Available online at www.ijpe-online.com

International Journal of Performability Engineering

Vol. 13, No. 5, September 2017, pp. 657-669
DOI: 10.23940/ijpe.17.05.p10.657669

Image Retrieval Method based on Multi-View Generating and Ensemble Learning

Huanyu Li*, Yunqiang Li, Yufei Zha

Air Force Engineering University, Xi'an City, China

Abstract

This paper addresses the problem of approximate nearest neighbors (ANN) search in large-scale image collections. Inspired by the idea of multi-view observation in daily life, we propose a novel unsupervised hashing method to solve large-scale image retrieval on the scenarios of single information source, dubbed Multi-view Ensemble Hashing (MEH). MEH is realized by ensemble learning and a parallel architecture. In our approach, MEH learns a set of convolution filters from abundant images by principal component analysis (PCA) off-line at first. Next, MEH filters the original image collection of single information source respectively via these convolution filters, to generate the multi-view data itself. Then, MEH uses a traditional hashing method to learn hash function and hash code respectively in each generated view. Finally, MEH merges the results of multi-view together to achieve a final retrieval result by voting. Extensive experiments on dataset CIFAR-10 and LabelMe show the superiority of our proposed approach over several state-of-the-art hashing methods. Compared to the original hashing methods that used as the operator in MEH, our proposed approach improves the retrieval precision over 100% at code size of 16-bit, and 10% at code size of 256-bit. Furthermore, the cost of MEH maintains an approximate level for its parallelizable structure.

Keywords: Image retrieval; Hashing; ensemble learning; Principal component analysis; Convolution filters

(Submitted on May 5, 2017; Revised on July 15, 2017; Accepted on August 22, 2017)

1. Introduction

With the advancement of the Internet, we are inundated by abundant data of image, document, music, and video. Data tends to have similar semantics, as the density of similar objects increases in data space. Approximate nearest neighbors search has been a very challenging problem in large-scale and high-dimensional data set [15].

Hashing method attracts extensive attention for its computational and storage efficiencies [13,17,19,21,22]. Specifically in image retrieval, a widely used application of ANN search. Hashing methods aim to map original high-dimensional data to compact binary codes, and distinguish the similarity of data points by their Hamming distance. An effective scheme of hashing method should have such properties [24]: First, short codes, the binary codes should be short to enable the storage of a large-scale image data set in a limited RAM. Second, similarity preserve, similar data (either in terms of feature space distance or semantic distance) should have binary strings of low Hamming distance. Third, efficient and scalable, the approach for parameters learning and new data encoding should be efficient and scalable.

There exists a dilemma between the hashing code length and performance in image retrieval. On one hand, hash code is expected to be short for fast calculation and low storage cost. Generally it is less than 256 bits. On the other hand, the intra-class variety of objects and the inter-class indistinction of their interface are aggravating while the data scale increases. Moreover, there is information loss when the original high-dimensional data is mapped to compact binary codes. Therefore,

* Corresponding author. Tel.: +86-13389258581; fax: +86-13389258581.
E-mail address: lihuanyu1984@163.com.

the hashing efficiency and codes discrimination have caused a performance bottleneck, and short hash code has been unable to provide enough discrimination to distinguish objects.

With the development of hashing, several novel hashing methods have been proposed to improve the retrieval performance, which are based on scenarios of multiple models or multiple labels [5,18,29,30]. These methods generally support cross-view searching by combining or concatenating hashing codes learned from data, and need multiple information sources as a precondition. Although some of these methods are dubbed "multi-view", they are indeed multi-source in information but not multi-view in vision. For example, an object "dog", these methods of two-view need an input data of both a text and an image description of "dog". Therefore, they generally work well when all the information sources are available, which is too demanding in the real world.

Overall, improving retrieval performance on the most common scenarios of single information source is still a challenging problem to be solved.

1.1. Motivation

When we want to recognize an object by vision, besides watching it carefully from a single viewing angle (like front), we would also watch it comprehensively from many other viewing angles (like top and bottom, left and right, etc.). That is, because we would obtain a more accurate recognition by analyzing all the multi-view results, even if we have not seen it clearly in any single-view. So, when we deal with a similar thing by computer (like image retrieval), one natural idea is whether we could use the same idea to resolve the conflict between hashing efficiency and codes discrimination.

In this paper, we propose a novel unsupervised hashing learning model based on the idea of multi-view observation, dubbed Multi-view Ensemble Hashing (MEH), to solve the large-scale image retrieval problem on scenarios of single information source.

1.2. Implementation

An overview of our proposed framework is illustrated in Figure 1. In this work, MEH can be roughly divided into two parts: off-line and on-line. The off-line part is to find a set of orthogonal filters, and the on-line part is to generate multi-view data and learn hashing. First, we learn filters from abundant images by principal component analysis (PCA). Second, we use these filters to do convolution filtering on the original image data, which is of single information, to generate new multi-view data. Third, we treat the data of each view as a new data set to start hash learning respectively. In this step, we treat each view as an individual learner of an ensemble learning system and use a traditional hashing method (like LSH) as the operator to learn the hash function and binary codes of this view. Fourth, we evaluate the similarity of original data by their hamming distance in each view and use the "voting" idea in ensemble learning to obtain a final similarity estimation. Thus, we accomplish retrieval by distinguish samples roundly.

1.3. Contributions

We evaluate our proposed MEH on dataset CIFAR-10 and LableMe, and compare it with several unsupervised hashing methods which are also based on single information source. Experimental results demonstrate the advantage of MEH. Our main contributions include:

- We propose a novel method for multi-view generating. Our method is extremely conducive to the demands of ensemble learning, and it supports parallel computing well. To the best of our knowledge, it is the first time to solve image retrieval by convolution filtering and ensemble learning.
- Our hashing method can boost the benchmark of image retrieval on scenarios of single information source. In our model, existing unsupervised hashing methods can be used as the operator in multi-view, and obtain a remarkable performance improvement by ensemble learning, especially at small code size. For instance, our method improves the performance of Locality Sensitive Hashing (LSH) and Iterative Quantization (ITQ) over 100% at code size of 16-bit.
- Our hashing method maintains an approximate complexity as traditional method. In MEH, the on-line parts are based on ensemble learning theory, which is inherently favorable to parallel computing. Its process can be assigned to multiple independent computation units. Therefore, it can maintain an approximate complexity and storage requirement as the traditional hashing method, which is used as the operator in multi-view.

