arXiv:1501.06272v1 [cs.CV] 26 Jan 2015

Deep Semantic Ranking Based Hashing for Multi-Label Image Retrieval

Fang Zhao Yongzhen Huang Liang Wang Tieniu Tan
Center for Research on Intelligent Perception and Computing
Institute of Automation, Chinese Academy of Sciences
{fang.zhao,yzhuang,wangliang,tnt}@nlpr.ia.ac.cn

Abstract

With the rapid growth of web images, hashing has received increasing interests in large scale image retrieval. Research efforts have been devoted to learning compact binary codes that preserve semantic similarity based on labels. However, most of these hashing methods are designed to handle simple binary similarity. The complex multilevel semantic structure of images associated with multiple labels have not yet been well explored. Here we propose a deep semantic ranking based method for learning hash functions that preserve multilevel semantic similarity between multi-label images. In our approach, deep convolutional neural network is incorporated into hash functions to jointly learn feature representations and mappings from them to hash codes, which avoids the limitation of semantic representation power of hand-crafted features. Meanwhile, a ranking list that encodes the multilevel similarity information is employed to guide the learning of such deep hash functions. An effective scheme based on surrogate loss is used to solve the intractable optimization problem of multivariate ranking measures involved in the learning procedure. Experimental results show the superiority of our proposed approach over several state-of-the-art hashing methods in term of ranking evaluation metrics when tested on multi-label image datasets.

1. Introduction

Representing images efficiently is an important task for large scale content-based image retrieval. Binary hashing has attracted extensive attention due to computational and storage efficiencies of binary hash codes. It aims to map high-dimensional image data to compact binary codes in a Hamming space while maintaining some notion of similarity in the original space (e.g., Euclidean space or semantic space).

Early hashing methods are data-independent, such as locality sensitive hashing [4] and its variants, which use random projections as hash functions without exploring the data distribution. Recently, various data-dependent hashing methods have been proposed, which learn hash functions according to the data distribution. Some of them mainly focus on preserving the neighborhood structure of the input data in the Euclidean space, such as spectral hashing [28] and binary reconstructive embedding [12]. However, Euclidean distance in the original feature space sometimes cannot measure well the semantic similarity that is essential for image retrieval.

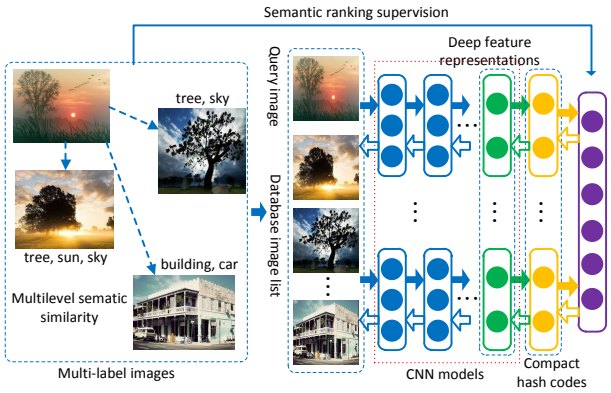


Figure 1. The proposed deep semantic ranking based hashing. Solid and hollow arrows indicate forward and backward propagation directions of features and gradients respectively. Hash functions consist of deep convolutional neural network (CNN) and binary mappings of the feature representation from the top hidden layers of CNN. Multilevel semantic ranking information is used to learn such deep hash functions to preserve the semantic structure of multi-label images.

To preserve semantic structure of the data, hashing methods with supervisory information in form of class labels have been further developed [20, 23, 6, 24, 16]. Through formulating hash function learning as a classification problem or as an optimization problem of pairwise relation based loss functions, these methods are able to learn hash codes which preserve binary semantic similarity. But in practice images are usually simultaneously associated with multiple semantic labels, and in this case the similarity re-

lationship between two images is more complex and is usually relevant to the number of common labels that images have. Consequently, a multilevel measure (such as very similar, normally similar and dissimilar) is required to describe the similarity, which cannot be handled well by the above methods and has not been studied well.

Besides, the learning capability of the standard pipeline followed by most hashing method, i.e., firstly extracting features like GIST [18] and SIFT [15] as image representations, and then learning mappings from these representations to binary codes, is inadequate for dealing with relatively complex semantic structure due to the semantic information loss in the hand-crafted features which are usually extracted in an unsupervised way. Thus more effective semantic feature representation is also desirable.

In this paper, we introduce a novel framework based on semantic ranking and deep learning model for learning hash functions that preserve multilevel similarity between multi-label images in the semantic space. An overall view of the proposed framework termed deep semantic ranking based hashing (DSRH) is illustrated in Fig. 1. Here we use deep convolutional neural network (CNN) [11] to construct hash functions to learn feature representations directly from images, which provides much richer sematic information than hand-crafted features. Meanwhile, we learn such deep hash functions with ranking supervision where multilevel semantic structure is characterized as the order of a ranking list derived from shared class labels between query and database images. This joint optimization of feature representation and mappings from them to hash codes is more effective than the conventional two-stage pipeline. A ranking loss defined on a set of triplets is used as surrogate loss to solve the multivariate optimization problem resulting from ranking measures, and then the stochastic gradient descent algorithm can be used to optimize model parameters. We evaluate the proposed DSRH method on a couple of multi-label image datasets and compare it with several state-of-the-art hashing methods based on both hand-crafted features and activation features from the CNN model. Experimental results demonstrate that our method is able to capture complex multilevel semantic structure and significantly outperforms other hashing methods in ranking quality.

