Multimed Tools Appl (2017) 76:21281–21302
DOI 10.1007/s11042-016-4057-z

Hierarchical deep semantic hashing for fast image retrieval

Xinyu Ou[1,2,3] · Hefei Ling[1] · Si Liu[3] · Jie Lei[1]

Received: 29 December 2015 / Revised: 3 August 2016 / Accepted: 10 October 2016/
Published online: 25 October 2016
© Springer Science+Business Media New York 2016

Abstract Content-Based large-scale image retrieval has recently attracted considerable attention because of the explosive increase of online images. Inspired by recent advances in convolutional neural networks, we propose a hierarchical deep semantic method for learning similarity function that solves the problems of precision and speed of retrieval in the setting of large-scale environments. The distinctive contribution of our work is a novel approach that can utilize previous knowledge of the semantic hierarchy. When semantic information and a related hierarchy structure are available, significant improvements can be attained. Exploiting hierarchical relationships is the most important thing for large-scale issues. Binary code can be learned from deep neural network for representing the latent concepts that dominate the semantic labels. Different from other supervised methods that require learning an explicit hashing function to map the binary code features from the images, our method learns Hierarchical Deep Semantic Hashing code (HDSH-code) and image representations in an implicit manner, making it suitable for large-scale datasets. An additional contribution is a novel hashing scheme (generated at the same time with seman-

✉ Hefei Ling
lhefei@hust.edu.cn

Xinyu Ou
ouxinyu@hust.edu.cn

Si Liu
liusi@iie.ac.cn

Jie Lei
lei_jie@hust.edu.cn

[1] School of Computer Science and Technology, Huazhong University of Science and Tecnnology, Wuhan, China

[2] Yunnan Province Cadres Online Learning College, Yunnan Open University, Kunming, China

[3] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

tic information) that is able to reduce the computational cost of retrieval. Comprehensive experiments were conducted on *Holidays*, *Oxford5k/105k*, *Caltech256* retrieval datasets, our HDSH performs competitively even when the convolutional neural network has been pre-trained for a surrogate unrelated task. We further demonstrates its efficacy and scalability on a large-scale dataset *Imagenet* with millions of images. With deep hierarchical semantic hashing, we report retrieval times are 0.15ms and 53.92ms on $Holidays$ and $Imagenet$ dataset, respectively.

Keywords Hierarchical deep semantic hashing · Similarity search · Large-scale image retrieval · Convolutional neural network

1 Introduction

Since the 2012 ImageNet competition [33] winning entry by Krizhevsky et al. [17], their network AlexNet has been successfully applied to a larger variety of computer vision tasks, for example to object-detection [7], segmentation [22], human pose estimation [35], video classification [15], object tracking [36]. This paper tackles the problem of similarity image retrieval. The objective is: given a query, find its most similar from a large assembly. As shown in Fig. 1, results depicted hierarchical relationships can dramatically enhance the discriminant ability. The integration of the feature presentation with semantic and hashing hierarchical relationships is more important for increasing retrieval accuracy as images grow larger.

In content-based image retrieval task, both image representation and computational cost play a key role. Image representation is the engine of image retrieval, and it has been a

Fig. 1 Images retrieved by utilizing hierarchy with 128-bit hash code versus those without considering hierarchy. The top query image is selected from the validation dataset of *Holiday* dataset, while the lower one is selected from the *Imagenet* dataset. Accordingly, the retrieval is performed on the training dataset of *holiday* and *Imageset* dataset, respectively. The query results show category hierarchy is good for preserving the semantics of search results, which make the images more similar to the query image

driving motivation for research in computer vision for many years. For much of the past decade, feature revolution was established through the introduction of SIFT [23], HOG [4], LBP, GIST [27], or Bag-of-Features (BoF). Nevertheless, learning mapping from these representations to binary code is inadequate for dealing with relatively complex semantic structure, because the representation ability of handcrafted features is still too low-level to capture the semantic information from the image. In fact, how to choose suitable features to represent objects, and use these features (or combination of features) to retrieve images is extremely difficult. Thus more effective semantic feature representation is also desirable.

Additionally, because of the explosive increase of online images, rapid similarity search is increasingly critical for large-scale image retrieval, therefore, search in a large database becomes an emerging need. Many studies aim at answering the question that how to efficiently retrieve the relevant image from the large-scale database. Owing to the high computational overhead, traditional linear search is facing the problem of inadequate performance. Instead of linear search, an alternative strategy is to take advantage of the technique of hashing based method [21, 25, 39] or Approximate Nearest Neighbor (ANN) speedup. These methods map the high-dimensional feature to a lower dimensional space, and then generate the binary hash code. Benefiting from the compact binary code, fast image search can be measured via Hamming distance and binary pattern matching, which significantly reduces the computational overhead and further optimizes the efficiency of the search. Other researchers [19] attach to the pair-wised method that uses the pairwise geometric relations between correspondence and propose a strategy to incorporate spatial relations to reduce the computational cost. However, these methods are requested to construct the correlation matrix and generate the codes which maybe a bottleneck in large-scale dataset.

Inspired by the advancement of end-to-end learning methods such as convolution neural network, we think a question that can we make use of CNN to produce compact binary code directly? To address this question, and handle such large scale data, computational efficiency and scalability problems, we propose a hierarchical deep semantic hashing method, which utilizes CNN model to learn semantic information and binary representation simultaneously. It demonstrates how to effectively integrate human knowledge in the form of a hierarchical structure defined on semantic information of images. As an example, given an image containing a semantic object “Monkey”, a predefined hierarchy might let us know that an image containing a “Lemur” would be more similar to one containing a “house” or a “person”. At this point, we can discard the images which belong to the irrelevant semantic categories.

Once semantic-level similarity is determined, the next challenge is the efficient retrieval. This paper presents a novel hashing learning strategy from CNN with semantic information. Without any hashing index(such as inverted file), retrieval of similar images in the Euclidean distance can be performed in 50 milliseconds per query on *Imagenet* dataset with competitive accuracy. In fact, using index to retrieve further decreasing retrieval time, but this is beyond the scope of this study.

The specific contributions of our work are as following:

- We present a simple but efficient supervised learning framework for rapid image retrieval.
- With slender modifications to the vanilla network model, our ConvNet learns probability-based semantic-level feature and hashing-level feature for image representation simultaneously.
- Exploiting the semantic hierarchy for similar image retrieval, can reduce search space dramatically for matching similar image from a large-scale datasets.

– Moreover, we use a simple but novel way to deal with the problems of samples few and imbalance.
– Our approach of learning hierarchical binary hash code is very stable. Compared with state-of-the-art methods, the performance attenuation is not obvious when reducing the feature dimension.

This paper is organized as follows: Section 2 briefly reviews previous related works. And we elaborate on details of our hierarchical similarity methodology in Section 3. Finally, Section 4 demonstrates the experimental results. Conclusions are provided in Section 5.

2 Related work

Recently, as the ever-growing large-scale web data make information retrieval and other problems more challenging, hashing becomes a popular solution [37, 39]. The shortly binary code makes retrieval efficient both on storage and computation. In many cases, search in millions of data will only consume constant time via tens-of-bit representations mapped from the hash code.