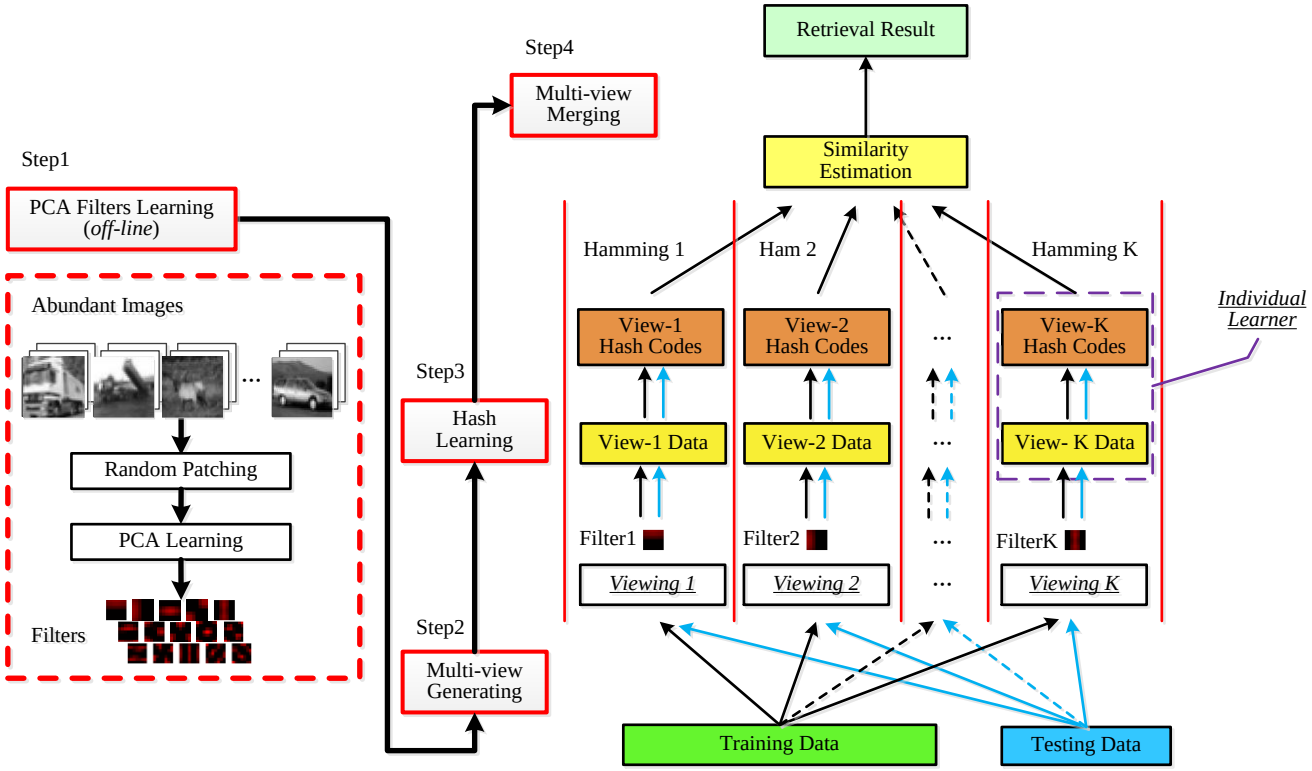


Figure 1. The proposed ensemble hashing. It is consisted of off-line and on-line two parts. Filters used for convolutional filtering are learned off-line. The processes of multi-view data generating, hash learning of each view, and multi-view merging are accomplished on-line. As the on-line processes are mainly parallel processing, the calculation can be distributed to multiple computers.

2. Related work

In this section, we give some backgrounds about hashing and introduce some related works, which attempt to handle ANN search problem with "multi-view", PCA, or bagging.

Existing hashing methods can be roughly divided into two categories according to the usage of label information: supervised and unsupervised. Here we mainly discuss the unsupervised methods that this paper focuses on. Some unsupervised hashing methods are data-independent, which use simple random projections as hash function, irrespective of the distribution of training data, like LSH and its extensions [1]. In contrast, most unsupervised hashing methods are data-dependent, whose hash functions are learned based on the distribution of training data, like Spectral Hashing (SH) [28], Isohash [11], ITQ [24], Weighted Hashing [27], SPEC Hashing [16], etc. These traditional hashing methods all face the dilemma between the code length and performance in image retrieval.

2.1. "Multi-view" hashing

As mentioned in the beginning of this paper, several cross-view hashing methods have been proposed to improve the retrieval performance in recent years, like multi-label, multi-modal, and "Multi-view" [5,18,29,30]. These cross-view hashing methods are mainly based on multiple information sources. They are indeed not multi-view in vision but multi-source in information, because they need training data of multiple labels that are tabbed artificially. Moreover, most of these methods focused on the process of joint hashing learning and did not discuss the process of multiple information sources generating. In addition, the demand of parallel processing on multiple information sources was not considered yet, which is important in retrieval problem of large-scale data set. Specifically, Compact Kernel Hashing with Multiple Feature (MFKH) [20] considers the process of multi-view generating. First, MFKH extracts a 384-dimension GIST feature and a 300-dimension visual words tag from each image of dataset CIFAR-10. Then, MFKH formulates the multiple feature hashing as a similarity preserving problem with optimal linearly combined multiple kernels. Finally, MFKH achieves image retrieval based on the similarities conveyed by the two different features.

2.2. Bagging PCA hashing

Bagging PCA Hashing (BPCAH) [14] is a hashing method based on eigendecomposition and bagging, which integrates the bootstrap sampling with PCA to learn effective binary codes. BPCAH randomly samples a small fraction of the training data to learn the PCA directions, and keeps only the top eigenvectors each time to generate one piece of short code. This process can be seen as a block each time, and BPCAH repeats it several times to form several blocks. Finally, BPCAH concatenates the obtained short codes of blocks into one piece of long code.

BPCAH uses PCA to reduce the dimension of original data, as the usage in PCAH [24] and many other applications, and series connects the short codes learned from blocks. Moreover, its bagging scheme is benefit for parallel computing.

