vec{s}_2), CSIM(\vec{v}_3, \vec{s}_3))) / 3 \\ &= (\max(0.7750, 0.7750, 0.8234) + \\ &\max(0.8933, 0.8625, 0.8009) + \\ &\max(0.8082, 0.7633, 0.8082)) / 3 \\ &= (0.8234 + 0.8933 + 0.8082) / 3 = \\ &0.8416. \end{aligned}$$

Similarly the similarity between F_1 and F_3 is $Dist_2 = 0.8625$. Thus we can conclude that the second image in Bob's collection is similar to Alice's image than the first one.

As shown in the above example, the main step to evaluate similarity between two images is the dot product between their corresponding normalized vectors. Therefore, once we know how to calculate the dot product in a privacy-preserving way, we can calculate the distance between any two images without sharing their contents.

Table 2: Bob collection of two images.

Bob collection											
F_2			F_3			$\vec{F_2}$			$\vec{F_3}$		
S_1	s_2	s_3	x_1	X_2	x_3	$\vec{s_1}$	$\vec{s_2}$	$\vec{s_3}$	$\vec{x_1}$	$\vec{x_2}$	$\vec{x_3}$
2	1	3	2	3	3	0.3592	0.1796	0.5388	0.417	0.7746	0.6882
1	2	2	1	2	2	0.1796	0.3592	0.3592	0.2085	0.5164	0.4588
3	4	1	0	1	1	0.5388	0.7184	0.1796	0	0.2582	0.2294
1	3	1	3	0	1	0.1796	0.5388	0.1796	0.6255	0	0.2294
4	1	4	3	1	2	0.7184	0.1796	0.7184	0.6255	0.2582	0.4588

In the following subsection, we will demonstrate a protocol [8] that is based on homomorphic encryption to compute the dot-product operation in a privacy-preserving mode. Then we show how to utilize such a protocol, as a tool, to design our proposed SIM.

3.2.1 Secure dot product based on homomorphic encryption

Homomorphic encryption system is a probabilistic public key encryption that allow to perform some mathematic operations like addition and multiplication over the encrypted ciphertext numbers without decryption. Let $HEpk(x)$ and $HDpr(y)$ be the encryption and decryption functions in this system with public key pk and private key pr . Without the private key pr , no adversary can guess the plaintext x in polynomial time. Furthermore, $HEpk(x)$ has a semantic security [9] property, which means that no adversary can compute any function of the plaintext from the ciphertext set. Specifically, we use additive homomorphic cryptosystem, where additive property allows adding two encrypted numbers, i.e., $HEpk(x1) * HEpk(x2) = HEpk(x1+x2)$. Furthermore, given a constant c and a ciphertext $HEpk(x)$, it can compute : $HEpk(x)^c = HEpk(c*x)$. In this paper, we adopt Paillier's system [10] for the practical implementation due to its efficiency. The security proof of Paillier cryptosystem is well presented in [11]

Let u and v are secure vectors of Alice and Bob, respectively. Both vectors are of the same size n . Below we show how to use the homomorphic encryption for computing the secure dot product between u and v . At the beginning, Alice encrypts her private vector component-wise, i.e., $z_i \leftarrow HEpk(u_i)$, and send the encrypted vector z to Bob. Upon

receiving z , Bob computes the encrypted component-wise product between z and v based on the multiplicative property, (i.e., $y_i = z_i^{v_i}$, for all $i = 1, \dots, n$). Then he sums up these products based on the additive homomorphic property to compute the encrypted dot product $EDot$ such as: $EDot = y_1 + y_2 + \dots + y_n$. After receiving $EDot$ from Bob, Alice use her private key pr to decrypt it to get the plaintext value of $u * v$, i.e., $HDpr(EDot) = u * v$. Note that Alice's private vector u is not revealed to Bob because only the encrypted values of u are sent to Bob.

4. Proposed Scheme

Before providing our proposed scheme, we explain briefly the method that we used to extract the feature vectors for the image collection.