In the early stage of hash, methods were mostly data-independent. For example, Locality Sensitive Hashing (LSH) [6] uses random projection to construct hash function. The property of LSH, that samples within short Hamming distance in hash space are most likely to near in their source space, makes it very attractive. But this metrics is asymptotically subjected to the increasing code length. Thus, to achieve high precision, LSH-related methods require large hash tables. And LSH works only with theoretic guarantees for some metric spaces, such as Euclidean distance. Learning-based Hashing unlike data-independent hashing, learning-based methods attempt to capture the inherent distribution of the task domain by learning. Unsupervised methods use only unlabeled data as training set, such as KLSH [18], Semantic Hashing [16, 34] and Spectral Hashing [38]. Semantic Hashing uses stacked RBMs (Restricted Boltzmann Machines) to learn binary hash code from raw inputs. After pre-trained layer by layer, the RBMs are unrolled and refined as a deep autoencoder. Spectral Hashing defines similarity on the feature space and attempts to minimize the weighted average Hamming distance between similar samples. The majority of these methods explicitly learn a hashing function to map the low feature to compact code. Another area is learning hash code in the implicit way [20, 26, 41]. Zhao et al. [41] extract the mid-level feature of the neural network as the hash code. Similarly, our proposed method also uses CNN to generate feature, but we do not need to explicitly learn a hashing function, the hash code is implicitly generated by feed-forward of CNN, this avoids the complexity of designing a hashing function.

Beside the research track of hashing, image representations also play an essential role in CBIR. Convolutional neural networks [17] have become very popular in recent years for many tasks in computer vision. Different from various traditional methods, CNN is a typical supervised learning method, it needs a large number labeled images to optimize model and learn the parameters. This is a considerable challenge. Fortunately, Russakovsky et al. [33] provide such a large-scale labeled dataset *ImageNet*. It can be utilized to achieve feature learning from the original images. DCNN-based feature has been applied on the task of image retrieval recently [2, 8, 20, 24, 30, 39, 41]. Babenko et al. [2] focus on exploring the feature of different layers, and improve the retrieval performance with dimensional reduction. Xia et al. [39] proposed a two-stage framework to accurately preserve the semantic similarity of image pairs, which demonstrates state-of-the-art retrieval performance.

However, in preprocessing stage, a matrix-decomposition algorithm is employed to learn the representation for data. It is unfavorable for the case when the data size is large (e.g., *imagenet* dataset in this paper) because it consumes both computational time and considerable storage. Zhao et al. [41] proposed employing multilevel semantic ranking supervision to learn deep hash function based on CNN. However, feature extraction directly from the compact hash code maybe increases the semantic information loss.

In contrast, we introduce a simple yet effective deep learning approach to learn an effective hash code from CNN directly, and it achieves more competitive results on the *Holidays* and *Oxford5k* image retrieval benchmark datasets. We further apply our approach to the larger *Imagenet*, *Oxford105k* and *Caltech256* datasets to demonstrate the scalability of our approach. In the next section, we will introduce our hierarchical deep semantic hashing method.

3 Exploiting hierarchy for retrieval

3.1 Similarity strategy

The objective of similarity retrieval is trying to find out the most similar images from an image gallery with the greatest similarity to a given image. But what is the most similar? In order to answer this question and exploit hierarchical knowledge in retrieval, we consider two subprocesses to evaluate the *similarity* between two images. One is how to express the data effectively. The other one is how to compute similarity quickly. Generally, a hash function $h(\cdot) : \mathbb{R}^D \rightarrow \{0, 1\}$ is treated as a mapping that maps a D-dimensional input onto a compact binary code. Supposed $I = \{(x_n, c_n)\}_{n=1}^N$ is a set of images and their labels, where each image $x_n \in \mathbb{R}^D$ regarding a subset of semantic labels $L = \{1, \ldots, C\}$. Our goal is to learn a hash function $h(\cdot)$ to map the image x_n to a binary hash codes $h(x_n)$ while preserving its semantic c. We hope the D can be instead of the new pairwise data $H = \{h(x_n), c)\}_{n=1}^N$ to express the images and their labels without any semantic loss. In previous works, learning a similarity function which maps low-level image presentation to a similarity value is the main objective of retrieval. Given a pair of images a and b, we can denote their low level presentation as $f_l(\cdot)$, then we can define their similarity as $sim(a, b) = Sim(f_l(a), f_l(b))$.

As illustrated in Fig. 2, compared to previous work, our approach first obtains the mid-level features $f(a)$ and $f(b)$ through feed-forward based on raw images a and b instead the low-level presentation $f_l(a)$ and $f_l(b)$, we use the mid-level features to estimate the probabilities $p(a) = (L(a)|f(a))$ and $p(b) = (L(b)|f(b))$ of semantic labels, then to compute the hash codes $h(a) = h(f(a))$ and $h(b) = h(f(b))$. Furthermore, we use the semantic probabilities $p(a)$, $p(b)$ and the hash codes $h(a)$, $h(b)$ to calculate the hierarchical similarity step by step. Now, we can obtain the similarity function: $Sim(a, b) = (p(a), h(a))\boldsymbol{S}(p(b), h(b))$. Here, $\boldsymbol{S}$ is a hierarchical correlation matrix, which can be achieved by two stages: semantic-level similarity and hashing-level similarity. This similarity function is not merely allowed utilizing hierarchical information to enhance accuracy, but also speeded up the similarity computing on the whole database.

It is noteworthy that we do not have explicitly to design and study a hashing function. It is implemented by deep neural network inference and a hierarchical fusion strategy, described in following section.

Probability-based semantic-level similarity The key to our method is to take advantage of semantic knowledge to simplify the computing, then use hash feature to compute

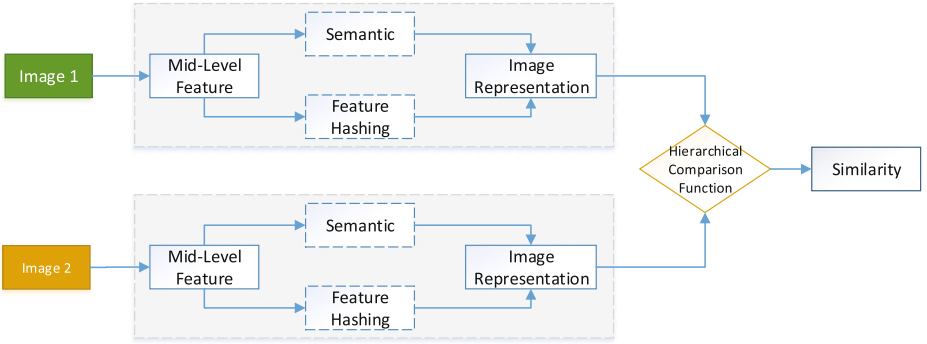


Fig. 2 Similarity evaluation between two images. Logically, our approach learns image representation with two parts, an probability estimation for matching semantic labels, and a hash function for mapping feature. The final similarity is computed by a hierarchical comparison function

similarity for retrieval. An easy method is by considering a label-based semantic-level (non-probabilistic) version of such a similarity, and describe an image a by a set of binary semantic information $\{1, \ldots, L\}$. For example, we can assign a label to an image, and calling it is a "house" or a "person". Nevertheless, measuring the similarity by semantic categories is a challenging problem in retrieval. On the one hand, natural semantics always categories overlap and inherently ambiguous, using hard-categories to recognize object always failure. On the other hand, perfect classification of semantic is unrealistic. For example, most of the time, we think "Lemur catta" and "Eulemur coronatus" are similar, it means both of them belong to "Lemur".

In order to solve these problems and improve performance, we proposed a probabilistic version similarity to instead of using simple label-based similarity. Given the semantic labels $L = \{1, \ldots, C\}$, the similarity between the two images a and b can be measured as how much they match. Let $\delta_i(I) \in \{0, 1\}$ be the indicator function of image I has semantic i. Similarly, we can use a probability $Y_i^C = P(\delta_i(a) = 1|a)$ to indicate the possibility of image a has semantic i. Obviously, the index of $i = max(Y_i^C)$ is the semantic label of image a. In this work, the probability Y_i^C can be obtained by CNN inference, and its value is equal to the input of the *Softmax* classifier. Follow the research of [3], we use the relevant information of query-image to define the similarity of the probabilistic version as follows.