2.3. Ensemble learning

Ensemble learning and parallel computation provide the realization foundation for our MEH. Ensemble learning is a normal form in machine learning [4,9]. It uses multiple homogeneous learners together to solve one question and works out a result by voting. Parallel computation [25] uses distributed memory multicomputer to execute one program, thus improves calculation efficiency and solves complex problem. Ensemble learning and parallel computation have shown their advantage in big data processing. Based on the idea of bagging in ensemble learning, Kleiner proposed a novel procedure (called the Bag of Little Bootstraps, BLB) to break the bottleneck of bootstrap on big data [10]. Based on multi-computer cluster, Gonzalez proposed a distributed graph-parallel computation frame (called "GraphLab") to realize machine learning on big data [8].

2.4. Difference

All these researches abovementioned inspire us to solve large-scale image retrieval problem by ensemble hashing.

Our MEH is quite different from the hashing methods abovementioned, though it uses PCA and multi-view combination yet.

Compared with cross-view hashing, MEH only needs a single information source as the training data, and obtains the diverse multi-view data by self-generating. Therefore, since MEH has a low requirement of dataset, it can adapt a more common scenario in real world.

Compared with other hashing methods with PCA, MEH does not use PCA to reduce the dimension of the original data, but to learn filters. Then, these filters are used to do convolution filtering on the original data. Therefore, the dimension of generated multi-view data is unaltered. There are two advantages about this:

First, bootstrap sampling idea samples a small fraction of the original training data several times, which is powerful to small data set, but weak to large-scale data set. By contrast, our PCA filtering method filters the whole training data set to generate new view training data each time, so there will be no degeneration when dealing with large-scale data set.

Second, as bootstrap methods, the performance of bagging is usually weaker than boosting for its random sampling, and boosting may lead to overfit. By contrast, our filtering method learns filters by PCA. For the eigenvectors of PCA are uncorrelated and ordered, there will be rigorous diversity and ordering of the generated multi-view training data, which is strictly in favor of ensemble learning and can avoid overfitting effectively.

3. Our method

Given an image dataset $\mathbf{O} = \{\mathbf{I}_i\}_{i=1}^{n}$, which is of n data points and $\mathbf{I}_i \in \Re^d$ is the i th data point. Hash function $h : \Re^d \to \{-1,1\}$ is treated as a mapping that projects a d-dimensional input onto a binary code. Moreover, $\mathbf{h}(\mathbf{I}) = \left[h_1(\mathbf{I}), h_2(\mathbf{I}), ..., h_r(\mathbf{I})\right]$ denotes a set of binary functions that projects a d-dimensional data point $\mathbf{I}$ I onto r-bit ($r \ll d$) binary codes while preserving the semantic similarity of data points. Accordingly, the hashing code for data point $\mathbf{I}$ is denoted as $\mathrm{b}(\mathbf{I}) \in \{-1,1\}^{r\times 1}$. Our goal is to learn a set of functions $\mathbf{H}(\mathbf{I}) = \left\{\mathbf{h}^i(\mathbf{I})\right\}_{i=1}^{K}$, where the hashing results of them can be merged

together to improve the discrimination on similarity, and both the learning processes of $\mathbf{h}^i(\mathbf{I})$ and the generating processes of their $\mathrm{b}^i(\mathbf{I})$ are mutual independent.

3.1. Multi-view generating

3.1.1. The requirements analysis of ensemble learning

Ensemble learning is the basis of our method. Each view in MEH corresponds an individual learner in ensemble learning. Therefore, the most important thing is to generate the multi-view data that in favor of ensemble learning. Moreover, the generating processes of views should be mutual independent, thus to fit parallel computation.

According to paper [12], the effect of ensemble learning can be expressed as (1):

$$E = \bar{E} - \bar{A} \tag{1}$$

Where E is the ensemble generalization error, $\bar{E}$ is the weighted average of the generalization error of the individual networks, $\bar{A}$ is the weighted average of the ambiguities that determined by the diversity of the individual learners.

Therefore, as shown in Figure 2, the better the diversity among individual learners is and the better the precision of each individual learner is, the better the effect of ensemble learning achieves.

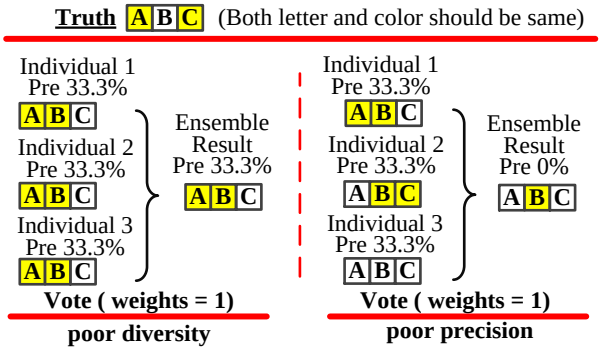


Figure 2. An example for illustrating the demands of ensemble learning, where "Pre" is short for "precision".

In fact, less correlation corresponds to more diversity. The output of individual learners will be uncorrelated, if the input of them are uncorrelated and the process on them are the same. Therefore, it is feasible to maximize the diversity by decorrelating the input of these individual learners.

In our method the multi-view data generated from the original image data set is the input of individual learner, so the multi-view data should have such properties:

1) Low correlation. The generated data of a view should be low correlation with others, to ensure the diversity of individual learners.

2) Powerful feature expression. The generated data of a view must represent a considerable characteristic of the original data, to ensure the precision of the individual learners after a series of effective processes.

According to the analysis abovementioned, we propose a novel method to generate multiview.

We filter the original image dataset by several different convolutional filters, which are learned from images by PCA. There are two advantages of this method [2,3]: First, the filtering result contains a considerable characteristic of the original image. Second, for the filters are orthogonal, the filtering results of an original image will be uncorrelated with each other absolutely.

