QUERY ORIENTED SUBSPACE SHIFTING FOR NEAR-DUPLICATE IMAGE DETECTION

Lei Wu[1], *Jing Liu*[2], *Nenghai Yu*[1], *Mingjing Li*[3]

[1]MOE-MS Key Lab of MCC, University of Science and Technology of China, Hefei 230026, China
[2]Institute of Automation, Chinese Academy of Sciences, Beijing 100080, China
[3]Microsoft Research Asia, 49 Zhichun Road Beijing 100080, China

ABSTRACT

Near-duplicate image detection is a critical task in copyright protection. More challenging than the common similarity search, this task requires not only the retrieval of the top similar images but also the detection of the entire near-duplicates collection from the internet. The common similarity search algorithms are not capable to undertake the latter demand. This paper proposes the query oriented subspace shifting algorithm. The algorithm measures the similarity in various subspaces, which are dynamically generated based on the correlation between samples and the query image. An adaptive threshold is generated automatically to filter the near-duplicates in each subspace. As these subspaces are query oriented, the near-duplicates are less likely to be missed. Experiments shows that this method can effectively improve the detection recall while keeps the similar precision, comparing with the common similarity search algorithm.

Index Terms— Subspace shifting, near-duplicate detection, image copyright protection, similarity search

1. INTRODUCTION

With the advances of web technology, the diffusion of web images has increased exponentially. This greatly aggravates the problem of image copyright infringement.

Although watermark schemes have been proposed [1] to protect the copyrighted images and trademarks, this kind of protection will become inefficacy when the content of the copyrighted image is slightly modified and then republished. To detect these slightly modified images, which is also called near-duplicates, the content-based image replica recognition scheme is proposed [2]. Given a copyrighted image as a query, the task is to find all the accessible duplicates and near-duplicates on the web by content analysis.

The main issues with the near-duplicates detection focus on two aspects, efficient image features and similarity measurement. Considering the efficiency, most features used in the large-scale near-duplicates detection task are simple, such as mean gray, color histogram, texture histogram etc. To measure the similarity, many distance functions are proposed, i.e Minkowski-like metrics, Histogram Cosine distance, Fuzzy logic etc. However, these methods frequently overlook the near-duplicate images. Later, some advanced methods are proposed, such as [4][5]. Although these methods are reasonable, they are not efficient enough for large-scale near-duplicates detection.

Recently, Bin et al. [6] proposed the large-scale duplicates detection algorithm. This method divides the image into patches, and uses the mean gray of each patch as the feature. The hash code is generated from the most distinguishing feature dimensions picked by principle component analysis (PCA) to facilitate fast similarity comparison. Hamming distance is adopted for similarity measurement. This algorithm is reported efficient and still capable to maintain high precision. Yet, as the distinguishing features picked by PCA only characterize the whole dataset, the specific property of the query image is not well utilized. In this paper, we suggest that the similarity measurement should be dynamic according to the query image, and propose the query oriented subspace shifting method to detect the near-duplicates.

The rest of this paper is organized as follows. In Sect.2, we give a brief overview of the proposed method. The two phases of the algorithm are addressed in Sect.3 and 4 respectively. Sect.5 presents the results of the experiment. We offer conclusion in Sect.6.

2. OVERVIEW OF THE APPROACH

Considering both the efficiency and effectiveness, the whole approach consists of two phases, offline indexing and online detection. In the offline indexing phase, the main objective is to provide efficient index of the whole dataset. To achieve this, we transform each image in the database into a low dimensional feature vector, which can be further represented as a compact hash code. The PCA projection matrix for feature dimension reduction is generated in advance from a static sufficiently large image collection. In the online detection phase, we aim at improving the effectiveness of the method without too much cost of efficiency. For this reason, firstly a rough filtering is performed based on the fast hash code matching to remove the major proportion of non-duplicates. Then on the relatively small remaining set, the proposed iterative subspace shifting algorithm is used to refine the roughly filtered

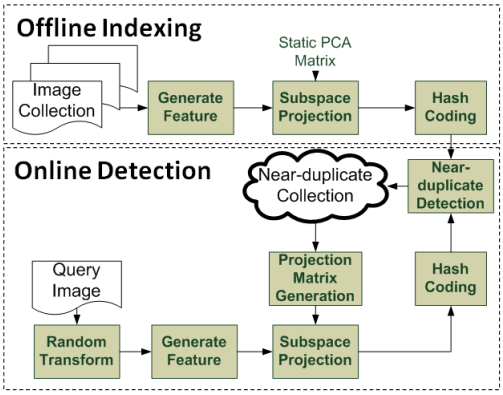

Fig. 1. Flow chart of the approach

results. The flow chart of the algorithm is illustrated in Fig.1.