For each $i \in L$, let I_i^+ denote the set of images that are relevant to the semantic i, and let I_i^- denote the set of irrelevant. The semantic-image relevance is defined by the matrix $\mathbf{R}_{IL} : I \times L \to \mathbb{R}^+$, and obeys $\mathbf{R}_{IL}(i, I_i^+ > 0)$ and $\mathbf{R}_{IL}(i, I_i^- = 0)$ for all $i \in L$, $I_i^+ \in I_i$, $I_i^- \in I_i$. In order to compute the image-image correlation matrix $\mathbf{R}_{II} : I \times I \to \mathbb{R}^+$, we treated images as being conditionally independent given image a, b and the semantic i, $P(I(a), I(b)|i) = P(I(a)|i)P(I(b)|i)$, and computed the joint image-image probability as a relevance measure

$$\begin{aligned} P(I(a), I(b)) &= \sum_{i \in L} P(I(a), I(b)|i) P(i) \\ &= \sum_{i \in L} P(I(a)|i) P(I(b)|i) P(i) \end{aligned} \tag{1}$$

To improve scalability, we considered two images to be related only if their joint distribution exceeded a cutoff threshold t.

$$\mathbf{R}_{II}(I(a), I(b)) = [P(I(a), I(b))]_t \tag{2}$$

That is to say,

$$x = \begin{cases} [x]_t, & x > t \\ 0, & otherwise \end{cases} \tag{3}$$

where $x = P(I(a), I(b))$.

Built on the above discussion, we can know the correlation matrix is a diagonal matrix, due to the image correlation only occurring when they both have semantic i. Moreover, most of the semantic probability is very low, hence after limited by cutoff threshold t, most of the probability forced to 0. Therefore, this correlation matrix is quite sparse, which makes the probability-based semantic-level similarity can filter a large number of irrelevant images. We take many experiments to select the threshold t, described in Section 4.4.1.

Hashing-level similarity In this section, we discuss hashing-level similarity. Given an image I, we first extract the output of the fully-connected layers as the image representation which is denoted by a D-dimensional feature vector $g(I)$, where $g(\cdot)$ is the convolution transformation over all of the previous layers. Then one can obtain a q-bit binary code by a simple hashing function $h(\cdot)$. For each bit $i = 1, \ldots, q$, we output the binary hash code of H by

$$H = h(x) = \begin{cases} 1, & f(x_i) - Avg_i^q(f(x_i)) > 0 \\ 0, & f(x_i) - Avg_i^q(f(x_i)) < 0 \end{cases}, \tag{4}$$

where $x = g(I)$ is the CNN feature and $x_i (i = 1, \ldots, q)$ is the activation value of the i-th neuron unit. $f(x)$ is a *Sigmoid* function defined by $sigmoid(v)=\frac{1}{1+e^{(-v)}}$, and $Avg(\boldsymbol{u})$ is an average function calculated on vector $\boldsymbol{u}$. Here the function *Sigmoid* is used to normalize the feature to [0, 1].

Let $I = \{I_1, I_2, \ldots, I_n\}$ denote the dataset consisting of n images for retrieval, and $H = \{H_1, H_2, \ldots, H_n\}$ with $H_i \in \{0, 1\}^q$ is the corresponding binary code of each image I. Given a query image I_q, we use its binary code H_q to identify the image from the images set I. We use Euclidean distance between H_q and $H_i \in H$ to define the hashing-level similarity:

$$d(q, i) = Dist(I_q, I_i) = ||H_q - H_i||. \tag{5}$$

The smaller the Euclidean distance is, the higher the similarity of the two images is. Each image I_i is ranked in descending order by the similarity. Hence, top k ranked images are identified.

Combined with semantic-level and hashing-level similarity Now, after defined the semantic-level similarity and hashing-level similarity, we can combine them to produce our hierarchical similarity between image a and b by:

$$\begin{aligned} Sim(a, b) &= \sum_i^C (p(a), H_a)\mathbf{S}(p(b), H_b) \\ &= \sum_i^C [P(I(a), I(b))]_t \times (1 - d(a, b)), \end{aligned} \tag{6}$$

where $\boldsymbol{R}_{II}(I(a), I(b)) = [P(I(a), I(b))]_t$ is the probability-based semantic-level similarity and $1 - d(q, i)$ is the hashing-level similarity. The former is a diagonal matrix, while the latter is a value. Operation "$\times$" combines the two similarities to a certain value to measure the similarity between images a and b. In practice, $\boldsymbol{R}_{II}$ is very sparse, which is beneficial to quickly create an image list relevant to the query image. Given a query image q, we go over the whole dataset for which image is semantic relevant to q. These images may be related to several relevant semantics or approximate semantics. We collect a list of all images those are related to these relevant semantics. The number of semantics relevant for a query image in our experiment about $1 \sim 10$. As a consequence, the total computation time of $Sim(a, b)$ can be calculated efficiently even for large-scale image sets.

3.2 Learning similarity

In the real world, the retrieval system needs to handle thousands of semantic categories and large-scale images, and the most important considerations are scalability and efficiency. As illustrated in Fig. 3, our approach includes two main steps. The first step is to get the image representation via a ConvNet which supervised pre-training on the *ImageNet* [17] dataset and fine-tuning on a *target* datasets. The pre-trained CNN model is proposed by Krizhevshy et al. [17] which consists of five convolutional layers, two fully-connected

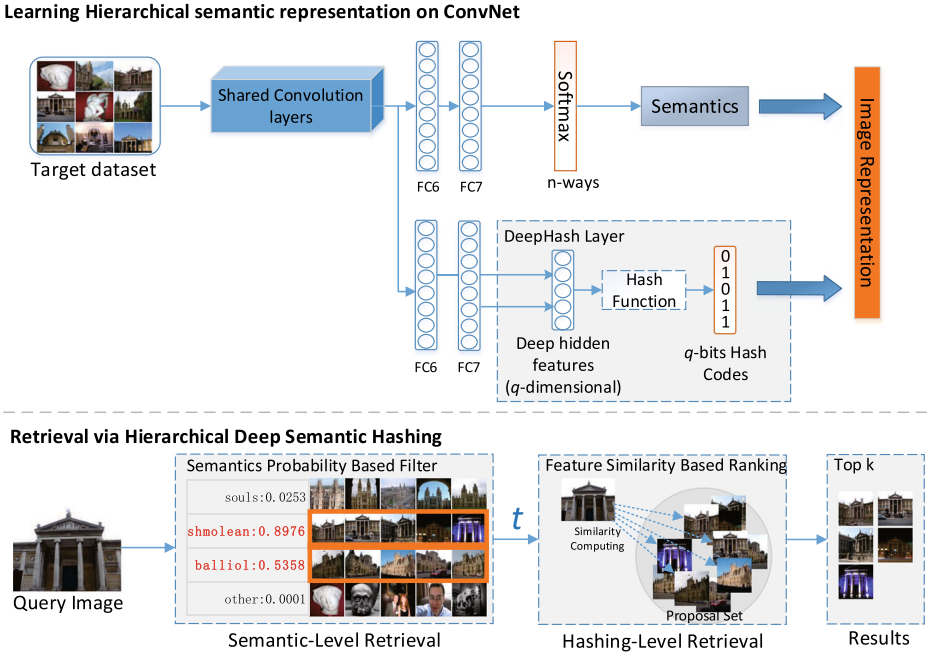


Fig. 3 The architecture of our proposed image retrieval method via hierarchical deep semantic hashing. *Top:* An image is input into a convolutional neural network which is supervised pre-training on *Imagenet* and fine-tuning on target dataset. The network has two output vectors per image: a *Softmax* probabilities and a hash-like representation. The two branches share the convolutional layers. Two CNN are both trained end-to-end with the groundtruth labels. The input of the two vectors are utilizing the fully-connected layers (FCs). *Down:* The images retrieval procedure is using a coarse-to-fine strategy that separate into two logistic components semantic-level retrieval and hashing-level retrieval

layers, and one 1000-ways one-hot encoding *softmax* classifier. Constructing hash function through incorporating the CNN model had been reported in some recent literature [41]. Different from [41], we compute the semantic from the last fully-connected layer directly, instead of hash layer. We contend that semantic feature extract only from the compact hash code will increase the semantic information loss. Hence, we branch the output of the final fully-connected layer into two streams. One is a n-ways *softmax* (n is corresponding to the number of categories of the target dataset) to generate semantic information. The other one is a hash-like function to make the CNN feature map to hash code.

