密级：_____

浙江大学

硕 士 学 位 论 文

论文题目 用于图像语义检索的深度哈希算法

作者姓名 任伟超

指导教师 陈刚 教授

学科(专业) 计算机技术

所在学院 计算机科学与技术学院

提交日期 2017 年 1 月

A Dissertation Submitted to Zhejiang University for the Degree of Master of Engineering

TITLE: Deep Hashing for Semantic Image Retrieval

Author: Ren Weichao

Supervisor: Prof. Chen Gang

Subject: Computer Application Technology

College: Computer Science and Technology

Submitted Date: 2017-01

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得 浙江大学 或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文作者签名：　　签字日期：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解 浙江大学 有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权 浙江大学 可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本授权书）

学位论文作者签名：　　导师签名：

签字日期：　　年　　月　　日　　签字日期：　　年　　月　　日

学位论文作者毕业后去向：

工作单位：　　电话：

通讯地址：　　邮编

摘要

随着互联网的飞速发展，网上的图像资源也在以爆炸的速度增长，为了在如此大规模的数据中根据不同用户各自的需求来高效迅速地检索相关的图像，哈希相关的检索算法应运而生。然而在现存的大多数传统哈希算法中图像特征提取和后续的哈希函数学习是割裂的，这使得特征无法与哈希函数学习很好地适配，影响了检索效果。基于深度学习的哈希算法的出现缓解了这一问题，但仍面临优化复杂难于训练，生成哈希码冗余性强等一系列问题。

在本文中，我们提出了基于深度学习的语义检索哈希算法（Deep Hashing for Semantic Retrieval，简称 DHSR），该方法通过使用卷积神经网络同时作为特征提取器和哈希函数学习器，实现了完全端到端的训练学习，将图像特征的提取和哈希函数的学习融入到同一个过程，使这两部分互相适配促进，并提出结合了样本点标签和样本对标签两种监督信息以及量化误差的优化目标，使得神经网络模型可以在学习较好的个体语义特征的基础上，生成保持样本对原始相对位置关系的哈希码。在此基础上，本文进一步提出了一种基于神经网络的复合哈希码层次检索策略，通过单次训练神经网络生成两组哈希码，结合哈希表查找和哈希码排序两种方法的优点，可根据实际情况灵活地使用提升检索准确率或缩短检索时间。另外，本文还采用分块全连接模块代替全连接部分，生成各位更加不相关的哈希码等策略，并通过模型融合以及微调训练的方式以及一系列优化技巧提升算法的稳定性和准确率。

大量实验结果证明，本文提出的 DHSR 方法在 MNIST，CIFAR-10 和 NUS-WIDE 三个公开图片数据集上都取得了比现有的哈希算法更好的效果。

关键词： 图像检索、哈希算法、深度学习、卷积神经网络

Abstract

With the rapid development of the Internet, online image resources are growing at an explosive rate. In order to retrieve relevant images efficiently according to the needs of different users in such a large-scale data, retrieval algorithms related to hashing came into being. However, in most of the existing hashing algorithms, the image feature extraction and subsequent hash function learning are isolated, which makes the feature can not be well adapted to the hash function learning, affecting the retrieval effectiveness. Some hashing methods based on deep learning alleviate this problem, but still face a series of problems such as the difficulty of optimization and training and the redundancy of generated hash codes.

In this paper, we propose a novel deep hashing method, called deep hashing for semantic retrieval (DHSR), to achieve complete end-to-end learning by using convolutional neural network as feature extractor and also hash function learner. In this way, image feature extraction and hash function learning are integrated into one stage, so that these two tasks can improve each other in the joint process. To this end, we design a loss function combining quantization error with two types of supervised information, including point-wise labels and pair-wise labels, in order that the convolutional neural network can generate hash codes which not only maintain the original relative position of samples, but also learn better individual semantic features. Furthermore, we devise a hierarchical retrieval strategy using compound hash codes based on the neural network. Specifically, we can generate two sets of hash codes with only one training session, and combine the advantages of hash table lookup and hash code ranking. It can be flexibly used to improve the retrieval accuracy or shorten the retrieval time according to the specific situation. In addition, we use the divide-and-encode module instead of fully-connected layer to generate less redundant hash codes, and adopt methods like ensemble and fine-tuning procedure, along with a series of techniques to improve the stability and accuracy of our method.

Extensive experiments on three large scale datasets MNIST, CIFAR-10 and

NUS-WIDE show that our DHSR method can outperform other state-of-the-art hashing methods in image retrieval applications.

Keywords： Image Retrieval, Hashing Methods, Deep Learning, Convolutional Neural Networks

目录

图目录

表目录

第1章 绪论

1.1 课题背景

近些年来，随着互联网的日常发展和普及，人们可以随时随地拍摄各种图片，并上传到网络上和朋友们分享，仅就国内外的社交网络平台 Facebook，微信，微博等，每天的总上传图片就数以亿计，这也使得在如此大规模的数据中根据不同用户各自的需求来检索相关的图像变得非常困难，也越发受到相关研究人员的广泛关注。

图像检索的核心在于根据用户给出的图像快速高效地从海量数据库中找到相似的图像，这里的相似有两层含义，既可以是图像看起来相似，也可以是图像语义上的相似，如图 1-1 所示。在实际应用中，基于语义相似性的检索应用较多，本文也重点关注图像的语义相似性。

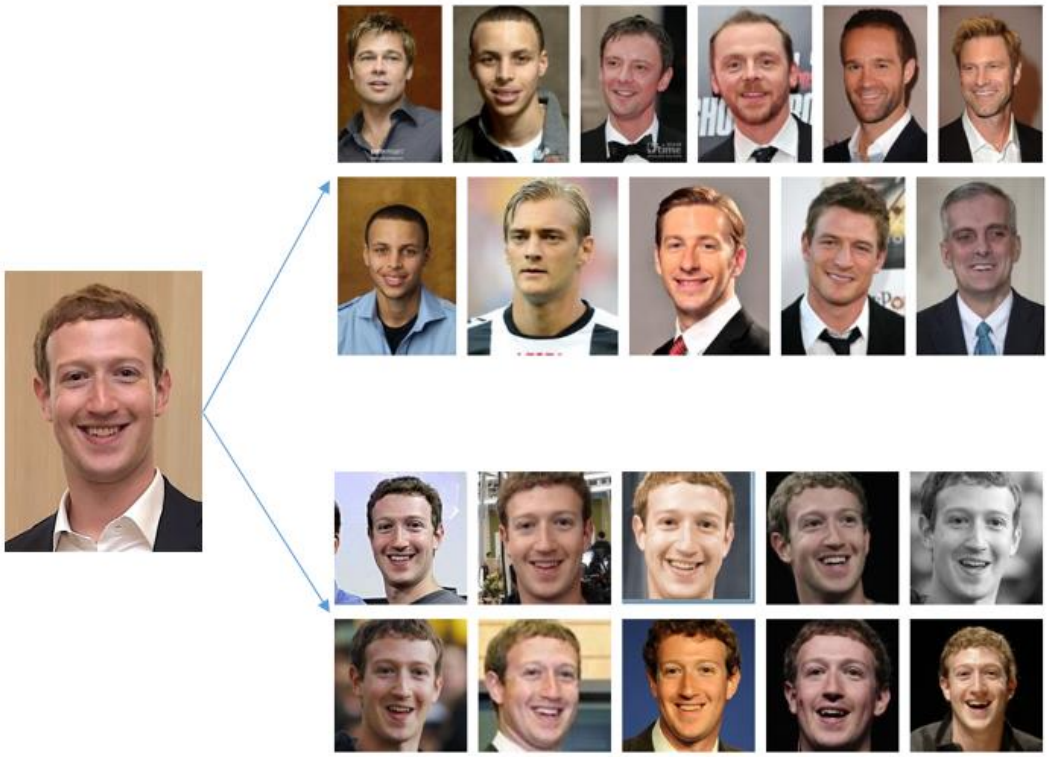

图 1-1 图片的外观相似和语义相似举例

假设所有数据库中的图像以及用户上传的图像都已经用实数特征向量表示，那么最直接的寻找相关图像的方法就是根据特征空间中与用户图像之间的距离来对数据库中所有图像排序，并返回距离最接近的结果。然而，对于当今常见的百万乃至上亿级别的图像数据库来说，上述解决最近邻问题的方法对整个数据库的线性穷举搜索，会占用过多的时间和空间。具体来说，如果使用长度 1000 维的单精度实数向量来表示每张图像，那么一百万张图像已经需要约 3.7G 的存储空间，而且计算每张图像和用户图像之间的距离，都需要 1000 次乘法和 2000 次加法运算，对一百万张图片都进行这样的计算会使得用户的等待时间过长。

为了解决这样的难题，人们提出了各种近似最近邻算法（Approximate Nearest Neighbor）来更高效地进行检索，哈希作为其中一种代表性方法，也引起了越来越多研究者的关注。哈希的目标通常是将数据样本映射成定长的二值编码，并且保证相似的样本具有相似的哈希编码。由于二进制编码的距离（汉明距离）可以通过异或位运算迅速完成，再加上更短的二进制码取代了之前的实数特征，节省了大量存储空间，哈希码在大规模图像检索中非常高效。

哈希算法可以分为两大类：数据无关方法和数据相关方法（也称为基于学习的哈希）。局部敏感哈希（Locality Sensitive Hashing）[1]是一种典型的数据无关的方法，使用简单的随机投影作为哈希函数，然而这样的方法通常需要很长的编码来实现满意的搜索准确率，因而会导致空间占用较大以及检索召回率低的问题[2]。基于学习的哈希则是一种数据相关的哈希算法，它通过对特定数据集学习得到哈希函数，使得哈希编码空间中的最近邻搜索结果与原始空间中的搜索结果尽可能接近，并减小时间和空间需求。此处原始空间中的相邻既可能是距离量度接近，也可能是指语义信息（分类）类似。

具体来说，基于学习的哈希方法又可以分为三类：无监督方法[3][4][5][6]，半监督方法[7][8]和监督方法[2][9]。无监督方法更加通用，只使用无标签数据学习哈希函数，然而它受限于语义鸿沟的难题[10]，即对象的高层的语义描述和低层的特征描述通常有很大区别。而半监督方法和监督方法通过把语义级别的信息（例如图像对是否相似的标签）与哈希学习过程相结合，减小语义鸿沟，可以用更少的位数

来实现更精确的检索，也更加适合用来进行语义相似性检索。

然而，现有的许多哈希算法中图像特征提取和后续的哈希函数学习是割裂的，即先对图像提取特征，再对所提取到的特征学习，这类方法的效果都非常依赖于使用的特征，由于这些特征往往都是通过无监督的方式提取的手工视觉描述符（例如 GIST[11]，HOG[12]），在这些特征空间中，距离远近并不能保证与图像语义是否相关等价，所以它们更适用于外观相似而不是语义相似搜索。因此，如何将特征提取和哈希函数的学习有机结合，将对检索系统的准确度提升产生重大影响。另一方面，深度学习以及卷积神经网络（CNN）已经在图像分类，对象识别，人脸识别等视觉任务上取得了重大的突破，并展现出了强大的特征学习能力。在不同的任务中，CNN 可以被看作是一个特征提取器，通过因任务而异的目标函数来指导 CNN 学习有用的特征。这些对 CNN 的成功应用都证明了 CNN 学习出的特征可以透过千变万化的外观形式捕捉到图像中深层的语义结构。