Our main contributions include: 1) A novel hash function learning formwork is proposed to combine semantic ranking and deep learning model to address the problem of preserving multilevel semantic similarity between multi-label images. To the best of our knowledge, it is the first time to exploit deep convolutional neural network with list-wise ranking supervision for hashing. 2) A scheme based on surrogate losses is applied to the proposed framework to effectively solve the optimization problem of multivariate ranking measures. 3) Our method boosts the benchmark of multi-label image retrieval, achieving the state-of-the-art performance in terms of ranking evaluation metrics.

The rest of this paper is organized as follows. Related work is briefly discussed in Section 2. The proposed deep semantic ranking based hashing is formulated and optimized in Section 3. Experimental evaluations are presented in Section 4. Finally, Section 5 concludes this paper.

2. Related Work

As described before, the existing hash methods can be roughly divided into two categories: data-independent and data-dependent. Here we mainly discuss data-dependent hash methods preserving the semantic structure which this paper focuses on. Iterative quantization with canonical correlation analysis (CCA-ITQ) [6] utilizes CCA with labels to reduce the dimensionality of input data and binarizes the outcome through minimizing the quantization error, where the pointwise label information is exploited to guide hash function learning. By comparison, some approaches try to preserve the semantic similarity based on pairwise relation. Boosted similarity sensitive coding (BSSC) [20] assigns each pair of data points a label to learn a set of weak classifiers as hash functions. Semi-supervised hashing (SSH) [24] minimizes an empirical error over the labeled pairs of points and makes hash codes balanced and uncorrelated to avoid overfitting. Motivated by latent structural SVM, minimal loss hashing (MLH) [16] proposes a pairwise hinge-like loss function and minimizes its upper bound to learn similarity-preserving binary codes.

Furthermore, order-preserving approaches, which are more related to this paper, explicitly use ranking information in objective functions to learn hash codes that preserve the similarity order in the original space. Order preserving hashing (OPH) [27] formulates an alignment between the similarity orders computed respectively from the original Euclidean space and the Hamming space, which can be solved using the quadratic penalty algorithm. On the basis of [16], hamming distance metric learning (HDML) [17] develops a metric learning framework based on a triplet ranking loss to preserve relative similarity. However, this triplet loss function only considers local ranking information and is limited in capturing information about multilevel similarity. By using a triplet representation for list-wise supervision, ranking-based supervised hashing (RSH) [25] minimizes the inconsistency of ranking order between the hamming and original spaces to keep global ranking order. Different from RSH, our method leverages deep learning model to discover deeper semantic similarity and can scale well on large training sets. Column generation hashing (CGH) [13] and StructHash [14] combine ranking information with the boosting framework to learn a weighted hamming embedding. In contrast, our method needs no extra weight to rank hash codes.

Deep learning models, particularly deep convolutional

neural networks (CNNs), have achieved great success in various visual tasks such as image classification, retrieval and object detection [11, 26, 5] due to their powerful representation learning capability. Through assuming the stationarity of statistics and the locality of pixel dependencies in images, CNNs have much fewer connections and parameters than standard deep neural networks, which makes them easier to train, while the performance is nearly not affected. There are a few hashing methods that also use deep models. Torralba et al. [23] model a deep network by using multiple layers of RBMs, and maximize the number of neighbors around each query belonging to the same class to learn model parameters. Given approximate hash codes learned from pairwise similarity matrix decomposition, Xia et al. [29] learn hash functions through using CNNs to fit the learned hash codes and class labels. However, these methods do not explicitly impose the ranking constraint on the deep models, which can not figure out the multi-level similarity problem.

3. Our Method

In general, a hash function $h : \mathbb{R}^D \rightarrow \{-1, 1\}$ is treated as a mapping that projects a D-dimensional input onto a binary code. Assume that we are given a set of class labels $\mathcal{L} = \{1, ..., C\}$ and a dataset $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$ where each data point $\mathbf{x} \in \mathbb{R}^D$ is associated with a subset of labels $\mathcal{Y} \subseteq \mathcal{L}$, our goal is to learn a set of hash functions $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), ..., h_K(\mathbf{x})]$ that generates K-bit ($K \ll D$) binary codes while preserving the similarity structure of data points with multiple labels.

3.1. Deep Hash Functions

A good form of hash functions is important for obtaining desirable hash codes. As mentioned earlier, most conventional hashing methods first extract visual features like GIST and SIFT from images and then learn "shallow" (usually linear) hash functions upon these features. However, these hand-crafted features have limited representation power and may lose key semantic information which is important to the task of similarity search. Here we consider designing deep hash functions using CNNs to jointly learn feature representations from raw pixels of images and their mappings to hash codes. This non-linear hierarchical hash function has more powerful learning capability than the shallow one based on features extracted in advance, and thus is able to learn feature representations more suitable for multilevel semantic similarity search.

