学习紧凑二进制码的深度哈希

Deep Hashing for Compact Binary Codes Learning

Venice Erin Liong, Jiwen Lu, Gang Wang, Pierre Moulin, and Jie Zhou
Advanced Digital Sciences Center, Singapore
School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
Department of ECE, University of Illinois at Urbana-Champaign, IL USA
Department of Automation, Tsinghua University, Beijing, China
venice.l@adsc.com.sg; jiwen.lu@adsc.com.sg; wanggang@ntu.edu.sg;
moulin@ifp.uiuc.edu;jzhou@tsinghua.edu.cn

CVPR 2015

摘要

在本文中，我们提出了一个新的深度哈希（DH）方法来学习用于大规模可视化搜索的紧凑二进制码。与大多数现有的二进制码学习方式（用单一的线性投影将每个样本投影到 0-1 向量上）不同，我们设计了一个深度神经网络来训练多层次的非线性变换以便学习这些二进制码，从而使样本的非线性关系可以得到很好的利用。我们的模型受到深度网络的三个约束：1）原始实值特征和所学到的二进制码之间的差异最小，2）二进制码的每一位分布均匀，3）不同位尽可能独立。为了进一步提高二进制码的辨别度，我们将 DH 扩展至监督型的 DH（SDH）。这可以通过在目标函数中加入辨别项来做到。同时，最大化类间差别最小化类内差异就可以学习出合适的二进制码。实验结果表明，所提出的方法的优越于现有的先进算法。

1. 简介

近几年来，大规模可视化搜索凭借其广泛的应用前景受到了计算机视觉的高度重视[4]。散列（哈希）是一种用于大规模视觉搜索的强大技术，各种以哈希为基础的方法已经被提出[6，7，20，34，36]。这些方法的基本思路是构造一系列散列函数将每个可视对象映射成一个二进制特征向量，同时保证视觉上相似的样本将被映射到相似的二进制码上。

基于散列的方法可分为两类：与数据无关的[1，3，11，23]和数据相关的[5，6，8，15，21，34]。对于第一类，先将样本随机投影到一个特征空间，然后进行二值化处理。这一类的代表性方法有局部性敏感哈希（LSH）[1]和它的核化或判别化扩展[11，16，23]。第二类方法使用各种统计学习技术来学习哈希函数，再用这些哈希函数将样本映射成二进制码。该类别下最先进的方法包括频谱哈希[36]，二进制重建嵌入（BRE）[15]，迭代量化（ITQ）[6]，k-means 哈希（KMH）[8]，最小损失哈希（MLH）[21]，以及序列投影学习哈希(SPLH) [34]。然而，大这些数据相关的哈希方法大多数不能很好捕获样品的非线性组合结构。虽然提出了一些核哈希方法[7,19]，但它们在可扩展性上具有一些问题。

在本文中，我们提出了一个新的深度哈希（DH）方法来学习用于大规模可视化搜索的紧凑二进制码。图 1 说明了该方法的基本思想。与大多数现有的二进制码学习方式（用单一的线性投影将每个样本投影到 0-1 向量上）不同，我们设计了一个深度神经网络来训练多层次的非线性变换以便学习这些二进制码，从而使样本的非线性关系可以得到很好的利用。我们的模型受到深度网络的三个约束：1）原始实值特征和所学到的二进制码之间的差异最小，2）二进制码的每一位分布均匀，3）不同位尽可能独立。为了进一步提高二进制码的辨别度，我们将 DH 扩展至监督型的 DH（SDH）。这可以通过在目标函数中加入辨别项来做到。同时，最大化类间差别最小化类内差异就可以学习出合适的二进制码。实验结果表明，所提出的方法的优越于现有的先进算法。