正因如此，许多研究者开始关注利用深度学习进行哈希的研究，这种方法利用 CNN 可以编码任何非线性哈希函数的强大能力，自动在哈希学习过程中学习充分保留语义相似性的特征表示，并且已经在很多数据集上达到了业界最好的效果。但即使如此，利用深度学习进行哈希的方法仍存在诸多问题，例如设计复杂，训练时间长，哈希函数之间不独立，未考虑连续值阈值化为二进制编码时的量化误差，有的仍需要分阶段，没能实现完全端到端的训练，有些则使用 sigmoid/tanh 等函数限制输出范围，导致模型训练缓慢。

综上所述，找到更为合理的方法将深度学习与哈希方法结合，对提升大规模图像检索场景下的效率和准确率至关重要。

1.2 本文工作及贡献

本文主要研究在图像检索领域，利用哈希算法实现在大规模样本检索的场景下，既保持较好准确率，又提升检索效率的具体方法。

具体来说，在综合调研相关研究工作的前提下，我们提出了一种基于深度学习的哈希方法，这种方法实现了完全端到端的训练学习，通过结合卷积神经网络的强大表达能力和特征学习能力，将对图像有用特征的提取以及哈希函数的学习

融入到同一个过程，使得两部分可以相互促进。另外，本文还结合了样本点以及样本对标签两种监督信息，与量化误差一起构成损失函数，一方面通过样本对标签直接学习与检索场景直接相关的相对距离问题，使得在哈希前后，不同样本对的基本位置和距离关系得到很好的保持；另一方面通过结合样本点标签，可以辅助 CNN 网络学习到更好的保持个体语义的特征，帮助提高模型的表达能力和检索性能。在此基础上，我们打破之前传统方法只使用哈希表查找或是哈希码排序其中一种的检索方式，综合两种检索方式的优点，并在神经网络的环境中，通过一次训练同时生成两组哈希码分别用于层次进行哈希表查找和哈希码排序。

综上所述，本文的主要贡献可以概括为以下几点：

1. 使用基于深度学习的哈希方法，实现了完全端到端的训练学习，对样本同时进行特征提取和哈希函数学习，两者可以相互促进。
2. 提出结合了样本点标签和样本对标签两种监督信息以及量化误差的优化目标，在学习到保持语义的个体特征的基础上，优化保持哈希前后样本对间的相对位置关系，直接提升语义检索的效果。
3. 提出了一种基于神经网络的复合哈希码层次检索策略，通过单次训练神经网络生成两组哈希码，并结合哈希表查找和哈希码排序的方法的优点，可根据实际情况灵活地使用提升检索准确率或减少检索时间。
4. 采用分块全连接模块代替全连接部分，生成各位更加不相关的哈希码等策略，并通过模型融合、微调训练等方式以及一系列优化技巧提升算法的稳定性和准确率。
5. 将本文算法应用于三个公开权威图片数据集，通过大量实验，与各种已有的传统哈希方法和基于深度学习的哈希方法对比，结果证明文中提出的 DHSR 方法相比现有方法，在准确率和检索时间方面都具有优势。

1.3 本文组织

本文共分为七章，每章的具体内容如下：

第一章为绪论，描述哈希算法的应用场景，并介绍它在图像检索领域的应用现状和前景，并指出当前相关工作的不足与问题，说明了本文针对这些问题的具

体工作和贡献，以及之后各章的组织结构。

第二章整理总结了目前已有的相关研究工作，分别介绍了各种经典传统哈希算法及其原理，深度神经网络和卷积神经网络的原理，最后介绍了将卷积神经网络结合入哈希学习算法过程中的相关算法，并说明了它们各自的弱点以及可能改进的方向。

第三章是对本文要解决问题的规范化定义和描述，包括文章中用到的符号的定义，待解决的核心问题的规范定义以及本文中提出的哈希方法预期要达到的效果。

第四章具体描述了本文提出的 DHSR 算法，从特征学习器 CNN，训练过程的损失函数与新样本的哈希码生成，复合哈希码层次检索策略三方面来说明算法的设计思路和合理性。

第五章介绍本文方法的实现细节和部分优化策略。

第六章展示了本文中的各项实验结果，通过将本方法在完全相同的评测指标，实验设定和测试环境下与其他已有哈希方法进行对比，以及整个方法中不同模块的分拆对比来充分证明本文实现方法的有效性。

第七章对全文工作进行总结，指出本文中提出方法的优缺点，并展望未来可以进一步探索研究的方向。

第2章 相关工作

2.1 哈希算法

2.1.1 最近邻检索

最近邻检索的定义是：从样本集合$\mathrm{X} = \{x_1, x_2, \ldots, x_N\}$中查找检索项 q 的最近邻项 NN(q)，满足：

$$\mathrm{NN(q)} = \arg\min_{\mathrm{x}\in\mathrm{X}} \mathrm{dist(q,x)} \qquad \text{(公式 2.1)}$$

其中dist(q,x)表示检索项 q 与样本集合中某项 x 的l_s范数距离，且有：

$$\mathrm{dist(q,x)} = \|\mathrm{x}-\mathrm{q}\|_\mathrm{s} = (\sum_{i=1}^{d}|x_i - q_i|^s)^{1/s} \qquad \text{(公式 2.2)}$$

在此基础上最直接的扩展是 K-最近邻检索，即需要找到 K 个而不是一个最近邻样本。

目前已有一些有效的算法对低维的数据进行最近邻检索，例如 KD 树算法[13]，通过对 d 维数据空间的分割，在特定空间的部分内进行相关检索以实现高效索引，在 KD 树之后，更是出现了 M 树[14]，cover 树[15]等树形结构进行最近邻检索，然而对于高维的情况，最近邻检索问题变得非常困难，而且大多数算法甚至会比最简单的线性搜索算法付出更大的计算代价。因此，近似最近邻检索（ANN）：$(1+\epsilon)$-最近邻检索[16]近来受到了很多关注。近似最近邻搜索关注的问题是：找到检索项 q 的近似最近邻 ANN(q) = p，且满足对任意$\mathrm{p}' \in \mathrm{X}$，都有：

$$\mathrm{dist(p,q)} \leq (1+\epsilon)\,\mathrm{dist(p',q)} \qquad \text{(公式 2.3)}$$

其他最近邻检索问题包括（近似）固定距离近邻问题，随机最近邻检索问题等则是局部敏感哈希算法相关研究者重点关注的问题。

定时近似最近邻检索目标是花费尽可能短的时间，并比较 K 个近似最近邻和 K 个真正的最近邻，并尽量提升检索的平均准确率。

2.1.2 哈希检索方法

哈希检索算法是解决最近邻问题的一种重要代表算法，它通过将被检索样本

和参照样本映射成目标样本，并借助于目标样本高效准确地进行近似最近邻检索，这样的目标样本就叫做哈希码。

哈希函数可以被规范地定义为：$y = h(x)$，其中$h(\cdot)$就是哈希函数，y 是哈希码。哈希码多为二进制值 1 和 0（或-1）。在近似最近邻检索的相关应用中，通常许多哈希函数会一起使用来计算复合哈希码：$y = h(x)$，其中$y = [y_1\ y_2 \dots\ y_M]^T$，$h(x) = [h_1(x)\ h_2(x) \dots\ h_M(x)]^T$。有两种基本的方法来使用哈希码进行最近邻检索：哈希表查找和哈希码排序。

2.1.2.1 哈希表查找

哈希表查找的核心思想是将距离计算的个数从N减少到$N'(N \gg N')$，此时时间复杂度就会从$O(Nd)$减小到$O(N'd)$，从而加速检索。此处用到的数据结构哈希表（一种形式的倒排索引）中的每一项都由一个哈希码索引，与传统的计算机科学中的哈希算法不同的是，这里的哈希表不避免冲突（将多个样本映射到同一个表项中），二是希望最大化相邻的样本发生冲突的概率。对于给定的检索项 q，哈希码$h(q)$对应的表项中的样本将成为检索 q 的最近邻样本的候选集，之后通常再对这一候选集中的样本，使用原始空间中的特征计算它们与 q 之间的真实距离，并据此排序得出 q 的 K 个最近邻。

上述方法在表项过多时可能由于冲突很少，造成召回率很低，有两种方法可以改善这种现象。第一种方法是再访问在哈希码空间当中距离$h(q)$最近的一些表项，如图 2-1[17]所示；第二种方法是构建更多个哈希表，这样在 L 个哈希表中的$h_1(q), \dots, h_L(q)$这 L 个表项中的所有样本共同作为 q 的最近邻样本的候选集，如图 2-2[17]所示。另一方面，为了保证准确率，L 个哈希码中的每一个都需要是一段比较长的编码。对比来看，第二种方法是实际上需要存储参照样本 id 的多个拷贝，因此占用的空间更多。第一种方法则只需要存储每个参照样本 id 的一个拷贝，因而占用空间更少，但它的缺点是需要访问更多的表项以达到和第二种方法同样的召回率。与之类似的还有一种多重赋值方法：只构建一个哈希表，但将同一个参照样本同时分配到多个哈希表项中去，其实上述的第二种方法也是多重赋值方法的一种特殊形式。

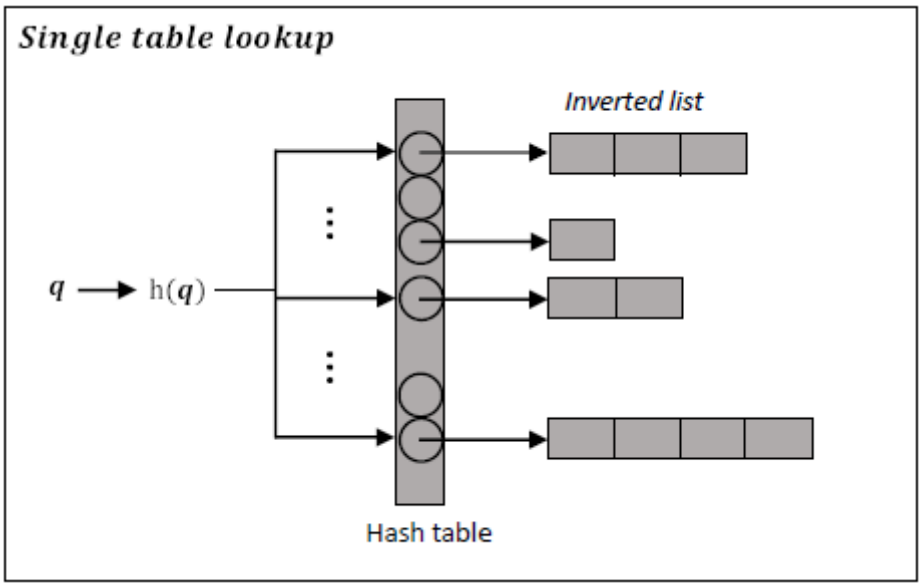

图 2-1 单哈希表查找

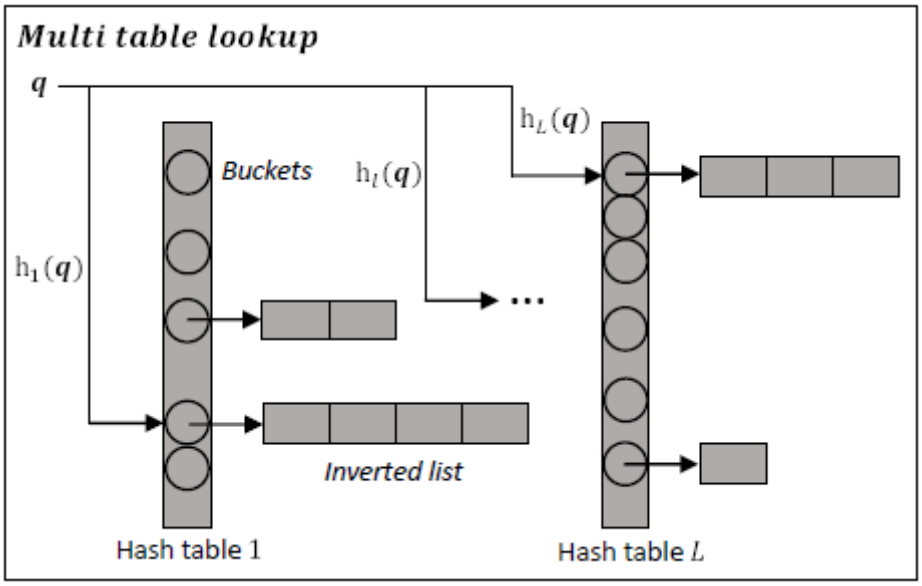

图 2-2 多哈希表查找

2.1.2.2 哈希码排序

哈希码排序算法进行的是穷举搜索，它的核心思想是通过使用哈希码，将每组距离计算的消耗从d减小到$d'(d \gg d')$，这样时间复杂度就会从$O(Nd)$减小到$O(Nd')$，从而加速检索。具体来说，这种算法根据检索项以及参照项的哈希码，通过使用距离表查询或汉明距离计算（可通过高效位异或操作实现）等方法，快