As shown in Fig. 2, we construct hash functions through incorporating the CNN model whose architecture is the same as [10]. A deep feature representation is computed by forward propagating a mean-subtracted 224×224 image through five convolutional layers and two fully connected layers, and then fed into the last hash layer to generate a compact binary code. Please refer to [11, 10] for more details about the geometry of the convolutional layers, local normalization and max pooling. Unlike image classification, hash codes are expected to contain global feature information within an image when used for retrieval. Thus instead of cropping an image, we warp all pixels in the image to the required size. Inspired by [22], we add a bypassing connection between the first fully connected layer (FCa) (called the skipping layer) and the hash layer to reduce the possible information loss. We argue that the features from the second fully connected layer (FCb) of CNN are dependent on classes too much and have strong invariance, which is unfavorable for capturing subtle sematic distinction. Thus we connect the hash layer to both the two fully-connected layers to enable it encoding more diverse information biased toward visual appearance. Accordingly we define a deep hash function as:

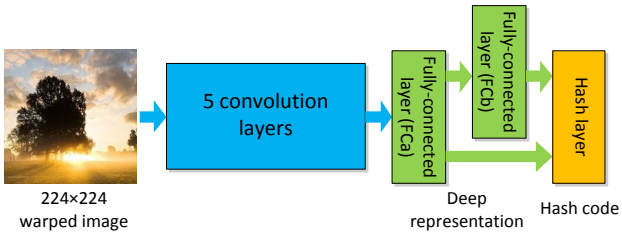


Figure 2. The structure of deep hash function. An input image is first transformed to a fixed size, and then goes through five convolution layers and two fully-connected layers, which provides a deep feature representation. Finally, the hash layer generates a compact binary code. The hash layer is also directly connected to the first fully-connected layer (FCa) in order to utilize diverse feature information biased toward visual appearance.

$$h(\mathbf{x}; \mathbf{w}) = \text{sign}(\mathbf{w}^{\mathrm{T}}[f_a(\mathbf{x}); f_b(\mathbf{x})]), \quad (1)$$

where $f_a(.)$ and $f_b(.)$ denote feature vectors from the outputs of the layers FCa and FCb respectively. They can be represented as the composition of the functions of previous layers and their parameters are omitted for the sake of concision. To obtain a K-bit binary code, $\mathbf{h}(\mathbf{x}; \mathbf{W}) = [h_1(\mathbf{x}; \mathbf{w}_1), h_2(\mathbf{x}; \mathbf{w}_2), ..., h_K(\mathbf{x}; \mathbf{w}_K)]$ can be computed.

3.2. Semantic Ranking Supervision

When each data point in $\mathcal{D}$ is associated with a single class label, pairs of points could be labelled as either similar or dissimilar according to whether they have the same label, and the learning procedure of hash functions is simply to make the Hamming distances between binary codes small (large) for similar (dissimilar) pairs. However, in the case of multiple labels, there exits multilevel similarity between data points depending on how many common labels they have. To preserve such multilevel semantic structure, one of the most essential ways is that for individual data points,

we keep the ranking order of neighbors computed by the Hamming distance consistent with those derived from semantic labels in terms of ranking evaluation measures.

Assume we have a sample point from $\mathcal{D}$ as a query $\mathbf{q}$. For the query $\mathbf{q}$, a semantic similarity level r of a database point $\mathbf{x}$ with $\mathbf{q}$ can be calculated based on the number of their common labels. The most similar database points are those sharing all the labels with $\mathbf{q}$, and assigned a level $r = |\mathcal{Y}_q|$. Accordingly, the second similar points, which share any $|\mathcal{Y}_q| - 1$ of the labels, are assigned a level $r = |\mathcal{Y}_q| - 1$. At last, the dissimilar points share none of the labels and are assigned a level $r = 0$. And then we can obtain a ranking list for $\mathbf{q}$ by sorting the database points in decreasing order of their similarity levels. According to the ground-truth ranking, various evaluation criteria can be used to measure the consistency of the rankings predicted by hash functions, such as the Normalized Discounted Cumulative Gain (NDCG) score [8], which is a popular measure in the information retrieval community and defined as:

$$NDCG@p = \frac{1}{Z}\sum_{i=1}^{p}\frac{2^{r_i}-1}{\log(1+i)}, \tag{2}$$

where p is the truncated position in a ranking list, Z is a normalization constant to ensure that the NDCG score for the correct ranking is one, and r_i is the similarity level of the i-th database point in the ranking list. Directly optimizing such ranking criteria is intractable, which involves minimizing multivariate ranking losses. One solution is to regard it as a problem of structured output learning and optimize the problem by structured SVM. But this framework is not suitable for deep learning model which is used to construct our hash functions. Next we will discuss a simple but effective scheme based on surrogate loss.

3.3. Optimization with Surrogate Loss

To circumvent the problem of directly optimizing the ranking criteria, we try to use a surrogate loss as the risk that the learning procedure minimizes in practice. Given a query $\mathbf{q}$ and a ranking list $\{\mathbf{x}_i\}_{i=1}^{M}$ for $\mathbf{q}$, we can define a ranking loss on a set of triplets of hash codes as follows:

$$L(\mathbf{h}(\mathbf{q}), \{\mathbf{h}(\mathbf{x}_i)\}_{i=1}^{M}) = \sum_{i=1}^{M}\sum_{j:r_j<r_i}[\delta d_H(\mathbf{h}(\mathbf{q}), \mathbf{h}(\mathbf{x}_i), \mathbf{h}(\mathbf{x}_j)) + \rho]_+, \tag{3}$$

where M is the length of the ranking list, $[.]_+ = \max(0, .)$, $\delta d_H(\mathbf{h}, \mathbf{h}_1, \mathbf{h}_2) = d_H(\mathbf{h}, \mathbf{h}_1) - d_H(\mathbf{h}, \mathbf{h}_2)$, $d_H(.,.)$ is the Hamming distance and ρ is a margin parameter which controls the minimum margin between the distances of the two pairs. This surrogate loss is a convex upper bound on the pairwise disagreement which counts the number of incorrectly ranked triplets. This type of surrogate loss has been used in Ranking SVM [9] for leaning to rank where a score function for ranking is learned.

