Deep Supervised Quantization by Self-Organizing Map

Min Wang[†], Wengang Zhou[†], Qi Tian[‡], Junfu Pu[†], Houqiang Li[†]

[†] CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, University of Science and Technology of China
[‡] Computer Science Department, University of Texas at San Antonio
{wm123,pjh}@mail.ustc.edu.cn,{zhwg,lihq}@ustc.edu.cn,qitian@cs.utsa.edu

ABSTRACT

Approximate Nearest Neighbour (ANN) search is an important research topic in multimedia and computer vision fields. In this paper, we propose a new deep supervised quantization method by Self-Organizing Map (SOM) to address this problem. Our method integrates the Convolutional Neural Networks (CNN) and Self-Organizing Map into a unified deep architecture. The overall training objective includes supervised quantization loss and classification loss. With the supervised quantization loss, we minimize the differences on the maps between similar image pairs, and maximize the differences on the maps between dissimilar image pairs. By optimization, the deep architecture can simultaneously extract deep features and quantize the features into the suitable nodes in the Self-Organizing Map. The experiments on several public standard datasets prove the superiority of our approach over the existing ANN search methods. Besides, as a byproduct, our deep architecture can be directly applied to classification task and visualization with little modification, and promising performances are demonstrated on these tasks in the experiments.

KEYWORDS

Approximate Nearest Neighbour Search, Supervised Quantization, Self-Organizing Map

1 INTRODUCTION

With the explosive growth of data in real multimedia applications, Approximate Nearest Neighbour (ANN) search has become an important research topic. Given a query, ANN search aims to find its nearest neighbours from a large-scale dataset in a sub-linear, or even constant time complexity. To address this problem, lots of methods have been proposed, including quantization-based methods [6, 17, 22, 27, 30, 32] and hashing-based methods [3, 13, 15, 16, 20, 21, 23, 25, 26, 29, 31].

The existing ANN search methods usually encode the input into a compact description, then the distance computation between the compact descriptions can be finished in a fast way even on a large-scale dataset. The key of ANN search methods is to keep the original distance measure for unsupervised cases or similarity measure for supervised cases after encoding. For hashing-based methods, they usually learn a projection function to map the data points into a low dimensional space at first, then binarize the mapped data points. Online distance computation in hashing is to calculate the Hamming distance between the binary codes. For quantization-based methods, they often train a codebook offline. Then for a new input sample, it will be quantized into one nearest codeword in the codebook. The distance between a pair of samples is approximated by the distance between the corresponding codewords. Online nearest neighbour search is realized by computing the distances between the codewords and ranking.

Although the above two kinds of ANN methods are both advantageous in terms of memory cost and search efficiency, there still exist some problems in each kind of methods. For hashing-based methods, the learning objective functions often involve discontinuous terms, which introduce difficulty into the optimization process. On the other hand, compared with the real-valued vectors used in quantization methods, the description capability of binary codes is limited, because many samples share the same Hamming distance to the query and cannot be accurately arranged in the resulted ranking list. For quantization-based methods, most of them only focus on learning in an unsupervised way, such as [6, 17, 27]. Using labels or classification information is convinced to improve the performance of ANN search because of more information used in the training process. As supervised hashing methods usually outperform unsupervised ones under the same bit length setting, utilizing supervised information in the quantization methods is a feasible way to improve the performance, which is verified by the experiments in [22].

With the great advance of deep learning techniques, especially Convolutional Neural Networks, many traditional multimedia tasks resort to deep architectures, such as image classification [10], object detection [18], and so on. For ANN search problem, nearly all the deep explorations concentrate on hashing-based methods[11, 13, 15, 24, 26, 28]. Deep hashing methods can simultaneously learn feature descriptions and hash functions in an end-to-end way, which are shown to outperform the classical two-step, hand-crafted feature based hashing methods. However, there is few research efforts on deep supervised quantization. In [22], a supervised quantization method is proposed, which includes separable feature extraction step and learning to quantization step. Different from [22], this paper explores the supervised quantization with a deep architecture.

This paper proposes a new deep supervised quantization method based on Self-Organizing Map (SOM), which is abbreviated as

MM '17, October 23–27, 2017, Mountain View, CA, USA.
 ISBN 978-1-4503-4906-2/17/10…$15.00
DOI: https://doi.org/10.1145/3123266.3123415

DeepSOM for conciseness. Self-Organizing Map projects multi-dimensional features to the two-dimensional map in a topology-preserving way. The weights of all the connection to one node in the two-dimensional map denote a codeword. The quantizer is intrinsically to find a nearest codeword, which also corresponds to the node with the maximal response on the map. Our method unifies Convolutional Neural Networks and Self-Organizing Map into one architecture. It aims to simultaneously minimize the supervised quantization loss together with the classification loss. Our supervised quantization objective is to minimize the differences on the maps between similar image pairs, and maximize the differences on the maps between dissimilar image pairs. After the optimization, the learned deep architecture can simultaneously extract deep features and quantize the features into the suitable node in Self-Organizing Map. We evaluate our method on several public datasets. The experimental results show the superiority of our approach over the existing ANN search methods. Besides, our deep architecture can be applied to classification and visualization tasks with promising performances in the experiments.

2 RELATED WORK

In this section, we review some related work from the following two aspects: (1) ANN search methods and (2) deep clustering methods.

Quantization is the traditional solution in many methods on ANN search. Recently, there have been some new explorations on quantization. As an efficient and scalable ANN search method, Product Quantization [6] decomposes the space into a Cartesian product of low-dimensional subspaces and quantizes each subspace separately. Then a vector is represented by a short code composed of its subspace quantization indices. Cartesian k-means [17] develops a new clustering model with a compositional parameterization of cluster centers to increase the representational capacity. Composite Quantization [27] makes use of the composition of several elements selected from the dictionaries to accurately approximate a vector and to represent the vector by a short code composed of the indices of the selected elements.