速得出检索项和参照样本之间的距离，并将其中距离最小的一部分参照项作为检索项 q 的最近邻样本的候选集。最后，与哈希表查找方法相似，我们对候选集中样本，使用原始空间特征计算它们与 q 之间的真实距离，并据此排序得出结果。过程示意图如图 2-3[17]所示。

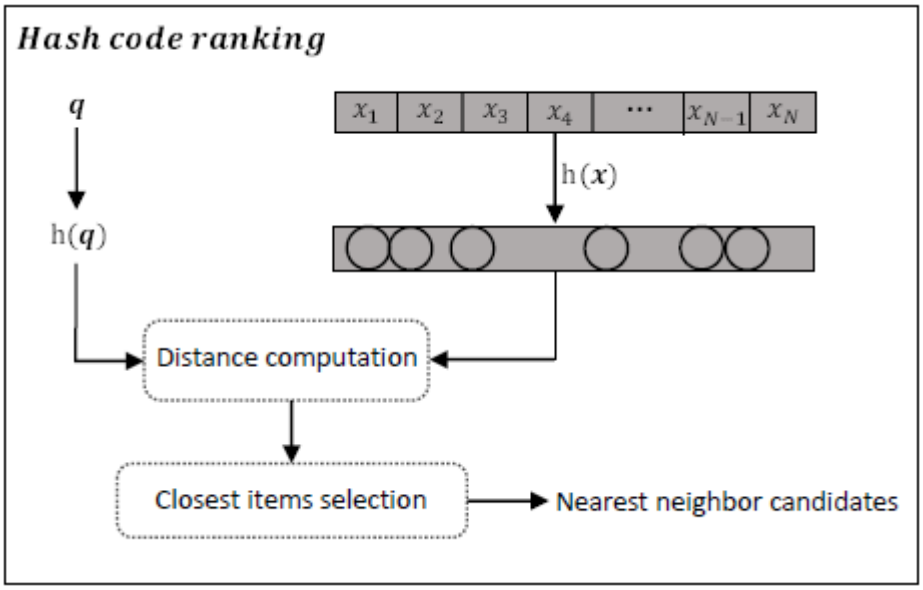

图 2-3 哈希码排序

2.1.2.3 小结

哈希表查询的方法主要在局部敏感哈希（LSH）中使用，也曾在一些文章中用来评价基于学习的哈希方法。已有相关研究证明，与哈希表查询方法相比，哈希码排序的方法比哈希表的查询的方法在搜索准确率上更好，但搜索效率更差，总体效果更好，所以被广泛应用于科研实验与实际应用中。

本文中正是提出了结合这两种方法的复合哈希码机制，综合利用两种基本方法的优势。

2.1.3 局部敏感哈希

局部敏感哈希（LSH）作为最典型的数据无关哈希算法，对后续一系列哈希算法的研究发展产生了深远的影响，它最早由 Indykzai[16]在 1998 年提出，之后在 1999 年，Gionis[1]对其进行了深入的分析证明和改进。2002 年，Charikar[18]提出了基于随机投影的随机哈希算法。该算法通过将样本投影到随机的超平面上来生

成二进制码，根据 Johnson-Lindenstrauss[19]定理产生特定分布的投影矩阵，再根据该矩阵进行线性变化，使得投影后的低维空间尽可能保持原始高维空间的位置信息[20]。理论上已证明局部敏感哈希算法中随着哈希码长度的增长，两个二进制哈希码的汉明距离渐进逼近他们在原始空间中的距离，但是这也造成为了使用该算法时为了达到满意的效果，往往要使用很长的哈希码，从而占据大量空间。近些年来，局部敏感哈希相关的研究主要集中在理论层面，包括提出对多种距离量度满足局部敏感性质的随机哈希函数[21][22][23][24][25][26]，证明更好的检索效率和准确率[23][27]，提出更好的搜索机制[28][29]，另外机器学习相关的相关研究者提出了对哈希函数的各种改进，包括可以提供小方差相似性估计[30][31][32][33]，更节省空间[34][35]，对哈希函数更快的计算[31][33][36][37]。

2.1.4 基于学习的哈希

2.1.4.1 概述

基于学习的哈希算法是数据相关的一类算法，它的任务是从特定数据集中学出适合该类数据的一个或一组哈希函数，使得对检索项的最近邻检索与真实的最近邻尽可能相似，另外由于它可以使用更少位数的哈希码来达到同样甚至更好的准确率，使得在哈希码空间的检索更加高效，同时在大样本数据集中可以节省大量空间，所以在实际应用中受到了比数据无关方法更多的关注。这里要学习的哈希函数可以是线性投影，核，球面函数，神经网络，无参数函数等。线性哈希函数是其中广泛使用的一种：

$$y = h(x) = sgn(w^T x + b) \quad \text{(公式 2.4)}$$

其中当$z \geq 0$时，$sgn(z) = 1$，否则$sgn(z) = 0$（或-1），w 是投影向量，b 是偏置项。

另外很常见的是核函数：

$$y = h(x) = sgn(\sum_{t=1}^{T} w_t K(s_t, x) + b) \quad \text{(公式 2.5)}$$

其中，$\{s_t\}$是从数据集中随机抽取的样本集或是数据集的聚类中心点，$\{w_t\}$是权重。

另外还有一种常见的哈希函数是基于最近向量赋值的无参数函数：

$$\mathrm{y} = \arg\min_{k\in\{1,\dots,K\}} \|x - c_k\|_2 \qquad \text{(公式 2.6)}$$

其中，$\{c_1,\dots,c_K\}$是由 K-means 等算法计算出的中心点的集合。与其他哈希算法不同的是，此处的哈希码是最近向量的下标，哈希码空间中的距离则是通过哈希码对应的中心点计算的。

哈希函数的选用会直接影响检索准确率和计算效率，线性函数计算更加高效，但是核函数和基于最近向量分配的方法具有更高的准确率且使用更为灵活。

2.1.4.2 无监督学习哈希算法

无监督的哈希算法不需要标签或者样本相似度等语义信息，只需提供样本的特征向量便可以从训练数据中学习哈希函数，并希望保留某种距离度量（例如l_2距离）下的近邻，常见的有学习最小化重建误差[6][38]和近邻保持的谱哈希（Spectral Hashing，简称 SH）[5]以及基于图的哈希算法锚点图哈希（Anchor Graph Hashing，简称 AGH）[5][39]，还有学习最小化量化误差的迭代量化法（Iterative Quantization，简称 ITQ）[3]，其他方法还有核局部敏感哈希（Kernelized Locality Sensitive Hashing，简称 KLSH）[4]，各向同性哈希（Isotropic Hashing，简称 IsoHash）[40]，可扩展图哈希（Scalable Graph Hashing，简称 SGH）[41]等。

SH 算法[5]最小化样本对间的加权汉明距离，这里的权重正是样本对的相似性度量，即它的目标函数为：

$$\min \sum_{(i,j)\in\gamma} s_{ij}^{o} d_{ij}^{h} \qquad \text{(公式 2.7)}$$

其中哈希空间中采用欧式距离$d_{ij}^{h} = \|y_i - y_j\|_2^2$以方便优化，原始空间中的相似度则定义为：

$$s_{ij}^{o} = \exp\left(-\frac{\|x_i - x_j\|_2^2}{2\sigma^2}\right) \qquad \text{(公式 2.8)}$$

将公式 2.7 写成矩阵形式即为：

$$\min \sum_{(i,j)\in\gamma} s_{ij}^{o} d_{ij}^{h} = trace(Y(D-S)Y^T) \qquad \text{(公式 2.9)}$$

其中$Y = [y_1\ y_2 \dots\ y_N]$是一个$M \times N$的矩阵，$[s_{ij}^o]_{N\times N}$是相似矩阵，$D = Diag(d_{11}, \dots, d_{NN})$是对角矩阵，其中$d_{nn} = \sum_{i=1}^{N} s_{ni}^o$。此问题有一个明显的解是$y_1 = y_2 = \cdots = y_N$，但明显这个解是没有意义的，为了避免这个解，SH 算法引入了编码平衡条件，即映射到每一个哈希码的数据样本个数要相同。位平衡（每一位都有约 50%的机会是 1 或-1）和位不相关（不同位之间不相关）可以用来近似编码平衡条件，这样两个条件可以记作：

$$Y1 = 0,\quad YY^T = I \qquad \text{(公式 2.10)}$$

其中 1 是 N 维的全 1 向量，I 是大小为 N 的单位矩阵。

由此得到的哈希算法步骤如下：

1. 使用主成分分析法（PCA）找到 N 维参照样本的主成分。
2. 在每个 PCA 方向上(共 d 个方向)计算 M 个一维的拉普拉斯特征函数(与 PCA 方向上最小的 M 个特征值对应)。
3. 从 Md 个特征函数中选择特征值最小的 M 个。
4. 对特征函数设置阈值为 0，得到二进制码。

该算法通过在给定数据的主成分分析（PCA）的方向上设置阈值来产生紧致（冗余度低）的二进制码，如此得到的哈希码除了距离相似性与原空间一致外，还是平衡且各分量间不相关的。而且文中给出了评价哈希函数的 3 个指标：

1. 保持原样本空间的相似性。
2. 需要的位数较少，生成紧致的哈希码（各函数间独立）。
3. 对新的输入易于求解哈希码。

但是，SH 算法由于可能在同一个 PCA 方向上选取超过一个特征函数，所以可能会破坏位不相关性，同时直接以 0 为阈值可能会导致接近阈值点的近邻被映射成不同的哈希值。

AGH 算法[39]使用锚点图来估计近邻图，并相应地用锚点的图拉普拉斯来近似原始图的图拉普拉斯，从而快速计算特征向量，并且利用分层哈希来解决边界问题。

ITQ 算法[3]本身可以用于无监督学习哈希算法和监督学习哈希算法，它将样本投影到二值超立方体顶点，并最小化这个过程中的量化错误，借此来减小由于原样本空间和映射后的二进制汉明空间之间偏差而造成的信息损失。它首先对数据进行预处理，使用主成分分析（PCA）方法将数据降至 M 维：

$$\mathrm{v} = \mathrm{P}^{T}\mathrm{x} \quad \text{(公式 2.11)}$$

其中 P 是根据 PCA 方法计算得到的$\mathrm{d} \times \mathrm{M}$的矩阵（$\mathrm{M} \ll \mathrm{d}$）。之后再找到一个最优旋转 R 并进行标量量化：

$$\min\|Y - R^{T}V\|_{F}^{2} \quad \text{(公式 2.12)}$$

其中 R 是$\mathrm{M} \times \mathrm{M}$的矩阵，$\mathrm{V} = [\mathrm{v}_1\ \mathrm{v}_2 \ldots \mathrm{v}_N]$，$\mathrm{Y} = [\mathrm{y}_1\ \mathrm{y}_2 \ldots \mathrm{y}_N]$。