To obtain the probability-based semantic representation, a deep mid-level feature is computed by forward a mean-substracted 227×227 image. We extract the mid-level feature from the last fully-connected layer, and learn a *softmax* classifier for each semantic simultaneously. Then we adjust the output of the classifier into probability. In our implements, we treat the scores of *softmax* as the probability of semantics.

To obtain the hashing-level feature representation, we concatenated the *FC6* and *FC7* layers to the DeepHash Layer to enable it encoding a much wider and various visual appearance, as shown in Fig. 3. We argue that the hash code only computed from the last fully-connected layer is dependent on categories too much and the strong invariance goes against to capture subtle semantic distinction. Accordingly, we redefine our deep hidden feature vector $g(I)$ as: $g(I) = [g_{fc6}(I); g_{fc7}(I)]$, where $g_{fc6}(\cdot)$ and $g_{fc7}(\cdot)$ denote the output features of the first fully-connected layer *FC6* and the second fully-connected layer *FC7*, respectively. Here bias terms and parameters of $g(\cdot)$ are omitted for the sake of concision. The deep hash function is still $H = h(x_i)$, but $i = 1, \ldots, q_{fc6} + q_{fc7}$. To generate a q-bit hash code, $H = [h(x_1), \ldots, h(x_{q_{fc6}+q_{fc7}})]$ can be computed.

3.3 Retrieval process

The second step of our procedure is image retrieval via hierarchical deep semantics hashing. In contrast to existing similarity learning algorithms that learn similarity from low-level feature, our similarity is the combination of semantic-level and hashing-level similarity. As (2) show, $Sim(a, b)$ is a joint algorithm. To calculate the similarity over all the images will consume considerable time with a simple similarity algorithm. Therefore, we adopted a hierarchical strategy to achieve retrieval task. As described in Fig. 3 (down), the semantic-level similarity is computed firstly, if the semantic relevance $\boldsymbol{R}_{II}(I(q), I(b))$ between query image q and target image b equal to zero, the image b will be discard, otherwise identify b as a candidate image. After the semantic relevance checking, we will obtain a pool of proposal set of m candidate images $P = \{I_m\}$, $m \ll N \in \mathbb{R}$. Then we compute the hashing-level similarity over proposal set P.

3.4 Efficiency analysis

Efficiency is the major challenge for large-scale retrieval. Thus computing the similarity between a query and each image is unacceptable.

Given a query image I_q and image dataset $I = \{I_1, I_2, \ldots, I_n\}$ with their semantic labels $L = \{1, \ldots, C\}$. Assuming the time complexity of calculating the similarity between image I_q and I_i is $O(1)$, then we can obtain the computational complexity over the whole dataset is $O(n)$. Our proposed method uses a hierarchy to reduce the system complexity. The number of the categories C^+ which semantics relevant to a query image is usually less than 10 (most of these are less than 5). Actually, computing the semantic relevance nearly cost-free, therefore, the system complexity depends on the size m of the proposal set P. If

the data distribution of each category is homogeneous, the computational complexity of our method can be denoted by $O(m)$ $\left(m << n, \frac{n}{C} = \frac{m}{C^+}\right)$ and the speedup ratio is $\frac{O(n)}{O(m)} = \frac{C}{C^+}$. For example, the speedup ratio of the *Holidays* dataset is about $500/(5 \sim 10) = 50 \sim 100$ times. We believe that this is a surprising result.

4 Experiments

In this section, we compare our CNN-based method to the current state-of-the-art retrieval pipelines including traditional SIFT-based methods [1, 5, 12, 19, 28, 40] and CNN-based methods [2, 8, 24, 30, 31, 41]. For a fair comparison, we only report results on representation with relevant dimensions and exclude post-processing methods such as spatial re-ranking or query expansion.

4.1 Datasets and evaluation criteria

We utilize the famous open source toolkit Caffe[14] to evaluate the performance of our method on several well-known instance-level image benchmark datasets listed below. We use *Holidays* [11], *Oxford5k* [29] to compare with existing similarity learning algorithm, use *Oxford105k* [29] and *ImageNet* [33] for large scale experiments, and use *Caltech256* [9] and *ImageNet* [33] to evaluate cross-category generalization. The *Holidays* [11] dataset consists of 1491 images corresponding to 500 groups, each of which represents a distinct scene or object. We use 500 queries images as our validation dataset, while the other 991 images as training dataset. The *Oxford5k* [29] consists of 5062 images collected from Flickr by searching for particular Oxford landmarks. The collection has been manually annotated to generate a comprehensive groundtruth for 11 distinct landmarks (some have complex structure and comprising several buildings), each represented by 5 possible queries. There are 512 images related to the 11 landmarks in "good" and "ok" setting, so we use these images to train our CNN, while use 55 queries images to validate the model. The *Oxford105k* dataset [29] is the combination of *Oxford5k* with a set of 100k negative images, in order to evaluate the search quality on a large scale. Following the standard evaluation protocol, the performance is measured by the mean average precision(mAP) over the provided queries. *Caltech256* dataset [9] consists of 30607 images associated with 256 object categories. In this work, we use 200 categories for training, 56 categories for validation. In the validation set, 30 % images for selecting threshold, 70 % images for testing. *ImageNet* ILSVRC 2012 dataset contains about 1.2 million training images, 50,000 validation images, roughly 1300 images in each of 1000 categories. For tuning parameters, we separate the original validation into two parts, 10,000 for parameter selection, 40,000 for testing and performance evaluation. We use the ranking based criteria [13] to evaluate this dataset.

In addition, we used the average query time to measure the time complexity of algorithms on all above datasets.

4.2 Data augmentation and preprocessing

Because of lack of training images of *Holidays* and *Oxford5k* dataset, training CNN from scratch becomes unrealistic. We use two effective ways to solve this problem, and to avoid over-fitting problem. Firstly, we use transfer learning to pre-train CNN on the large-scale *ImageNet* dataset and fine-tune on *target* datasets. Secondly, we take a novel data augmentation to reduce the effects of over-fitting. The *Holidays* and *Oxford5k* dataset have 991

and 512 raw training images, respectively. The imbalance problem of each category is quite prominent. As shown in Table 2, e.g. the category of *Oxford5k* "pitt_rivers" has only 1 training image, but the "radcliffe_camera" has 216 training images. To address this issue, we expand the number of each category to 1000, utilizing horizontal flipping, rotation and brightness transformations. We define an expansion ratio n for each category, where n = *floor(1000/N)*, N is the number of raw image corresponding to category c. Then we perform n/3 times random rotation on the N images in angle [-25, 25], and their horizontal flipping. Similarly, we perform n/6 times random luminance transformation by function f. Assuming L is the brightness of the original image range in $(low_{in}, high_{in})$, then the new brightness $L'(low_{out}, high_{out}) = f(L(low_{in}, high_{in}))$, function f is a mapping function, it maps the brightness range from $(low_{in}, high_{in})$ to $(low_{out}, high_{out})$. We define the largest range of luminance L is from 0 to 1, then the low_{out} is randomly selected from 0 to 0.2 and $high_{out}$ is randomly selected from range 0.8 to 1. In this way, we not only increase training data, but also balance the samples of each category. Furthermore, similar to [17], we also execute image crop and horizontal reflection on the whole training set.