Most research efforts in quantization have been devoted to the unsupervised quantization methods. There are only a few supervised quantization methods. In [22], it proposes to map the data points into the low dimensional space, and quantize the data points in the transformed space with a new quantization criterion. It not only minimizes the quantization loss, but also keeps the semantic discriminability. Although this supervised quantization method shows superior performance, it still is a two-step method, which includes separable feature extraction and quantization stages. Compared with [22], our DeepSOM method simultaneously learns the features description and quantizer. The unified architecture is demonstrated to perform better than the separable two-step method.

On the other hand, binary hashing methods have attracted more and more research efforts because of their compact description and efficient distance computation. The binary hashing methods can be divided into three categories according to whether using supervised information (such as labels) in the training process: unsupervised, semi-supervised, and supervised hashing methods. Since our DeepSOM method is supervised, we only review the deep supervised hashing methods in this section.

In [24], a deep supervised hashing method CNNH is proposed for image retrieval, which can simultaneously learn feature representation and the corresponding hash functions. It involves two stages, including the similarity matrix learning and deep networks optimization. Deep Pairwise-Supervised Hashing (DPSH) [13] proposes to simultaneously learn features and hash functions by CNN with pairwise labels. The objective of DPSH is to make the Hamming distance between two similar points as small as possible, and make the Hamming distance between dissimilar points as large as possible. In [26], a deep supervised hash learning framework, *i.e.*, Deep Regularized Similarity Comparison Hashing (DRSCH), is proposed, which takes triplet samples as input. It maximizes the margin between the similar pairs and the dissimilar pairs, which intuitively guarantees learned binary codes to preserve the ranking orders of images. In addition to preserving the image ranking, it introduces the adjacency consistency as regularization. In [28], a deep hashing method is proposed to learn hash functions based on semantic ranking based method that preserves multilevel similarity between multi-label images. In [11], a carefully designed deep neural network is proposed to simultaneously learn features and hash functions. It also uses a triplet ranking loss to train the deep architecture. DSH [15] presents a new deep hashing method based on Siamese CNN, which aims to maximize the discriminability of the output space by encoding the supervised information from the input image pairs, and simultaneously minimizes the quantization loss during binarization.

Different from the above deep supervised binary hashing methods, our proposed DeepSOM is a deep supervised quantization method, which projects the input samples to the suitable nodes on the SOM rather than binary codes.

Recently, there has been some work involving deep clustering methods in different tasks. In [4], it presents a neural network-based end-to-end clustering framework. The network is trained with the contrastive criteria, *i.e.*, the distributions between the softmax output for a similar pair should be similar, while the distributions over the class labels should be dissimilar. In [5], a novel end-to-end network is presented to map unlabeled images to categories as a clustering network. The training task in [5] is a two-step process: first, it trains a similarity prediction network on an existing dataset to predict whether a pair of images is semantically similar; then on the new unlabeled data, it trains the category discovery network as a unsupervised clustering problem, whose outputs are expected to be distinguishable distributions. In [14], a clustering based regularization method is proposed to encourage parsimonious representations to facilitate generalization for deep networks.

Compared with the above work, our DeepSOM is different in three aspects: 1) we focus on a different task, *i.e.*, ANN search. In contrast, they are used for clustering and classification tasks; 2) since we use SOM as quantizer, the number of the output units in our architecture is not constrained as the number of the accurate cluster centroids, which allows us to solve different tasks flexibly; 3) our DeepSOM unifies the Convolutional Neural Networks and Self-Organizing Map into one architecture, to our best knowledge, which has not been explored before. On the classification task, the experiments prove that our DeepSOM method obtains better or comparable results with those deep clustering methods.

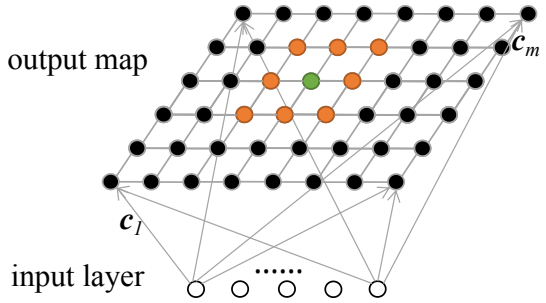

Figure 1: The structure of Self-Organizing Map. The orange nodes are the neighbours of the green node.

3 OUR METHOD

In this section, we first introduce the problem formulation. Then we briefly review Self-Organizing Map. After that we present our DeepSOM method. Finally we introduce the out-of-sample extension of our DeepSOM method.

3.1 Problem Formulation

In a large dataset $\mathbf{X} \in \mathbb{R}^{n \times d}$, given a query $\mathbf{q} \in \mathbb{R}^d$, ANN search aims to find its nearest neighbours in a sub-linear or even constant time complexity. As an efficient solution to ANN search problem, quantization-based method usually first learns a codebook offline, then encodes each data point in the database by assigning it to the closest codeword in the codebook. After quantization, each data point is represented by the index of the closest codeword. The original distance for each pair of data points is approximated by the distance of the corresponding codewords. We can compute the distances between all the pairs of codewords offline and save them in a table, so online ANN search only involves quantization, looking up table, and ranking steps. Our DeepSOM method simultaneously learns the feature descriptions and the quantizer in a unified framework. For a query, the proposed framework can directly output the indices of suitable nodes (*i.e.*, the result of quantization) on SOM. Then our method only needs extra looking up table and ranking manipulations to give the nearest neighbours of the query.

3.2 Self-Organizing Map Review

Self-Organizing Map (SOM) [7, 8] is a neural network which is trained by unsupervised learning.[1] The network in SOM only contains one single fully-connected layer, which learns the structure of input data space. It usually produces low-dimensional representation for each input vector, as illustrated in Fig. 1. The weights of the connections to each node in the output map present a codeword, and the value of each node on the output map denotes the similarity between the input vector and the corresponding codeword. The dimension of the output map is usually two or three, which makes it convenient for dimension reduction, vector quantization and visualization.