这一问题通过交替优化的方式分两步解决，R 保持不变，$\mathrm{Y} = \mathrm{sign}(\mathrm{R}^{T}\mathrm{V})$，Y 保持不变，则问题变成了标准的正交 Procrustes 问题，解是$\mathrm{R} = \hat{\mathrm{S}}\mathrm{S}^{T}$，其中S和$\hat{\mathrm{S}}$通过矩阵$\mathrm{YV}^{T}$的奇异值分解（SVD）得到：

$$\mathrm{YV}^{T} = \mathrm{S}\Lambda\hat{\mathrm{S}}^{T} \quad \text{(公式 2.13)}$$

KLSH 算法[4]的提出是为了弥补 LSH 的不足，它在核函数对应的空间而不是原始空间中随机构造哈希函数，将 LSH 的思想推广到可以使用任意核函数。

IsoHash 算法[40]同样是要寻找经过 PCA 预处理之后的数据的一个旋转方向 R，不同的是，它限制矩阵$\Sigma = \mathrm{R}^{T}\mathrm{VV}^{T}\mathrm{R}$中对角线上的元素相等，即有$[\Sigma]_{11} = [\Sigma]_{22} = \cdots = [\Sigma]_{MM}$，它的目标函数是：

$$\|\mathrm{R}^{T}\mathrm{VV}^{T}\mathrm{R} - \mathrm{Z}\|_{F} = 0 \quad \text{(公式 2.14)}$$

其中矩阵 Z 的对角线上元素全部等于一个未知的变量σ，该问题可用投影和梯度流的方法解决。限制矩阵对角线上元素相等（即使得 M 个方向的方差相同）是为了使得哈希码中的每一位在距离度量中的贡献相等，在样本满足各向同性高斯分布的情况下，该算法的解与 ITQ 算法的解相同。

无监督学习虽然可以在没有语义标签或相关性信息时使用，通用性更好，但往往受限于语义鸿沟问题（高层语义与低层特征描述无法对应），无法实现很好的哈希检索效果。

2.1.4.3 监督学习哈希算法

监督学习的哈希方法（细分的话包括有监督学习哈希方法和半监督学习哈希方法）可以结合语义标签，相对相似度和相关性反馈等信息来减小语义鸿沟，因此可以使用更短的哈希码来达到比较高的准确率，编码更为紧致（Compact Hash Code）。通常来说，检索效果会随着编码长度的增加而提升，但是当长度超过一定阈值时，再增加长度甚至可能使结果变差。

最小损失哈希（Minimal Loss Hashing，简称 MLH）[42]和汉明距离度量学习（Hamming Distance Metric Learning）[43]都基于数据点间的相对相似度来最小化折叶损失（Hinge Loss）函数。MLH 方法中的损失函数是不连续的，无法直接优化，所以作者采用了类似于结构化 SVM 的方法，优化折叶损失的上界。它的优化目标函数与 SH 算法的形式相同（公式 2.7），其中如果(i,j)相似，那么$s_{ij}^{o}=1$，如果不相似则$s_{ij}^{o}=-1$。但距离则是以类折叶（Hinge-like）的形式定义的：

$$\mathrm{d}_{ij}^{h}=\begin{cases}\max(\left\|\mathrm{y}_i-\mathrm{y}_j\right\|_1+1,\rho), & (i,j)\text{相似}\\ \min(\left\|\mathrm{y}_i-\mathrm{y}_j\right\|_1-1,\rho), & (i,j)\text{不相似}\end{cases} \quad \text{(公式 2.15)}$$

在这样的损失函数中，相似样本对的汉明距离足够小或者不相似样本的汉明距离足够大，都不会得到任何惩罚。公式在ρ固定的情况下，相当于：

$$\mathrm{d}_{ij}^{h}=\min\sum_{(i,j)\in\gamma^{+}}max\left(\left\|\mathrm{y}_i-\mathrm{y}_j\right\|_1-\rho+1,0\right)+\sum_{(i,j)\in\gamma^{-}}\lambda max\left(\rho-\left\|\mathrm{y}_i-\mathrm{y}_j\right\|_1+1,0\right) \quad \text{(公式 2.16)}$$

其中超参数ρ是在哈希值后的汉明空间中区分相似样本对和不相似样本对的阈值，超参数λ则是控制由相似样本对（或不相似样本对）引起的比例，哈希函数是线性形式的$\mathrm{y}=\mathrm{sgn}(\mathrm{W}^{T}\mathrm{x})$，投影矩阵 W 通过使用结合隐变量的结构化预测来进行优化，超参数ρ和λ都通过交叉验证选出。

此外，典型关联分析迭代量化法（CCA-ITQ）[3]是 ITQ 方法的一种扩展，使用标签信息寻找最佳投影方向。可预测的区别二进制码（Predictable Discriminative Binary Codes，简称 DBC）[44]寻找能以大间隔区分不同类别样本的超平面作为哈

希函数。半监督哈希（Semi-Supervised Hashing，简称 SSH）[45]利用大量无标签数据来对哈希函数正则化。上述方法都将线性投影作为哈希函数，因此无法很好地解决不线性可分数据的情况。

监督核哈希（Supervised Hashing with Kernels，简称 KSH）[2]是一种基于核的方法，可以处理样本不线性可分的情况，它通过最小化相似样本对间的汉明距离，最大化不相似样本对间的汉明距离来生成紧实的二进制码。而与其他监督学习方法不同的是，它通过最小化哈希码的内积来学习，并证明了此做法相当于最小化汉明距离，使问题得到简化。它的优化目标函数是：

$$\min \sum_{(i,j)\in\gamma}\left(s_{ij}^{o} - s_{ij}^{h}\right)^2 \quad \text{(公式 2.17)}$$

$$= \min \sum_{(i,j)\in\gamma}\left(s_{ij}^{o} - \frac{1}{M}y_i^T y_j\right)^2 \quad \text{(公式 2.18)}$$

其中如果(i,j)相似，那么$s_{ij}^{o}=1$，如果不相似则$s_{ij}^{o}=-1$，y = h(x)是核哈希函数。

如果我们展开公式 2.17，可以进一步得到：

$$\min \sum_{(i,j)\in\gamma}\left(s_{ij}^{o} - s_{ij}^{h}\right)^2 \quad \text{(公式 2.17)}$$

$$= \min \sum_{(i,j)\in\gamma}\left(\left(s_{ij}^{\mathrm{o}}\right)^2 + \left(s_{ij}^{\mathrm{h}}\right)^2 - 2s_{ij}^{o}s_{ij}^{h}\right) \quad \text{(公式 2.19)}$$

$$= \min \sum_{(i,j)\in\gamma}\left(\left(s_{ij}^{\mathrm{h}}\right)^2 - 2s_{ij}^{o}s_{ij}^{h}\right) \quad \text{(公式 2.20)}$$

由此可见，相似度间误差最小化（即公式 2.17）和相似度间的乘积最大化的区别就在于$\min \sum_{(i,j)\in\gamma}\left(s_{ij}^{\mathrm{h}}\right)^2$这一项，即最小化哈希空间中样本点间的相似度，即使得哈希码尽可能地不相同，这一项可以看作对相似度最大化项$\max \sum_{(i,j)\in\gamma} s_{ij}^{\mathrm{o}}s_{ij}^{\mathrm{h}}$补充的正则化项，可以帮助模型避开所有样本哈希码全相同这样明显无意义的解。

二值重建嵌入（Binary Reconstruction Embedding，简称 BRE）[9]目标是最小化距离间误差，它通过最小化数据点间距离和相应哈希码距离的二次误差来学习哈希函数，并使用坐标梯度下降发迭代更新哈希函数以得到局部最优解。

具体来说，它的优化目标是：

$$\min \sum_{(i,j)\in\gamma}\left(d_{ij}^{o} - d_{ij}^{h}\right)^2 \quad \text{(公式 2.21)}$$

使用核哈希函数可得：

$$y_{nm} = h_m(x) = sgn(\sum_{t=1}^{T_m} w_{mt}K(s_{mt}, x)) \quad \text{(公式 2.22)}$$

其中$\{s_{mt}\}_{t=1}^{T_m}$是抽样样本集，$K(\cdot,\cdot)$是哈希函数，$\{w_{mt}\}$是要学习的权重。

BRE 算法在优化过程中没有采用松弛或者直接去掉 sgn 函数（见 2.1.4.4），而是使用了一个交替优化的两步机制：保持除了某个权重w_{mt}之外的其他权重不变，并对于w_{mt}来优化公式 2.21 中的目标函数，该方法的时间复杂度为O(NlogN + $|\gamma|$)。

如果我们展开公式 2.21，可以进一步得到：

$$\min \sum_{(i,j)\in\gamma}\left(d_{ij}^{o} - d_{ij}^{h}\right)^2 \quad \text{(公式 2.21)}$$

$$= \min \sum_{(i,j)\in\gamma}\left(\left(d_{ij}^{o}\right)^2 + \left(d_{ij}^{h}\right)^2 - 2d_{ij}^{o}d_{ij}^{h}\right) \quad \text{(公式 2.23)}$$

$$= \min \sum_{(i,j)\in\gamma}\left(\left(d_{ij}^{h}\right)^2 - 2d_{ij}^{o}d_{ij}^{h}\right) \quad \text{(公式 2.24)}$$

由此可见，距离的乘积最大化项$\max \sum_{(i,j)\in\gamma} d_{ij}^{o}d_{ij}^{h}$的目的是使得原始空间距离大的样本在哈希空间中距离也大，而距离间误差最小化（即公式 2.21）和距离的乘积最大化的区别就在于$\min \sum_{(i,j)\in\gamma}\left(d_{ij}^{h}\right)^2$这一项，即最小化哈希空间中样本点间的距离，这一项可以看作对乘积最大化项$\max \sum_{(i,j)\in\gamma} d_{ij}^{o}d_{ij}^{h}$补充的正则化项。

另外，如果根据监督信息的形式，有监督的哈希学习算法还可以归类为样本点标签，样本对标签和排序标签等监督信息。使用样本点标签的包括 CCA-ITQ[3]和监督离散哈希（Supervised discrete hashing，简称 SDH）[46]；使用样本对标签的包括 MLH[42]，KSH[2]以及最近提出的序列投影学习哈希（Sequential Projection Learning for Hashing，简称 SPLH）[8]，两阶段哈希（Two-Step Hashing，简称 TSH）[47]，快速有监督哈希（Fast Supervised Hashing，简称 FastH）[48]，隐因子哈希（Latent Factor Hashing，简称 LFH）[49]和基于列采样的离散有监督哈希（Column Sampling

based Discrete Supervised Hashing，简称 COSDISH）[50]等；使用排序标签的包括基于排序的有监督哈希（Ranking-based Supervised Hashing，简称 RSH）[51]，列生成哈希（Column Genaration Hash，简称 CGHash）[52]，排序保持哈希（Ranking Preserving Hashing，简称 RPH）[53]和保序哈希（Order Preserving Hashing，简称 OPH）[54]。