4.1 Feature Extraction

In this paper, we utilized the SURF algorithm [2], which is a novel scale- and rotation-invariant detector and descriptor. SURF approximates or even outperforms previously proposed SIFT algorithm [1], which is patented, with respect to repeatability, distinctiveness, and robustness, yet can be computed and compared much faster. Generally speaking, SURF extracts the feature vectors of the providing image as follows. First, it selects some interest points at distinctive locations in the image, such as corners, blobs, and T-junctions. Such points are selected in such a way that enables the detector for finding the same physical interest points under different viewing conditions. Next, the neighborhood of every interest point is represented by a feature vector. This descriptor has to be distinctive and at the same time robust to noise, detection displacements

and geometric and photometric deformations. The descriptor vectors are matched between different images. The matching is based on a distance between the vectors, e.g., the Euclidean distance, or cosine similarity.

Formally, given the image Im , we use the SURF algorithm to generate its feature vectors $F=\{v_1, v_2, \dots, v_k\}$, where k is number of interest points in the provided image. Note that different images may differ in the number of descriptors k . Figure 1 illustrates the interest points of Lena image and their counterparts in the same image after rotation.



Figure 1: SURF interest point of two images

4.2 Secure Image Matching (SIM)

The implementation of SIM utilizes the homomorphic encryption for evaluating similarity. Main steps are highlighted in Algorithm 2. Our proposed protocol distributes scores calculation between the two participant parties and it is composed of two phases, namely: initialization and matching phases. In the first phase, each party computes the feature vector set for each image in its own collection and then normalizes each vector to enable evaluating the cosine similarity. We demonstrate the proposed scheme using SURF descriptors in this paper, although this scheme is applicable to other feature vectors. To match her private image, Alice goes into two rounds. In the first round, she encrypts her feature vector set and sends them to Bob. Once receiving Alice's encrypted vectors, Bob employs the *secure_dot_product* subroutine (as explained in Algorithm 3) to return the dot product matrix of the input vector set and the feature vector set of each image in Bob's collection. The details of the above listed subroutine are explained in Section 3.2.1. Without loss of generality, we assume that all

Bob's images are of the same number p of descriptors to make the presentation more clear. At the second round, Alice uses its private key to decrypt the dot product terms and get the actual values, which will be employed for evaluating the similarity scores as explained before in Algorithm 1.

4.3 Complexity Analysis

In this section, we measure the complexity of our proposed scheme in terms of computing and complexity. At the first round of Alice's side, the encryption represents the most expansive operation, which is bounded by $O(k)$, where k is the number of descriptors in the input image. At Bob side, the secure dot product subroutine is run m times, each time it takes $O(kpn)$ complexity. Thus the overall complexity of this step is $O(mkpn)$. The decryption represents the most expansive operation in the second round of Alice's side, which bounded by $O(mkp)$ operations. With respect to the communication cost, we can summarize it as follows: at the first round, Alice sends $k*n$ values to Bob, Bob sends back $m*p*k$ values to Alice. Suppose that each value has b -bit long, then the total complexity is bounded as: $O(b*(kn+mpk))$ bits.

5. System Evaluation

5.1 Security Analysis

In our proposed scheme, Alice encrypts the descriptors of her private image and sends the encrypted descriptors to Bob. Bob computes the encrypted dot products for every image in his private collection and sends them to Alice. For each encrypted dot product, Alice decrypts it to get the actual dot product. Thus the actual descriptors of Alice are well protected due to encryption. Similarly, Bob's collection is protected because it does not moved out.

Recall that our work returns a set of numerical scores to Alice. From these scores, Alice and Bob may decide what images should be retrieved later. This may require the disclosure of a small subset of Bob's images and Alice's targeted image. Although we assume images are kept private, this disclosure is inevitable due to the nature of such application. Thus, we assume disclosure of a small subset of images is allowed provided that these images are very

likely to be similar to the targeted image, and provided both parties agree.

Algorithm 2: Secure Image Matching

Input: I : Alice's image, $D = \{Img_1, \dots, Img_m\}$: Bob's collection.

Output: $\alpha_1, \alpha_2, \dots, \alpha_m$: the similarity scores.

Initialization:

Alice:

- Generate the homomorphic encryption public key pair (pr, pk) .
- Send pk to Bob.
- Use SURF algorithm to extract the feature vector set $F = \{v_1, v_2, \dots, v_k\}$ for the image I , all vectors v_i are of the same size n .
- Compute \vec{v}_i as in (2), for $i=1, \dots, k$, and replace it with v_i in F .

Bob:

For each image $Im_j \in D, \forall j = 1, \dots, m$

- Use SURF algorithm to extract the feature vector set $F_j = \{s_1, s_2, \dots, s_p\}$
- Compute \vec{s}_i as in (2), for $i=1, \dots, p$, and replace it with s_i in F_j .