4.3 Model training

Similarly to the Faster-RCNN [32], we adopt a pragmatic 2-step training algorithm to learn shared feature via step-by-step program. In the first step, we train the CNN by initialized with an *ImageNet*-pre-trained model and fine-tuned end-to-end on the target dataset. In the second step, we fix the semantic branch, then branch a new hash branch to generate hash code. The hash branch consists of two full-connected layers, a DeepHash layer, a hash function layer (include a Sigmoid activation function and an average-subtracted function, described in Section 3.1) and a new *Softmax* classifier. We train this network end-to-end with the same images until convergence, but we do not care about the accuracy of the hash branch. In inference stage, we combine the two branches to a unified framework and generate the semantic-level feature and hashing-level feature, simultaneously, as shown in Fig. 3(Top).

4.4 Hierarchical retrieval

4.4.1 Comparison of hierarchical

Feature layer in HDSH We first study the performance of convolutional feature from different layers with our Hierarchical Deep Semantic Hashing (HDSH) method. For simplicity, we denoted the last convolutional layer and the first, second fully-connected layers of CNN by *Conv5*, *FC6* and *FC7*, respectively. The output before the *softmax* classifier is denoted by *FC8*. As illustrated in Fig. 3, we extract the features from these layers as our deep CNN feature. Moreover, we use the *FC6* and *FC7* features to generate three kinds of synthetic features. We compute the average value and maximum value between the *FC6* and *FC7* features, and denote the results as *Mean* and *Max*. Finally, we denote a cascade feature by *Cas*, which concatenated with the *FC6* and *FC7* features. On the other side, we extract the DeepHash layer feature as our hashing feature (denote as Hq, q is the length of hash code, which is equivalent to the length of DeepHash Layer) to save storage space and accelerate retrieval. In all, we total obtain seven kinds of full-size features (HDSH-*Conv5*(*FC6*, *FC7*, *FC8*, *Mean*, *Max*, *Cas*)) and six kinds of compact features (HDSH-(H16, H32, H64, H128, H256, H512)). Figure 4 (Left) shows the performance for both hierarchical (HDSH-0.2) and non-hierarchical (HDSH-0) methods with different setting. The MAP first increases

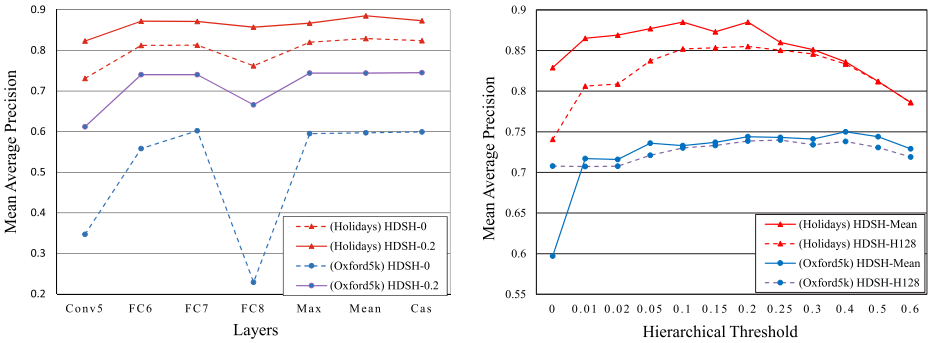


Fig. 4 The performance of different feature on both Holidays and Oxford5k datasets. *Left:* Retrieval Precision of our Hierarchical Deep Semantic Hashing method against non-hierarchical method as described in Section 4.4.1. *Solid* and *dash lines* correspond to the hierarchical and non-hierarchical methods respectively. *Right:* Combining Probability-based Semantic-Level Similarity with different threshold in retrieval step as described in Section 4.4.1. The *solid* and *dash lines* indicate the results of Mean feature and 128-bits deep hash codes extracted from CNN directly

until reaching a peak, because the deep network can achieve more invariance. However, the performance went an apparent decline at the last layer since the features become too generalized and less discriminative for instance-level retrieval. The best result of our approach on the *Holidays*, *Oxford5k* datasets is the HDSH-*Mean* which is the average value of the layer *FC6* and *FC7*, but the feature of *FC6*, *FC7*, *Max*, *Cas* are not that bad. On the *Holidays* dataset, the performance of hierarchical *Mean* feature is much better than that of the *Conv5* layer and the last layer(0.885 vs 0.823 and 0.857). Similar trend can be obviously seen on *Oxford5k* dataset. We then perform a similar experiment with the non-hierarchical feature, we still see this tendency. To our delight, the hierarchical feature is always better than non-hierarchical feature on both *Holidays* and *Oxford5k* datasets. On *Holidays*, the hierarchy has about $6 \sim 10$ percent performance boosts for all layers. In contrast, more than 30 percent improvements on *Oxford5k* datasets. In our opinions, more images in each category on *Oxford5k* dataset may be good for our HDSH method. This verifies that across different network layers and datasets, hierarchical perform the best and should be used for instance-level retrieval.

Parameters in HDSH The main function of threshold t is to filter the categories with lower semantic similarity, thus to improve the retrieval precision. The threshold t determines the number of the filtered categories. Therefore, in the probability-based semantic-level similarity, we need to determine a cutoff parameter t to filter the images which are not similar to the query image Q.

In order to choose appropriate threshold t, we perform image retrieval over a set of fixed threshold search space T, $t \in T$, $T = [0, 0.01, 0.02, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6]$. Due to *Holidays* and *oxford5k/105k* datasets are too small, threshold search and performance evaluation are all performed on the validation set directly. Besides, on the *Imagenet* and *Caltech256* datasets, threshold search worked on the validation set, but performance evaluation is implemented on testing set. In the threshold search space, $t = 0$ means the similarity between the image I and query image Q is only computed by the feature of the last fully-connected layer, this means a non-hierarchical method. As Section 3.1 described, higher threshold can filter more semantic dissimilar images, thus increase accuracy and

accelerate retrieval. But a higher threshold maybe excludes similar categories potentially, especially some complex pictures. Figure 4 illustrates the MAP curves on *Holidays* and *Oxford5k* with different thresholds t. We also compare the full-size feature HDSH-Mean with compact feature HDSH-H128 using 128bit hash code. We discover that MAP first increases with the increasing threshold, until reaching a peak it gradually drops. It shows that $t = 0.2$ is a great choice, because it gives a high MAP and filter appropriate images. In practice, threshold t probably dissimilar in different datasets, but in a certain range it is stable and acceptable. E.g., *Holidays*: $0.01 \sim 0.2$ and *Oxford5k*: $0.05 \sim 0.6$ are the acceptable choice. For convenience, we set $t = 0.2$ in this work for all the remaining experiments. It should be pointed out that most of the hierarchical methods are better than non-hierarchical methods. On *holidays* dataset, the HDSH-Mean-0.2 brings 6.8 % accuracy improvement (from 0.829 to 0.885). The same for compact version, the HDSH-H128-0.2 bring 15.4 % improvement (from 0.74 to 0.855). *Oxford5k* give a more surprising result, 24.6 % for HDSH-Mean-0.2 and 4.4 % for HDSH-H128-0.2.

Another very important key to set the cutoff threshold t is it reduce the retrieval space. Usually, computing the similarity is done on the whole dataset. That is to say all the categories will be scanned. By setting the cutoff threshold t, we only need to search a few categories, usually less than 10, even only one category. This strategy greatly increases the speed of retrieval. Table 4 shows the experimental results.

4.4.2 Comparison to state-of-the-art

Since our Hierarchical Deep Semantic method combined simple CNN hashing-level similarity with semantic-level similarity to express images, we only compare to other classic SIFT-based and CNN-based approaches.