Different from the classical multi-layer neural networks, SOM applies competitive learning strategy for training. Once an output node obtains the maximal response value, the response of other nodes will be suppressed to zero except its neighbouring nodes. The most interesting point of SOM is its topology-preserving property because of the use of neighbourhood function in the training process. The input vectors selected from the same region in the input space will activate the same node on the output map. On the other hand, the distances between the input vector and the codewords, whose corresponding nodes locate in the small region on the map, are similar.

[1]Note that the SOM used in our paper refers as Kohonen map.

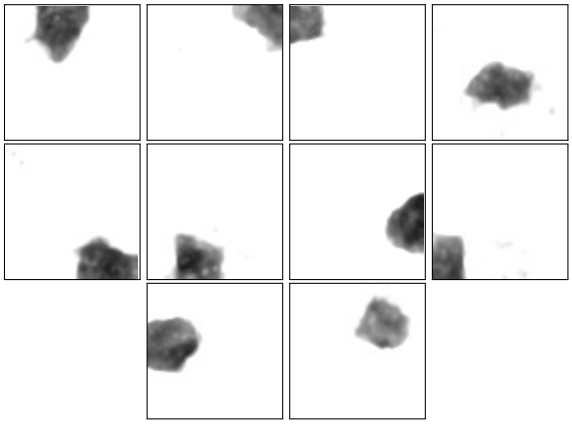

Figure 2: The output maps of our architecture for one randomly selected image from each class in CIFAR-10 dataset. Each output map is computed based on the inner product between the feature description of the selected image and all the codewords in SOM. We linearly project the values of the output maps into the range of $[0, 255]$. The black color denotes 255, while the white color denotes 0. The darker the point is, the more similar the image description is to the codeword corresponding to the node on the map.

The general objective function for training SOM is formulated as follows:

$$\min_{\mathbf{C}} \sum_i \sum_{j \in N(k), \mathbf{c}_k = q(\mathbf{x}_i)} \theta(j) ||\mathbf{x}_i - \mathbf{c}_j||_2^2, \tag{1}$$

where $q(\mathbf{x}_i)$ denotes the quantization function, which returns the nearest codeword $\mathbf{c}_k$ for the input $\mathbf{x}_i$. Each element in $\mathbf{c}_k$ denotes a weight of the connection between the corresponding input node to the k-th node on the map. $\theta(j)$ denotes neighbourhood function on the two-dimensional map, which weights the j-th neighbouring node according to the distance to the k-th node with maximal response. $N(k)$ denotes the set of neighbours for the k-th node according to the definition of neighbour radius. During the training process, the neighbour radius becomes smaller and smaller. The effect of neighbour function is that once a node is updated, its neighbour nodes are also updated.

The SOM is learnt by stochastic gradient descent method. After the learning process, we can get the set of codewords $\mathbf{C} = [\mathbf{c}_1^T, \mathbf{c}_2^T, \cdots, \mathbf{c}_m^T]^T \in \mathbb{R}^{m \times d}$, where m denotes the node number on the map. These codewords are also the weight vectors of the network. Finally, each learned codeword represents a small region of input space in a topology-preserving way. The distances between pairs of data points, which locate closely in the input space, are

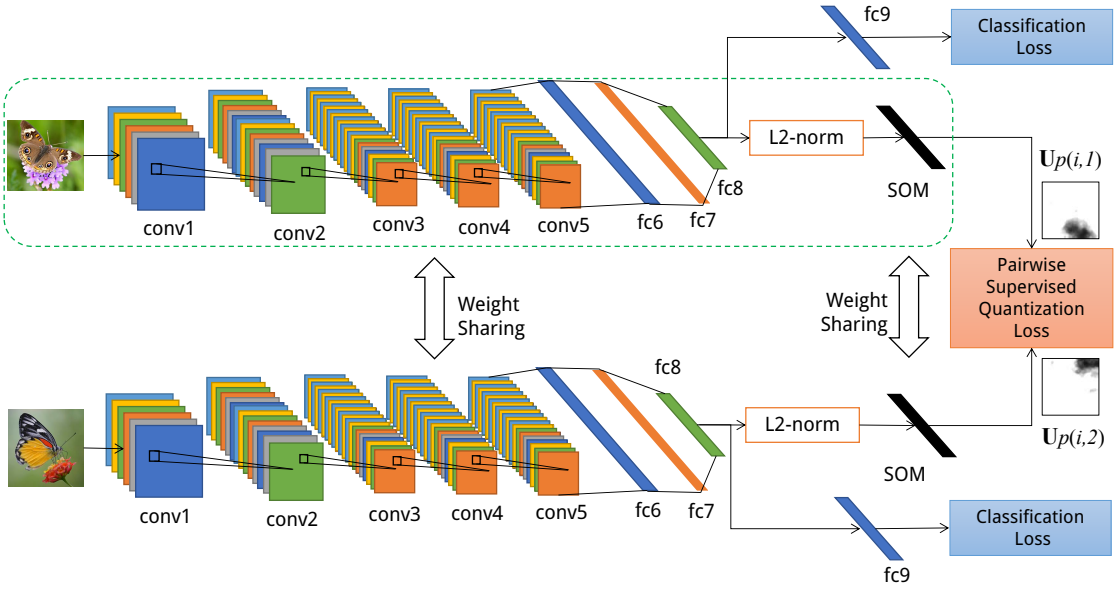

Figure 3: The architecture of our proposed DeepSOM in the training process. The content in the green dashed rectangle denotes the architecture used in the testing process.

small on the map. On the contrary, the distances between pairs of far data points in the input space are large on the map.

In the above formulation, the objective function is defined based on Euclidean distance. In our DeepSOM, for the concern of the combination with CNN, we use the inner product based SOM instead, *i.e.*, $< \mathbf{x}_i, \mathbf{c}_j >= \mathbf{x}_i \cdot \mathbf{c}_j^T$, where $\mathbf{x}_i$ and $\mathbf{c}_j$ are unit vectors.