3. OFFLINE INDEXING

3.1. Feature

The meaning of an image is effectively expressed by color, texture, and structure information. The commonly used image modification (discussed in introduction) will directly change the color. So texture and structure information, although may be indirectly altered a little bit, seems more robust in the duplication detection task. Therefore, we propose to adopt the patch-based texture histogram feature for near-duplicate detection.

In the generation of this feature, each image I is firstly divided into 8×8 equal-sized patches. For each patch, the 8-bin texture histogram $h(k), k = 0, \cdots, 7$ are calculated.

$$h(k) = \sum_{d_{ij} \in d_k} m_{ij} \tag{1}$$

$$m_{ij} = \sqrt{dx_{ij}^2 + dy_{ij}^2} \tag{2}$$

$$d_{ij} = \arctan \frac{dy_{ij}}{dx_{ij}} - D \tag{3}$$

$$dx_{ij} = I_{ij} - I_{i+1,j} \tag{4}$$

$$dy_{ij} = I_{ij} - I_{i,j+1} \tag{5}$$

$$D = \arctan \frac{\sum_{ij} dy_{ij}}{\sum_{ij} dx_{ij}} \tag{6}$$

where I_{ij} is the i, j pixel value of each image. The gradient magnitude of the i, j pixel is m_{ij}. d_{ij} is the gradient direction at pixel i, j. The k^{th} dimension $h(k)$ of the texture histogram represents the total intensity of the pixel gradient whose direction lies in the k^{th} direction bin $d_k, k = 0, \cdots, 7$. The direction bins are defined by the relative angle to the dominant gradient direction D of the whole image as shown in Fig.2.

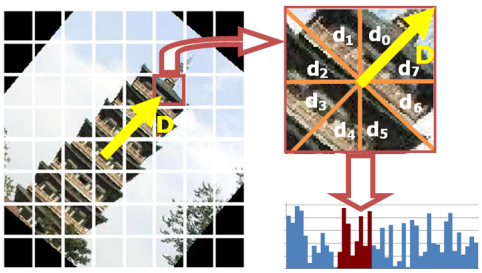

Fig. 2. The generation of texture histogram feature

Finally, putting all the patch texture features of the image together, a 512-dimensional feature vector is formed. The feature generation process is illustrated in Fig. 2.

3.2. Subspace hash coding

For the consideration of efficiency and robustness to noise, PCA is applied to project the image feature to lower dimensional space. The projection matrix is prepared based on a sufficiently large image collection. The property of PCA ensures that the features are projected along the most distinguishing d dimensions. As this projection matrix does not change with the query image, it is called static projection matrix.

In the lower dimensional space, the image is able to be represented by a further compact hash code to reduce the calculation burden in similarity measurement. We adopt the same hash coding method as [6], which is briefly listed as follows.

$$C_{ik} = \begin{cases} 0, & \text{if } v_{ik} > mean_k; \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

where C_{ik} is the k^{th} bit of the hash code for image i, and v_{ik} is the k^{th} dimension of the feature vector for image i. $mean_i$ is the mean of i^{th} dimension of the feature vector over all the images.

4. ONLINE DETECTION

4.1. Rough filtering

For large scale data set, complex similarity search algorithm can not be applied for the limitation of memory and computational time. So it is necessary to reduce the scale of the collection by efficient rough filtering. For this purpose, hash code matching by Hamming distance is adopted. We set a loose threshold on the hash code distance to exclude the images which are obviously different from the query.