Matching:

Alice:(first round)

- For $i=1, \dots, k$
 Encrypt the elements of vector \vec{v}_i as:

$$z_{ij} \leftarrow HEnc_{pk}(\vec{v}_{ij}) \text{ for all } j=1, \dots, n$$
- Send z to Bob

Bob:

For $t=1, \dots, m$

Get the feature vector set F_m of image m .

- Compute the secure dot product set between the Alice's vector set and the vector set of image t as:

$$Dot\{t\} = Secure_dot_product(Z, F_t);$$
- Send $Dot\{t\}$ to Alice.

Alice:(second round)

- Receive Dot from Bob, where each element in Dot is a matrix of $[k, p]$ dimensions.
- For $t=1, \dots, m$

Set X to be matrix t of Dot .

$Sum=0;$

For $i=1, \dots, k$

For $j=1, \dots, p$

$sub_j = HDec_{pr}(X_{ij})$

endfor//j

$min = maximum(sub)$

$Sum = Sum + min;$

endfor//i

Compute the distance with image m as:

$\alpha_t = sum / k$

endfor//t

Algorithm3: Secure_dot_product(Z, F)

Input: two feature vector sets Z and F of sizes k and p , respectively.

Output: $Dot[k,p]$: the encrypted sup-product terms between Z and F vectors.

For $i=1, \dots, k$

For $j=1, \dots, p$

$Dot_{ij} = HDec_{pk}(0)$; // initial value

For $t=1, \dots, n$

$Dot_{ij} = (Z_{it}^{v_{jt}}) * Dot_{ij}$

Endfor

Endfor

Endfor

5.2 Experimental Results

In this section, we report the experimental results of the proposed scheme on a real image database containing 1000 color images from the Corel dataset [12]. These images are grouped by content into 10 categories, with 100 images in each category: African, Beach, Architecture, Buses, Dinosaurs, Elephants, Flowers, Horses, Mountain, and Food. Image sizes are either $256*384$ or $384*256$. Our experiments were conducted on a 2.5GHz Intel i5-3210m processor, Windows 7 operating system of 64-bits, with a RAM of 4GB. We used MATLAB R2008a to implement our experiments. We used Java class to implement Paillier cryptosystem. For the SURF descriptors, the size of each descriptor is 64 element, i.e., $n=64$. The normalized vectors are actually scaled by a user specific factor to convert the normalization (between 0 and 1) into an integer numbers. This is because the encryption function is only applied on integer values. Table 3 shows the common symbols used in our experiments. We would like to mention that SURF vectors are already normalized to vector units. So, there is no need for normalizing such vectors.

Table 3: common symbols used in our

Symbol	Meaning
n	Size of descriptors
m	Number of images in Bob's collection
k	Number of descriptors of Alice's image.
p	Number of descriptors of each one of Bob's images.

5.2.1 Effectiveness

In this experiment, we test the ability of our proposed scheme to retrieve the most similar images to the provided query. Figure 2 shows samples of our results. The first column represents the provided image queries. The other columns are the returned images arranged according to their similarity to their corresponding query. It is easy to see that, almost of times, our scheme is able to retrieve the images that are of the same category as of the query image.

5.2.2 Retrieval Accuracy

During the matching step, scores of Bob's database will be returned in a descending order of their similarity to the Alice's query q , as computed using the secure cosine similarity measure. Retrieval performance is usually evaluated using precision-recall curve, where precision and recall terms are defined as:

$$Precision(q) = \frac{|R \cap A|}{|A|}, \quad recall(q) = \frac{|R \cap A|}{|R|},$$

Where R is the set of relevant images, and A is the set of retrieved images. We define the relevant images as those images in the same category as the submitted query q .