Uncompressed representation We first compare our approach with other advanced approaches using uncompressed full-size representation in Table 1. In Fig. 4(Left), the feature HDSH-Mean performing very well, so we use this feature with/ without hierarchical deep semantic (HDSH-Mean-0/ HDSH-Mean-0.2) to compare with other methods. Although we do not focus on producing state-of-the-art results on full-size representation, our system gives competitive result compared to state-of-the-art methods. Specifically, our approach significantly outperforms most of the classic SIFT-based approaches with BoW encoding, which verifies the powerful representation ability of the CNN feature. Although better results are reported by other methods, such as a traditional approach [19] using Pairwise Geometric Matching, Hamming Embedding and Multiple assignment, and a CNN-based approach [31] using multi-resolution spatial search. Our approach still produces better or comparable results. Like the best method, our framework is not confined to the current setting, and can easily be extended to other re-ranking techniques and multi-resolution methods. In addition, compared to recent CNN-based approaches, our approach outperforms its opponent that retrain the entire network using extra data [2] or use time-consuming multi-scale sliding windows to extract feature [8]. [30] and [31] produce competitive method with ours, we believe that spatial search on multi-scale features can improve our performance.

To further demonstrate the performance of the proposed hierarchical deep semantic hashing (HDSH) algorithm, we show the MAP of each category of *Oxford5k* dataset in Table 2. Because CNN is a typical data-driven approach, enough raw data is especially important for training the network. There are 4 categories lower than the average value, but we can discover that the number of images of these categories are very few. Especially, class "pitt_rivers" and "keble" only have 1 and 2 pictures, respectively. Although we extend every

Table 1 The MAP of uncompressed representation

Method	Holidays	Oxford5k	Oxf105k
SIFT-based methods			
BoW 200k-D [12]	0.54	0.364	–
Improved FV [28]	0.626	0.414	–
VLADintra [1]	0.653	0.558	–
LCS+RN [5]	0.658	0.517	0.456
CVLAD [40]	0.827	0.514	-
HE+MA+PGM [19]	**0.892**	**0.737**	–
CNN-based methods			
Neural Codes [2]	0.793	0.545	0.512
MOP-CNN [8]	0.808	–	–
LFDN [24]	0.840	0.581	54.2
CNNaug-ss [30]	0.843	0.68	–
Spatial Pooling [31]	**0.896**	**0.843**	**0.795**
DHRS [41]	0.858	0.712	0.603
HDSH-Mean-0	0.829	0.597	0.523
HDSH-Mean-0.2	**0.885**	**0.744**	**0.712**

the bold indicate the best results in the comparison experiments.

category to 1000 images manually, the number of raw images is more important than the extended version, because these raw images can bring more invariant of categories. This implies that the accuracy can be improved effectively by increasing some raw images. In addition, it can be observed that the proposed hierarchical HDSH algorithm can get better retrieval application than the non-hierarchical algorithm for most queries.

Low-dimensional representation To trade-off retrieval accuracy, retrieval speed and storage space, most approaches compress the low-level feature to a low-dimensional representation. Unlike these methods, we generate hash code by CNN feature. As illustrated in Fig. 3, we focus on end-to-end method to product compact hash code and semantic feature simultaneously.

In Table 3, we compare with the previous best results. Our approach HDSH with threshold $t = 0.2$ achieves very competitive accuracy on all datasets with minimal performance loss. We demonstrate six kinds of different low-dimension feature to verify the algorithm validity. For all SIFT-based approaches, our HDSH outperforms them by a large margin,

Table 2 The detailed results of each category on *Oxford5k* dataset

Method	Souls	Ashm	Ball	Bodle	Christ	Corn	Hert	Keble	Magd	Pitt	Red	Avg
HDSH-mean(0)	0.566	0.486	0.439	0.802	0.597	0.385	0.936	0.434	0.285	0.680	0.958	0.597
HDSH-mean(0.2)	0.905	0.794	0.375	0.959	0.913	0.245	0.990	0.476	0.843	0.679	0.999	0.743
HDSH-H128(0)	0.963	0.917	0.521	0.802	0.833	0.290	1.000	0.440	0.739	0.280	1.000	0.708
HDSH-H128(0.2)	0.968	0.939	0.491	0.907	0.936	0.257	1.000	0.439	0.913	0.287	1.000	0.740
Numbers	72	20	7	19	73	4	49	2	49	1	216	512

Table 3 Comparison of low dimensional descriptors

Method	D	Holidays	Ox5k	Ox105k
LCS+RN [5]	16	0.323	0.27	0.222
Neural Codes [2]	16	0.609	0.418	0.354
HDSH-H16(0.2)	16	0.815	0.722	0.665
Neural Codes [2]	32	0.729	0.515	0.467
HDSH-H32(0.2)	32	0.858	0.723	0.665
Neural Codes [2]	64	0.777	0.548	0.508
HDSH-H64(0.2)	64	0.856	0.737	0.671
FV + T [10]	128	0.617	0.433	–
VLADintra [1]	128	0.625	0.448	–
LCS+RN [5]	128	0.335	0.322	0.262
Neural Codes [2]	128	0.789	0.557	0.523
LFDN [24]	128	0.836	0.558	52.9
HDSH-H128(0.2)	128	**0.855**	**0.739**	**0.676**
Neural Codes [2]	256	0.789	0.557	0.524
DHRS [41]	256	0.818	0.574	0.488
Spatial Pooling [31]	256	0.742	0.533	0.511
HDSH-H256(0.2)	256	**0.858**	**0.754**	**0.688**
MOP-CNN [8]	512	0.784	–	–
Neural Codes [2]	512	0.789	0.557	0.522
DHRS [41]	512	0.838	0.672	0.563
HDSH-H512(0.2)	512	**0.86**	**0.768**	**0.693**

the bold indicate the best results in the comparison experiments.

which again demonstrates the power of CNNs. Moreover, our low-dimension result outperforms [2] on every scale of dimension reduction version, even though [2] fine-tune the pre-trained model on a mess of extra images. Most interesting is the largest performance gap at the end, such as 16,32,64-bits. It is important to use sophisticated encoding methods to capture the local information of convolutional feature instead of simple max-pooling as in [31], but with low-dimensional descriptors it drops notably compared to our 256-bits representation, showing that elimination of spatial search greatly reduces the power of CNN representation. Besides that, our approach still outperforms MOP-CNN [8] using a larger 512-bits VLAD encoding, which further verifies that extracting convolutional feature from DeepHash layer is more suitable for instance-level image retrieval. To be sure, HDSH method has been no significant decline in accuracy when reducing the length of hashing-level code. It means the low-dimensional representation is robust and can retain good discriminative ability.

4.5 Result on imagenet large-scale dataset

Finally, we assessed the similarity on precision to demonstrate its efficacy and scalability on the *imagenet* dataset. We split the images to training set and validation set. To learn the Hierarchical Deep Semantic Hashing, we extract binary code using carefully crafted CNN described as Fig. 3. We compare our HDSH with several state-of-the-art binary methods [13]: **B-Hie**: a bilinear similarity hash with prior matrix; **Cosine-Hie**: same as B-Hie except with L2 normalized probability vectors; **B-Flat**: bilinear similarity without encoding

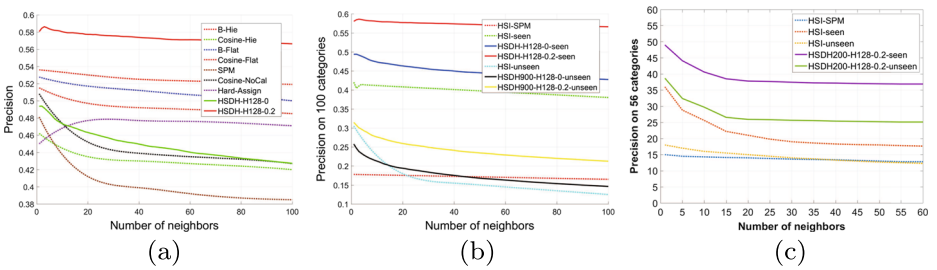


Fig. 5 *Left*: Retrieval precision of our Deep Semantic Hashing with hierarchy encoded against other methods on Imagenet dataset. For all curves of our approach, the threshold of hierarchical semantic is 0.2. *Middle*: HDSH use training data from 1000 categories and test on the same data ("seen in training") performs the best, but train on 900 categories and test on the other 100 categories ("unseen in training") compares favorably to other methods are very competitive. *Right*: Retrieval on Caltech256 dataset. HDSH use training data from 256 categories ("seen in training") performs the best, and train on 200 categories and test on the other 56 categories ("unseen in training") is always better than the other methods

the hierarchy prior matrix; **Cosine-NoCal**: cosine similarity of semantic classifiers without probability calibration; **Cosine-Flat**: cosine similarity without probability calibration; **SPM**: a no learning method which rank the images and representing low level feature by intersection kernel on SPM histograms of visual words; **Hard-Assign**: classifying the query image to the most likely category and ranking others by their probabilities related to this category.

