Memory Efficient Recognition of Specific Objects with Local Features

Koichi Kise, Kazuto Noguchi, Masakazu Iwamura
Dept. of Computer Science and Intelligent Systems,
Graduate School of Engineering, Osaka Prefecture University
{*kise, masa*} *@cs.osakafu-u.ac.jp, noguchi@m.cs.osakafu-u.ac.jp*

Abstract

Balancing the recognition rate, processing time and memory requirement is an important issue for object recognition based on local features. For the task of recognizing not generic but specific objects (object instances), a larger number of local features enable us to improve the recognition rate but pose a problem of processing time and memory requirement. For the problem of processing time, approximate nearest neighbor search is known to be extremely effective. In this paper, we propose a method of memory reduction by applying scalar quantization to local features. From experimental results on 100,000 images, we have found that the scalar quantization is of great help; As compared to the original representation with 16 bit/dimension, the recognition rate of 98.1% was kept unchanged using the representation with 2 bit/dim.

1. Introduction

Local features such as SIFT [4] enable us to make object recognition robust to occlusion and variations of imaging conditions [7]. A common approach is called "bag of words" or "bag of features" which describes images without taking into account physical arrangement nor co-occurrence of local features. An unknown image is recognized by matching its local features with those of modeled objects.

Because hundreds to thousands of "raw" feature vectors are generally extracted from a single image, both memory requirement for recording them and computational burden of their matching are important problems for this approach. A typical solution to these problems is to define "visual words" by applying vector quantization to the above raw feature vectors. Visual words are then employed to describe images. The above problems can be solved if the number of visual words is small enough, e.g., a few thousand. For the task of generic object recognition and indexing of video data, the above holds to obtain good performance [6, 7, 8]. For the task of specific object recognition, on the other hand, it is reported that accuracy is improved with a larger number of visual words [5]. An extreme is to store all raw feature vectors as visual words. In such a case, both storage and computation of matching are problematic.

The problem of computation can be solved by approximating the nearest neighbor search for matching feature vectors. In [3], it is reported that matching of 600 vectors with 200M vectors (from 100,000 images) can be done in 4.5 ms, which is about 10^7 times faster than the brute force matching. On the other hand, a currently available method for the problem of storage is to employ a smaller number of visual words, though it lowers the recognition rate.

In this paper we propose a simple method of memory reduction that is effective to a large number of feature vectors. Our proposal is to reduce the amount of memory required for the storage by scalar quantization which limits the number of bits for each dimension. Experiments using 100,000 images of specific objects demonstrated that the recognition rate was kept unchanged with down to 2 bit/dimension.

2. Proposed Method

2.1. Recognition by voting and its problems

We are concerned here with a simple way of recognizing objects by voting. A specific object (instance) to be recognized is stored in the database as a set of feature vectors extracted from its image with its object label. An unknown image, which we call a "query", is also represented as a set of feature vectors (query vectors). For each query vector, its nearest neighbor is found and the label is used for voting. The object with the maximum number of votes is the result of recognition.

In this scenario, there are at least two problems to be solved for realizing a large scale and real-time ob-

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

ject recognition. First, due to a large number of feature vectors in the database, the simple brute force matching by linear search is prohibitive; We need to introduce a sophisticated way of matching to reduce its burden. Second, a large number of feature vectors also cause a problem of storage. It is necessary for a recognition method to keep them on the main memory for fast matching. However, many high dimensional feature vectors require a large amount of memory, which is also prohibitive.

For the first problem, we employ a hash-based fast access to "approximate" nearest neighbors proposed in [3]. For the second problem, we propose scalar quantization of feature vectors. A representation with a smaller number of bits per dimension is expected to deteriorate the accuracy of recognition. In this paper, we experimentally validate to what extent the memory reduction is admissible.

2.2. Scalar quantization

In the proposed method, we utilize feature vectors obtained by PCA-SIFT [2] whose dimensions are 36. Due to the nature of PCA-SIFT, the histogram of values of each dimension is either unimodal or bimodal, whose average is almost 0. For each dimension, thresholds are determined to obtain a quantized representation. For example, three thresholds t_1,...,t_3 are used for the 2 bit/dim. representation. Then, quantized values 0,...,3 are assigned to intervals $[-\infty, t_1)$,...,$[t_3, \infty]$, respectively. All thresholds are determined in such a way that each quantized value is equally likely for all feature vectors in the database.

2.3. Hash function

For fast access to a feature vector, we employ a hash function as follows. Let $\boldsymbol{x} = (x_1, ..., x_{36})$ be an original feature vector obtained by PCA-SIFT. Its binary representation $\boldsymbol{u} = (u_1, ..., u_d)$ is defined using the first d dimensions of $\boldsymbol{x}$ as follows:

$$u_i = \begin{cases} 1 & \text{if } x_i - \theta_i \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where θ_i is the average of the value of x_i in the database. The hash function is defined by

$$H_{\text{index}} = \left(\sum_{i=1}^{d} u_i 2^{i-1} \right) \mod H_{\text{size}}, \tag{2}$$

where H_{size} is the size of the hash table.