The difference between SOM and traditional k-means algorithm mainly lies in: 1) k-means algorithm needs to initialize cluster number in advance, which largely impacts the performance of quantization. On the contrary, SOM does not need to set this kind of parameters. 2) SOM uses the neighbour function to update neighbour nodes while updating the node with maximal response. In this way, the update implicitly embeds the concept of soft quantization. In contrast, k-means algorithm only identifies the nearest centroid for each input data point, and updates it with the corresponding data points. 3) SOM uses the stochastic gradient descent algorithm to learn the codewords, while k-means learns the centroids by a heuristic iterative algorithm, which is similar with the expectation-maximization algorithm for mixtures of Guassian distribution. Based on these reasons, we use SOM in our architecture directly.

3.3 Deep Supervised Quantization

The proposed deep supervised quantization aims to simultaneously learn feature representations from raw images and quantizer. To this end, we combine the Convolutional Neural Networks (CNN) and Self-Organizing Map (SOM) into one unified deep architecture.

3.3.1 Architecture. Fig. 3 shows our architecture including the feature extraction part and quantization part. Our CNN architecture for feature extraction part follows CNN-F model in [1] with a few modifications. We remove the last fully-connected layer and softmax layer, and add a new fully-connected layer (fc8). The node number in fc8 layer is set to twice the class number in the database. Overall, our feature extraction part contains five convolutional layers and three fully-connected layers.

After the feature extraction part, there are two branches in our architecture which is determined by our final objective. The first branch contains a fully-connected layer, in which the node number equals class number in the dataset. After that a softmax layer is used to compute the classification loss. The second branch considers the supervised quantization objective. In the second branch, there is a L2-normalization layer, which normalizes the length of input vector to a unit for the concern of using inner product based SOM. Then SOM is appended to the L2-normalization layer, which outputs the distance map to each codeword. Note that we use the black parallelogram to denote SOM, which is in the same structure with the common fully-connected layers in Fig. 3. The difference between SOM and common fully-connected layers lies in that the output of SOM is two-dimensional and has the topology-preserving property.

Note that there are two identical DeepSOMs in Fig. 3. They share the same parameters, structures, and weights. The output maps of the two SOMs are used to compute the supervised quantization objective.

3.3.2 Objective Function. In our final objective function, there are three losses, of which two are about classification error while one is related to the supervised quantization error.

The classification loss follows the traditional softmax classification loss, which is formulated as follows:

$$l(\mathbf{W}) = -\frac{1}{n}\left[\sum_{i=1}^{n}\sum_{j=1}^{c} 1(y_i = j)\log\frac{e^{\mathbf{W}_j \mathbf{x}_i^T}}{\sum_{l=1}^{c} e^{\mathbf{W}_l \mathbf{x}_i^T}}\right], \quad (2)$$

where $\mathbf{W}$ denotes the parameter of our architecture, and $1(\cdot)$ is the indicator function.

The quantization loss is oriented for the supervised quantization objective. The proposed supervised quantization objective aims to minimize the differences on the maps between similar image pairs, while maximizing the differences on the maps between dissimilar pairs. It is formulated as follows:

$$g(\mathbf{W},\mathbf{C}) = \frac{1}{k}\Bigg[\sum_{i=1}^{k}\Big\{1(y_{p(i,1)} = y_{p(i,2)})||\mathbf{U}_{p(i,1)} - \mathbf{U}_{p(i,2)}||_2^2 - \Big[1 - 1(y_{p(i,1)} = y_{p(i,2)})\Big]\,||\mathbf{U}_{p(i,1)} - \mathbf{U}_{p(i,2)}||_2^2\Big\}\Bigg], \tag{3}$$

where k denotes the number of image pairs, and $\mathbf{p} \in \mathbf{R}^{k\times 2}$ denotes the indices of pairs in the database. $\mathbf{U}$ is the output map of SOM, which measures the distances between the input sample and all the codewords in SOM. While using inner product in SOM, $\mathbf{U}_i = \mathbf{x}_i\mathbf{C}^T$, where $\mathbf{C} = [\mathbf{c}_1^T, \mathbf{c}_2^T, \cdots, \mathbf{c}_m^T]^T \in \mathbb{R}^{m\times d}$ denotes the weight matrix of SOM, and each of them denotes a codeword. Fig. 2 shows the output maps of randomly selected sample in each class in CIFAR-10 dataset after the learning process. The size of output map is 75×75. We linearly transform the value range of the output maps into $[0, 255]$. The black color denotes 255, while the white color denotes 0. From Fig. 2, we can find that different classes generate response at different regions on the map. On the other hand, there is nearly only one dark region in each map, which implies the codewords, whose corresponding nodes locate in the small dark region on the map, are similar.

The overall objective function is given as follows:

$$\begin{aligned}\min J(\mathbf{W},\mathbf{C}) &= \min l(\mathbf{W}) + \lambda g(\mathbf{W},\mathbf{C}) \\ &= \min_{\mathbf{W},\mathbf{C}} -\frac{1}{n}\left[\sum_{i=1}^{n}\sum_{j=1}^{c} 1(y_i = j)\log\frac{e^{\mathbf{W}_j\mathbf{x}_i^T}}{\sum_{l=1}^{c} e^{\mathbf{W}_l\mathbf{x}_i^T}}\right] \\ &+\frac{\lambda}{k}\Bigg[\sum_{i=1}^{k} 1(y_{p(i,1)} = y_{p(i,2)})||\mathbf{U}_{p(i,1)} - \mathbf{U}_{p(i,2)}||_2^2 \\ &- \Big[1 - 1(y_{p(i,1)} = y_{p(i,2)})\Big]\,||\mathbf{U}_{p(i,1)} - \mathbf{U}_{p(i,2)}||_2^2\Bigg],\end{aligned} \tag{4}$$

where λ is the parameter for controlling the relative importance of the two parts. In the experiments, we set $\lambda = 1.25e-6$.