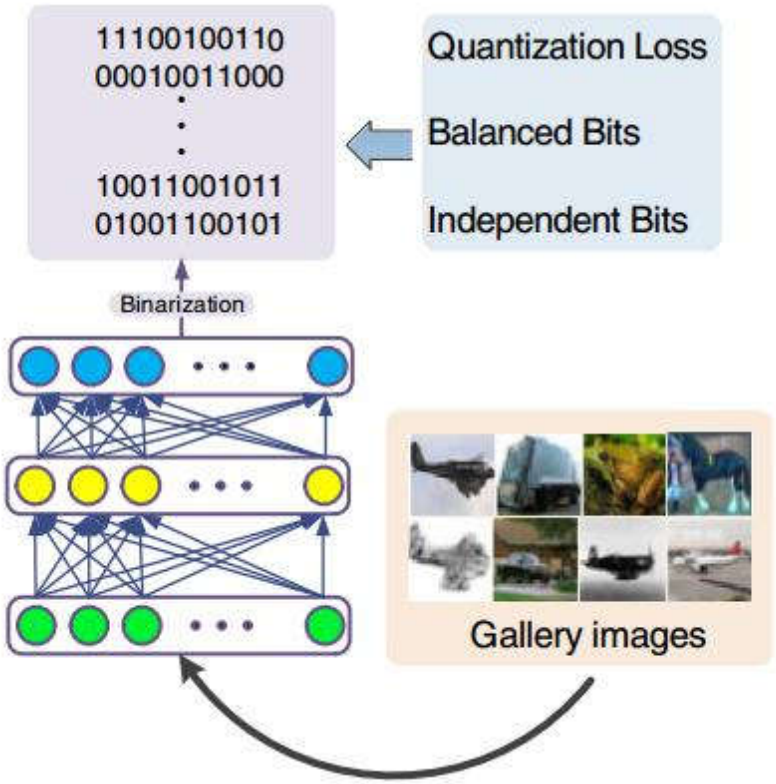


图 1 所提出的学习紧凑二进制码方法的基本想法

2. 相关工作

基于学习的哈希：现有基于学习的哈希方法可分为三大类：无监督、半监督和监督型。无监督哈希方法在学习过程中不需要训练集含有标签信息。例如，Weiss 等人[36]提出了一种谱哈希方法通过求解一个光谱图划分问题来获得平衡的二进制码。Gong 等人[6]提出了同时最大化每个二进制位的方差和最小化二值化损失的 ITQ 方法。He 等人[8]提出了通过最小化单一点和聚类中心之间的汉明距离的 KMH 方法。Heo 等人[9]提出了通过最小化原始实值特征和二进制特征之间的球面距离的散列方法。半监督哈希使用了成对标签信息来学习散列函数。Wang 等人[33]设计了一种半监督哈希算法(SSH)，该算法最小化成对标记的训练

样本的经验误差，同时最大化标记和未标记训练样本的方差。Kulis 和 Darrell[15]提出了一种 BRE 方法，该方法最小化原始空间上的欧几里得距离和重构的汉明距离之间的误差。 Norouzi 和 Fleet[21]提出了一种 MLH 方法，该方法最大限度地减少了汉明距离和量化误差之间的损耗。监督型哈希算法要求在哈希函数学习中使用每个样品的类别标签。Stretcha 等人[27]提出了一种 LDA 哈希算法啊，该算法最小化类内变化同时最大化二进制码的类间变化。Rastegari 等人[25]提出了一种基于线性支持向量机的监督型哈希算法。虽然这些哈希方法已在许多应用中取得了相当不错的成绩，但是其中大部分通常要求找到一个单一的线性投影方式，而不能很好地捕捉到样本间的非线性关系。

深度学习：深度学习的目的是通过从原始数据中提取高层次特征，以学习特征的分层表示。近年来，在计算机视觉和机器学习领域人们已经提出了各种深度学习算法[2，10，12，18，17，24，30]，其中一些已经成功地应用于可视化分析中的图像分类[14]、实体检测[28]、动作识别[17]、人脸识别[29]和视觉跟踪[35]。深度学习的代表方法包括深度堆叠自动编码器[17]、深度卷积神经网络[12]、深度信念网络[10]。虽然深度学习已经在各种视觉应用中取得了巨大的成功，但是在以哈希为基础的大型视觉搜索领域中，深度学习进展甚微。据我们所知，语义哈希[26]是第一个采用深度学习技术的哈希算法。它采用堆叠的受限波尔兹曼机（RBM）来学习用于可视化搜索的紧凑二进制代码。然而，这种模式是复杂的，并且需要预先训练，因而在实际应用中效率不高。

3. 所提出的方法

在本节中，我们首先介绍了学习型散列法的一些基本知识，然后详细说明了所提出的 DH 和 SDH 方法。

3.1 哈希

设 $X=[x_1,x_2,\ldots,x_N]\in\Re^{d\times N}$ 是训练集中，包含 N 个样本，其中 $x_n\in\Re^d(1\le n\le N)$。学习型散列方法的目标是寻求多个哈希函数映射，使用这些函数令每个样本量化成一个紧凑的二进制码。假定有 K 个哈希函数，那么，该组哈希函数将每个 x_n 映射成位二进制码 $b_n=[b_{n1},\ldots,b_{nK}]\in\{-1,1\}^{K\times 1}$，其中 b_{nk} 计算如下：

$$\mathrm{b}_{nk}=f_k(\mathbf{x}_n)=\mathrm{sgn}\left(g_k(\mathbf{x}_n)\right)=\mathrm{sgn}(\mathrm{w}_k^T\mathbf{x}_n) \qquad (1)$$

其中 f_k 是第 k 个哈希函数，$w_k\in\Re^d$ 是它的投影矩阵；当 v>0 时，sgn(v)=1，否则为-1。 设 $W=[w_1,w_2,\ldots,w_K]\in\Re^{d\times K}$ 是投影矩阵。x_n 的映射可以计算为：$g(x_n)=W^Tx_n$，其可以进一步二值化以获得二进制码：

$$\mathbf{b}_n \quad = \quad \mathrm{sgn}(\mathbf{W}^T\mathbf{x}_n) \qquad (2)$$

近年来，各种学习型哈希方法已经被提出[5，6，8，15，21，34]，它们旨在学习不同的投影矩阵 W 和不同的目标函数。然而，大多数现有的哈希方法只是学习单个投影矩阵，它本质上是线性的，因而不能很好地捕捉样本间的非线性关系。另一方面，一些基于核的哈希方法已被提出[7,19]，但是可扩展性依然是个问题，因为这些核方法不能得到明确的非线性映射。在这项工作中，我们提出了一个深度学习方法，去学习多项式非线性变换，以获得紧凑的二进制码。