2.4. Storage

For each image of a specific object, feature vectors are extracted and their scalar quantized version are stored in the hash table. The collision is resolved by chaining: Feature vectors with the same hash key are stored in the list. A lengthy list generally indicates that the feature vectors stored in the list are less discriminative. Thus we employ the maximum length c for the list length n and if $n > c$ the list is deleted with its entries from the hash table.

2.5. Recognition

A query is recognized as follows. First, the query is represented as a set of feature vectors (query vectors). For each query vector, the hash table is accessed to obtain a set X of feature vectors in the database. The feature vector $\boldsymbol{x}_* \in X$ nearest to the query vector is found by calculating the Euclidean distance on the *scalar quantized domain*, and the object label of $\boldsymbol{x}_*$ is employed for vote.

The most important step is how to determine X. It should include the exact nearest neighbor with a high probability. The trivial solution is to select all feature vectors as X. However the number of feature vectors $|X|$ should be small enough to keep the computational cost low. Another simple solution is to define X with the feature vectors having the same hash key. However the exact nearest neighbor may have a different hash key due to variations.

A good compromise is achieved using a limited perturbation of bit vectors. Let $\boldsymbol{q} = (q_1, ..., q_d)$ be a query vector. For the dimension i such that

$$|q_i - \theta_i| \leq e, \tag{3}$$

the nearest neighbor of $\boldsymbol{q}$ is considered to have a different bit representation because q_i is close to θ_i. Thus not only its binary representation u_i but also the other one ($u_i' = (u_i + 1) \bmod 2$) are employed for the access to the hash table to determine X. In order to avoid the combinatorial explosion, we limit the number of dimensions for this process to b. In the case that the number of dimensions which satisfy Eq.(3) exceeds the limit b, those with smaller indices are adopted up to the limit.

3. Experiments

3.1. Experimental setting

We evaluated the proposed method using the following image databases and queries.

(a) (b) (c)

Figure 1. Examples of images in DBs.

As the images in the databases we employed in total 100,000 images collected using Google image search (Fig. 1(a)) and Flickr (Fig. 1(b)), as well as images which were available at the site of PCA-SIFT [1] such as in Fig. 1(c). Images were resized to have their longest side less than 640 pixels. The average number of feature vectors extracted from an image was about 2,000. Using these images, we prepared the five databases including 1,000, 5,000, 10,000, 50,000, and 100,000 images, respectively, where all images in a smaller database were included in a larger database.

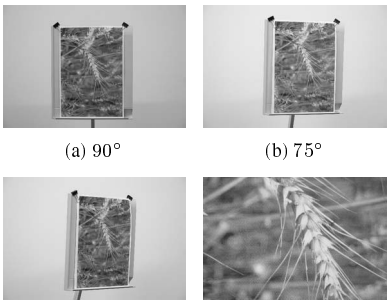
Images used as queries were prepared as follows. We first selected at random 500 images from the smallest database. Then these images were printed out and converted into images by taking their pictures in four different ways as shown in Fig. 2. The size of the images were reduced to 512×341 pixels and then PCA-SIFT was applied to extract feature vectors. About 600 feature vectors were obtained on average from an image.

Experiments were performed using a computer with AMD Opteron 2.8GHz CPU and 32GB RAM. In the following, processing time means the average time needed for recognition of a single query excluding the time for extracting local features.

3.2. Required memory and the recognition rate

First we investigated how the memory reduction by scalar quantization affected the recognition rate. Figure 3 shows the results where the horizontal axis indicates the number of bits for representing a single dimension. For example, since the number of dimensions of a PCA-SIFT vector is 36, 2 bit/dim. requires $2 \times 36 = 72$bits for a single vector. 16 bit/dim. means that the original PCA-SIFT vectors were employed. For the case of 0 bit/dim., no distance calculation was employed; All the feature vectors retrieved from the hash table were simply employed for voting. The values of parameters were $b = 10, c = 10, d = 28, e = 400$, which showed a good performance for 16 bit/dim.

[1] http://www.cs.cmu.edu/~yke/pcasift/



(a) 90° (b) 75°

(c) 60° (d) Part

Figure 2. Examples of queries.

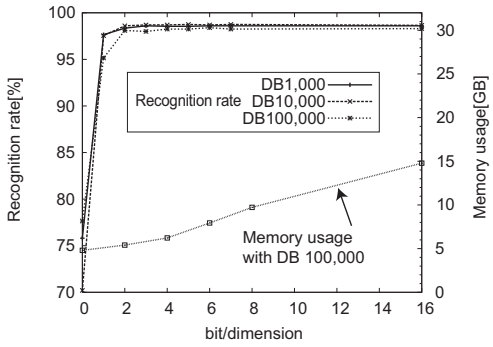


Figure 3. Required memory and recognition rate.