We illustrate the results in Fig. 5(Left). Our HDSH with threshold $t = 0.2$ achieves significantly better precision than all others. It also demonstrates that hierarchy is important to our hash code. Compare to the runner-up, the HSDH-H128-0.2 is often beyond 7 % in precision. Furthermore, this performance has been done in 128-bits instead of thousands of bits.

4.6 Hierarchical efficiency

The most important improvement of our hierarchical deep semantic hashing method dramatically reduces search time. In this paper, we use the Hamming distance to calculate the similarity between two images. In Table 4, HDSH-xxx-0 means using the last fully-connected layer feature of CNN to compute similarity, while HDSH-xxx-0.2 means

Table 4 Comparison of retrieval times(ms) on all dataset

Method	D	(1)	(2)	(3)	(4)	(5)
Categories		500	12	12	257	1000
HDSH-mean-0	4K	138	693	3504	1121	45558
HDSH-mean-0.2	4K	0.83	167	666	13	333
Speedup Ratio		167.3	4.14	5.26	87	134.9
HDSH-H128-0	128	8.1	114	613	198	8900
HDSH-H128-0.2	128	0.15	28	140	5.4	54
Speedup Ratio		54	4.15	4.37	36.7	165.1

Dataset: (1) Holidays, (2) Oxford5k, (3) Oxford105k, (4) Caltech256, (5) Imagenet

using hierarchical strategy with threshold $t = 0.2$. The results indicate that hierarchy is very useful for image retrieval. Nevertheless, we found hierarchical method for speed improvement is not consistent. It is obvious that speedup ratio depends heavily on the number of categories. Actually, compute the semantic relevance nearly cost-free, so the retrieval time mainly depends on the proposal set size. Therefore, the more categories the dataset has, our proposed method can filter the more unrelated categories, and decreasing the search space is more significant.

Besides, we investigate our proposed method from the perspective of resource consumption. Following the most common implementation methods, we first use the GPU to achieve inference and feature extraction. In this process, other resource consumption can be ignored. Then, in the retrieval stage, we load the feature of images in advance. In this process, the consumption of memory is the most significant problem. Figure 6 shows the consumption of memory space with each 10,000 images. Increasing the length of the hash code can improve the performance (as shown in Table 3), but with the unwanted result of seriously increased memory consumption. It is noteworthy that, the result without hierarchy consumed 240MBytes memory each 10,000 images, but only achieves 0.829 MAP on *Holidays*, which is much worse than the low-dimensional HDSH methods, e.g. 128bit HDSH can achieve 0.855 MAP. The same phenomenon occurs in the *Oxford5k*, *Oxford105k* and *Imagenet* datasets.

4.7 Cross-category generalization

A potential advantage of using hierarchical deep semantic hashing is the ability to generalize to unseen categories in training. We use two large-scale dataset *Imagenet* and *Caltech256* to achieve this task. For fair, on the Imagenet dataset, the model is not using the pre-training model, but train from scratch. In the classification setting, only 900 semantic categories are used to train and build the semantic feature, and retrieval is only evaluated for other 100 categories for which no images are seen in the course of training ("unseen in training" curve in Fig. 5 (Middle)). Although the performance is much lower than the result which categories are seen in training("seen in training" curve in Fig. 5 (Middle)), it is still better than other baseline [13] with the same setting, even using compact version on 128-bit Hash Code. Encouragingly, our hierarchical deep semantic hashing with threshold $t = 0.2$ is always better than non-hierarchical version with threshold $t = 0$. It means hierarchical information is helpful to cross-category generalization so much. Similar to the *Imagenet*, we divide the Caltech256 dataset into two parts, 200 categories are used to train, and the rest of 56 categories are used to retrieval. In other words, these 56 categories are unseen in training. Figure 5 (Right) shows the search results. The unseen images are far worse than the seen images, but it is still better than the other methods. It proves our proposed method

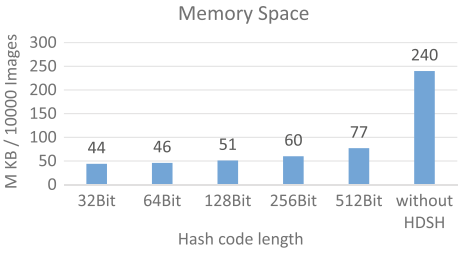


Fig. 6 The consumption of memory space. The vertical coordinate shows the memory requirement each 10,000 images, while the horizontal coordinate list the length of hashing feature. "without HDSH" means directly calculating similarity by Hamming distance without hierarchical methods

has the ability to generalize the unseen categories in training. Note here that training and testing image sets are mutually exclusive in all experiments.

In the real world application, retrieval on an open environment is a very challenging task. For example, if we want to retrieve a concept "Mexican pork soup" from *Imagenet* dataset, which belongs to a nonexistent category. Thanks to the *Imagenet* provide a rich category space, the system maybe returns a set of images which are related to a container, such as "soup bowl". However, if the semantics of a query is far apart with all the categories, the search result may be a tragedy, even if the proposal set consists of the "most similar" categories. As a consequence, the HDSH model based on supervision training, is more applicable to retrieve images in limited specific categories environment, even if its generalization ability is better than the other traditional and deep methods.

5 Conclusion

We have present an approach that can utilize the semantic hierarchy for similar image retrieval, and can be expanded to process large-scale image retrieval. The results presented in the previous section indicate the suitability of the proposed Hierarchical Deep Semantic Hashing method, even when the feature dimension reduces to a small scale such as 128-bit hash code. Combined with semantic-level similarity and hashing-level similarity, HDSH provides strong priors for computing distance, and reduces search space dramatically for matching similar image from a large-scale datasets. Moreover, we use a simple but innovative way to deal with quite a small dataset and imbalanced problem. Our final contribution is completing a unified architecture for generating probability-based semantic-level feature and hashing-level feature for image representation simultaneously. The results shown that it provide efficient retrieval, and may be useful in a wide range of applications, and better accuracy and scalability than the state of the art. Actually, in this paper, we mainly focus on the investigation whether the semantics can help us to improve retrieval performance. Therefore, replace the AlexNet with more powerful models(like VGG-16 or ResNet) will further improve performance.

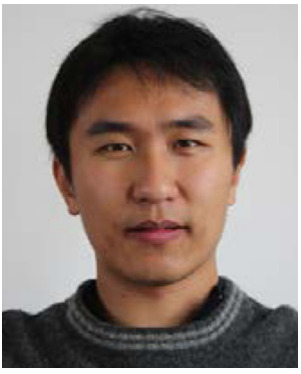
Acknowledgments This work was supported in part by the Natural Science Foundation of China under Grant U1536203 and 61272409, in part by the Major Scientific and Technological Innovation Project of Hubei Province under Grant 2015AAA013, the Nature Science Foundation of the Open University of China under Grant G16F3702Z and G16F2505Q. The authors appreciate the valuable suggestions from the anonymous reviewers and the Editors.