3.2 深度哈希

如图 1 所示，对于给定的样品 x_n，我们将它传递给包含非线性变换的多层网络以获得二进制码 b_n。假设我们的深度网络有 M+1 层，并且第 m 层有 p^m 个单元。对于给定的样品 x_n，第一层的输出为：$h_n = s(W^1 x_n + c^1)$，其中 $W^1 \in \Re^{p^1 \times d}$ 是第一层的投影矩阵。$c^1 \in \Re^{p^1}$ 是偏置，$s(\cdot)$ 是一个非线性激活函数。第一层的输出作为第二层的输入，所以 $h_n^2 = s(W^2 h_n^1 + c^2) \in \Re^{p^2}$，其中 $W^2 \in \Re^{p^2 \times p^1}$ 和 $c^2 \in \Re^{p^2}$ 分别第二层的投影矩阵和偏置矢量。同样，m 层的输出为：$h_n^m = s(W^m h_n^{m-1} + c^m)$，我们网络的顶层输出为：

$$g_{DH}(\mathbf{x}_n) = \mathbf{h}_n^M = s(\mathbf{W}^M \mathbf{h}_n^{M-1} + \mathbf{c}^M) \qquad (3)$$

现在，我们对 h^M 进行哈希操作获得二进制码：

$$\mathbf{b}_n = \mathrm{sgn}(\mathbf{h}_n^M) \qquad (4)$$

设 $B = [b_{n1}, \ldots, b_N] \in \{-1,1\}^{K \times N}$ 和 $H^m = [h_1^m, h_2^m, \ldots, h_N^m] \in \Re^{p^m \times N}$ 分别是二进制码向量的矩阵表示和网络第 m 层的输出，我们制定以下优化问题来学习网络参数：

$$\begin{aligned} \min_{\mathbf{W},\mathbf{c}} J &= J_1 - \lambda_1 J_2 + \lambda_2 J_3 + \lambda_3 J_4 \\ &= \frac{1}{2}\|\mathbf{B} - \mathbf{H}^M\|_F^2 - \frac{\lambda_1}{2N}\mathrm{tr}((\mathbf{H}^M \mathbf{H}^M)^T) \\ &+ \frac{\lambda_2}{2}\sum_{m=1}^{M}\|\mathbf{W}^m(\mathbf{W}^m)^T - \mathbf{I}\|_F^2 \\ &+ \frac{\lambda_3}{2}(\|\mathbf{W}^m\|_F^2 + \|\mathbf{c}^m\|_2^2) \qquad (5) \end{aligned}$$

第一项 J_1 的目的是最小化学习到的二进制码和原实值向量之间的量化损失。第二项 J_2 旨在最大化到二进制码的方差，以确保码位的平衡。第三项 J_3 强制些

投影矩阵相互正交。最后一项J_4是正则项，用以控制参数。$\lambda_1,\lambda_2,\lambda_3$三个参数起到平衡不同项的效果。

为了解决这个优化问题，我们采用随机梯度下降的方法来学习参数$\{W^m,c^m\}_{m=1}^M$。（5）中的目标函数相对于不同参数的梯度计算如下：

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{W}^m} &= \Delta^m(\mathbf{H}^{m-1})^T \\ &+ \lambda_2\mathbf{W}^m(\mathbf{W}^m(\mathbf{W}^m)^T-\mathbf{I})+\lambda_3\mathbf{W}^m \quad (6)\\ \frac{\partial J}{\partial \mathbf{c}^m} &= \Delta^m+\lambda_3\mathbf{c}^m \quad (7)\end{aligned}$$

其中：

$$\begin{aligned}\Delta^M &= (-(\mathbf{B}-\mathbf{H}^M)-\lambda_1\mathbf{H}^M)\odot s'(\mathbf{Z}^M) \quad (8)\\ \Delta^m &= ((\mathbf{W}_1^{m+1})^T\Delta^{m+1})\odot s'(\mathbf{Z}^m) \quad (9)\end{aligned}$$

这里，$Z^m=W^mH^{m-1}+c^m$。

参数可以用如下梯度下降公式进行更新，直至收敛：

$$\begin{aligned}\mathbf{W}^m &= \mathbf{W}^m-\eta\frac{\partial J}{\partial \mathbf{W}^m} \quad (10)\\ \mathbf{c}^m &= \mathbf{c}^m-\eta\frac{\partial J}{\partial \mathbf{c}^m} \quad (11)\end{aligned}$$

其中，η是步长。Algorithm1 描述了 DH 算法的详细实现过程。

Algorithm 1: DH

Input: Training set $\mathbf{X}$, network layer number M, learning rate η, iterative number R, parameters λ_1, λ_2 and λ_3, and convergence error ε.

Output: Parameters $\{\mathbf{W}^m, \mathbf{c}^m\}_{m=1}^{M}$.

Step 1 (Initialization):

Initialize $\mathbf{W}^1$ by getting the top p^1 eigenvectors from the covariance matrix.

Initialize $\{\mathbf{W}^m\}_{m=2}^{m=M} = \mathbf{I}^{p_{m-1} \times p_m}$ and $\{\mathbf{c}^m\}_{m=1}^{m=M} = \mathbf{1}^{p_m \times 1}$.

Step 2 (Optimization by back propagation):

for $r = 1, 2, \cdots, R$ **do**

  Set $\mathbf{H}^0 = \mathbf{X}$

  for $m = 1, 2, \cdots, M$ **do**

    Compute $\mathbf{H}^m$ using the deep networks from (3).

  end

  for $m = M, M-1, \cdots, 1$ **do**

    Obtain the gradients according to (6)-(7).

  end

  for $m = 1, 2, \cdots, M$ **do**

    Update $\mathbf{W}^m$ and $\mathbf{c}^m$ according to (10)-(11).

  end

  Calculate J_t using (5).

  If $r > 1$ and $|J_r - J_{r-1}| < \varepsilon$, go to **Return**.

end

Return: $\{\mathbf{W}^m, \mathbf{c}^m\}_{m=1}^{M}$.