OPH 算法[54]是学习保序的哈希函数，即使编码空间和原始空间中的排序对齐。对于样本x_n来说，整个数据库γ被分为 M 个类目，（C_{n0}^h，$C_{n1}^h,\dots,C_{nM}^h$），其中C_{nm}^h对应的是与被检索项距离为 m 的样本，而（C_{n0}^o，$C_{n1}^o,\dots,C_{nM}^o$）则是在原始空间中的距离对应的 M 个类目，它是根据将样本分配到任何哈希码的概率相同这一理想情况构造的。它的目标函数最大化两种类目的顺序对齐情况：

$$\sum_{n\epsilon\{1,\dots,N\}}\sum_{m=0}^{M}\left(\left|C_{nm}^o - C_{nm}^h\right| + \left|C_{nm}^h - C_{nm}^o\right|\right) \quad \text{(公式 2.25)}$$

其中$|C_{nm}^o - C_{nm}^h|$是两个集合的差集的基数，算法中使用线性哈希函数h(x)并且忽略了符号函数 sgn 以简便优化过程。

监督学习的方法往往能通过学习额外的语义级别的信息获得更好的检索效果，但是通常需要一个很大的系数矩阵来存储训练集中数据点的相似性。

2.1.4.4 优化中的松弛条件

哈希函数因为最终要产生离散的二进制码，但是如公式 2.4 和公式 2.5 中的 sgn 函数这样的离散值会导致优化求解的困难，常见的方法是连续性松弛，即将离散约束去掉，当作实数值进行优化，之后再对模型的输出进行量化来产生二进制码，例如 sigmoid 松弛：

$$\mathrm{sgn}(z) \approx \emptyset_\alpha(z) = \frac{1}{1+e^{-\alpha z}} \quad \text{(公式 2.26)}$$

另一种则是直接去掉符号函数sgn(z) ≈ z，还有一种较为复杂的做法是两步机制[47][48]，反复迭代进行两阶段优化[55]：1）在不考虑哈希函数的情况下优化二进制哈希码；2）通过优化后的哈希码估计函数的参数。

然而在实际情况中，我们并不能保证将优化目标松弛为实数之后求得的最优解，在量化成为离散值之后仍是最优的，所以离散图哈希 DGH（无监督方法）[56]

和监督离散哈希 SDH（有监督方法）[46]被提出用于直接针对二进制码进行优化，以此避免松弛条件带来的缺点，从而达到更好的检索效果。

2.2 深度神经网络

2.2.1 深度学习概述

深度学习作为机器学习中的一个新兴领域，通过模拟人脑的工作机制，建立神经网络来模拟人类处理图像，音频和文本等的能力。与传统的非深层常规机器学习方法相比，它的核心思想是通过层次的网络结构，组合低层特征构成抽象的高层表示和概念，并发现数据的分布式表征（Distributed Representation）。分布式表征是神经网络和深度学习研究的核心课题，具体来说，神经网络中神经元和高层概念是多对多的关系，某一概念可由多个神经元共同表达，一个神经元也可以表达多个不同的概念，这种方式与传统的局部表示方式相比存储效率增大很多，可以表达指数级别增加的概念数。另外这种对数据自动学习的分层抽象表示，可以显著减少特征工程的工作，并能广泛地应用到多个领域中。

深度学习的概念最早起源于对神经网络的研究，神经网络是一种模拟大脑进行感知和认知的结构，最早研究见于[57]，该文中提出生物神经元作为大脑中神将完了过的子节点，可以类比成二值逻辑门，变量信号到达树突之后，当输入信号在细胞体内超过阈值就会激活输出信号。1957 年，Frank 提出了感知机（Perceptron）模型[58]提出定义算法来学习权重 w，再用权重 w 与特征的乘积确定神经元是否被激活，这一想法规范化地将神经网络定义成数学模型，是之后无数种新兴神经网络的基础。含多个隐层的多层感知器实际上就是深度学习的原型，但在当时，由于多层模型非常复杂，具有太多局部极值点，参数量大难以优化等理论问题，以及当时的计算条件无法满足相关算法需要的大量计算，且没有大量数据支撑如此复杂的模型导致严重过拟合等问题，神经网络的相关研究进入了低潮。

之后在 2006 年深度学习的概率被 Hinton 等人[59]正式提出，他们发明了深度置信网络（Deep Belief Nets，简称 DBN）使用非监督的机制贪心地对网络的每一层逐层训练，并将每一层学习到的表示作为下一层的输入，最后用监督训练来调整所有层，这种方法极大缓解了深层网络和结构难以优化的问题，给深度学习的

发展带来了希望。

在那之后，随着计算机性能的大幅提升甚至 GPU 等强大计算设备的出现，再加上移动互联网等爆发引起的大数据时代到来，深度学习发展的客观条件已经日趋成熟，相关研究人员又从理论层面进一步提出了许多新的训练方法和网络结构，例如广泛应用于图像领域的卷积神经网络（Convolutional Neural Networks，简称 CNN）[60]和应用于自然语言处理于音频的循环神经网络（Recurrent Neural Network，简称 RNN）[61]等。相关实验已经证明在图像视频处理，自然语言处理，音频等多个领域，使用深度学习的方法从训练数据出发，设计端到端的模型，直接输出得到最终结果的模式都取得了远超传统方法的效果，这也标志着深度学习的研究取得了突破性的进展。

2.2.2 卷积神经网络概述

卷积神经网络（CNN）[60]是神经网络的一种，是指至少在一层网络中使用卷积运算代替矩阵乘法的神经网络，目前已广泛应用于语音和图像等领域。它作为第一个真正成功训练并得到应用的深层神经网络，利用样本在空间上的邻近关系通过权值共享的机制减少权值数，降低模型复杂度，使图像的像素信息可以直接作为输入，实现端到端的训练，避免了传统方法中人工特征的提取过程。另外它使用了局部区域感知和空间上的采样等方法，使得神经元能够捕捉到物体边缘和角点等基础特征，还在每层通过卷积操作获得样本最显著的特征，使得网络本身可以获取样本的平移，缩放旋转不变特征。

最早在 1962 年，Hubel 等人提出了感受野[62]的概念，之后 1983 年 Fukushima 等人基于感受野提出了神经认知机[63]，该模型也被看作是 CNN 的第一个实现网络，它将视觉特征分解成一系列子特征，之后在分层相连的特征平面上处理，并希望通过建立模型化系统，保证识别系统的位移不变性。这里的神经认知机正可以看作是卷积神经网络的前身和一种特例。

1998 年，Lecun 等人[64]发明了一种卷积神经网络，取名为 LeNet-5 进行手写数字分类，并使用梯度回传方法训练，实现了从原始像素中自动得出有效表征，并完成识别任务，但由于训练数据和机器计算能力的缺乏，LeNet-5 对于复杂问

题仍无法很好地处理。

在这之后，人们又提出了很多方法来解决深层的 CNN 难以训练的问题，包括 Alexnet[65]，ZFNet[66]，VGGNet[67]，GoogleNet[68]，ResNet[69]等，网络的层数逐渐增多，并利用增加的非线性学习目标函数，并学习更好的特征。这些方法都使 CNN 在一系列计算机视觉任务上取得了突破的进展。

2.2.3 卷积神经网络结构

卷积神经网络主要由卷积层（Convolutional Layer）和池化层（Pooling Layer）两种类型的网络层组成，一个典型的卷积神经网络结构如图 2-4[1]所示。前段是几组卷积层和池化层的组合，最后接入几个全连接层提取高层特征并连接损失函数层进行学习。

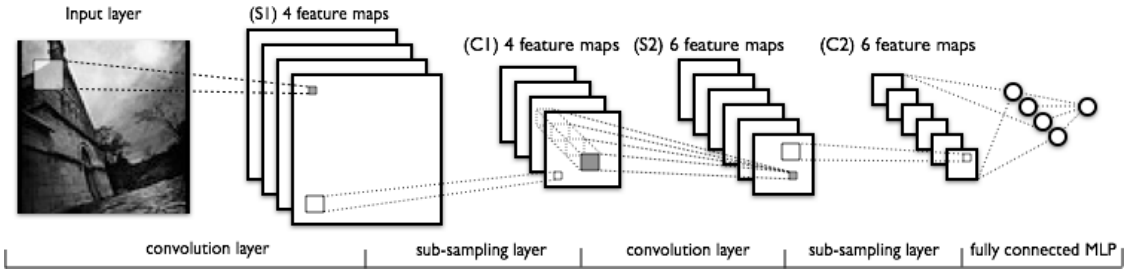

图 2-4 卷积神经网络结构

卷积层中每个神经元的输入与上一层的局部感受野相连，提取局部特征（如物体边缘等），并确定与其他特征的位置关系，每一个卷积层都由多个特征图组成，同一个特征图中的所有神经元共用卷积核（即同一组参数）使得需要的参数减少，提高模型泛化能力。池化层进行空间或时间上的采样，主要是为了模糊特征的具体位置，因为真正需要保留的只是它与其他特征的相对位置，这种对具体位置的模糊可以使模型更鲁棒，对旋转，平移，放缩等处理后的图片一样可以准确识别。卷积层和采样层多交替设置，从而由卷积层提取特征，之后由池化层组合形成更高层抽象的特征，这样组合可以检测更多的特征信息。

[1] 图片来源：http://deeplearning.net/tutorial/lenet.html

2.3 基于深度神经网络的哈希

随着深度神经网络近年来来的迅速发展，许多相关领域都开始引入深度学习相关的研究方法，哈希相关的研究领域也不例外。因为一方面现有哈希方法中很多都需要先通过无监督方式提取 GIST[11]，HOG[12] 等特征，再针对这些特征学习哈希函数，这使得哈希结果依赖于使用特征的好坏，并且不能很好地区分最终的结果由特征和算法本身影响，再加上特征提取过程和哈希函数学习过程完全独立，使得提取的特征并不能与哈希过程最优适配，导致这些特征往往无法保持语义上的相似性，因而不能很好地适用于语义相似搜索。另一方面，深度学习及卷积神经网络（CNN）已经在图像分类，对象识别，人脸识别等计算机视觉任务上取得了重大突破，而且它最大的优点是可以自动学习到样本的高效特征表示，省去了许多特征工程的工作。CNN 在不同的任务中根据目标函数的限制学习需要的特征，这些任务中的成功都证明了 CNN 可以学习到有用的语义特征。因此近些年来，许多研究者开始关注利用深度神经网络进行哈希的方法，利用 CNN 强大的特征表达能力和学习能力学习更为高效准确的哈希函数，比传统哈希算法取得了显著的效果提升。

最早的相关出现是由 Hinton 等人在 2009 年提出的 Semantic Hashing 方法[70]，该方法利用了神经网络的强大表达能力学习复杂的模型，但是该方法中仍然无法直接利用图像的像素信息进行学习，而是仍需要像传统方法一样使用人造特征，而且特征的学习和哈希码学习仍然是分离的两部分，并没有从根本上解决之前提到的依赖手工特征选择等问题。

2014 年，CNNH 方法[71]的提出改变了这一状况，它作为一种监督学习哈希方法，可以利用 CNN 的特征学习能力同时自动学习出用于哈希的图像特征表示和一系列哈希函数。它的过程分两个阶段，整个流程如图 2-5 所示。第一个阶段中，作者先构建训练集$\mathbb{I} = \{I_1, I_2, \dots, I_n\}$的相似度矩阵 S，其中当$I_i$和$I_j$语义相似时，$S_{i,j} = 1$，否则$S_{i,j} = -1$，之后提出一种可扩展的坐标下降方法将 S 分解为$HH^T$，这里的 H 中的每一行就是一幅图片对应的近似哈希码。在第二阶段，该方法使用第一阶段得到的近似哈希码作为 CNN 的标签使用交叉熵损失函数进行学习，指