3.3.3 Optimization. Our overall architecture contains two parameters to be optimized: the CNN parameter $\mathbf{W}$ and SOM parameter $\mathbf{C}$. We adopt an iterative method to train the whole architecture. First, we pretrain the feature extraction part of our architecture and the branch of classification (*pretrain_step*). Adopting the output of the feature extraction part, *i.e.*, the features extracted by the CNN, we train SOM individually (*train_som_step*). We call this simple process as DeepSOM-0 (*pretrain_step* + *train_som_step*). Then we initialize the overall architecture with the pretrained CNN and SOM. Since SOM is also optimized by stochastic gradient descent (SGD) method, the overall architecture in our method can be directly trained by SGD method (*train_all_step*). After the training process of the overall architecture, we use the feature extraction part, and retrain SOM (*retrain_som_step*). Then replacing the SOM layer in the architecture with the retrained SOM, we train the whole architecture again (*train_all_step*). These two steps (*i.e.*, *train_all_step* and *retrain_som_step*) are iteratively alternated until convergence. Usually, ten iterations guarantee to a good result. Fig. 4 shows the curve of mAP convergence with respect to the iteration number on CIFAR-10 dataset.

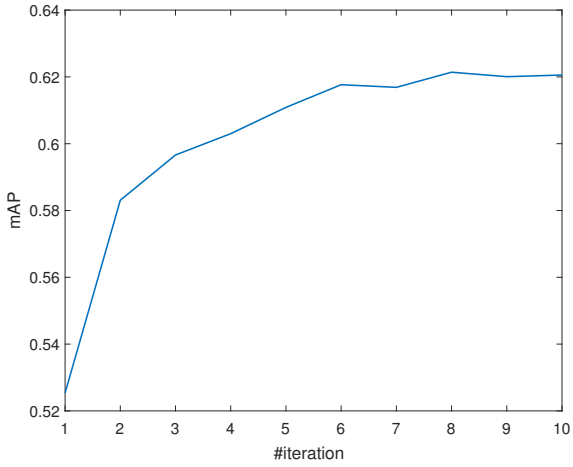

Figure 4: The curve of mAP convergence with respect to the iteration number on CIFAR-10 dataset.

In these two steps, only SGD algorithm is used. We give the key equations in different steps:

1) SOM Training (*train_som_step* and *retrain_som_step*): the codewords are updated using the neighbour function and the update rule as follows:

$$\mathbf{c}_j^{t+1} = \mathbf{c}_j^t + \alpha \cdot \theta(j,t) \cdot (\mathbf{x}_i \cdot \mathbf{c}_j^T)^T\mathbf{c}_j, \tag{5}$$

where we use the inner product based SOM, and t denotes the iteration number. $\theta(j,t)$ denotes the neighbourhood function for j-th node during the t-th iteration. α denotes the learning rate. We repeat Eqn. (5) until a fixed iteration number is reached.

2) DeepSOM Training (*train_all_step*): Since SOM is a neural network with single fully-connected layer, the combination of CNN and SOM is still a network, which also can be directly solved by SGD method. For the fully connected layer in the classification branch, its weights $\mathbf{W}_j$ only are updated by the classification loss. The gradient with respect to $\mathbf{W}_j$ is

$$\frac{\partial l(\mathbf{W}_j)}{\partial \mathbf{W}_j} = -\frac{1}{n}\sum_{i=1}^{n}\left[\left(1(y_i = j) - \frac{e^{\mathbf{W}_j\mathbf{x}_i^T}}{\sum_{l=1}^{c} e^{\mathbf{W}_l\mathbf{x}_i^T}}\right)^T \mathbf{x}_i\right]. \tag{6}$$

For the fully connected layer in SOM, the gradient with respect to its weights is

$$\frac{\partial g(\mathbf{W},\mathbf{C})}{\partial \mathbf{C}} = \frac{\lambda}{k}\Bigg[\sum_{i=1}^{k}\Big\{1(y_{p(i,1)} = y_{p(i,2)})\left[\left(\mathbf{x}_{p(i,1)} - \mathbf{x}_{p(i,2)}\right)\mathbf{C}^T\right]^T\mathbf{C} - \Big[1 - 1(y_{p(i,1)} = y_{p(i,2)})\Big]\left[\left(\mathbf{x}_{p(i,1)} - \mathbf{x}_{p(i,2)}\right)\mathbf{C}^T\right]^T\mathbf{C}\Big\}\Bigg]. \tag{7}$$

Table 1: Comparison on mAP performance for ANN search problem. For NUS-WIDE, we calculate the mAP values within the top 5000 returned results. All the hashing methods are evaluated under 24 bits. #query:1K/2100, #training:5K/21*500

Methods	CIFAR-10	NUS-WIDE
CNNH [24]	0.521	0.630
NINH [11]	0.566	0.697
DeepSOM-0	0.505	0.680
DeepSOM	**0.622**	**0.730**

In the experiments, the pairs of images are randomly selected in each epoch. The number of pairs of similar images equals the number of dissimilar pairs. As for the layers in the feature extraction part, their weights can be updated by the chain rule.

3.3.4 Out-of-Sample Extension. After the optimization, we obtain the whole deep supervised quantization architecture. In the online ANN search, we only need to use the feature extraction part and the second branch (SOM branch), as the green box shows in Fig. 3. For a query, the output of the architecture is a map, on which each value denotes the inner product to the corresponding codeword in SOM. We can directly identify the maximum from the map and obtain index of the node with maximal response on the map. In other words, the query is quantized to the most similar codeword based on the inner product computation. The distances between the query and database images is approximated by the distances between the corresponding codewords. The distances between all pairs of the codewords can be computed in advance and saved in a look-up table. Then through looking up table and ranking all the images in the database according to the distances, the nearest neighbours of the query are identified.