From the definition of NDCG, it can be observed that the top ranked items have a larger gain factor for the score, which better reflects the performance of the ranking models in practical image retrieval systems, because users usually pay most of their attentions to the results on the first few pages. Thus we wish that the ranking of these items could be predicted more accurately than others. However, (3) treats all triplets equally, which is not desired. Inspired by [1], we modify the ranking loss by adding adaptive weights related to the similarity levels of database points:

$$L_\omega(\mathbf{h}(\mathbf{q}), \{\mathbf{h}(\mathbf{x}_i)\}_{i=1}^{M}) = \sum_{i=1}^{M}\sum_{j:r_j<r_i}\omega(r_i, r_j)[\delta d_H(\mathbf{h}(\mathbf{q}), \mathbf{h}(\mathbf{x}_i), \mathbf{h}(\mathbf{x}_j)) + \rho]_+. \tag{4}$$

According to NDCG, the weight ω can be given by:

$$\omega(r_i, r_j) = \frac{2^{r_i} - 2^{r_j}}{Z} \tag{5}$$

where Z is the normalization constant in (2). The higher the relevance of $\mathbf{x}_i$ and $\mathbf{q}$ is than that of $\mathbf{x}_j$ and $\mathbf{q}$, the larger decline the NDCG score would suffer if $\mathbf{x}_i$ is ranked behind $\mathbf{x}_j$. And thus the larger weight should be assigned to this triplet. When $\omega(r_i, r_j) \equiv 1$, it corresponds to (3).

Given the dataset $\mathcal{D}$ as a training set, we wish to learn hash functions that optimize the rankings for all query points $\mathbf{q}$ from $\mathcal{D}$. Based on the surrogate loss (4) and the hash function (1), the objective function can be given by the empirical loss subject to some regularization:

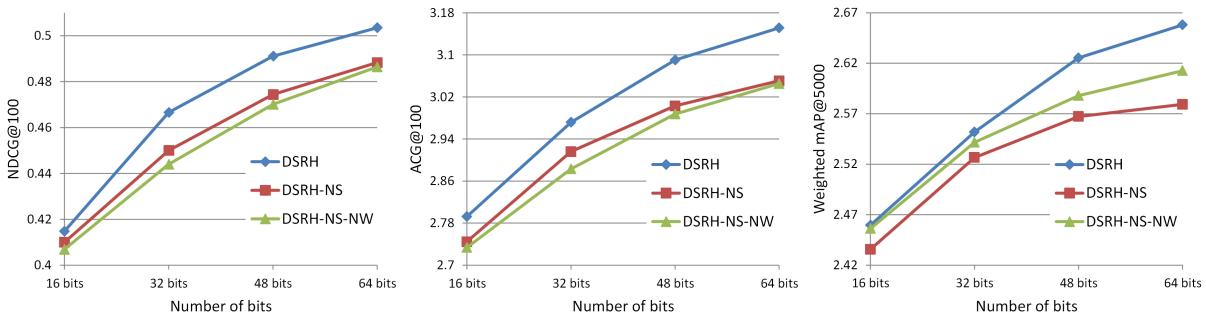
$$\mathcal{F}(\mathbf{W}) = \sum_{\mathbf{q}\in\mathcal{D}, \{\mathbf{x}_i\}_{i=1}^{M}\subset\mathcal{D}} L_\omega(\mathbf{h}(\mathbf{q};\mathbf{W}), \{\mathbf{h}(\mathbf{x}_i;\mathbf{W})\}_{i=1}^{M}) + \frac{\alpha}{2}\left\|\operatorname{mean}_{\mathbf{q}}(\mathbf{h}(\mathbf{q};\mathbf{W}))\right\|_2^2 + \frac{\beta}{2}\|\mathbf{W}\|_2^2. \tag{6}$$

Similar to [17], the second term is the balance penalty which is used to encourage each bit averaged over the training data to be mean-zero and to make sure more stable convergence of the learning procedure. And the third term is the L_2 weight decay which penalizes large weights [10]. Due to the non-smooth sign function in (1), the optimization of (6) is difficult. To address this issue, we relax $h(\mathbf{x};\mathbf{w})$ to:

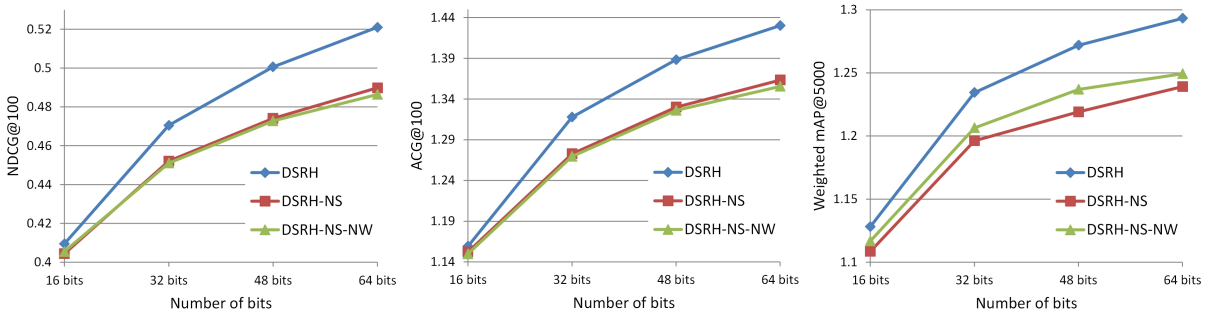
$$h(\mathbf{x};\mathbf{w}) = 2\sigma(\mathbf{w}^{\mathrm{T}}[f_a(\mathbf{x}); f_b(\mathbf{x})]) - 1, \tag{7}$$

where $\sigma(t) = 1/(1+\exp(-t))$ is the logistic function. In order to facilitate the gradient computation, we rewrite the hamming distance as the form of inner product:

$$d_H(\mathbf{h}(\mathbf{q};\mathbf{W}), \mathbf{h}(\mathbf{x};\mathbf{W})) = \frac{K - \mathbf{h}(\mathbf{q};\mathbf{W})^{\mathrm{T}}\mathbf{h}(\mathbf{x};\mathbf{W})}{2} \tag{8}$$



(a)



(b)

Figure 3. Ranking performance evaluations (NDCG, ACG and weighted mAP) of different components using various numbers of hash bits on two datasets: (a) MIRFLICKR-25K and (b) NUS-WIDE.

where K is the number of hash bits.