3.3 有监督的深度哈希

由于 DH 是一种无监督的学习方法，我们无法通过训练样本的标签信息进一步提高其性能。在这一小节中，我们提出了一个有监督的深度哈希（SDH）方法。这是 DH 在有监督领域的扩展版本。对于每对训练样本(x_i, x_j)，我们已经知道它们是否来自于同一类。因此，我们可以从训练集中构建两个集合 S 和 D，它们代表了训练集中的正样例和负样例。然后，我们制定了 SDH 的优化问题，如下：

$$\begin{aligned}\arg\min_{\mathbf{W},\mathbf{c}} J &= \frac{1}{2}\|\mathbf{B} - \mathbf{H}^M\|_F^2 \\ &- \frac{\lambda_1}{2}(\mathrm{tr}(\frac{1}{N}\mathbf{H}^M(\mathbf{H}^M)^T) + \alpha\mathrm{tr}(\Sigma_B - \Sigma_W)) \\ &+ \frac{\lambda_2}{2}\sum_{m=1}^{M}\|\mathbf{W}^m(\mathbf{W}^m)^T - \mathbf{I}\|_F^2 \\ &+ \frac{\lambda_3}{2}\sum_{m=1}^{M}(\|\mathbf{W}^m\|_F^2 + \|\mathbf{c}^m\|_2^2) \qquad (12)\end{aligned}$$

其中，

$$\begin{aligned}\Sigma_W &= \frac{1}{N_S}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}}(\mathbf{h}_i^M-\mathbf{h}_j^M)(\mathbf{h}_i^M-\mathbf{h}_j^M)^T \\ &= \frac{1}{N_S}\mathrm{tr}((\mathbf{H}_{s1}^M-\mathbf{H}_{s2}^M)(\mathbf{H}_{s1}^M-\mathbf{H}_{s2}^M)^T) \quad (13)\\ \Sigma_B &= \frac{1}{N_D}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}}(\mathbf{h}_i^M-\mathbf{h}_j^M)(\mathbf{h}_i^M-\mathbf{h}_j^M)^T \\ &= \frac{1}{N_D}\mathrm{tr}((\mathbf{H}_{d1}^M-\mathbf{H}_{d2}^M)(\mathbf{H}_{d1}^M-\mathbf{H}_{d2}^M)^T) \quad (14)\end{aligned}$$

N_S 和 N_D 分别是邻居对和非邻居对的个数，$\{H_{s1}^m,H_{s2}^m\}_{m=1}^M$ 是 S 中样品的潜在表示，$\{H_{d1}^m,H_{d2}^m\}_{m=1}^M$ 是 D 中样品的潜在表示。J_2 的目标是最小化类内变化最大化类间变化，α是平衡这两个部分的参数。J_1,J_3,J_4 的目的与 DH 是一致的。

与 DH 类似，我们可以使用随机梯度下降的方法来学习 SDH 参数。（12）中的目标函数相对于这些参数的梯度的计算如下：

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{W}^m} &= \Delta^m\mathbf{H}^{m-1}+\frac{\alpha}{\lambda_1}(\Delta_{s1}^m\mathbf{H}_{s1}^{m-1} \\ &- \Delta_{s2}^m\mathbf{H}_{s2}^{m-1}-\Delta_{d1}^m\mathbf{H}_{d1}^{m-1} \\ &+ \Delta_{d2}^m\mathbf{H}_{d2}^{m-1})+\lambda_3\mathbf{W}^m \\ &+ \lambda_2\mathbf{W}^m(\mathbf{W}^m(\mathbf{W}^m)^T-\mathbf{I}) \quad (15)\\ \frac{\partial J}{\partial \mathbf{c}^m} &= \Delta^m+\frac{\alpha}{\lambda_1}(\Delta_{s1}^m-\Delta_{s2}^m \\ &- \Delta_{d1}^m+\Delta_{d2}^m)+\lambda_3\mathbf{c}^m \quad (16)\end{aligned}$$

其中，第一层的梯度计算如下：

$$\begin{aligned}\Delta_{s1}^M &= \frac{1}{N_S}(\mathbf{H}_{s1}^M-\mathbf{H}_{s2}^M)\odot s'(\mathbf{Z}_{s1}^M) \quad (17)\\ \Delta_{s2}^m &= \frac{1}{N_S}(\mathbf{H}_{s1}^M-\mathbf{H}_{s2}^M)\odot s'(\mathbf{Z}_{s1}^M) \quad (18)\\ \Delta_{d1}^M &= \frac{1}{N_D}(\mathbf{H}_{d1}^M-\mathbf{H}_{d2}^M)\odot s'(\mathbf{Z}_{d1}^M) \quad (19)\\ \Delta_{d2}^m &= \frac{1}{N_D}(\mathbf{H}_{d1}^M-\mathbf{H}_{d2}^M)\odot s'(\mathbf{Z}_{d1}^M) \quad (20)\end{aligned}$$

隐藏层的梯度计算如下：

$$\begin{aligned}\Delta_{s1}^m &= ((\mathbf{W}^{m+1})^T\Delta_{s1}^{m+1})\odot s'(\mathbf{Z}_{s1}^m) \quad (21)\\ \Delta_{s2}^m &= ((\mathbf{W}^{m+1})^T\Delta_{s2}^{m+1})\odot s'(\mathbf{Z}_{s2}^m) \quad (22)\\ \Delta_{d1}^m &= ((\mathbf{W}^{m+1})^T\Delta_{d1}^{m+1})\odot s'(\mathbf{Z}_{d1}^m) \quad (23)\\ \Delta_{d2}^m &= ((\mathbf{W}^{m+1})^T\Delta_{d2}^{m+1})\odot s'(\mathbf{Z}_{d2}^m) \quad (24)\end{aligned}$$

Algorithm 2 描述了 SDH 方法的实现过程。

Algorithm 2: SDH

Input: Training set $\mathbf{X}$, pairwise sample indices, network layer number M, learning rate η, iterative number R, parameter λ_1, λ_2, λ_3 and α, and convergence error ϵ.