4 EXPERIMENTS

In this section, we conduct experiments on ANN search problem, classification, and visualization tasks.

We evaluate our model on several public available datasets: MNIST [12], CIFAR-10 [9], and NUS-WIDE [2].

The MNIST dataset is a set of handwritten digits from '0' to '9' with 70,000 images in total. The training set contains 60,000 examples, and the test set contains 10,000 examples. Each example in MNIST is a 28×28 grey-level image. The CIFAR-10 dataset consists of 60,000 32×32 color images in 10 classes, with 6000 images per class.

In the NUS-WIDE dataset, there are 269,648 images collected from Flickr. It is a multi-label dataset, in which each image is annotated with one or more labels from 81 concepts. Following [11, 13, 24], we only use the images associated with the 21 most frequent concepts, where the number of images associated with each concept is at least 5000. We resize each image into 32×32 to save memory.

In all the experiments, we set the size of SOM to 75×75. The initial neighbour radius is 65, and the number of iteration in training SOM is 5000. In the training process of the whole architecture, the batch size is 40. The whole architecture is realized based on MatConvNet [19]. Although our DeepSOM method adopts an alternative method for training, on a machine with one NVIDIA K40c GPU, and one Intel Xeon CPU E5-2620@2.40GH, it can be trained within 12 hours.

4.1 Evaluation on ANN search

For ANN search problem, since our DeepSOM is supervised, we compare it with several state-of-the-art supervised deep hashing methods including CNNH [24], NINH [11], DSRH [28], DRSCH [26], DPSH [13], DSH [15], and one existing supervised quantization method, *i.e.*, Supervised Quantization (SQ) [22]. Following these methods, we conduct the experiments on MNIST, CIFAR-10 and NUS-WIDE datasets to evaluate the performance of ANN search. We use the exactly same database setting (including the image numbers for query set, database, and training set) with the compared methods on these datasets, and directly compare our DeepSOM method with the experimental results published in their papers. Since different existing ANN search methods conduct their experiments under the different database setting, for fair comparison, we make experiments under all the settings used in these paper, and show the results in Table 1 , Table 2 and Table 3, respectively. We use mAP (mean Average Precision) to evaluate the performance of ANN search.

Table 1 shows the results of mAP on CIFAR-10 and NUS-WIDE datasets compared with CNNH [24] and NINH [11]. Following CNNH and NINH, we randomly select 1000 images (100 images per class) as the query set in CIFAR-10. And 5000 images (500 images per class) from the rest images are randomly selected as training set. In NUS-WIDE, 2100 query images from the 21 most frequent labels (100 images per class) are randomly selected as query set. 500 images per class are randomly selected from the rest images as training set. Since it is a multi-label dataset, two images will be considered to be similar if they share at least one concept.

On NUS-WIDE, mAP value is computed within the top 5000 returned results. From Table 1, we can find our DeepSOM outperforms these two compared deep supervised hashing methods. "DeepSOM-0" denotes a simple way to train our architecture. In DeepSOM-0, we first use the feature extraction part and the classification branch to pretrain CNN (*pretrain_step*), then use the pretrained feature extraction part and the SOM branch to train SOM (*train_som_step*). DeepSOM-0 is trained without the iterative training process on the whole architecture like DeepSOM. Through comparing the results of DeepSOM and DeepSOM-0 in Table 1, we can find the iterative training process (*i.e.*, *train_all_step* and *retrain_som_step*) does improve the performance of ANN search.

In Table 2, we compare our DeepSOM method with other deep supervised hashing method under different dataset setting. In CIFAR-10, following the setting in [13, 15, 26, 28], we randomly select 10,000 images (1000 images per class) as query set and use the rest images as training set. For NUS-WIDE dataset, we randomly sample 2100 images from the 21 most frequent classes (100 images per class) as query set, and use the rest images as training set. The mAP value is computed within the top 50,000 returned images in NUS-WIDE dataset. From Table 2, we can find our DeepSOM performs better than those compared methods under this experiment setting. Our DeepSOM improves the performance of ANN search by 9.9% and

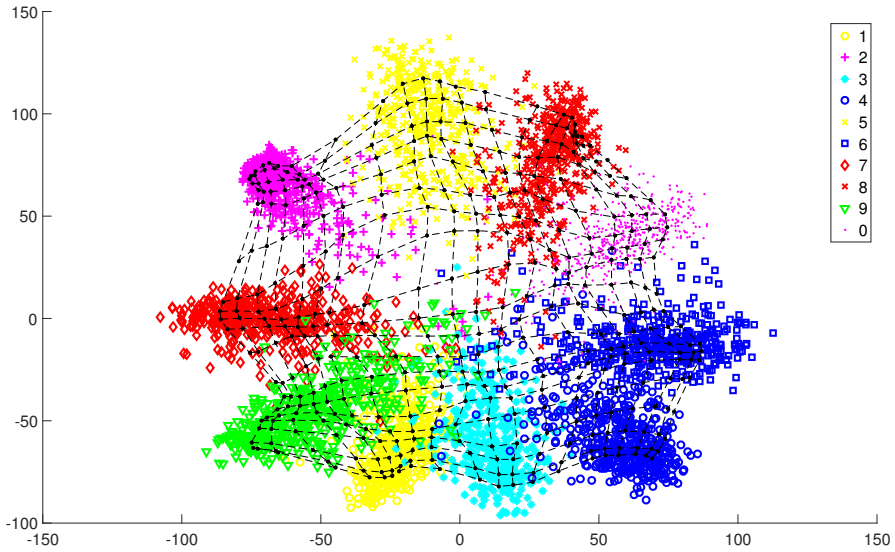

Figure 5: Visualization for MNIST dataset. The black points and dark lines denote the learned nodes and their links in SOM.