Stochastic gradient descent is used to minimize the objective function. It can be observed that the loss function (4) is actually a summation of a sequence of weighted triplet losses. For any one triplet $(\mathbf{q}, \mathbf{x}_i, \mathbf{x}_j)$, if

$$\frac{1}{2}(\mathbf{h}(\mathbf{q};\mathbf{W})^{\mathrm{T}}\mathbf{h}(\mathbf{x}_j;\mathbf{W}) - \mathbf{h}(\mathbf{q};\mathbf{W})^{\mathrm{T}}\mathbf{h}(\mathbf{x}_i;\mathbf{W})) + \rho > 0,$$

the derivatives of (6) with respect to hash code vectors are given by:

$$\frac{\partial \mathcal{F}}{\partial \mathbf{h}(\mathbf{q};\mathbf{W})} = \frac{\alpha}{N_q}\operatorname*{mean}_{\mathbf{q}}(\mathbf{h}(\mathbf{q};\mathbf{W})) + \frac{1}{2}\omega(r_i, r_j)(\mathbf{h}(\mathbf{x}_j;\mathbf{W}) - \mathbf{h}(\mathbf{x}_i;\mathbf{W})) \quad (9)$$

$$\frac{\partial \mathcal{F}}{\partial \mathbf{h}(\mathbf{x}_i;\mathbf{W})} = -\frac{1}{2}\omega(r_i, r_j)\mathbf{h}(\mathbf{q};\mathbf{W}), \quad (10)$$

$$\frac{\partial \mathcal{F}}{\partial \mathbf{h}(\mathbf{x}_j;\mathbf{W})} = \frac{1}{2}\omega(r_i, r_j)\mathbf{h}(\mathbf{q};\mathbf{W}). \quad (11)$$

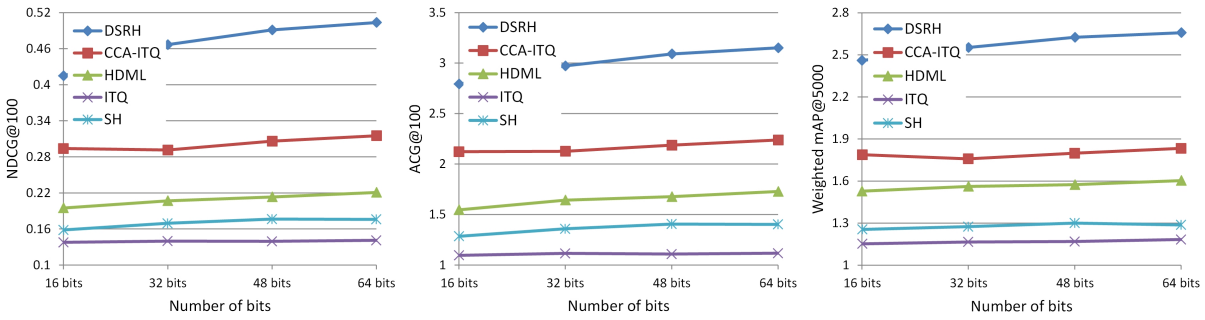
where the mean value is computed over one mini-batch and N_q is the size of a mini-batch. These derivative values can be fed into the following CNN via the back-propagation algorithm to update the parameters of each layer.

4. Experiments

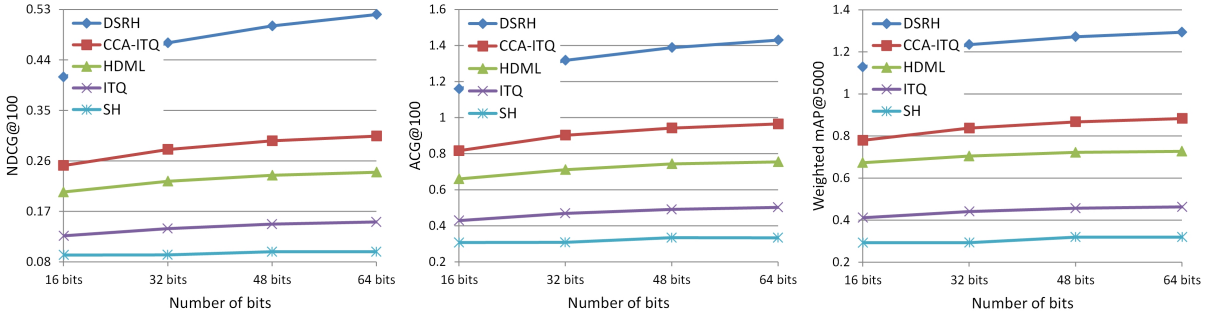
We test the proposed hashing method on two multi-label benchmark datasets, i.e., MIRFLICKR-25K [7] and NUS-WIDE [2]. We present quantitative evaluations in terms of ranking measures and compare our method with unsupervised methods: iterative quantization (ITQ) [6], spectral hashing (SH) [28], and supervised methods using labels: CCA-ITQ [6], hamming distance metric learning (HDML) [17].