Output: Parameters $\{\mathbf{W}^m, \mathbf{c}^m\}_{m=1}^{M}$.

Step 1 (Initialization):

Initialize $\mathbf{W}^1$ by getting the top p^1 eigenvectors from a semi-supervised hashing method in [33].

Initialize $\{\mathbf{W}^m\}_{m=2}^{m=M} = \mathbf{I}^{p_{m-1} \times p_m}$ and $\{\mathbf{c}^m\}_{m=1}^{m=M} = \mathbf{1}^{p_m \times 1}$.

Step 2 (Optimization by back propagation):

for $r = 1, 2, \cdots, R$ **do**

  Set $\mathbf{H}^0 = \mathbf{X}$, $\{\mathbf{H}_{s1}^0, \mathbf{H}_{s2}^0\}$ and $\{\mathbf{H}_{d1}^0, \mathbf{H}_{d2}^0\}$ for pairwise samples in $\mathcal{S}$ and $\mathcal{D}$ respectively from set $\mathbf{X}$

  for $m = 1, 2, \cdots, M$ **do**

    Compute $\mathbf{H}^m$, $\mathbf{H}_{s1}^m$, $\mathbf{H}_{s2}^m$, $\mathbf{H}_{d1}^m$, and $\mathbf{H}_{d2}^m$ using the deep networks.

  end

  for $m = M, M-1, \cdots, 1$ **do**

    Obtain the gradients according to (15)-(16).

  end

  for $m = 1, 2, \cdots, M$ **do**

    Update $\mathbf{W}^m$ and $\mathbf{c}^m$ according to (10)-(11).

  end

  Calculate J_t using (12).

  If $r > 1$ and $|J_r - J_{r-1}| < \varepsilon$, go to **Return**.

end

Return: $\{\mathbf{W}^m, \mathbf{c}^m\}_{m=1}^{M}$.

4. 实验

我们在三个广泛使用的数据集上进行实验，以评估所提出的 DH 和 SDH 方法，并同几个先进的哈希算法进行比较。以下描述实验和结果的详细情况。

4.1 CIFAR-10 上的实验结果

CIFAR-10 数据集[13]含有 10 个类，共计 60000 幅彩色图像。每个图像的尺寸为 32×32。采用与[34]相同的设置，我们随机取样 1000 个样本，每个类 100 幅，作为查询数据，用剩余的 59000 图像作为库集。每幅图像被表示为 512-D GIST 特征向量[22]。

对于 DH 和 SDH 方法，我们设置 M=2 训练了我们的三层深度模型，层数维度根据经验设定为[60→30→16]，[80→50→32]和[100→80→64]。这三组维度分

别对应于 16，32 和 64 位哈希码。参数λ1，λ2 和λ3 分别凭经验设定为 100，0.001 和 0.001。使用双曲正切函数作为我们模型中的非线性激活函数。参数α设定为 1。

我们将我们的方法与最先进的九种哈希方法进行比较，其中六个是无监督方法，其他三个是有监督的。六种无监督方法是 PCA-ITQ[6]，KMH[8]，Spherical [9]，SH[36]，PCAH[34]和 LSH[1]。三种监督方法是 SPLH[34]，MLH[21]，和 BRE[15]。对于这九种方法，我们采用由原作者提供的方法实现方式和默认参数。

我们使用了以下三种评价指标来衡量的不同方法的性能：1）平均精确度（MAP）：其计算的是 P-R 曲线下的区域面积，可以用来评估不同的散列算法的总体性能; 2）N 样本上的精确度：返回的头 N 个样本中真实相似图片所占的比例; 3）当汉明半径设置为 N 时的查找精度：将与查询样例相距 2 个汉明距离的所有图片作为查询结果，在这个结果上计算精确度。一个失败的查询表现为没有图片返回，精确度为 0。我们重复实验 10 次，并把平均值作为最终结果。

表 1 展示的是不同散列方法在 CIFAR-10 的数据集上的搜索结果。图 2 展示的是在 16、32 和 64 位上不同方法的 P-R 曲线。从图中可以看出，我们的 DH 法优于其他无监督哈希方法，我们的 SDH 优于其他监督型哈希方法。此外，我们还和语义哈希[26]进行了比较，它也使用了深度模型。我们方法的性能比语义哈希好很多，因为我们的模型不仅最大限度地降低了重建成本，还保证了每个哈希码的位平衡性和独立性。

图 3 给出了 CIFAR-10 上一些查询图像的返回结果，这是在 64 位码长设置下不同哈希方法返回的检索结果。可以看出，我们的 DH 和 SDH 方法具有更好的搜索性能，这是因为它有更高的语义关联性。