3.1.2. Off-line filters learning

We download abundant image data from internet, and normalize them to size 32×32, and randomly sample m images to constitute a training dataset, denoted as $\{\tilde{\mathbf{I}}_i\}_{i=1}^{m}$. Then we take P_n patches of size $k_1\times k_2$ from each image $\tilde{\mathbf{I}}_i$ randomly, and vectorize them. We denote the patching sample set of image $\tilde{\mathbf{I}}_i$ as matrix $\mathbf{X}_i=[\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_{p_n}]\in\Re^{k_1k_2\times p_n}$, where $\mathbf{x}_i\in\Re^{k_1k_2\times1}$ is the i th patch of image $\tilde{\mathbf{I}}_i$. Then the patching sample set of dataset $\{\tilde{\mathbf{I}}_i\}_{i=1}^{m}$ can be denoted as matrix $\mathbf{X}=[\mathbf{X}_1,\mathbf{X}_2,...,\mathbf{X}_m]\in\Re^{k_1k_2\times p_nm}$. Then we use PCA to do eigendecomposition on matrix $\mathbf{X}$, to learn the convolutional filters.

PCA minimizes the reconstruction error within a family of orthogonal filters, and we can get (2):

$$\min_{\mathbf{V}\in\Re^{k_1k_2\times K}}\left\|\mathbf{X}-\mathbf{V}\mathbf{V}^{\mathrm{T}}\mathbf{X}\right\|_F^2,\ \text{s.t. }\mathbf{V}^{\mathrm{T}}\mathbf{V}=\mathbf{I}_K \tag{2}$$

Where K is the number of filters, and $\mathbf{I}_K$ is identity matrix of size $K\times K$. The solution $\mathbf{V}$ is known as the K principal eigenvectors of $\mathbf{X}\mathbf{X}^{\mathrm{T}}$, and one column of $\mathbf{V}$ is an eigenvector, we denote it as $\mathbf{v}$, $\mathbf{v}\in\Re^{k_1k_2\times1}$.

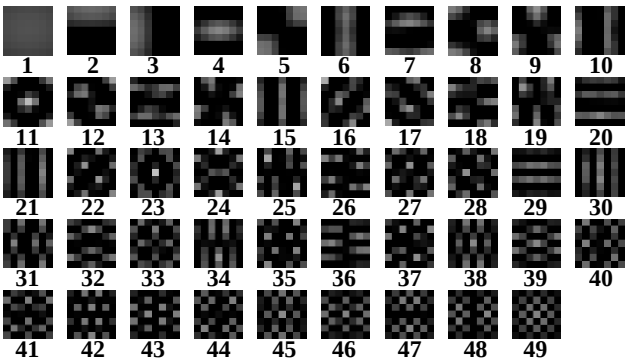


Figure 3. Convolution filters learned by PCA, where m = 3000, P_n = 20, k_1 = k_2 = 7, and K = 49. The filters are ranged according to the order of eigenvalues. For the eigenvectors of PCA are orthogonal, the filters are mutual independent with each other. Moreover, the generalization of these filters will be discussed in chapter 4.3

If we arrange $\mathbf{v}$ to a matrix $\mathbf{m}$, $\mathbf{m}\in\Re^{k_1\times k_2}$, $\mathbf{m}$ would denote a convolution filter learned offline. Consequently, according to $\mathbf{V}$, all the K convolution filters learned off-line can be denoted as a set $\{\mathbf{m}_k\}_{k=1}^{K}$. The typical filters of size 7×7 have been shown in Figure 3.

3.1.3. On-line multi-view data generating

We orderly use the convolution filters learned off-line to filter each image that given in training data or testing data. One filter corresponds to one view, thus we can generate multi-view data based on the original image data.

Assuming that total K view data generated from image $\mathbf{I}_i$ is denoted as $\mathbf{T}_i^1\sim\mathbf{T}_i^K$, we can get (3):

$$\mathbf{T}_i^k=\mathbf{I}_i*\mathbf{m}_k \tag{3}$$

Where $*$ denotes 2D convolution, and k denotes the sequence number of filter.

In this way, corresponding to the given image dataset $\mathbf{O}=\{\mathbf{I}_i\}_{i=1}^{n}$, we can obtain the multiview data $\{\mathbf{T}_i^1\}_{i=1}^{n}\sim\{\mathbf{T}_i^K\}_{i=1}^{n}$. As the discussion mentioned above, the correlations among $\mathbf{T}_i^1\sim\mathbf{T}_i^K$ are very low because of the orthogonality of the filters. Therefore, we have maximized the diversity of the input of individual learners, the generated multi-view data $\{\mathbf{T}_i^1\}_{i=1}^{n}\sim\{\mathbf{T}_i^K\}_{i=1}^{n}$ quite meet the requirements of ensemble learning. Figure 4 shows an instance of multi-view generating.

Moreover, the generating processes of different views are independent, so it is capable to compute in different units concurrently.

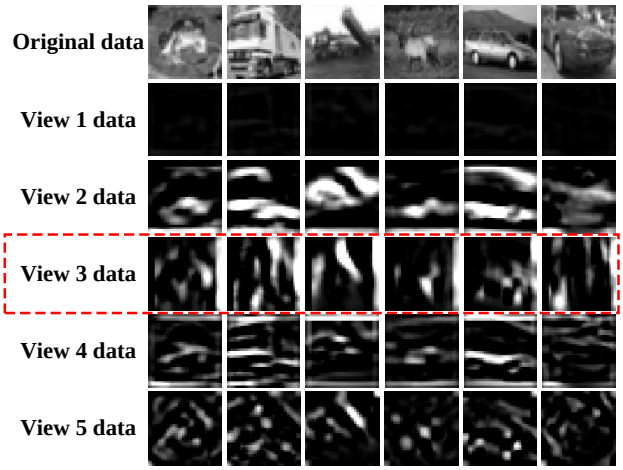


Figure 4. An instance of multi-view generating. In this instance, there are six original images and five views, the filters used to convolutional filtering are the first five that shown in Figure 3. One row is the data of one view, which data is generated from original images by the same filter (we have outlined the generated dataset of view 3 in red. Moreover, one column is the multiview data ref1ref1of one original image. It is observed that, the generated data of each view includes a considerable characteristic of the original image (e.g. borders, outlines, saliency regions of different directions and positions). Moreover, it is obvious that the characteristics represented in different views are quite different.

3.2. Multi-view merging

3.2.1. Individual learning

After generating multi-view data, the next on-line step is learning to hash in each view. As a view corresponds to an individual learner of the ensemble learning system, it has to work out a result about similarity predication, to prepare for the next step of voting. The similarity predication of hashing is achieved by calculating the hamming distance of data, so we need to learn the hash function and hash code of each view based on the generated multi-view data.