Suppose the query image is $q \in R^n$. An image I_j is believed close to the query image if and only if

$$\|H(Pq) - H(PI_j)\|_\kappa < \epsilon \tag{8}$$

where P is the static projection matrix. $H(\bullet)$ is the hash coding function. κ represents the corresponding subspace, and ϵ is the threshold to determine whether the image is close to the query or not in the subspace. The set of samples which are close to the query are called *query surrounding samples*. All the query surrounding images form the *query surrounding collection* Q_s.

In order to determine the loose threshold ϵ for rough filtering, several random transformations are generated from each query image and represented in hash code in the same subspace projected with the static PCA matrix. The largest Hamming distance between the query and its transformations is set as the threshold.

$$\epsilon = \max_l \|Pq_j - Pq_j^{(l)}\|_\kappa \tag{9}$$

where $q_j^{(l)}$ is the l^{th} random transformation of the query image q_j.

4.2. Query oriented subspace shifting

Since the hash code matching has provided a much smaller query surrounding collection, we can use an iterative scheme to detect the near-duplicates from this collection. For each iteration, PCA eigenspace of the query surrounding samples is selected as the optimal subspace for measuring the similarity among the query surrounding samples. This subspace keeps as much of the variance of the collection as possible. The remote samples will then be excluded from the query surrounding collection. As the collection is updated, the eigenspace will of course shift. So in the next iteration, the similarity measurement will be performed in another eigenspace. It is more probably that the near-duplicates would remain close to the query after the subspace has shifted, while non-duplicated images which may form a cluster in a previous subspace will scatter in the subsequent spaces. This scheme is presented in detail as follows.

Step 1: Calculate the closeness threshold ϵ in the subspace κ by the same means in rough filtering;

Step 2: Select the query surrounding samples and update the Q_s;

$$Q_s = \{I_j | \|PQ - PI_j\|_\kappa < \epsilon\} \tag{10}$$

Step 3: Update the projection matrix P based on the query surrounding collection;

$$P_i \leftarrow eigenvector(cov(Q_s), i) \tag{11}$$

$$P = [P_0, P_1, \cdots, P_d] \tag{12}$$

where $eigenvector(cov(Q_s), i)$ is the i^{th} sorted eigenvector of the covariance matrix for query surrounding collection, and d is the dimension of the low dimensional space.

Step 4: Repeat Step 1 and 3, until the query surrounding collection Q_s does not change. So far, we believe all the non-duplicates surrounding the query image are filtered, and the algorithm finishes.

Threshold ϵ in each iteration is calculated in the same way as the rough filtering step. The only variation is the projection matrix P. So the threshold is adaptive to the query in different subspaces.

5. EXPERIMENT

5.1. Testbed

To evaluate the effectiveness of the proposed method, a testbed consists of a large number of images with variety of near-duplicates should be built. Although there are huge amount of web images as well as their near-duplicates in the internet, the ground truth is difficult to obtain. For this reason, we have to construct a huge artificial data set to simulate the realistic internet environment.

The dataset used in the experiment is the 2,600,000 photos from Flickr. We randomly pick 500 images from the dataset as query images. For each query image, 43 kinds of transformations defined in [2] are constructed and mixed into the data collection as duplicates. So in total there are 44 near-duplicates including the original image for each query.

5.2. Parameter setting

All the images are normalized into either size of 240×160 or 160×240. Each image is divided into 8×8 patches. Image features are all projected into the 32-dimensional subspace for hash coding. Hamming distance is adopted to compare two hash codes.

We take Bin's method as the baseline. For this method, these is one parameter θ, which is the threshold for near-duplicates validation. The range of the threshold is [0,32]. We try the thresholds from 0 to 32 with interval of one, and make sure of the optimal performance. In order to distinguish the contribution of feature from that of algorithm, we use both gray feature and texture feature in Bin's approach.

5.3. Evaluation

To evaluate the performance, we input 500 query images into the two systems and calculate the average precision and recall.

For each query, the precision and recall are defined as follows.

$$Precision = \frac{T_+}{T_+ + T_-} \tag{13}$$

$$Recall = \frac{T_+}{T_+ + F_+} \tag{14}$$

where T_+ is the number of true near-duplicates in the search result. T_- is the number of non-duplicates in the search result. F_+ is the number of true near-duplicates that are not in the search result. The average precision and average recall are calculated over the 500 queries.