We set the mini-batch size for gradient descent to 128, and impose dropout with keeping probability 0.5 on the fully connected layers to avoid overfitting. The regularization parameter α and β in the objective function (6) are set to 1 and $5e^{-4}$ respectively. The length of the ground-truth ranking list used for training is set to 3, which can be created by taking one item sharing all the labels with a query, one item without any common label and one item having at least one common label. For all compared methods, we use the best settings reported in their literatures.

The ImageNet ILSVRC-2012 dataset [19] is utilized to pre-train the CNN model by optimizing multinomial logistic regression objective function in the image classification task. This dataset contains about 1.2 million training images and 50,000 validation images, roughly 1000 images in each of 1000 categories. We use the pre-trained parameters of convolutional layers and fully-connected layers to



(a)



(b)

Figure 4. Comparison of ranking performance of our DSRH and other hashing methods based on hand-crafted features on two datasets: (a) MIRFLICKR-25K and (b) NUS-WIDE.

initialize the CNN part of hash functions in our method. In order to ensure fairness, we apply the features from the pre-trained CNN model to the compared hashing methods as well. Furthermore, the compared methods are carried out using the features that are learned through fine-tuning the CNN model on the multi-label datasets in our retrieval task.

4.1. Datasets

The MIRFLICKR-25K dataset [7] consists of 25,000 images collected from the social photography website Flickr. All images are annotated for 24 semantic concepts including various scenes and objects categories such as sky, night, food and tree. Moreover, 14 of these concepts are used for a stricter labeling, i.e., an image is annotated with a concept again only if the concept is salient. Thus we have total 38 semantic labels where each image may belong to several labels. 2000 images are randomly selected as testing queries and the remaining images are used as the database for training and retrieval. Following [21], a 3857-dimensional feature vector for each image is extracted by concatenating Pyramid Histogram of Words (PHOW) features, Gist and MPEG-7 descriptors, which will be used for the compared methods.

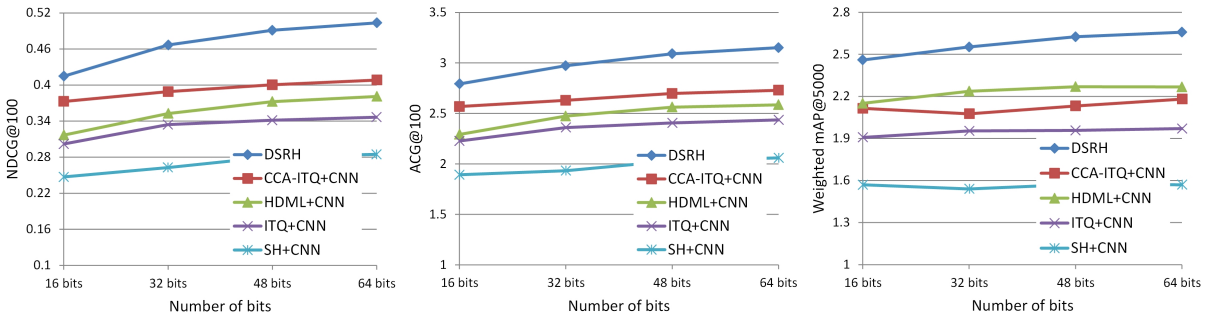
The NUS-WIDE dataset [2] is a relatively larger image dataset containing 269,648 images annotated with 81 concepts. It is also from Flickr, but more challenging than MIRFLICKR-25K due to more images and more diverse semantic concepts. Since the website of the dataset does not provide raw image data but the URLs of images, for learning deep feature representations we download these images ourselves from the web. However, we only collected 226,265 images as some of those URLs have been invalid now. We randomly sample 5000 images for testing queries and the rest is used for training and retrieval. The dataset includes six types of low-level features extracted from these images: bag of words based on SIFT feature, color histogram, color correlogram, edge direction histogram, wavelet texture and block-wise color moments. We concatenate them all and get a 1134-dimensional feature representation for each image.

4.2. Evaluation Criteria

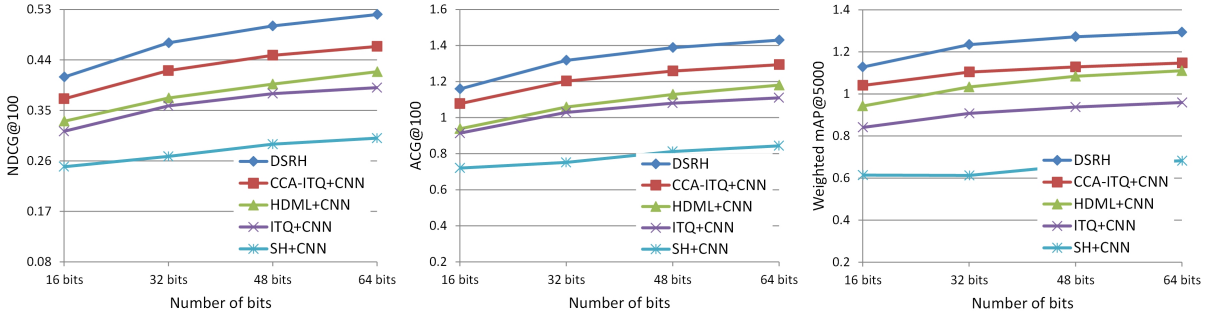
As mentioned before, NDCG defined in (2) is a popular measure in the information retrieval community which evaluates the ranking of data points by penalizing errors in higher ranked items more strongly. Besides, We use average cumulative gain (ACG) [8] and weighted mean average precision (mAP) to measure the ranking quality of retrieved database points.