导 CNN 来同时学习图像的特征表示以及一系列哈希函数。同时，文中还提出了 CNNH⁺的方法，在训练集中包含图像的离散分类标签信息时，将其与第一阶段得到的近似哈希码共同作为CNN 的标签（分类标签采用标准的 softmax 损失函数），意在将样本点标签和样本对标签这两种监督信息相结合，强化学习特征的语义相关性。但该方法由于是分段的所以并不能实现真正的端到端学习，而且由于训练集的哈希码在第一阶段就已确定，深度学习学到的特征表示并不能影响其哈希码的更新。

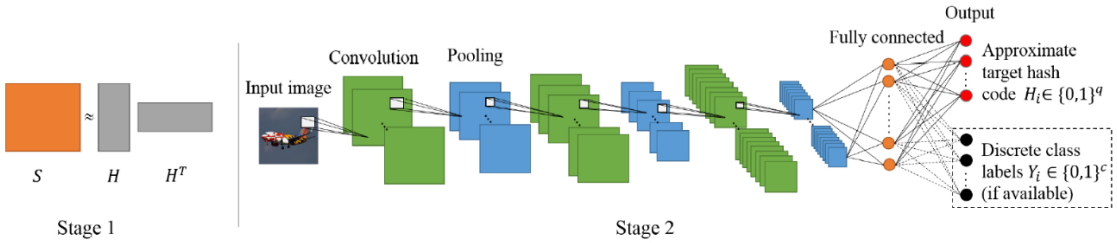

图 2-5 CNNH 算法流程

之后在 2015 年，该文的作者又提出了新的方法 DNNH[72]，该方法使用一个比 CNNH 更深的子网络，实现了在一个共同学习的过程中使得图像的特征表示和哈希码可以互相促进提升。该算法提出的深度神经网络结构分为三部分，如图 2-6 所示：1）CNN 子网络来产生图像的高效中间表示；2）拆分编码模块用部分连接代替常用的全连接，将图像的中间表示分成若干分支，每个分支之后编码成一位哈希码，并加入分段量化函数，这种策略可以有效地降低最终哈希码的不同位之间的相关性，即使得哈希码更紧致；3）三元组损失函数来表征图像 A 比图像 B 更近似于图像 C（这种关系在网络中的点击日志等场景下非常容易获得），目标是使得哈希码空间中相似样本间距小于不相似样本。

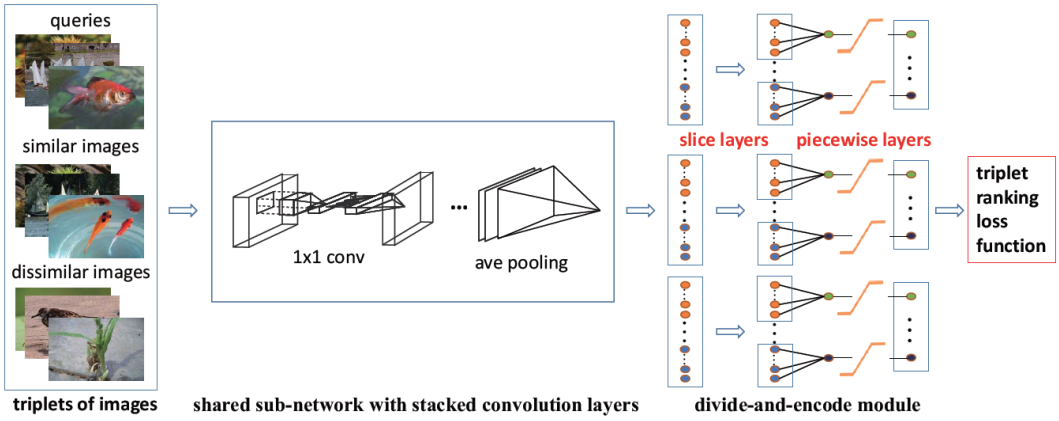

图 2-6 DNNH 网络结构

在预测阶段，该算法的工作过程如图 2-7 所示。

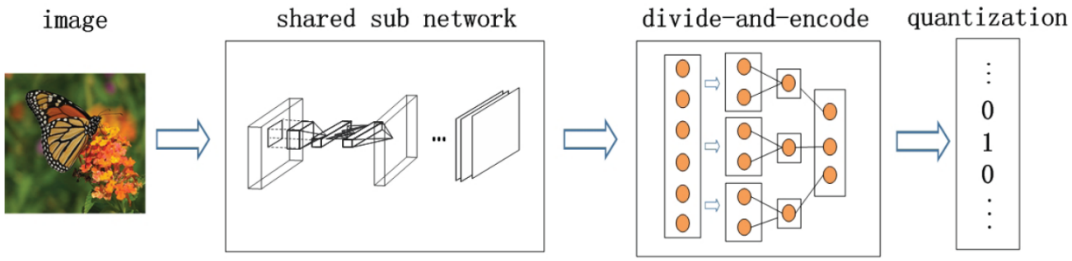

图 2-7 DNNH 预测过程

但这种选取三元组的方法工作量较大，而且三元组的质量直接影响检索的效果，而在网上生成的点击数据等质量又往往难以控制。

与以上方法中多使用样本对标签的监督信息不同，Kevin 等人提出的 DLBHC 方法[73]只利用样本点的类标签信息，先在大规模数据集上训练用于分类任务的 CNN 网络，之后在其输出层前加入一层全连接隐含层，并在目标数据集上进行微调（fine-tuning）训练。该隐含层被证明可以表征类标签表示的语义信息，所以通过对该层的输出值（sigmoid 激活函数限制在(0,1)之间后）二值化即可得到最终的哈希码，这一方法同样可以同时学习图像的特征表示和哈希码，同时由于不需要样本对标签的信息，通用性更强。另外由于使用样本对标签的算法往往需要构建矩阵，在相同数据集情况下，算法的整体复杂度等从$O(N)$增长到$O(N^2)$甚至是$O(N^3)$，因而在大规模数据集中效率较低。但该方法虽然据此提升了效率，且方法更加简单易于实现，但也相应地损失了样本点间的相对位置信息，并不能完全

保证哈希码的汉明距离较小的点在语义上是相似的。

在搜索阶段，该方法采用了 2.1.2.2 中提出的哈希码排序的思想，分粗粒度和细粒度两个层次进行图像检索，整体架构如图 2-8 所示。

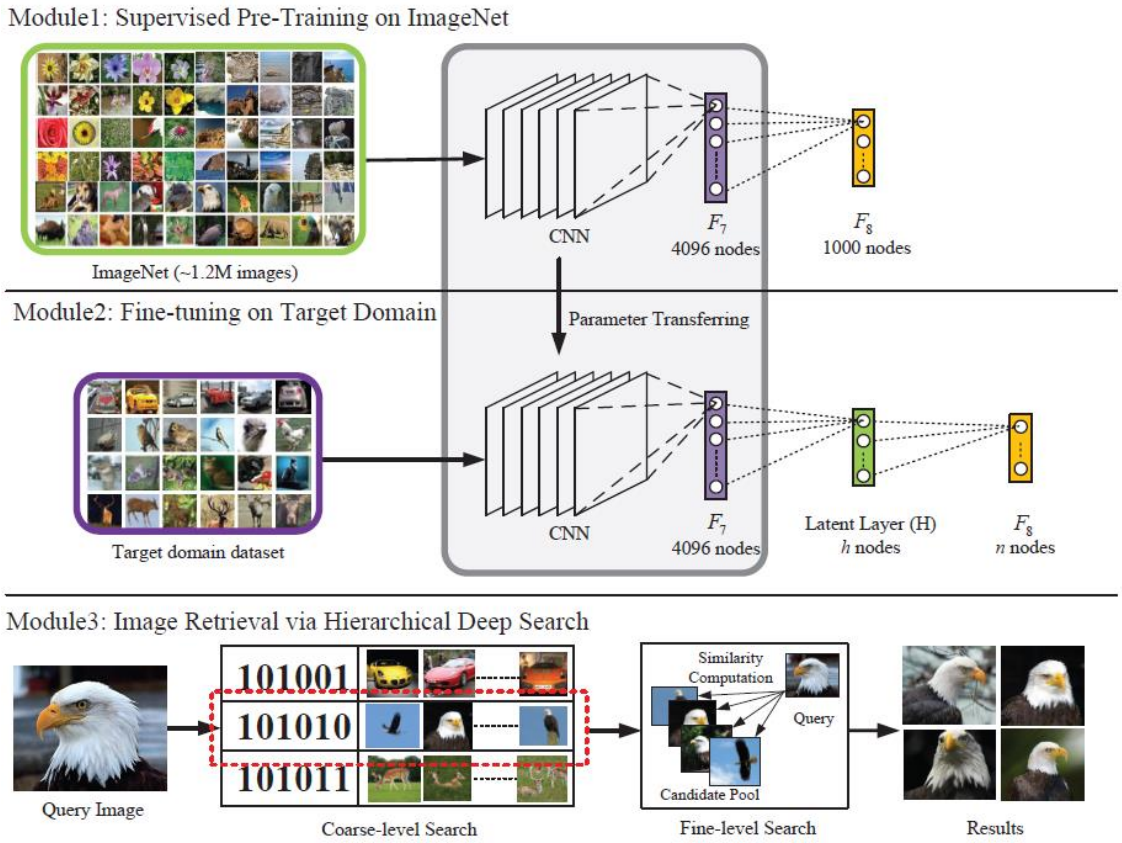

图 2-8 DLBHC 方法整体框架

除此之外，还有一系列方法[74][75][76]关注激活函数限制模型输出和将连续值量化为二进制码的过程，并提出了各种解法。

可以看出，大多数基于深度的哈希算法都基于如图 2-9 所示的框架。

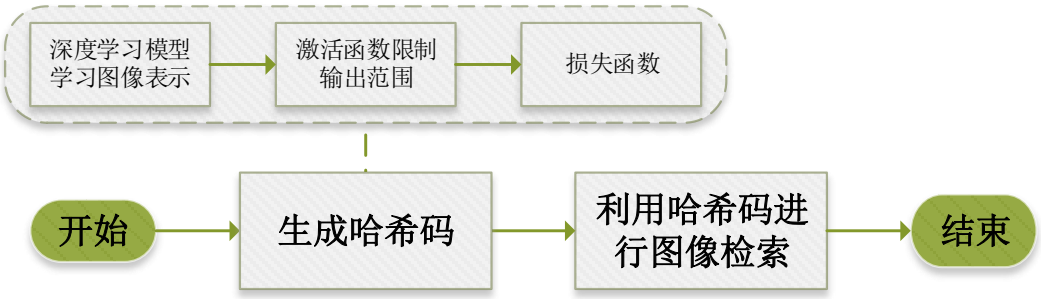

图 2-9 基于深度学习的哈希算法流程

本文正是为了解决上述环节中仍然存在的各种问题，提出了一系列将深度学习与哈希方法更合理地结合的方法，实验证明其对提升图像检索场景下的效率和准确率效果明显。

2.4 本章小结

本章是本文研究课题相关工作的综述，2.1 节介绍了各种哈希算法，先形式化地介绍最近邻检索问题，并说明了用哈希码进行检索的几种方法，接着分数据无关和数据相关（又可分为有监督和无监督两类）两大类列举了各种哈希算法，最后介绍了相关问题中优化的松弛条件。2.2 节从深度学习的发展进程切入，说明了深度学习的优势，之后重点介绍了本课题中使用的卷积神经网络的发展以及结构特点。2.3 节列举了各项最新的结合深度学习进行哈希的研究进展，分析它们尚存的问题，并结构化整个流程，说明了本文要解决的问题及其重要意义。

第3章 问题描述

本文主要解决的是将深度学习与哈希方法结合，更高效准确地进行大规模图像检索的问题。在介绍具体方法之前，本章将首先给出这一问题的形式化定义，同时对后文中用到的符号提前进行标注和说明，并明确本文的预期研究目标，方便后文的阅读理解。