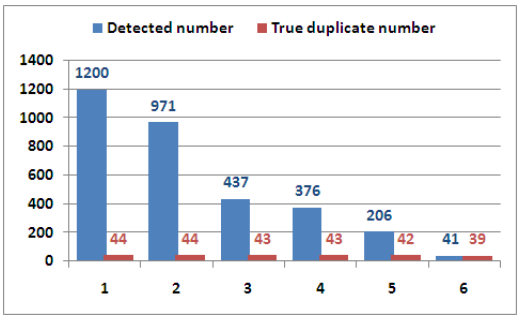

Fig. 3. Detection rate under each iteration

5.4. Performance

In order to facilitate the comparison, we adopt the following abbreviation. G-HC represents Bin's hash coding approach with gray feature. T-HC denotes Bin's approach using the texture histogram feature. QOSS is the query oriented subspace shifting algorithm with texture histogram feature. The results are given in Table 1.

Table 1. Near-duplicates detection performance

Methods	G-HC	T-HC	QOSS
Precision	96.57	96.82	96.85
Recall	69.97	73.15	90.34

Table 1 shows that comparing with Bin's method, the QOSS method has greatly improved the recall while keeps similar precision. For Bin's method, the similarity measure is done in a single subspace. In order to keep relatively high precision, the near-duplicate criterion should be strict. Even some near-duplicates may not follow. For the proposed method, the similarity is measured on multiple subspaces iteratively, and in each subspace the criterion may not necessarily be strict to maintain high precision.

Table 1 also shows that the texture histogram feature is superior than the mean gray feature. It is obvious that texture histogram feature contains more discriminative information than the simple mean gray feature. This superiority is obtained with the cost of computational complexity. Using the PC with 3GHz CPU and 2G memory, the generation of mean gray feature needs 0.013 second/image, while the texture histogram feature costs 0.237 second/image. However, the most time consuming process of feature extraction of the large image collection can be performed offline. For the online query processing, 0.237 second does not matter so much either. So the calculation of texture histogram does not form the bottle neck in this task.

Fig.3 shows the average number of detected images and average number of true duplicates in each iteration. The horizontal axis is the iteration number. It shows that after several iterations, the majority of non-duplicate images are eliminated, while most of the truth duplicates are preserved.

6. CONCLUSION

In this paper, we proposed the query oriented subspace shifting algorithm to detect the near-duplicate images. Superior to the common hash coding duplicate detection method, the proposed method dynamically choose the optimal subspace for similarity measure according to the property of the query image. By using this algorithm, the detection recall has been greatly improved. Meanwhile, the precision keeps the same with traditional method. With both high precision and high recall, this algorithm is more capable for pirate image detection than the commonly used similarity search algorithm.

7. ACKNOWLEDGEMENT

The research is supported in part by National 863 Project (2006AA01Z315), National Natural Science Foundation of China (60675003), National Natural Science Foundation of China(60672056) and Specialized Research Fund for the Doctoral Program of Higher Education (20070358040).

8. REFERENCES

[1] S. Ketchpel H. Garcia-Molina and N. Shivakumar, "Safeguarding and charging for information on the Internet," in *Proc. of International Conference on Data Engineering*, 1998.

[2] E.Y.Chang A.Qamra, Y.Meng, "Enhanced perceptual distance functions and indexing for image replica recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 379–391, March 2005.

[3] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition", year = "2004"*.

[4] L. Huston Y. Ke, R. Sukthankar, "Efficient near-duplicate detection and sub-image retrieval," in *Proc. of ACM Multimedia (MM'04), 2004*, 2004.

[5] D-Q. Zhang and S-F. Chang, "Detecting image near-duplicate by stochastic attributed relational graph matching with learning," in *Proc. of the 12th ACM Multimedia*, 2004.

[6] B. Wang, Z. W. Li, M. J. Li, and W. Y. Ma, "Large-scale duplicate detection for web image search," *Proc. of IEEE International Conference on Multimedia & Expo (ICME'06)*, 2006.