We use a traditional unsupervised hashing method as the homogeneous operator of multiview. Thus, we can purchase the hash function and hash code of each view by the operator and the generated data of this view. Here, we use formulation (4) to express the learning process:

$$(\mathbf{h}^k, \mathbf{B}^k) = f_{\mathrm{M}}\left(\{\mathbf{T}_i^k\}_{i=1}^n\right) \tag{4}$$

Where $f_{\mathrm{M}}(*)$ is the learning function of the operator, for example, f_{ITQ} denotes the learning process of ITQ. Superscript k is the sequence number of view. $\{\mathbf{T}_i^k\}_{i=1}^n$ denotes the generated data of view k. $\mathbf{h}^k$ and $\mathbf{B}^k$ are respectively the learned hash function and the learned hash codes corresponding to dataset $\{\mathbf{T}_i^k\}_{i=1}^n$.

Theoretically, any traditional unsupervised hashing method can be used as the operator in MEH. Moreover, the learning processes of different views are independent, so they are parallelizable.

3.2.2. Results merging

The last step of MEH is merging the hashing results of multi-view. Judging how the similarity of samples is the essence of ANN search problem, so as image retrieval. According to the preceding work, we have purchased several groups of hash codes that correspond to multi-view. Each view can work out a similarity predication of any two original image data by calculating the Hamming distance between their hash codes. Therefore, we can purchase final similarity estimation of two images by merging their multi-view similarity predication results.

We use voting to realize merging, which is a common approach in ensemble learning. This process can be expressed as:

$$S(\mathbf{I}_x, \mathbf{I}_y) = e^{-\frac{1}{K}\sum_{k=1}^{K}\omega_k D_k(\mathbf{I}_x, \mathbf{I}_y)} \tag{5}$$

Where $S(\mathbf{I}_x, \mathbf{I}_y)$ denotes the final similarity estimation between image $\mathbf{I}_x$ and $\mathbf{I}_y$, $D_k(\mathbf{I}_x, \mathbf{I}_y)$ denotes the Hamming distance between their hash codes in view k , ω_k is the weight of view k , and K is the amount of views.

Furthermore, the retrieval result for a query image $\mathbf{I}$ would be obtained by ranging its S in order.

4. Experiments

4.1. Multi-view merging

We evaluated our method on two widely used datasets, CIFAR-10 and LabelMe.

The first dataset CIFAR-10 is consists of 60; 000 color images, which are manually labeled into 10 classes. The second dataset is a 22K LabelMe used in [23] and [26], it contains 22; 019 images, which are sampled from the large LabelMe dataset.

Moreover, we scaled the images of these two datasets to size 32×32 , and converted them to gray images. Thus, each image can be represented by a 1024-dimensional grayscale descriptor, we use that as the input of all methods.

4.2. Protocols and baseline methods

We used Euclidean neighbors in the original space as the ground truth. Then we computed the precision-recall curve on code sizes of 16, 32, 64, 128, and 256-bit. Specifically, a threshold of average distance to the 50th nearest neighbor was used to determine whether a point returned for a given query is a true positive.

We sampled 2000 points of the dataset randomly as the queries, and another 20000 points of the rest as the training data set. All experimental results were averaged over five random training/test partitions.

Although much work has been done to hashing, they are mostly supervised or based on multiple information sources. In contrast, our method is essentially an unsupervised method based on single information source. Therefore, for fair comparison, we selected several representative unsupervised methods for evaluation, which are also based on single information source and the program codes can be found from internet. They are LSH [1], ITQ [24], SITQ [7], BITQ-rand [6], BITQ-opt [6], and BPCAH [14]. Moreover, we chose LSH and ITQ as the operator of our method, and denoted them as Ensemble-LSH and Ensemble-ITQ.

4.3. Parameters setting about MEH

(1) Filters

The filters used to generate multi-view are the most important parameters in our method. As discussed in the front of this paper, an effective hashing method should be efficient and scalable. Therefore, for MEH, we must insure the filters learned off-line are of well generalization. Effective filters should be capable to deal with new image without updating or learning repeatedly. Therefore, we implemented sufficient experiments to prove the generalization of the filters learned off-line.

We downloaded about 100,000 images from Internet, and sampled portions randomly to constitute several data sets of different scale. Then we learned filters from these data sets by PCA respectively. Treating the filters that learned from data set of scale 30000 as the baseline, thus to calculate the variance between it and the filters of other scale data sets. We repeated this process several times and calculated the average value of variance, which can be expressed as (6):

$$\Delta V_i = \frac{\sum_{j=1}^{t} \left\| \mathbf{V}_i^{\,j} - \mathbf{V}_{30000}^{\,j} \right\|_F^2}{t} \tag{6}$$

Where $\mathbf{V}_i^{\,j}$ denotes the filters learned at the *j*th time from the dataset of scale *i*, and *t* demotes the repeating amount. The result is shown in Figure 5.

We found that, with the scale of data set increasing, the variance between V30000 and other scales tended to be stable in a set patch size. This indicates that the filters learned from different image data sets are trend to invariable while the scale of the training data is large enough. As shown in Figure 5, the variance between V_{5000} and V_{30000} is less than 4%, and the variance between V_{20000} and V_{30000} is even less than 1%, in patch size of 3×3, 5×5, 7×7, and 9×9. This indicates that the filters learned from 20000 images are able to fit another 30000 images. Therefore, we are convinced that the filters learned from 30,000 images are able to deal large-scale dataset effectively. Consequently, the filters learned by our method are of well generalization.

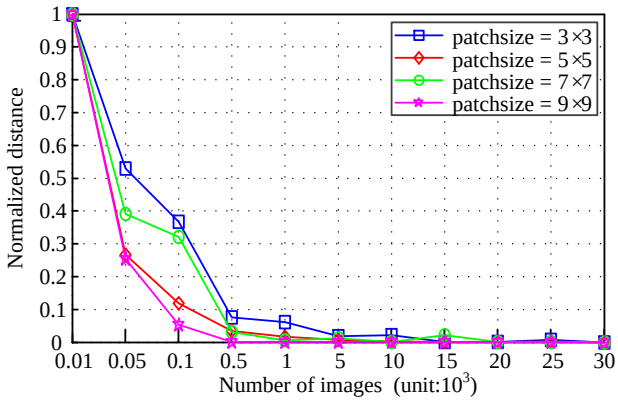


Figure 5. Comparison of filters that learned from datasets of different scales in several patch sizes.