Method	Hamming ranking (mAP, %)			precision (%) @ sample = 500			precision (%) @ r=2	
	16	32	64	16	32	64	16	32
PCA-ITQ [6]	15.67	16.20	16.64	22.46	25.30	27.09	22.60	14.99
KMH [8]	13.59	13.93	14.46	20.28	21.97	22.80	22.08	5.72
Spherical [9]	13.98	14.58	15.38	20.13	22.33	25.19	20.96	12.50
SH [36]	12.55	12.42	12.56	18.83	19.72	20.16	18.52	**20.60**
Semantic [26]	12.95	14.09	13.89	14.79	17.87	18.27	11.49	13.78
PCAH [34]	12.91	12.60	12.10	18.89	19.35	18.73	21.29	2.68
LSH [1]	12.55	13.76	15.07	16.21	19.10	22.25	16.73	7.07
DH	**16.17**	**16.62**	**16.96**	**23.79**	**26.00**	**27.70**	**23.33**	15.77
SPLH [34]	17.61	20.20	20.98	25.32	29.43	32.22	23.05	30.47
MLH [21]	18.37	20.49	21.89	24.43	29.60	33.01	23.52	28.72
BRE [15]	14.42	15.14	15.88	20.68	22.86	25.14	20.89	20.29
SDH	**18.80**	**20.83**	**22.51**	**26.32**	**30.42**	**33.60**	**23.26**	**31.48**