It is amazing that from 16 bits down to 2 bits, the memory reduction has almost no harmful effect on the recognition rate. This means that $2/16 = 1/8$ reduction of the memory for a feature vector was achieved. The total memory requirement was reduced to 1/3 for the case of 100,000 image database. Further reduction is possible with the representation of 1 bit/dim. but the recognition rate suffers especially for larger databases. The use of 0 bit representation cannot be recommended due to the decline of the recognition rate.

3.3. Scalability

Next we evaluated how the method scales with respect to the processing time and the recognition rate. Figure 4 illustrates the results obtained using the same parameter values shown above. Both the recognition rate and the time for 2 bit/dim. are very close to those for 16 bit/dim., independently of the size of databases. For the case of the 100,000 image database (DB100,000), for example, the recognition rate and the time for 2 bit/dim. were 98.1% and 120ms, and those for 16 bit/dim. were 98.4% and 107ms, respectively.

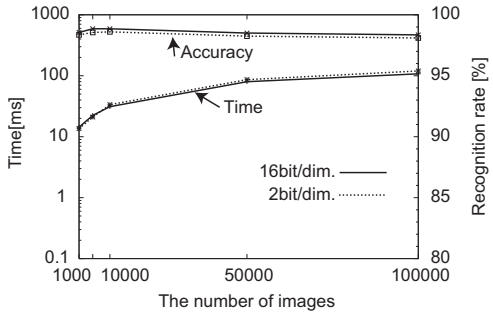


Figure 4. Time and recognition rates for databases of various sizes.

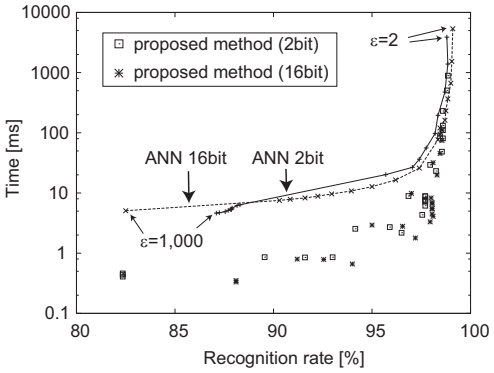


Figure 5. Recognition rate and time.

From this experiment, we have confirmed that the scalar quantization with 2 bit/dim. has little harmful effect not only on the recognition rate but also on the time, for this task of specific object recognition.

3.4. Comparison to recognition with ANN

To test whether the scalar quantization is effective only for the proposed hash-based method, we employed ANN [1], a well-known method of approximate nearest neighbor search, for recognition by voting and compared it to the proposed method. The number of images of specific objects in the database was fixed to 10,000 and various parameters were applied. For the parameter of ANN, the level of approximation ϵ was changed from 2 to 1,000. Figure 5 and Table 1 show the results. For the performances of both methods with 2 bit/dim. at the recognition rate of 90% or more, we can find similar performances with 16 bit/dim. Thus the scalar quantization is also effective to the recognition with ANN. Note also that the proposed method with 2 and 16 bit/dim. were superior to the method with ANN; In order to obtain the same recognition rate, the processing time of the proposed method was about 5–10 times faster.

Table 1. Results of the proposed method.

bit/dim.	parameters	recog. rate	time
16	$b = 14, e = 600$	98.9%	333.8 ms
16	$b = 10, e = 400$	98.9%	32.1 ms
16	$b = 4, e = 200$	96.3%	1.6 ms
2	$b = 14, e = 400$	98.6%	300.2 ms
2	$b = 10, e = 400$	98.6%	27.0 ms
2	$b = 4, e = 200$	95.8%	1.8 ms

($c = 10$, $d = 28$ were commonly employed.)

4. Conclusion

We have presented simple scalar quantization for a large scale recognition of specific objects with less amount of memory. The experimental results have shown that the representation of 2 bit/dim. allows us to achieve the memory reduction to 1/3 with the recognition rate and the time comparable to their original (16 bit/dim.). Future work is further improvement of the recognition rate without increasing time and memory.

Acknowledgment

This work was supported in part by the Grant-in-Aid for Scientific Research (B)(19300062) from Japan Society for the Promotion of Science(JSPS).

References

[1] S. Arya, D. M. Mount, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45(6):891–923, 1998.

[2] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *CVPR2004*, volume 2, pages 506–513, 2004.

[3] K. Kise, K. Noguchi, and M. Iwamura. Simple representation and approximate search of feature vectors for large-scale object recognition. In *Proc. BMVC2007*, pages 182–191, 2007.

[4] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[5] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proc. CVPR2006*, pages 775–781, 2006.

[6] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Proc. ECCV2006*, volume 4, pages 490–503, 2006.

[7] J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, editors. *Toward Category-Level Object Recognition*. Springer, 2006.

[8] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. ICCV2003*, volume 2, pages 1470–1477, 2003.