According to the discussion above, we chose the filters learned from a dataset of scale 30,000 to generate the multi-view in our method, where patch size is 7×7 and patch amount of each image is 20.

(2) Multi-view number

The amount of filters used is another important parameter in our method. According to PCA, the upper limit of filter amount is determined by the patch size (e.g. 49 corresponds to patch size 7×7). However, how many filters used to ensemble learning is still a question, whether it is the more the better?

Therefore, we implemented sufficient experiments to analyze the effects of different filter amounts. We compared the retrieval precision of merging different view amounts. The result is shown in Figure 6.

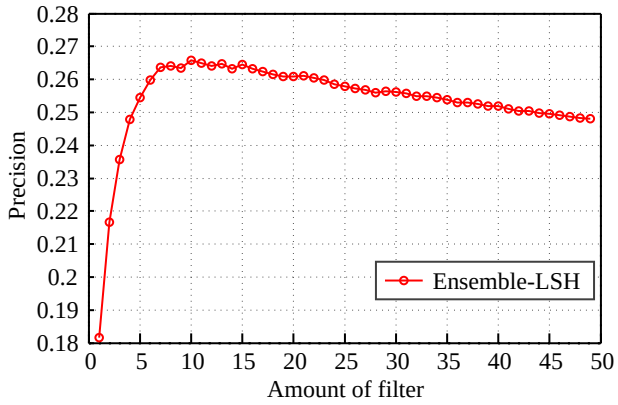


Figure 6. Comparison of retrieval precision of merging different view amounts (patch size is 7×7, hash code size is 32). The filters are used in order of the PCA eigenvalues.

We found that, more filters merged did not cause better performance achieved. With the increase of merged views, the performance increased rapidly at the beginning, and reached a ceiling at abref1ref1ref1out the amount $8\sim14$, then it declined slowly.

It is probable that, according to the requirements of ensemble learning, both the diversity and the precision of learners influence the effect of ensemble learning, as the example shown in Figure 2. Moreover, the latter views may be of restricted precision. If we merged them, it is equivalent to use a flawed individual learner to work out a judgment, which may be wrong and make the final recognition worse.

Therefore, we implemented future experiment to analyze this phenomenon. We used only one filter to generate one view each time, and then compared the retrieval precision of them. The result is shown in Figure 7.

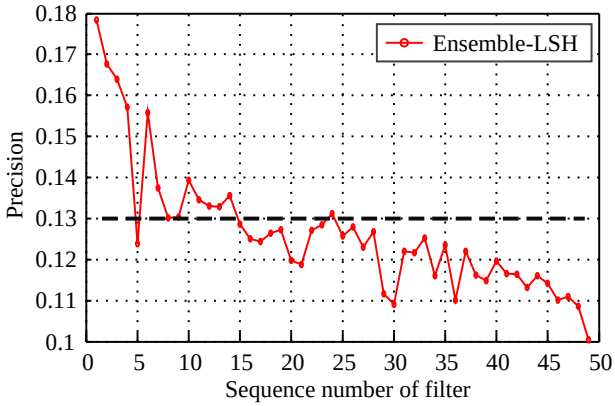


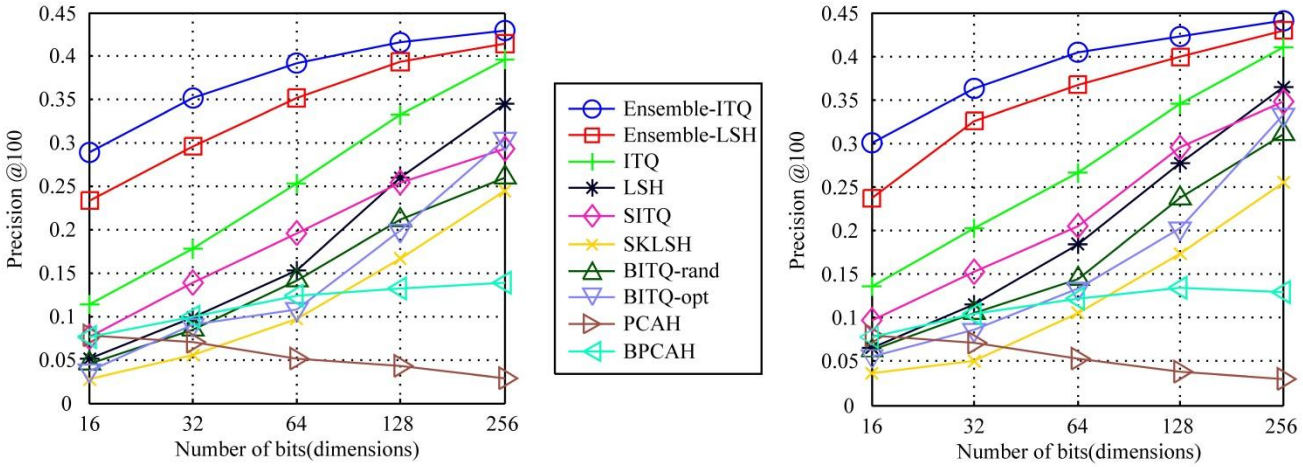
Figure 7. Retrieval precision of each view. The parameters are same as that of Figure 6.

We found that, the retrieval performances of different filters were quite different, and it confirmed the conjecture above. As shown in Figure 7, the overall trend is decreasing corresponding to the order of PCA eigenvalues, though there is little fluctuation in adjacent filters. The performances of filters after 15 are mainly unsatisfactory; they are all under the baseline marked in bold dotted line. This is mainly because most of the information is typically contained in the top eigenvectors of PCA while the remainders are usually less informative or even noisy. Therefore, the convolutional filtering results of latter eigenvectors contain little energy of the original image. Merging these views is equal to use a severely noise polluted image to recognize, which would certainly bring a worse recognition.

Moreover, the fluctuation in adjacent filters is strong probably because the filters learned from a large-scale training dataset may not match the small testing dataset absolutely. However, once again, the overall trend is decreasing corresponding to the order of PCA eigenvalues. Therefore, it is feasible to generate multi-view data orderly, according to the sequence of the filter.