References

1. Arandjelovic R, Zisserman A (2013) All about vlad. In: IEEE conference on computer vision and pattern recognition (CVPR), 2013. IEEE, pp 1578–1585
2. Babenko A, Slesarev A, Chigorin A, Lempitsky V (2014) Neural codes for image retrieval. In: Computer vision–ECCV 2014. Springer, pp 584–599
3. Chechik G, Sharma V, Shalit U, Bengio S (2010) Large scale online learning of image similarity through ranking. J Mach Learn Res 11:1109–1135
4. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE computer society conference on computer vision and pattern recognition, 2005. CVPR 2005, vol 1. IEEE, pp 886–893
5. Delhumeau J, Gosselin PH, Jégou H, Pérez P (2013) Revisiting the vlad image representation. In: Proceedings of the 21st ACM international conference on multimedia. ACM, pp 653–656

6. Gionis A, Indyk P, Motwani R et al (1999) Similarity search in high dimensions via hashing. In: VLDB, vol 99, pp 518–529
7. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE conference on computer vision and pattern recognition (CVPR), 2014. IEEE, pp 580–587
8. Gong Y, Wang L, Guo R, Lazebnik S (2014) Multi-scale orderless pooling of deep convolutional activation features. In: Computer vision–ECCV 2014. Springer, pp 392–407
9. Griffin G, Holub A, Perona P (2007) Caltech-256 object category dataset. California Institute of Technology
10. Jégou H, Zisserman A (2014) Triangulation embedding and democratic aggregation for image search. In: IEEE conference on computer vision and pattern recognition (CVPR), 2014. IEEE, pp 3310–3317
11. Jegou H, Douze M, Schmid C (2008) Hamming embedding and weak geometric consistency for large scale image search. In: Computer vision–ECCV 2008. Springer, pp 304–317
12. Jégou H, Perronnin F, Douze M, Sanchez J, Perez P, Schmid C (2012) Aggregating local image descriptors into compact codes. IEEE Trans Pattern Anal Mach Intell 34(9):1704–1716
13. Jia D, Berg AC, Li FF (2011) Hierarchical semantic indexing for large scale image retrieval. Cvpr 32(14):785–792
14. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T (2014) Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the ACM international conference on multimedia. ACM, pp 675–678
15. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: IEEE conference on computer vision and pattern recognition (CVPR), 2014. IEEE, pp 1725–1732
16. Krizhevsky A, Hinton G (2011) Using very deep autoencoders for content-based image retrieval. In: ESANN. Citeseer
17. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
18. Kulis B, Grauman K (2009) Kernelized locality-sensitive hashing for scalable image search. In: IEEE 12th international conference on computer vision, 2009. IEEE, pp 2130–2137
19. Li X, Larson M, Hanjalic A (2015) Pairwise geometric matching for large-scale object retrieval. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR)
20. Lin K, Yang HF, Hsiao JH, Chen CS (2015) Deep learning of binary hash codes for fast image retrieval. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 27–35
21. Liu W, Wang J, Ji R, Jiang YG, Chang SF (2012) Supervised hashing with kernels. In: IEEE conference on computer vision and pattern recognition (CVPR), 2012. IEEE, pp 2074–2081
22. Long J, Shelhamer E, Darrell T (2014) Fully convolutional networks for semantic segmentation. arXiv preprint arXiv:1411.4038
23. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
24. Ng JY, Yang F, Davis LS (2015) Exploiting local features from deep networks for image retrieval. In: 2015 IEEE conference on computer vision and pattern recognition workshops, pp 53–61
25. Norouzi M, Blei DM (2011) Minimal loss hashing for compact binary codes. In: Proceedings of the 28th international conference on machine learning (ICML-11), pp 353–360
26. Ntalianis K, Tsapatsoulis N, Doulamis A, Matsatsinis N (2014) Automatic annotation of image databases based on implicit crowdsourcing, visual concept modeling and evolution. Multimedia Tools Appl 69(2):397–421
27. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. Int J Comput Vis 42(3):145–175
28. Perronnin F, Liu Y, Sánchez J, Poirier H (2010) Large-scale image retrieval with compressed fisher vectors. In: IEEE conference on computer vision and pattern recognition (CVPR), 2010. IEEE, pp 3384–3391
29. Philbin J, Chum O, Isard M, Sivic J, Zisserman A (2007) Object retrieval with large vocabularies and fast spatial matching. In: 2007 IEEE conference on computer vision and pattern recognition, pp 1–8
30. Razavian AS, Azizpour H, Sullivan J, Carlsson S (2014) Cnn features off-the-shelf: an astounding baseline for recognition. In: IEEE conference on computer vision and pattern recognition workshops (CVPRW), 2014. IEEE, pp 512–519
31. Razavian AS, Sullivan J, Maki A, Carlsson S (2014) Visual instance retrieval with deep convolutional networks. arXiv preprint arXiv:1412.6574
32. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. Computer Science
33. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M et al (2014) Imagenet large scale visual recognition challenge. Int J Comput Vis 1–42

34. Salakhutdinov R, Hinton G (2009) Semantic hashing. Int J Approx Reason 50(7):969–978
35. Toshev A, Szegedy C (2014) Deeppose: human pose estimation via deep neural networks. In: IEEE conference on computer vision and pattern recognition (CVPR), 2014. IEEE, pp 1653–1660
36. Wang N, Yeung DY (2013) Learning a deep compact image representation for visual tracking. In: Advances in neural information processing systems, pp 809–817
37. Wang J, Kumar S, Chang SF (2012) Semi-supervised hashing for large-scale search. IEEE Trans Pattern Anal Mach Intell 34(12):2393–2406
38. Weiss Y, Torralba A, Fergus R (2009) Spectral hashing. In: Advances in neural information processing systems, pp 1753–1760
39. Xia R, Pan Y, Lai H, Liu C, Yan S (2014) Supervised hashing for image retrieval via image representation learning. In: Proceedings of the AAAI conference on artificial intelligence, pp 2156–2162
40. Zhao WL, Jégou H, Gravier G (2013) Oriented pooling for dense and non-dense rotation-invariant features. In: BMVC-24th British machine vision conference
41. Zhao F, Huang Y, Wang L, Tan T (2015) Deep semantic ranking based hashing for multi-label image retrieval. In: IEEE conference on computer vision and pattern recognition, CVPR 2015, pp 1556–1564



Xinyu Ou received B.E. degree in electronic information science and technology and M.S. degree in software engineering from Yunnan University (YNU), Kunming, China, in 2004 and 2009 respectively. He pursuing the Ph.D degree in computer science, Huazhong University of Science and Technology (HUST) and he is currently a visitor student in Institute of Information Engineering, Chinese Academy of Sciences (CASIIE). He is an associate professor in Yunnan Open University (YNOU), Kunming, China. His research interests include deep learning, image retrieval and object detection and recognition.

Hefei Ling received the B.E. and M.S. degree in energy and power engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1999 and 2002 respectively, and the Ph.D. degree in computer science from HUST in 2005. Since 2006, he has been a professor with the college of computer science and technology, HUST. From 2008 to 2009, he joined in the department of computer science, University College London (UCL) as a visiting scholar. His research interests include digital watermarking and fingerprinting, copy detection, content security and protection. Dr. Ling has co-authored over 50 publications including journal and conference papers. He received Excellent Ph.D. dissertation of HUST in 2006 and Foundation Research Contribution Award of HUST in 2005, the best graduate student award from HUST, Wuhan, China in 1999.

Si Liu is an associate professor in Institute of Information Engineering, Chinese Academy of Sciences. She used to be a research fellow at Learning and Vision Group of National University of Singapore. She received her Ph.D. degree from National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences in 2012. Her research interests include computer vision and multimedia.

Jie Lei received her B.S. degree in computer science from Hainan University, Haikou, China, in 2003 and M.S. degree in computer science from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2006 and currently pursuing the Ph.D degree in computer science, Huazhong University of Science and Technology (HUST). She is currently an assistant professor in Nanchang University, Nanchang, China. Her research interests include computer vision, machine learning and deep learning.