ACG is calculated by taking the average of the similarity levels of data points within top-p positions:

$$ACG@p = \frac{1}{p}\sum_{n=1}^{p} r_i, \tag{12}$$



(a)



(b)

Figure 5. Comparison of ranking performance of our DSRH and other hashing methods based on activation features of pre-trained CNN on two datasets: (a) MIRFLICKR-25K and (b) NUS-WIDE.

where r_i is the similarity level of the data point on the i-th position of a ranking. ACG actually is equivalent to the precision weighted by the similarity level of each data point.

mAP is the mean of average precision for each query. Since the similarity level r can be bigger than one, we compute a weighted mAP using ACG:

$$mAP_w = \frac{1}{Q}\sum_{q=1}^{Q} AP_w(q), \tag{13}$$

$$AP_w = \frac{\sum_{p=1}^{M} \Pi(r_p > 0) ACG@p}{M_{r>0}}, \tag{14}$$

where $\Pi(.) \in \{0, 1\}$ is an indicator function and $M_{r>0}$ is the number of relevant data points.

4.3. Evaluation of Different Components

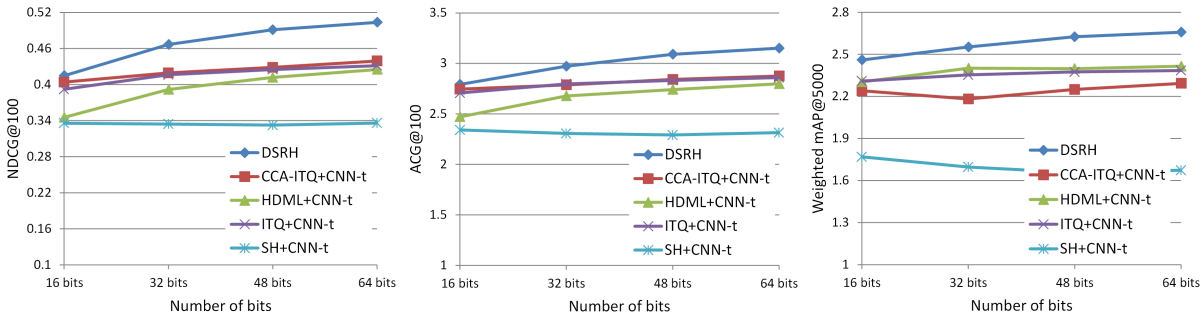
To analyze the effectiveness of several important components in the proposed DSRH, we remove the connection between the hash layer and the skipping layer and set $\omega(r_i, r_j) \equiv 1$ in (4) to evaluate their influence on the final performance. These two models are called DSRH-NS and DSRH-NS-NW. Here NS denotes no skipping layer and NW denotes no adaptive weight. Fig. 3 shows the results in the ranking measures.

We can see that using the surrogate loss with adaptive weights can improve the ranking quality of top-100 relevant items in terms of NDCG and ACG at the expense of the averaged ranking performance because it assigns larger weights to more relevant database points and weakens the effect of the less relevant ones. Connecting the first fully-connected layer to the hash layer can also improve the performance because more information biased toward visual appearance can be utilized which may be important for capturing multilevel semantic similarity.

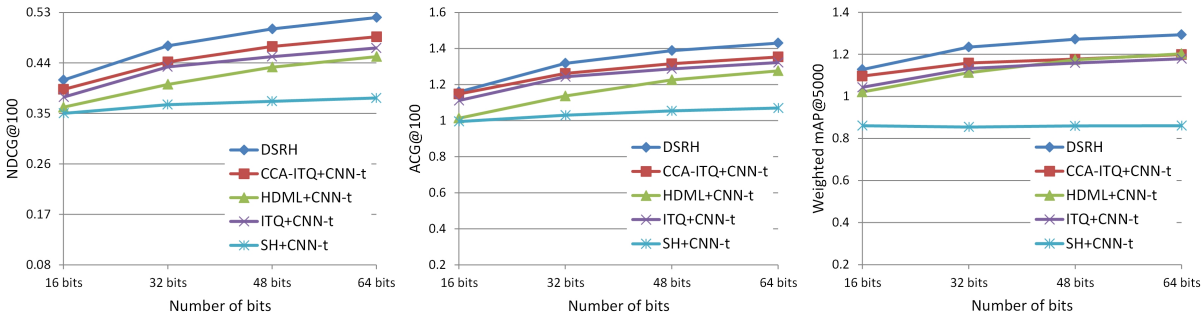
4.4. Method Comparison

We also compare the proposed DSRH with other hash methods based on hand-crafted features. Fig. 4 illustrates the scores of NDCG, ACG and weighted mAP of these methods using various numbers of bits. We can see that the performance of our method is significantly better than other methods based on hand-crafted features in all cases. By using the CNN model to construct hash functions, our method have higher learning capability and is able to exploit more semantic information than the hashing methods trained on hand-crafted features which are usually extracted by shallow models.

We further evaluate the compared hashing methods on the features obtained from the activation of the last hidden layer of the CNN model pre-trained on the ImageNet dataset. This activation feature can be seen as a generic visual feature and has been used in object recognition, domain adaption and scene recognition [3]. The results of NDCG, ACG and weighted mAP are shown in Fig. 5. Although



(a)



(b)

Figure 6. Comparison of ranking performance of our DSRH and other hashing methods based on activation features of fine-tuned CNN on two datasets: (a) MIRFLICKR-25K and (b) NUS-WIDE.

the activation features boost the ranking performance of the compared methods by a large margin, our method still has better performance because we construct the deep hash function to jointly learning feature representations and hash codes, which can utilize semantic supervision information to obtain features more fitted to the retrieval datasets. It is more effective than learning hash codes from the features learned in advance.