表 1 不同散列方法在 CIFAR-10 的数据集上的搜索结果

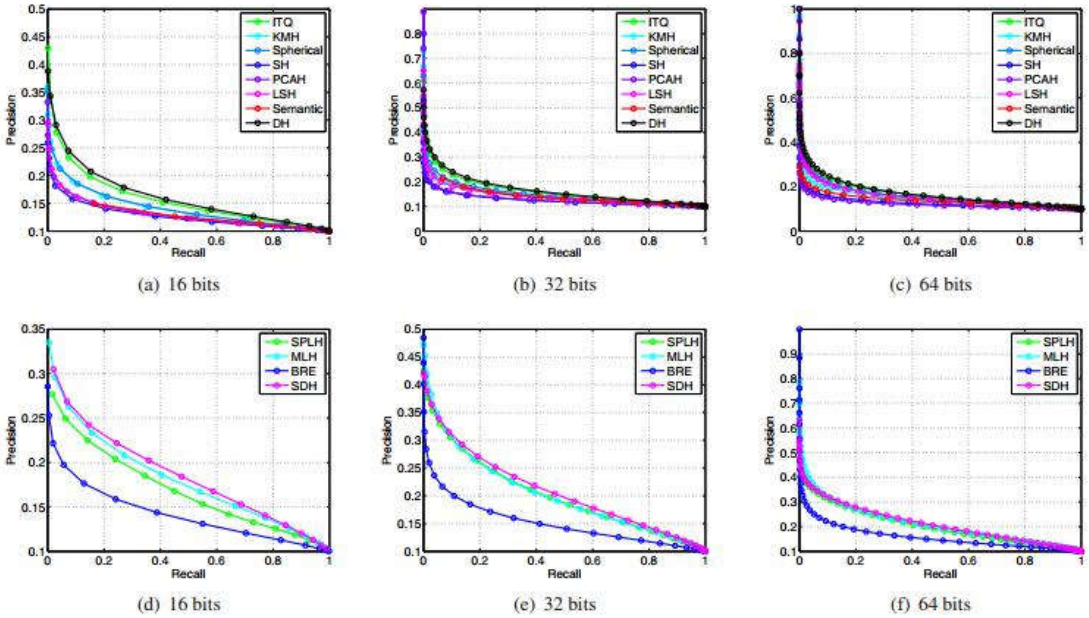


图 2 16、32 和 64 位码位下不同方法的 P-R 曲线

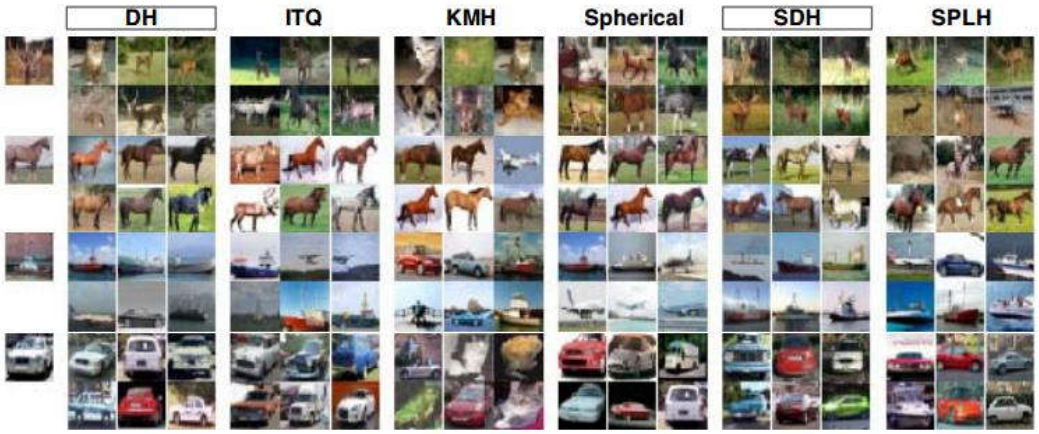


图 3 CIFAR-10 上不同方法查询图像的返回结果

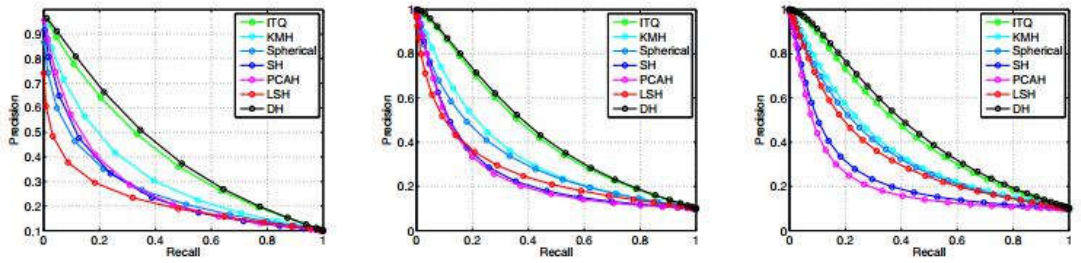
4.2 MNIST 上的实验结果

MNIST 数据集由 70000 幅 28*28 的灰色图像组成。每幅图都是 0-9 之间的一个手写体数字。我们随机取样 1000 个样本，每类 100 个，作为查询数据，用剩余的 69000 图像作为库集。每个图像被表示为一个 784-D 的灰度特征矢量。我们遵循与 CIFAR-10 集相同的设置，也采用相同的三个评价指标来比较不同的散列方法。表 2 展示的是不同哈希方法在数据集 MNIST 搜索结果。图 4 显示了 16，32 和 64 位上的 P-R 曲线。如图所示，我们的 DH 法优于其它无监督散列方法，我们的 SDH 优于其它有监督散列方法。

4.3 LabelMe22k 上的实验结果

LabelMe 数据集[32]由 22000 个实物图像组成，每个图像被表示为 512-D GIST 特征向量。对于每个样例，最多有 50 个语义邻居。采用与[15]相同的评价协议，我们随机抽样 2000 幅图像作为查询数据，使用其余 20000 图像作为库集。我们通过计算头 N 个返回结果的召回率以评估不同的方法，这种召回率这被定义为

检索到真实邻居在全部邻居中的占比。图 5 展示了不同散列方法在 LabelMe22k 数据集上召回率@ N 的结果。我们可以看出，和最先进的无监督哈希算法相比，DH 方法取得了非常有竞争力的结果；同时，SDH 优于有监督哈希方法。

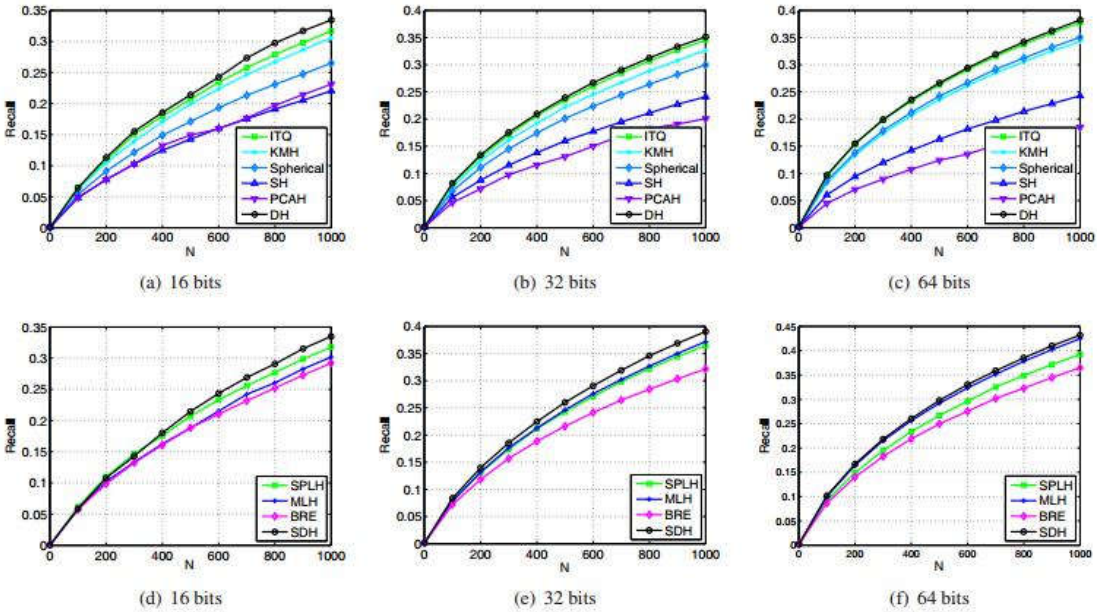


(a) 16 bits (b) 32 bits (c) 64 bits

图 4 16，32 和 64 位码位下不同方法的 P-R 曲线

Method	Hamming ranking (mAP, %)			precision (%) @ sample = 500			precision (%) @ r=2	
	16	32	64	16	32	64	16	32
PCA-ITQ [6]	41.18	43.82	45.37	66.39	74.04	77.42	65.73	73.14
KMH [8]	32.12	33.29	35.78	60.43	67.19	72.65	61.88	68.85
Spherical [9]	25.81	30.77	34.75	49.48	61.27	69.85	51.71	64.26
SH [36]	26.64	25.72	24.10	56.29	61.29	61.98	57.52	65.31
PCAH [34]	27.33	24.85	21.47	56.56	59.99	57.97	36.36	65.54
LSH [1]	20.88	25.83	31.71	37.77	50.16	61.73	25.10	55.61
DH	**43.14**	**44.97**	**46.74**	**67.89**	**74.72**	**78.63**	**66.10**	**73.29**
SPLH [34]	44.20	48.29	48.34	62.98	67.89	67.99	63.71	74.06
BRE [15]	33.34	35.09	36.80	60.72	68.86	**73.08**	34.09	64.21
SDH	**46.75**	**51.01**	**52.50**	**65.19**	**70.18**	72.33	**63.92**	**77.07**