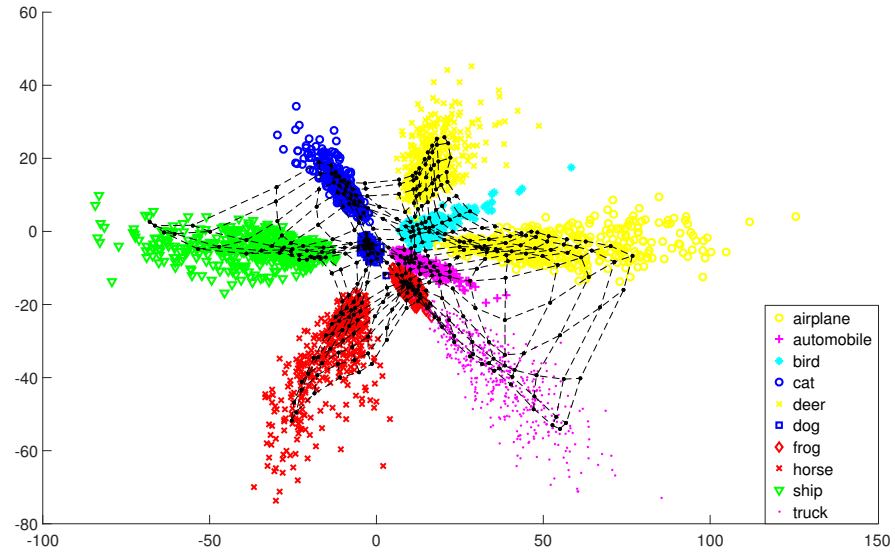

Figure 6: Visualization for CIFAR-10 dataset. The black points and dark lines denote the learned nodes and their links in SOM.

4.7% over the existing best results on CIFAR-10 and NUS-WIDE datasets, respectively.

To compare with supervised quantization method proposed in [22], we make experiments on MNIST and CIFAR-10 dataset. The numbers of images in query set, database and training set exactly follow the setting in [22]. For both of MNIST dataset and CIFAR-10 dataset, we randomly select 1000 images as query set, and use the rest dataset as training set. From Table 3, we can find that both our DeepSOM and DeepSOM-0 obtain the higher mAP value than SQ method even in 128 bits. These results show the capability of deep architecture for feature extraction in ANN search. Simultaneously learning feature description and quantizer is better than conducting these two steps separately.

Table 2: Comparison on mAP on the public datasets for ANN search problem. For NUS-WIDE, we calculate the MAP values within the top 50000 returned neighbors. All the hashing methods are evaluated under 24 bits. #query:10K/2100, #training:50K/the rest

Methods	CIFAR-10	NUS-WIDE
DSRH [28]	0.611	0.618
DRSCH [26]	0.622	0.622
DPSH [13]	0.781	0.722
DSH [15]	0.651	0.551
DeepSOM-0	0.762	0.691
DeepSOM	**0.868**	**0.723**

Table 3: Comparison on mAP on the public datasets for ANN search problem. The SQ method is evaluated under 16, 32, 64 and 128 bits. #query:1K/1K, #training:69K/59K

Methods	MNIST	CIFAR-10
SQ-16 bits [22]	0.933	0.605
SQ-32 bits [22]	0.937	0.686
SQ-64 bits [22]	0.938	0.704
SQ-128 bits [22]	0.940	0.712
DeepSOM-0	0.957	0.760
DeepSOM	**0.967**	**0.900**

Table 4: Comparison on classification accuracy (%). #query:10K/10K, #training:5K/50K

Methods	MNIST	CIFAR-10
[4]	98.80	73.70
[14]	N/A	81.05
DeepSOM-0	98.28	91.00
DeepSOM	**98.90**	**91.00**

4.2 Evaluation on Classification

In this section, we compare our DeepSOM method with deep clustering methods on the classification task. We do not pursue the optimal results on the dataset, but instead to compare the differences between our method and deep clustering methods.

For MNIST dataset, we randomly select 10,000 images as query set, and 5000 images as training set (for [4], it uses 6000 image as training set). In CIFAR-10 dataset, we randomly select 10,000 images as query set, and the rest images as training set. The results on the accuracy of classification are listed in Table 4. From this table, we can see that our DeepSOM method performs better on MNIST and CIFAR-10 datasets.

4.3 Visualization Study

In this section, we present an interesting application on visualization of our DeepSOM method. Since our method is targeted at simultaneously minimizing the classification error, and minimizing the supervised quantization objective, the proposed supervised objective is to minimize the differences on the map between similar image pairs, and maximize the differences on the map between dissimilar image pairs. It keeps the classification information in the process of quantization. So when the output of fc8 layer in our architecture is in a two-dimensional space, we can expect that using this architecture visualizes the dataset.[2]

Fig. 5 and Fig. 6 show the results of visualization for MNIST and CIFAR-10 datasets, respectively. Both of them visualize 5000 samples from the datasets. Different colors and markers denote the feature description from fc8 layer for different semantic classes. The black markers and lines denote the quantized codewords in SOM learned by our DeepSOM method and their locations on the map. In this experiment, the size of SOM is set to 20×20 for clear visualization. These two figures show the capability of our method on visualization. On the other hand, these two figures prove that the images which locate at a small region in the input space will activate the same node on SOM, and the nodes which locate closely on SOM share the same semantic label. This reflects the effect of the supervised quantization objective in our DeepSOM method.

5 CONCLUSIONS

In this paper, we propose a new deep supervised quantization method by Self-Organizing Map (SOM) for Approximate Nearest Neighbour search problem. Our method combines the Convolutional Neural Networks (CNN) and Self-Organizing Map into a unified deep architecture, which is abbreviated as DeepSOM in this paper. Our DeepSOM method simultaneously optimizes the supervised quantization objective and minimizes the classification error. The proposed supervised quantization objective is to minimize the differences on the map between similar image pairs, and maximize the differences on the map between dissimilar image pairs. By optimization, the deep architecture can simultaneously extract deep features and quantize the features into the suitable node in Self-Organizing Map. The experiments on several public standard datasets prove the superiority of our approach over the existing ANN search methods. Besides, our deep architecture shows great potential to classification and visualization tasks in the experiments.