3.1 符号定义

在形式化地定义问题之前，我们先给出本文将用到的各个符号的定义，如表 3-1 所示。

表 3-1 本文中所使用的符号

符号	符号描述
N	样本个数
K	哈希码长度
n	模型融合中的模型个数
$\mathcal{X}$	样本集合
$\mathcal{S}$	样本对相似度标签集合
$L(\cdot), CELoss(\cdot)$	损失函数，交叉熵损失函数
$x_i (1 \le i \le N)$	第i个样本
$S_{ij} (1 \le j \le N)$	第i个样本和第j个样本的相似度，1 为相似，0 为不相似
$f_i\ (1 \le i \le K)$	第i维哈希函数
$y_i (1 \le i \le N)$	第i个样本量化为哈希码之前在网络全连接层的输出值
$h_i (1 \le i \le N)$	第i个样本的哈希码
$h_i^{(1)}, h_i^{(2)} (1 \le i \le N)$	复合哈希码策略中对于第i个样本的两组哈希码
$D_h(h_i, h_j)$	哈希码h_i和h_j的汉明距离
$\alpha, \beta, \varepsilon$	权重因子及平滑因子
$Conv_i$	第i个卷积层
$Pool_i$	第i个池化层
FC_i	第i个全连接层

3.2 标签类型

在本文关注的图像语义相似性检索领域，图像的标签作为一种语义层级的监督信息，可以更直接地引导模型学习保持语义相似性的哈希码。基于监督学习的哈希方法由于可以利用这样的语义标签信息，已经被证明可以取得比基于无监督学习的哈希方法更好的检索效果，因此本文也主要关注基于监督学习的哈希方法。在这类方法中，数据集中的样本会带有语义标签信息，常见的标签包括样本点标签，样本对标签和样本三元组标签。样本点标签是指对每个样本都有其对应的标签，通常为该样本的语义分类；样本对标签形式则为$\mathcal{S} = \{s_{ij}\}$，其中$s_{ij} \in \{0,1\}$，当样本x_i和x_j语义相似时$s_{ij} = 1$，不相似时$s_{ij} = 0$；样本三元组标签则是对于三元组(I, I^+, I^-)来定义的，这样的三元组的含义是I和I^+的相似度要比I和I^-的相似度更大。容易发现，它们的关系如图 3-1 所示，即根据样本点标签可以容易地得到样本对标签，同样根据样本对标签可以得到样本三元组标签。前者可以简单地通过同一语义类中的样本对标签为相似，不同语义类中为不相似得到，后者则可以通过对相似样本对(I_1, I_2)和不相似样本对(I_1, I_3)合并得到样本三元组标签(I_1, I_2, I_3)，即I_1和I_2的相似度要比I_1和I_3的相似度更大，可见研究使用样本三元组的方法是最通用的。

图 3-1 各种类型标签之间的关系

然而在实际应用场景下，数据集中起初大多只具有样本点标签信息，所以直接选用样本点标签更为直接，但在哈希检索相关研究的场景下，我们最关心的问题是样本在哈希过程前后的空间中的相对位置关系是否得到保持，所以只使用点标签而不考虑样本间相对位置关系的做法与最终的检索任务和目标有所偏离，且其只针对单个样本点，并不能直接保证汉明距离近的点也语义近似。另一方面，样本三元组标签虽然通用，但是由于样本三元组本身较为复杂，实际应用中优化和模型收敛都比较缓慢，而且在原始只有样本点标签的数据集上经过转化得到的

样本三元组标签并没有体现比样本对标签更多的信息（在检索点击数据等场景下，三元组标签可能包含更多信息并有较好的应用，但其不在本文的研究范围），所以本文选用样本对标签作为监督信息展开研究，既可以直接以样本对为单位进行优化，使得样本在哈希前后的两两相对位置关系得到保持，又相对简单易用，模型容易收敛。

3.3 问题定义

在相似图像检索中，我们有 N 个样本作为训练集$\mathcal{X} = \{\mathrm{x}_i\}_{i=1}^{N}$，其中每一个都代表一个 D 维的特征向量$\mathrm{x} \in \mathbb{R}^{D}$，它们可能是人为提取的手工特征，也可能是像素点信息。

基于样本对语义标签进行监督哈希学习算法的目标就是学习哈希函数$\mathrm{f}: \mathrm{x}_i \longmapsto \mathrm{h}_i \in \{-1,1\}^{\mathrm{K}}$，将每个样本点$\mathrm{x}_i$编码成紧致的 K 位哈希码$\mathrm{h}_i = \mathrm{f}(\mathrm{x}_i)$，且这样的哈希码$\mathcal{H} = \{\mathrm{h}_i\}_{i=1}^{n}$需要保持$\mathcal{S}$中的语义相似性，即如果$\mathrm{s}_{ij} = 1$，那么哈希码$\mathrm{h}_i$和$\mathrm{h}_j$应当有较小的汉明距离，反之则它们之间的汉明距离应较大。并且我们希望学习到的哈希函数f对于训练集之外的新样本，也同样可以生成保持样本两两间相似性的哈希码，从而使得直接在哈希之后的空间中基于汉明距离进行检索可以得到准确的相似样本。

我们可以把二进制哈希码通用地记作：

$$\mathrm{h}_i = f(x_i) = [f_1(x_i), f_2(x_i), \dots, f_K(x_i)]^T \qquad \text{(公式 3.1)}$$

3.4 预期目标

本文的预期目标是针对现有的基于深度学习的哈希算法进行从检索策略到网络结构和优化目标等各方面的改进，并通过加入与哈希检索场景相符的一些优化技巧使得检索系统更加稳定，在 MNIST，CIFAR10，NUS-WIDE 三个权威的公开图像数据集上将本文的方法与利用手工提取特征的传统哈希算法，以及最近几年提出的几种基于深度学习的哈希监督学习算法进行比较，希望其在平均准确率 MAP（Mean Average Precision），同码长（即占用相同空间）时的准确率以及检索用时等方面较现有方法有所提高。

3.5 本章小结

本章内容是对本文的研究课题进行形式化的定义，在 3.1 节中定义了全文中用到的各种符号，3.2 节中介绍了在基于监督学习的哈希算法中，可能使用的不同种类的标签信息，并说明了本文选择样本对标签信息进行研究的原因，3.3 节中对本文研究的哈希检索问题进行形式化定义，3.4 节中则详细说明了文章希望达到的目标和评测指标等。

第4章 基于深度学习的语义检索哈希算法

尽管已经有许多现有的哈希算法可以学习保持相似性的二进制码，但它们往往受限于手工提取特征与哈希学习阶段分离无法完全适配，以及线性投影表达能力差等问题。卷积神经网络（CNN）作为一种非线性模型，已经在计算机视觉领域的各个任务上已经展现出了很好的效果，最近几年也有一些基于深度学习的哈希方法应运而生，然后它们也各自存在训练收敛慢，哈希码冗余性高，无法表达语义相似性等各种问题。本文中，我们综合之前出现的各种基于深度学习的哈希算法，从搜索策略，网络结构等多方面对其进行改进，同时学习紧致的二进制码和具有语义信息的图像表示。本文中提出的基于深度学习的语义检索哈希算法（Deep Hashing for Semantic Retrieval，简称 DHSR）一共包括三部分，1）使用包含多个卷积-池化层的 CNN 来学习图像的语义特征；2）设计结合了样本对标签和样本点标签，并考虑量化损失的损失函数层来快速学习合理的相似性度量； 3）提出结合神经网络设计的复合哈希码层次化检索策略。我们提出的方法首先使用样本对和他们的相似度标签来训练 CNN，之后通过损失函数学习保持相似性的二进制哈希码，对于新的图像将 CNN 的输出量化可得到对应的二进制码，再结合层次哈希检索策略得到检索结果。所有这三部分都可以无缝整合到同一个深度架构中，而且以端到端的方式直接将图像从像素直接映射到样本对标签，这样不同的部分之间可以互相反馈并形成促进效果，CNN 提取出的特征也更适合当前哈希任务，从而学习到比非端到端的方式更好的哈希函数。在本章的剩余部分，我们将依次详细对这几部分进行介绍。

4.1 基于 CNN 的特征学习

我们的 DHSR 方法对图像直接使用 CNN 来提取学习特征，目前在计算机视觉相关领域已经涌现出了多种高效的 CNN 网络结构设计，而由于在我们的方法中，CNN 只是起到特征学习器的作用，具体的算法并不依赖于具体的网络结构，所以事实上，大多数已有的 CNN 结构都可以用在本方法中进行特征提取，例如

2.2.2 中提到的 Alexnet[65]，ZFNet[66]，VGGNet[67]，GoogleNet[68]，ResNet[69]等结构，但 CNN 的具体设计并不是本文研究的重点，所以为了方便说明，我们使用最简单的 Alex Krizhevsky 在 Cuda-convnet 中使用的网络结构来阐释本文方法的有效性，实际情况中可以根据数据集选择已在该数据集上展现出较好效果的网络结构。我们的 DHSR 方法的整体框架如图 4-1 所示。

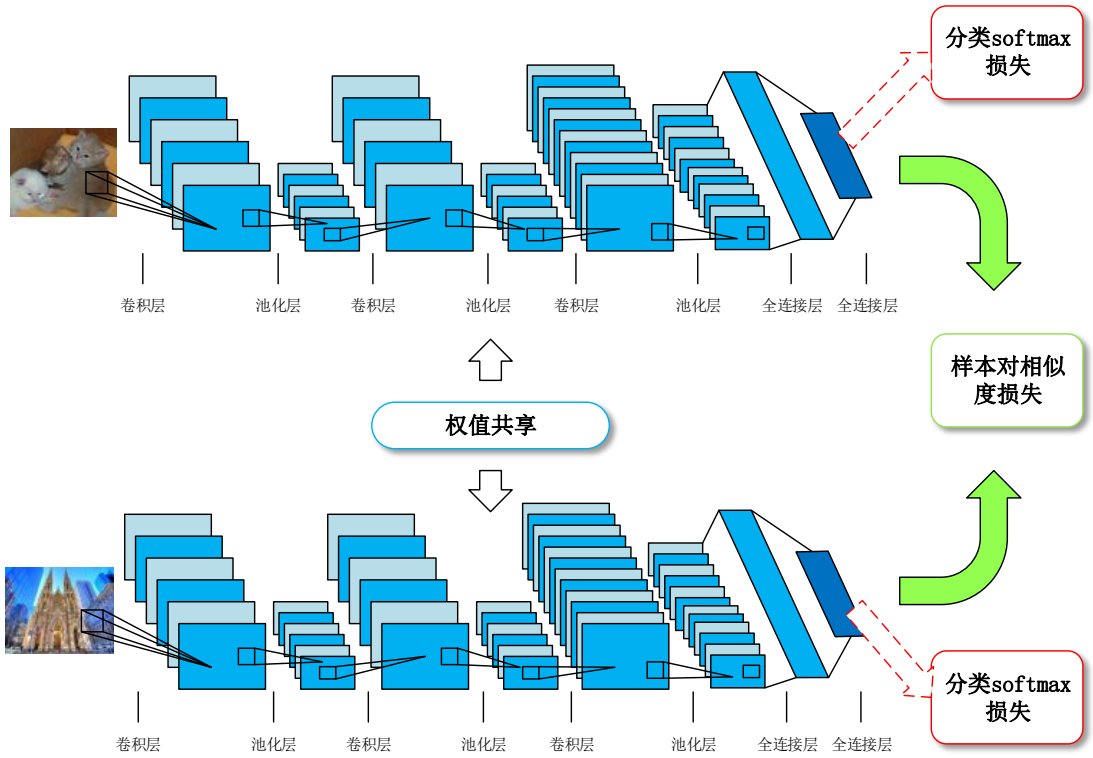

图 4-1 DHSR 方法的端到端学习框架示意图

可以看到，图 4-1 中其实有两个 CNN，分别对应一个样本对中的两个图片在神经网络中学习的过程，它们的结构完全相同，并且共享完全相同的权重，这种方法可以大大减少模型中的参数数量，可见我们的输入和损失函数都是基于样本对来考虑的。另外，与之相对的是对一个样本对使用两个不同的哈希码映射，类似论文[77]中的非对称哈希思想，但我们的实验证明了这样的权值共享机制不仅使模型易于训练，同时也可以取得更好的效果。