4.4. Results on CIFAR dataset and LabelMe dataset

We compared the retrieval performance of methods on several code sizes. The precision comparison result is shown in Figure 8. Figure 8(a) is the result of CIFAR-10, and Figure 8(b) is the result of LabelMe.



(a) CIFAR-10 (b)LabelMe

Figure 8. Comparative evaluation on dataset CIFAR-10 and dataset LabelMe.

We found that, our method MEH, either Ensemble-LSH or Ensemble-ITQ, has outperformed other compared hashing methods, both single-view and multi-view that based on single information resource. Specially, they obviously outperform the original hashing method which is used as the operator. The performance even boosts over 100% on code size of 16-bit. A 32-bit Ensemble-LSH has outperformed the performance of a 128-bit LSH. Moreover, even if the operator is a weak hashing method, the perfor mance can outperform other methods yet, like that shown in Ensemble-LSH. However, it is obvious that,

MEH based on effective original hashing method performances better than that based on a weak one, for example, Ensemble-ITQ performances better than Ensemble-LSH.

Figure 9 shows the complete recall-precision curves corresponding to Figure 8, where (a) ~ (d) are the curves on dataset CIFAR-10 and (e) ~ (h) are the curves on dataset LableMe. We found that, these recall-precision curves confirmed the results of Figure 8; our proposed MEH achieved a performance which was far better than other unsupervised methods, whether it is the original hashing method used as operator or the state-of-the-art methods based on single information source.

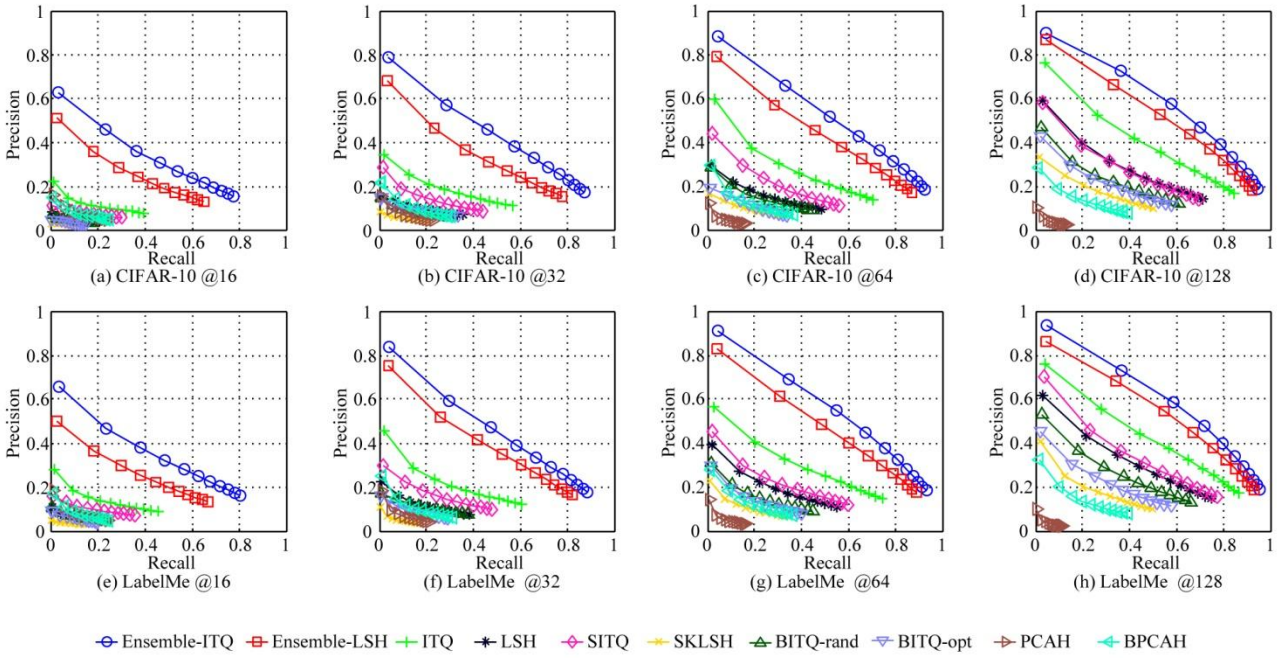


Figure 9. The complete recall-precision curves of comparison with some state-of-the-art unsupervised methods on dataset CIFAR-10 and dataset LabelMe. Refer to Figure 8.

Moreover, it is apparent in Figure 8 that, contrast to the performance of the original methods, the performance boosting degree of MEH seems to decline while the code size increases, like Ensemble-LSH to LSH, Ensemble-ITQ to ITQ. It is highly likely that the similarity discrimination of traditional hashing methods have been much stronger at a big code size (the size of binary hash code has been 256 bits while the dimension of original data is only 1024), so the boosting of multiview ensemble learning seems no longer obvious. However, once again, our MEH still performs best in all evaluating methods on each code size, and provides much more gain on small code size especially

There is no practical significance to analyzing the algorithm complexity based on real time because the configuration of computers and the algorithm optimizing degree of procedures are different. Therefore, we only analyze the time complexity in theory.

In MEH, the filters used for multi-view generating are learned off-line, so the main calculation is brought by the three on-line steps, multiview data generating, individual learners learning, and multi-view results merging.

For the data generating and the hashing learning of multi-view are independent, the calculation and storage requirement of these multiple views can be assigned to multiple distributed compute units, and achieved by parallel computing. Therefore, the time cost of MEH can be expressed by (7):

$$T_{en} = T_{sv} + T_m = (T_c + T_{om}) + T_m \tag{7}$$

Where T_{en} denotes the total time cost, T_{sv} denotes the cost of a single view, T_m denotes the cost of multi-view results merging, T_c denotes the cost of convolutional filtering, and T_{om} denotes the cost of hash learning of the original method that used as the operator.

Actually, $T_m \ll T_c \ll T_{om}$. Therefore, the time complexity of MEH is approximate to the original method that used as operator, that is $O_{en} \approx O_{om}$.