6 ACKNOWLEDGEMENTS

This work was supported in part to Dr. Houqiang Li by 973 Program under contract No. 2015CB351803, NSFC under contract No. 61325009 and No. 61390514, in part to Dr. Wengang Zhou by NSFC under contract No. 61472378 and No. 61632019, and the Fundamental Research Funds for the Central Universities, and in part to Dr. Qi Tian by ARO grant W911NF-15-1-0290 and Faculty Research Gift Awards by NEC Laboratories of America and Blippar. This work was supported in part by NSFC under contract No. 61429201.

[2]In the experiments in this section, we remove the L2-norm layer in the architecture to visualization. If we keep this layer, all the data point will distribute on a unit circle.

REFERENCES

[1] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531* (2014).

[2] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. 2009. NUS-WIDE: a real-world web image database from National University of Singapore. In *ACM International Conference on Image and Video Retrieval*. 48.

[3] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. 2013. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 12 (2013), 2916–2929.

[4] Yen-Chang Hsu and Zsolt Kira. 2015. Neural network-based clustering using pairwise constraints. *arXiv preprint arXiv:1511.06321* (2015).

[5] Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. 2016. Deep Image Category Discovery using a Transferred Similarity Function. *arXiv preprint arXiv:1612.01253* (2016).

[6] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2011. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 1 (2011), 117–128.

[7] Teuvo Kohonen. 1982. Self-organized formation of topologically correct feature maps. *Biological Cybernetics* 43, 1 (1982), 59–69.

[8] Teuvo Kohonen and Timo Honkela. 2007. Kohonen network. *Scholarpedia* 2, 1 (2007), 1568.

[9] Alex Krizhevsky and Geoffrey Hinton. 2009. Learning multiple layers of features from tiny images. *Technical report, University of Toronto* (2009).

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.

[11] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan. 2015. Simultaneous feature learning and hash coding with deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*. 3270–3278.

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[13] Wu-Jun Li, Sheng Wang, and Wang-Cheng Kang. 2015. Feature learning based deep supervised hashing with pairwise labels. *arXiv preprint arXiv:1511.03855* (2015).

[14] Renjie Liao, Alex Schwing, Richard Zemel, and Raquel Urtasun. 2016. Learning deep parsimonious representations. In *Advances in Neural Information Processing Systems*. 5076–5084.

[15] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. 2016. Deep supervised hashing for fast image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2064–2072.

[16] Zhen Liu, Houqiang Li, Wengang Zhou, Ruizhen Zhao, and Qi Tian. 2014. Contextual hashing for large-scale image search. *IEEE Transactions on Image Processing* 23, 4 (2014), 1606–1614.

[17] Mohammad Norouzi and David J Fleet. 2013. Cartesian k-means. In *IEEE Conference on Computer Vision and Pattern Recognition*. 3017–3024.

[18] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229* (2013).

[19] Andrea Vedaldi and Karel Lenc. 2015. Matconvnet: Convolutional neural networks for matlab. In *ACM International Conference on Multimedia*. ACM, 689–692.

[20] Jianfeng Wang, Jingdong Wang, Nenghai Yu, and Shipeng Li. 2013. Order preserving hashing for approximate nearest neighbor search. In *ACM International Conference on Multimedia*. 133–142.

[21] Min Wang, Wengang Zhou, Qi Tian, Zhengjun Zha, and Houqiang Li. 2016. Linear Distance Preserving Pseudo-Supervised and Unsupervised Hashing. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 1257–1266.

[22] Xiaojuan Wang, Ting Zhang, Guo-Jun Qi, Jinhui Tang, and Jingdong Wang. 2016. Supervised quantization for similarity search. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2018–2026.

[23] Yair Weiss, Antonio Torralba, and Rob Fergus. 2009. Spectral hashing. In *Advances in Neural Information Processing Systems*. 1753–1760.

[24] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. 2014. Supervised Hashing for Image Retrieval via Image Representation Learning.. In *Association for the Advancement of Artificial Intelligence*, Vol. 1. 2.

[25] Lei Zhang, Yongdong Zhang, Jinhui Tang, Xiaoguang Gu, Jintao Li, and Qi Tian. 2013. Topology preserving hashing for similarity search. In *ACM International Conference on Multimedia*. 123–132.

[26] Ruimao Zhang, Liang Lin, Rui Zhang, Wangmeng Zuo, and Lei Zhang. 2015. Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *IEEE Transactions on Image Processing* 24, 12 (2015), 4766–4779.

[27] Ting Zhang, Chao Du, and Jingdong Wang. 2014. Composite Quantization for Approximate Nearest Neighbor Search. In *International Conference on Machine Learning*. 838–846.

[28] Fang Zhao, Yongzhen Huang, Liang Wang, and Tieniu Tan. 2015. Deep semantic ranking based hashing for multi-label image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*. 1556–1564.

[29] Wengang Zhou, Houqiang Li, Richang Hong, Yijuan Lu, and Qi Tian. 2015. BSIFT: Toward data-independent codebook for large scale image search. *IEEE Transactions on Image Processing* 24, 3 (2015), 967–979.

[30] Wengang Zhou, Yijuan Lu, Houqiang Li, and Qi Tian. 2012. Scalar quantization for large scale image search. In *ACM International Conference on Multimedia*. 169–178.

[31] Wengang Zhou, Ming Yang, Houqiang Li, Xiaoyu Wang, Yuanqing Lin, and Qi Tian. 2014. Towards codebook-free: Scalable cascaded hashing for mobile image search. *IEEE Transactions on Multimedia* 16, 3 (2014), 601–611.

[32] Wengang Zhou, Ming Yang, Xiaoyu Wang, Houqiang Li, Yuanqing Lin, and Qi Tian. 2016. Scalable feature matching by dual cascaded scalar quantization for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 38, 1 (2016), 159–171.