To further verify the superiority of our method, we use the activation features fine-tuned on the retrieval datasets, i.e., MIRFLICKR-25K and NUS-WIDE, to evaluate the compared methods as well. Specifically, we retrain the CNN model for multi-label image classification by using cross-entropy cost function. We report the results in Fig. 6. It can be seen that our method still achieves the best ranking performance, which shows that the multilevel semantic ranking supervision can make hash function learning better preserve the semantic structure of multi-label images. CCA-ITQ also uses multi-label information, but dose not explicitly learn with the ranking. HDML performs even worse than the unsupervised ITQ because it only considers binary similarity relationship which harms the semantic structure in the fine-tuned features. Note that using the features fine-tuned by multi-label supervision, the unsupervised ITQ performs almost as well as CCA-ITQ which is its supervised version, even better in terms of weighted mAP.

Similar to the skipping layer in the hash function of DSRH, we also attempt to concatenate the activations of the last two hidden layers of the CNN model as feature representations and apply them to the compared hashing methods. However, the performance of the compared methods trained using these features become even worse. It further validates the effectiveness of the structure of our hash function which has a tight coupling with CNN.

5. Conclusion

In this paper we have proposed to employ multilevel semantic ranking supervision to learn deep hash functions based on CNN which preserves the semantic structure of multi-label images. The CNN model with listwise ranking supervision is used to jointly learn feature representations and mappings from them to binary codes. The resulting optimization problem of multivariate ranking measure is solved by using a ranking loss on a set of triplets as the surrogate loss, which makes stochastic gradient descent could be used to optimize model parameters effectively. Extensive experiments demonstrate that the proposed method outperforms other state-of-the-art hashing methods in terms of ranking quality.

References

[1] Y. Cao, J. Xu, T. Liu, H. Li, Y. Huang, and H. Hon. Adapting ranking svm to document retrieval. In *Proc. Annual ACM*

SIGIR Conf., 2006. 4

[2] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *Proc. of the ACM Int. Conf. on Image and Video Retrieval*, 2009. 5, 6

[3] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proc. Int. Conf. Mach. Learn. (ICML)*, 2014. 7

[4] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. Int. Conf. Very Large Data Bases (VLDB)*, 1999. 1

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn. (CVPR)*, 2014. 3

[6] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. *IEEE T. Pattern Analysis Mach. Intelli. (TPAMI)*, 2012. 1, 2, 5

[7] M. J. Huiskes and M. S. Lew. The mir flickr retrieval evaluation. In *Proc. ACM Int. Conf. Multimedia Info. Retrieval*, 2008. 5, 6

[8] K. Jarvelin and J. Kekalainen. Ir evaluation methods for retrieving highly relevant documents. In *Proc. Annual ACM SIGIR Conf.*, 2000. 4, 6

[9] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. ACM Knowledge Discovery and Data Mining*, 2002. 4

[10] A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. *CoRR*, abs/1404.5997, 2014. 3, 4

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Adv. Neural Info. Process. Syst. (NIPS)*, 2012. 2, 3

[12] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *Proc. Adv. Neural Info. Process. Syst. (NIPS)*, 2009. 1

[13] X. Li, G. Lin, C. Shen, A. van den Hengel, and A. R. Dick. Learning hash functions using column generation. In *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013. 2

[14] G. Lin, C. Shen, and J. Xin. Optimizing ranking measures for compact binary code learning. In *Proc. Eur. Conf. Comp. Vis. (ECCV)*, 2014. 2

[15] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. Int. Conf. Comp. Vis. (ICCV)*, 1999. 2

[16] M. Norouzi and D. J. Fleet. Minimal loss hashing for compact binary codes. In *Proc. Int. Conf. Mach. Learn. (ICML)*, 2011. 1, 2

[17] M. Norouzi, D. J. Fleet, and R. Salakhutdinov. Hamming distance metric learning. In *Proc. Adv. Neural Info. Process. Syst. (NIPS)*, 2012. 2, 4, 5

[18] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision (IJCV)*, 2001. 2

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F. Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. 5

[20] G. Shakhnarovich, P. A. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proc. Int. Conf. Comp. Vis. (ICCV)*, 2003. 1, 2

[21] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Proc. Adv. Neural Info. Process. Syst. (NIPS)*, 2012. 6

[22] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn. (CVPR)*, 2014. 3

[23] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn. (CVPR)*, 2008. 1, 3

[24] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn. (CVPR)*, 2010. 1, 2

[25] J. Wang, W. Liu, A. X. Sun, and Y. Jiang. Learning hash codes with listwise supervision. In *Proc. Int. Conf. Comp. Vis. (ICCV)*, 2013. 2

[26] J. Wang, Y. Song, T. Leung, C. Rosenberg, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn. (CVPR)*, 2014. 3

[27] J. Wang, J. Wang, N. Yu, and S. Li. Order preserving hashing for approximate nearest neighbor search. In *Proc. of the 21st ACM Int. Conf. on Multimedia*, 2013. 2

[28] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proc. Adv. Neural Info. Process. Syst. (NIPS)*, 2008. 1, 5

[29] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan. Supervised hashing for image retrieval via image representation learning. In *Proc. Twenty-Eighth AAAI Conf. on Arti. Intel. (AAAI)*, 2014. 3