From the 1000 images, we selected 100 images as queries. Figure 3 show the precision and recall for different similarity threshold measures for two cases: *Secure cosine* and *plain text cosine* similarities. Secure cosine similarity represents the similarity measured over encrypted vectors. Plain text cosine similarity represents the baseline retrieval performance, where the cosine similarity is evaluated without any encryption. The baseline achieves little better precision and recall than the secure cosine similarity measure. This is because the involved vectors were rounded into integer values to be suitable inputs for the encryption procedure. It is easy to see that encryption of the image feature vectors has very little negative impact on the retrieval performance, and the precision-recall curves before encryption and after encryption are very close to each other. For both cases a precision of 0.9 is achieved with greater than 0.8 recall.





























































Query	Retrieved images				
					
					
					
					
					
					
					
					
					
					

Figure 2: Selected result of retrieved images

5.2.3 Efficiency

In this experiment, we investigate the performance of our proposed scheme in term of matching time. Recall, that our scheme requires n exponentiations and n homomorphic additions to compute the distance between each two vectors. Such expansive operations make our scheme much slower than the non secure scheme. Figure 4 illustrates the average time cost of our scheme against the non secure scheme. In both cases, results are drawn as the number of image queries increases. Every query image is matched against 1000 images.

The average time cost of our scheme, to match a single query, is about 44s, while the other scheme requires 3.6s. The additional time cost of our work can be deemed as a reasonable cost for achieving a secure matching. Our on going research focuses on reducing the feature vector set of each image to improve the efficiency.

Evaluating the secure dot product at Bob's side is bounded to $O(mkpn)$ as explained in Section 4.3. Figures 5 and 6 report the effect

of both p and n on the run time for comparing an query image to an image collection of different sizes. In both experiments, we see that increasing p and n requires more matching time. The primary cause for such cases is that larger vectors and longer vectors requires more encryption operations, respectively. To securely compare an image of 50 vectors against 1000 images, we need approximately 10 minutes.

5.2.4 Security

Recall that our scheme uses a private key to encrypt the feature vectors of Alice party. Without the knowledge of this key no adversary is able to get the right matching scores. In this experiment, we try to see how it's difficult for the adversary Bob, if try, to know the matching scores when using a set of invalid private keys. The first row of Figure 4 shows retrieved image under the valid private key. The remaining rows show the retrieved images under invalid keys. The first column is the provided image queries.

6. Conclusion

Perform image matching while preserving confidentiality is a challenging task. This paper presents a secure scheme to evaluate similarity between image collections of two parties without compromising their privacy. We have utilized the homomorphic properties to design a secure protocol for achieving the cosine similarity between two feature vector sets. Specifically, we used SURF descriptor to extract feature vectors. Interestingly, our proposed framework for secure image matching is not limited to a specific feature vectors. But, instead, it can work under different features. The practical value of our work is demonstrated with several experimental results. Following this line of research, our future work tries to improve the performance of the matching time to scale for massive databases. We could apply clustering techniques to select the representative descriptors for each image. Since clustering selects few descriptors, it is possible to reduce

the distance calculation to a larger extent thereby reducing the matching cost.