5. Conclusion

Our proposed MEH is a novel unsupervised hashing method for large-scale image retrieval. The approach of our method is realized by multiview self-generating and ensemble learning, it can improve the retrieval performance remarkably on the scenarios of single information source. As our best knowledge, this is the first work to solve image retrieval problem by ensemble learning. For the consideration of distributed computation and storage, the architecture of MEH is almost parallel on the whole process, so it maintains an approximate complexity to the original hashing method that used as operator in individual learners. Sufficient experiments have been done to demonstrate the advantages of MEH than other unsupervised hashing methods. In conclusion, MEH can support image retrieval problem on large-scale dataset better than ever. Moreover, theoretically, the approach of MEH may also extend to the field of supervised field by inserting grouped labels.

Acknowledgements

This research was supported by the national natural science fund (No.61472442, No.61502522, No.61502523).

References

1. A. Andoni, P. Indyk. “Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions,” *Foundations of Computer Science Annual Symposium on*, vol. 51, no. 1, pp. 459–468, 2006
2. P. Baldi, K. Hornik. “Neural Networks and Principal Component Analysis: Learning from Examples without Local Minima,” *Neural Networks*, vol. 2, no. 1, pp. 53–58, 1989
3. T. H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, Y. Ma. “PCANET: A Simple Deep Learning Baseline for Image Classification,” *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society*, vol. 24, no. 12, pp. 5017–5032, 2015
4. T. G. Dietterich. “Machine Learning Research: Four Current Directions,” *AI Magazine*, vol. 18, no. 4, pp. 97–136, 2000
5. G. Ding, Y. Guo, J. Zhou. “Collective Matrix Factorization Hashing for Multimodal Data,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2083–2090, 2014
6. Y. Gong, S. Kumar, H. A. Rowley, S. Lazebnik. “Learning Binary Codes for High-dimensional Data Using Bilinear Projections,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 484–491, 2013
7. Y. Gong, S. Kumar, V. Verma, S. Lazebnik. “Angular Quantization-based Binary Codes for Fast Similarity Search,” *Advances in Neural Information Processing Systems*, 2012
8. J. E. Gonzalez, Y. Low, H. Gu, D. Bickson, C. Guestrin. “Powergraph: Distributed Graph-parallel Computation on Natural Graphs,” *In Usenix Conference on Operating Systems Design and Implementation*, pp. 17–30, 2012
9. L. K. Hansen, P. Salamon. “Neural Network Ensembles,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990
10. A. Kleiner, A. Talwalkar, P. Sarkar, M. Jordan. “The Big Data Bootstrap,” *Computer Science*, pp. 1759–1766, 2012
11. W. Kong, W. J. Li. “Isotropic Hashing,” *Advances in Neural Information Processing Systems*, vol. 2, no. 1, pp. 1646–1654, 2012
12. A. Krogh, J. Vedelsby. “Neural Network Ensembles, Cross Validation, and Active Learning,” *Advances in Neural Information Processing Systems*, vol. 7, no. 10, pp. 231–238, 1995
13. Z. Kuang, J. Sun, K. Wong. “Learning Regularized, Query-dependent Bilinear Similarities for Large Scale Image Retrieval,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 413–420, 2013
14. C. Leng, J. Cheng, T. Yuan, X. Bai, H. Lu. “Learning Binary Codes with Bagging PCA,” *Springer Berlin Heidelberg*, 2014.
15. W. J. Li, Z. Z. Zhou. “Learning to Hash for Big Data: Current Status and Future Trends,” *Chin. Sci. Bull*, vol. 60, no. 5-6, pp. 485–490, 2015
16. R. S. Lin, D. A. Ross, J. Yagnik. “Spec Hashing: Similarity Preserving Algorithm for Entropybased Coding,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 848–854, 2010
17. W. Liu, R. Ji. “Supervised Hashing with Kernels,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2074–2081, 2012
18. X. Liu, J. He, C. Deng, B. Lang. “Collaborative Hashing,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2147–2154, 2014
19. X. Liu, J. He, B. Lang. “Hash Bit Selection: A Unified Solution for Selection Problems in Hashing,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1570–1577, 2013
20. X. Liu, J. He, D. Liu, B. Lang. “Compact Kernel Hashing with Multiple Features,” *In ACM International Conference on Multimedia*, pp. 881–884, 2012
21. C. Ma, C. Liu. “Two Dimensional Hashing for Visual Tracking,” *Computer Vision and Image Understanding*, vol. 135, no. C, pp. 83–94, 2015
22. Y. Mu, J. Shen, S. Yan. “Weakly-supervised Hashing in Kernel Space,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3344–3351, 2010

23. M. Norouzi. “Minimal Loss Hashing for Compact Binary Codes,” *In International Conference on Machine Learning*, pp. 353-360, June, 2011
24. Y. Gong, S. Lazebnik, A. Gordo, F. Perronnin. “Iterative Quantization: a Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval,” *IEEE TPAMI*, vol. 35, no. 11, pp. 2916–2929, 2013
25. D. B. Skillicorn, D. Talia. “Models and Languages for Parallel Computation,” *ACM Computing Surveys*, vol. 30, no. 2, pp. 123–169, 1998
26. A. Torralba, R. Fergus, Y. Weiss. “Small Codes and Large Image Databases for Recognition,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008
27. Q. Wang, D. Zhang, L. Si. “Weighted Hashing for Fast Large Scale Similarity Search,” *In ACM International Conference on Conference on Information and Knowledge Management*, pp. 1185–1188, 2013
28. Y. Weiss, A. Torralba, R. Fergus. “Spectral hashing,” *In Conference on Neural Information Processing Systems*, pp. 1753–1760, December, 2008
29. D. Zhang, F. Wang, L. Si. “Composite Hashing with Multiple Information Sources,” *In Proceeding of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 225–234, July, 2011
30. F. Zhao, Y. Huang, L. Wang, T. Tan. “Deep Semantic Ranking Based Hashing for Multi-label Image Retrieval,” *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1556–1564, 2015

Huanyu Li graduated from Air Force Engineering University, for the degree of Bachelor, Master and Ph. D. Now he is a lecturer of Air Force Engineering University. His current research interests include machine learning and pattern recognition.

Yunqiang Li is a doctor student of Air Force Engineering University. His research interest is image retrieval.

Yufei Zha is an associate professor of Air Force Engineering University. His research interests include machine learning and computer vision.