表 2 MNIST 数据集上的实验结果



(a) 16 bits (b) 32 bits (c) 64 bits

(d) 16 bits (e) 32 bits (f) 64 bits

图 5 不同散列方法在 LabelMe22k 数据集上的 Recall@ N

Method	Recall @ $N = 1000$		
	16	32	64
PCA-ITQ [6]	31.29	34.54	37.97
KMH [8]	30.58	32.65	34.31
Spherical [9]	26.51	29.96	35.04
SH [36]	22.04	24.09	24.28
PCAH [34]	23.11	20.06	18.49
LSH [1]	22.01	28.22	32.42
DH	**32.60**	**35.51**	**38.42**
SPLH [34]	32.29	36.90	37.82
MLH [21]	32.05	37.21	36.66
BRE [15]	28.46	33.02	38.30
SDH	**33.79**	**39.10**	**42.98**

表 3 N 为 1000 时不同方法在 LabelMe22k 上的 Recall@N

Method	Training (seconds)	Test (microseconds)
PCA-ITQ [6]	5.0	2.6
KMH [8]	72.4	29.0
Spherical [9]	6.7	6.5
SH [36]	0.7	6.9
Semantic [26]	0.0023	43.1
PCAH [34]	0.7	0.06
LSH [1]	0.1	0.08
DH	27.5	4.1
SPLH [34]	8.0	2.5
MLH [21]	1770.0	2.5
BRE [15]	37.6	4.3
SDH	54.8	4.8

表 4 二进制码长设置为 16 时，不同哈希方法在 CIFAR-10 数据集上训练和测试时间

4.4 计算时间

最后，我们研究了所提出的 DH 和 SDH 方法的计算时间，并且与其他散列方法进行比较。我们的电脑配置了 3.40GHz 的 CPU 和 24.0 GB 的 RAM。表 4 示出了二进制码长设置为 16 时，不同哈希方法在 CIFAR-10 数据集上训练和测试时间，其中测试时间为计算每个查询图像所用的时间。我们看到，DH 的训练时间比之前的哈希方法长，不过测试时间还是不错的。

5. 总结

在本文中，我们提出了深度哈希（DH）和有监督的深度哈希（SDH）两种哈希算法。它们都可以用于紧凑型哈希码的学习。在三个广泛使用的数据集的实验结果表明了该方法的有效性。如何让我们提出的方法使用到视觉应用的其他方面，如实体识别和视觉追踪，将是未来一个有趣任务。

参考文献

[1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest

neighbor in high dimensions. In FOCS, pages 459–468, 2006.
[2] Y. Bengio. Learning deep architectures for AI. Foundations and trends○ R in Machine Learning, 2(1):1–127, 2009.
[3] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In BMVC, pages 812–815, 2008.
[4] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. ACM CSUR, 40(2):5, 2008.
[5] Y. Gong, S. Kumar, V. Verma, and S. Lazebnik. Angular quantization-based binary codes for fast similarity search. In NIPS, pages 1196–1204, 2012.
[6] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In CVPR, pages 817–824, 2011.
[7] J. He, W. Liu, and S.-F. Chang. Scalable similarity search with optimized kernel hashing. In KDD, pages 1129–1138, 2010.
[8] K. He, F. Wen, and J. Sun. K-means hashing: an affinitypreserving quantization method for learning binary compact codes. In CVPR, pages 2938–2945, 2013.
[9] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In CVPR, pages 2957–2964, 2012.
[10] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.
[11] P. Jain, B. Kulis, and K. Grauman. Fast image search for learned metrics. In CVPR, pages 1–8, 2008.
[12] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. PAMI, 35(1):221– 231, 2013.
[13] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, Univ. of Toronto, 2009.
[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, pages 1097–1105, 2012.
[15] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In NIPS, pages 1042–1050, 2009.
[16] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In ICCV, pages 2130– 2137, 2009.
[17] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In CVPR, pages 3361–3368, 2011.
[18] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In ACM, pages 609–616, 2009.
[19] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In CVPR, pages 2074–2081, 2012.
[20] X. Liu, J. He, B. Lang, and S.-F. Chang. Hash bit selection: a unified solution for selection problems in hashing. In CVPR, pages 1570–1577, 2013.
[21] M. Norouzi and D. M. Blei. Minimal loss hashing for compact binary codes. In ICML, pages 353–360, 2011.
[22] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic

representation of the spatial envelope. IJCV, 42(3):145–175, 2001.
[23] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In NIPS, pages 1509– 1517, 2009.
[24] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In CVPR, pages 1–8, 2007.
[25] M. Rastegari, A. Farhadi, and D. Forsyth. Attribute discovery via predictable discriminative binary codes. In ECCV, pages 876–889, 2012.
[26] R. Salakhutdinov and G. Hinton. Semantic hashing. IJAR, 50(7):969–978, 2009.
[27] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua. Ldahash: Improved matching with smaller descriptors. PAMI, 34(1):66–78, 2012.
[28] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In NIPS, pages 2553–2561, 2013.
[29] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In CVPR, pages 1701–1708, 2014.
[30] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In ECCV, pages 140–153, 2010.
[31] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. PAMI, 30(11):1958–1970, 2008.
[32] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In CVPR, pages 1–8, 2008.
[33] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In CVPR, pages 3424–3431, 2010.
[34] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large-scale search. PAMI, 34(12):2393–2406, 2012.
[35] N. Wang and D.-Y. Yeung. Learning a deep compact image representation for visual tracking. In NIPS, pages 809–817, 2013.
[36] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In NIPS, pages 1753–1760, 2008.