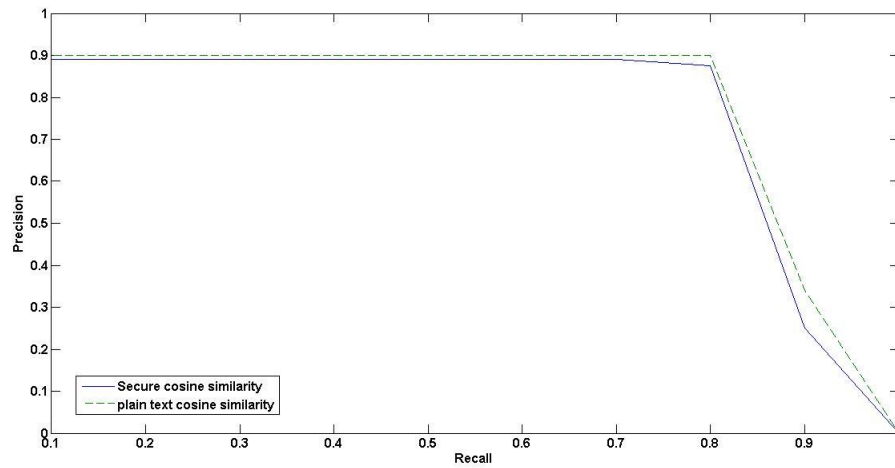


Figure 3: Retrieval Accuracy

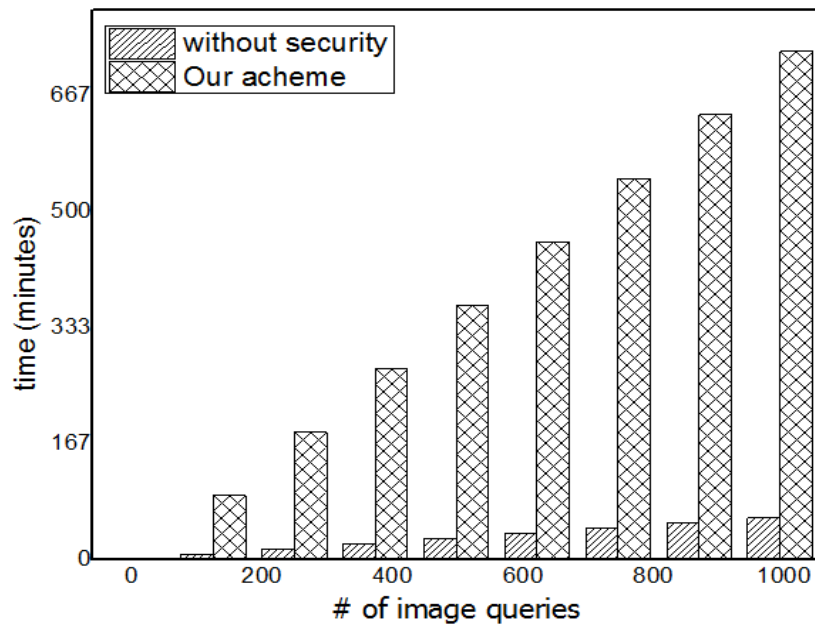
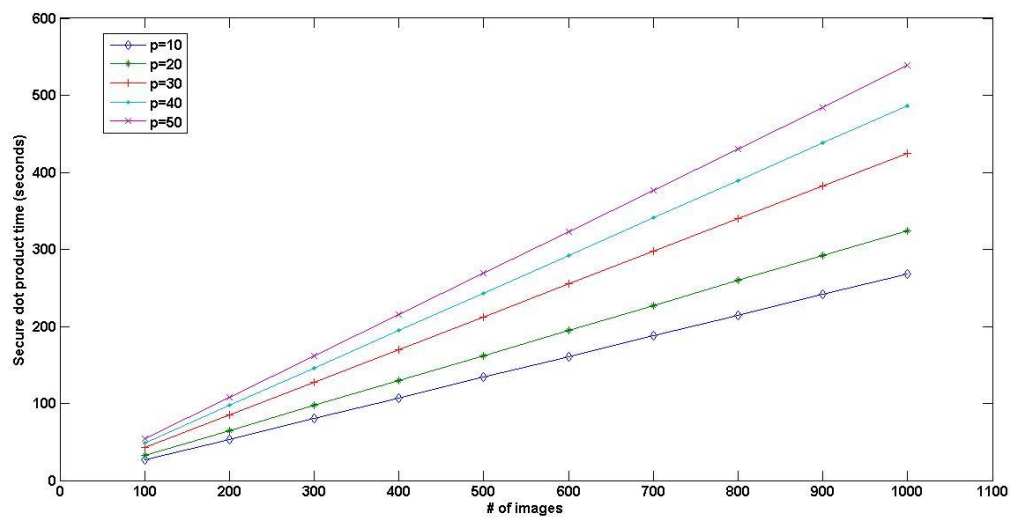
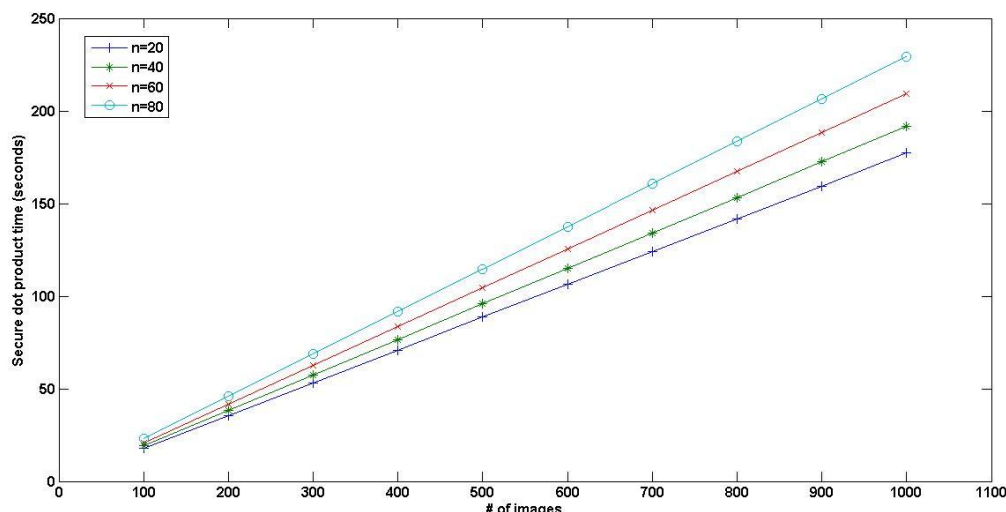


Figure 4: Matching time

Figure 5: effect of p on the matching time

Figure 6: effect of n on the matching time

References

- [1] D. Lowe. *Distinctive image features from scale-invariant key points*. IJCV, 60(2):91–110, 2004.
- [2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. *Speeded-up robust features (SURF)*. Computer Vision and Image Understanding, 110:346–359, 2008.
- [3] O. Chum, J. Philbin, M. Isard, and A. Zisserman. *Scalable near identical image and shot detection*. In Proc. CIVR, 2007.
- [4] O. Chum, J. Philbin, and A. Zisserman. *Near Duplicate Image Detection: min-Hash and tf-idf Weighting*. In Proc. British Machine Vision, 2008.
- [5] J. Shashank, P. Kowshik, K. Srinathan, and C. Jawahar. *Private content based image retrieval*. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2008.
- [6] W. Lu, A. Swaminathan, A. L. Varna, and M. Wu. *Enabling search over encrypted multimedia databases*. In Proc. of SPIE Media Forensics and Security'09, 2009.
- [7] O. Goldreich, S. Micali, and A. Wigderson. *How to play any mental game—a completeness theorem for protocols with honest majority*. In Proc. of the 19th ACM Symposium on Theory of Computing, New York, NY, USA, pp. 218–229, 1987.
- [8] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen. *On secure scalar product computation for privacy-preserving data mining*. In Proc. The 7th Annual International Conference in Information Security and Cryptology (ICISC 2004), Seoul, Korea, pp. 104–120, Dec 2–3, 2004.
- [9] S. Goldwasser, S. Micali, and C. Rackoff. *The knowledge complexity of interactive proof systems*. In Proc. of the 17th Annual ACM Symposium on Theory of Computing (STOC'85), Providence, Rhode Island, USA, pp. 291–304, May 6–8, 1985.
- [10] P. Paillier. *Public key cryptosystems based on composite degree residuosity classes*. In Proc. Advances in Cryptology—Eurocrypt '99. Lecture Notes in Computer Science, vol. 1592, pp. 223–238, Prague, Czech Republic, May 2–6, 1999.
- [11] J. Katz, and Y. Lindell. *Introduction to Modern Cryptography: Principles and Protocols*. 1st edition, CRC Press, 2007.
- [12] Corel test set. [Online]. Available: <http://wang.ist.psu.edu/~jwang/test1.tar>

تمكين المطابقة الآمنة لتجمعات الصور الخاصة

قسم علوم الحاسبات، كلية التربية للعلوم الصرفة، جامعة البصرة، البصرة، العراق
البريد الإلكتروني:

mraiadibraheem@yahoo.com

المستخلص:

تلعب تقنيات مطابقة الصور دوراً أساسياً في العديد من التطبيقات اليومية كاسترجاع الصور المعتمد على المحتوى، رؤية الحاسوب، واكتشاف تكرار الصور القريبة. تفترض الطرق التقليدية أن محتوى الصور يكون غير خاص. وهذا ما يقلل من مدى الاستفادة من تلك الطرق ليكون ملائماً فقط للعمل للبيئات التي تكون فيها الصور عامة الوصول. بصورة أساسية، هذا الافتراض يحدد الكثير من التطبيقات العملية، مثلاً مطابقة الصور بين وكالتي أمن حيث تكون الصور سرية. في هذا البحث قمنا بالأخذ بعين الاعتبار مسألة التطابق الحافظ للخصوصية للصور بين طرفين بحيث لا تكشف الصور للطرف الآخر. في البداية قمنا باستخلاص مجموعة الواصفات لكل صورة. يتم تشفير مجموعة واصفات الصورة المدخلة قبل كشفها إلى الطرف الآخر. قمنا بتطوير أسلوب أمن لقياس التشابه الجيبي بين مجاميع الواصفات بدون الحاجة إلى فك الشفرة. قمنا بإجراء عدة تجارب لتقييم انجازيه الأسلوب المقترح.