4.1.1 整体网络结构

具体的 CNN 网络的配置结构如表 4-1 所示，它共含有 3 个卷积层（Conv1-3），

三个池化层（Pool1-3）和两个全连接层（FC1-2），每个卷积层通过三个方面描述："filter"说明了卷积核的数量和它们的局部接受野的大小，用"数量*卷积核宽*卷积核高"表示，"stride"代表在输入图像上进行的相邻卷积操作的空间间隔，"pad"代表在输入图像的每一侧添加的像素数。每个池化层同样通过三方面来描述，首先是下采样的操作类型，通常有"Max"（求最大值）和"Ave"（求平均值），之后的"size"代表采样的窗口大小，"stride"代表相邻池化操作的空间间隔，"LRN"代表局部归一化层（Local Response Normalization）[65]。对全连接层，表中的数字代表输出的维数，其中 K 代表要生成的 hash 码的位数，这代表全连接层的输出需要与目标哈希码长度对应，α则是一个参数，具体将在 4.1.2 节中介绍。同时可以看出由于 CNN 在本文方法中更多的是以特征提取器的形式出现，所以网络结构受不同哈希场景的影响很小，通用性较强，具体相关内容将在 4.3 节说明。另外网络中卷积层部分的所有激活函数都使用 ReLU（Rectification Linear Unit）。

表 4-1 DHSR 方法的 CNN 网络配置图

类型	配置
卷积层 1（Conv1）	filter32*5*5, stride1*1, pad2
池化层 1（Pool1）	Max, size3*3, stride2*2, LRN
卷积层 2（Conv2）	filter32*5*5, stride1*1, pad2
池化层 2（Pool2）	Ave, size3*3, stride2*2, LRN
卷积层 3（Conv3）	Filter64*5*5, stride1*1, pad2
池化层 3（Pool3）	Ave, size3*3, stride2*2
全连接层 4（FC1）	$\alpha * K$
全连接层 5（FC2）	K

4.1.2 分块全连接模块

在经过 CNN 网络模块直接通过图像的像素信息学习提取有效的特征之后，我们需要通过全连接层对这些特征进行进一步地处理，综合提取出语义相关的高层抽象特征。此处为了减少哈希码的冗余性，使生成的哈希码更紧致（各位不相

关），我们借鉴了[72]中的想法，引入了拆分编码模块，当目标哈希码位数为 K 时，两个全连接层 FC1 和 FC2 层的输出长度分别为αK和K，此处的α是一个参数，在具体使用中可以通过交叉验证的方式选择当前场景最合适的值。首先我们将 FC1 层中的中间特征进过分片处理分为分成 K 个分支，每分支长度为α，在分支内部，所有的α个神经元通过全连接的方式与 FC2 层中的一位相连，最后 FC2 层中的 K 维经过符号函数 sgn 之后生成最后的哈希码，具体情况如图 4-2 所示。

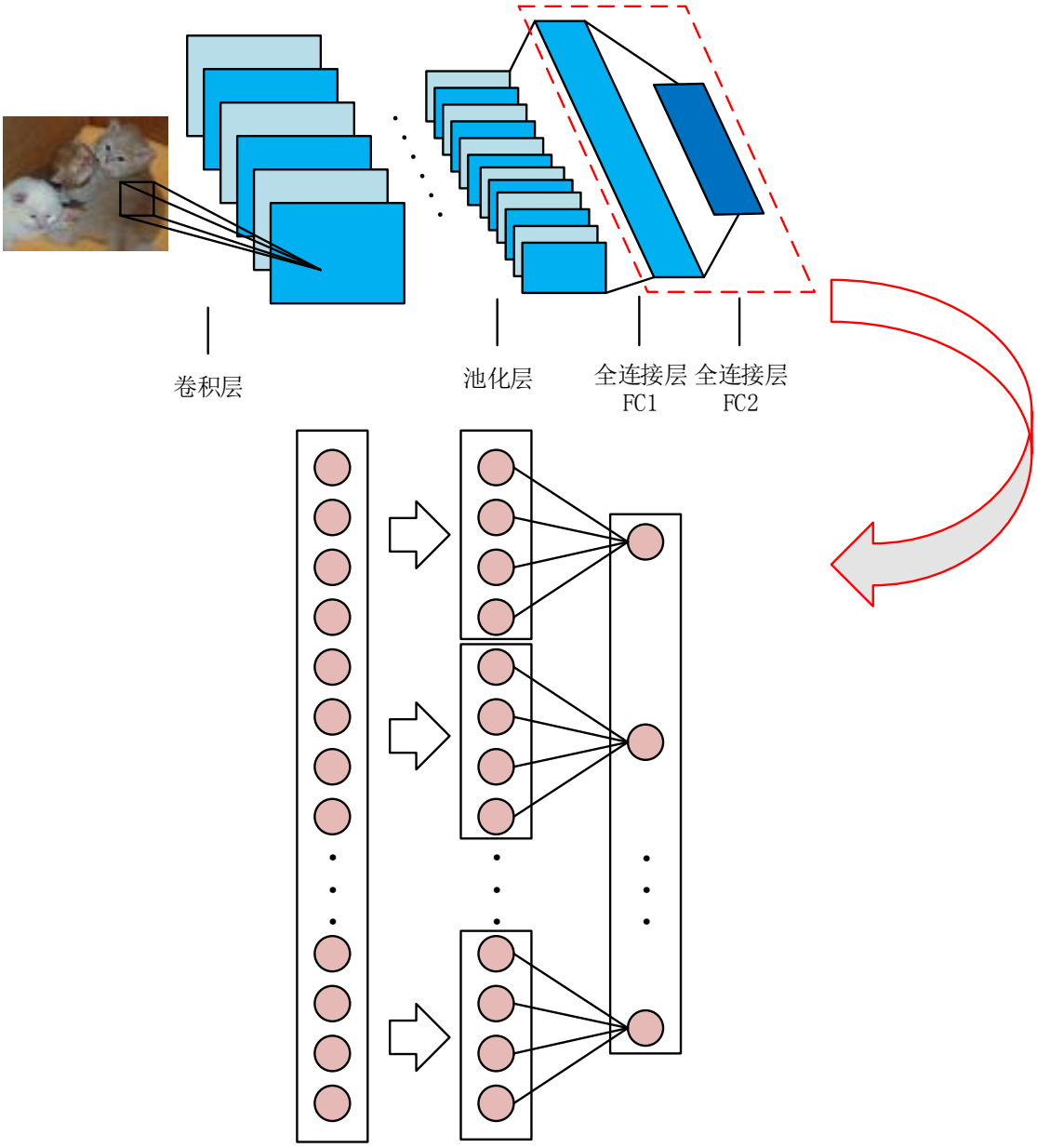

图 4-2 分块全连接模块示意图

与原先的完全连接方式相比，分块全连接的方式可以减少哈希码各位之间的冗余性，因为完全连接的方式中每一位哈希码都是由全部当前样本的全部中间特征生成的，这样生成的哈希码不同位之间是相关的，然而相比之下，分块全连接的方式中，每一位哈希码都是由一部分中间特征生成，且不同位哈希码的来源特征完全没有重叠，这可以减小不同位之间的相关性。

在文章[5]中曾提到的哈希学习的目标之一是用更少位数的哈希码取得更好的检索效果，而最近的研究[78]也在理论和实践方面进一步证明了具有更少冗余性的哈希码的检索效果更优。在我们的实验中也证明了分块全连接的方式比完全连接具有更好的检索效果。

4.2 损失函数与哈希码生成

根据 3.3 节中的问题定义可知，为了使哈希码空间同样能够保留原始空间中的样本对相似性，语义相似的样本对的哈希码同样应当尽可能地近，同理语义不相似的样本对的哈希码的汉明距离应当尽可能大。所以损失函数作为神经网络学习的目标，也正应以此为目标。

我们的 DHSR 方法中的损失函数设计结合了样本对标签和样本点标签，并考虑量化损失，来综合学习合理的相似性度量。

4.2.1 样本对距离保持

对于样本对标签部分，我们仿照[74]中的思想，对于样本对$\mathrm{x}_i, \mathrm{x}_j \in \mathcal{X}$，它们的语义相似度为$\mathrm{s}_{ij}$，生成的哈希码分别对应为$\mathrm{h}_i, \mathrm{h}_j$，遵循前文中的定义可得：

$$L_1\left(h_i, h_j, s_{ij}\right) = \frac{1}{2} s_{ij} D_h\left(h_i, h_j\right) + \frac{1}{2}\left(1 - s_{ij}\right)\left(\frac{1}{D_h\left(h_i, h_j\right) + \varepsilon}\right) \quad \text{(公式 4.1)}$$

其中$D_h(\cdot,\cdot)$代表汉明距离两个二进制向量之间的汉明距离，这种损失函数的思想非常直接，公式中第一项中对于相似的样本对直接以哈希码汉明距离的一半

$\frac{1}{2}D_h(h_i,h_j)$作为该项的损失，距离越大损失越大，第二项同理，使得不相似的样本对距离越小损失越大，并在分母中加入一个平滑项ε处理分母为 0 的情况。但经过实验发现，由于在现实场景中，训练集中不相似的样本对往往远远多于相似的样本对，这会造成公式 4.1 中第二项占得比重很大，使神经网络尽量将不相似的样本推远，造成神经网络在训练一段时间之后全部输出 0，无法训练学习到有用的信息，所以此处使用了对比损失函数（Contrastive Loss）的形式如下：

$$L_1(h_i,h_j,s_{ij}) = \frac{1}{2}s_{ij}D_h(h_i,h_j) + \frac{1}{2}(1-s_{ij})\max(t-D_h(h_i,h_j),0) \quad \text{(公式 4.2)}$$

其中 t 是一个阈值参数，式子第一项与公式 4.1 中相同，限制相似样本哈希后的距离，而在第二项中引入了一个阈值参数 t，只在不相似样本的哈希后距离比 t 还小的情况下才引入损失，否则在距离超过 t 的情况下均视为离得足够远，不计入损失。

4.2.2 量化误差

对于公式 4.2，其中的$h_i \in \{-1,1\}^K$是离散值，是对神经网络输出的结果用符号函数 sgn 二值量化之后得到的，所以无法直接被神经网络用来学习，通常需要引入松弛条件使得h_i取值放宽到整个实数范围（一些直接对离散值进行优化的方法[46][56]已经在[74]中被指出由于使用批量梯度下降无法保证产生哈希码的全局最优性）。但从另一个角度来说，只引入松弛条件，优化得到的结果对于量化之后的离散编码来说并不一定是最优的，如果不考虑量化这一步造成的误差，会导致对于量化阈值两边很近的点其实语义相似，但因为量化误差被分配不同的哈希码增大汉明距离，破坏相似度保持性的原则。为了解决这一问题，我们在对h_i加入松弛条件之后，在损失函数项中加入[74]提出的一种正则项，如图 4-3 所示。

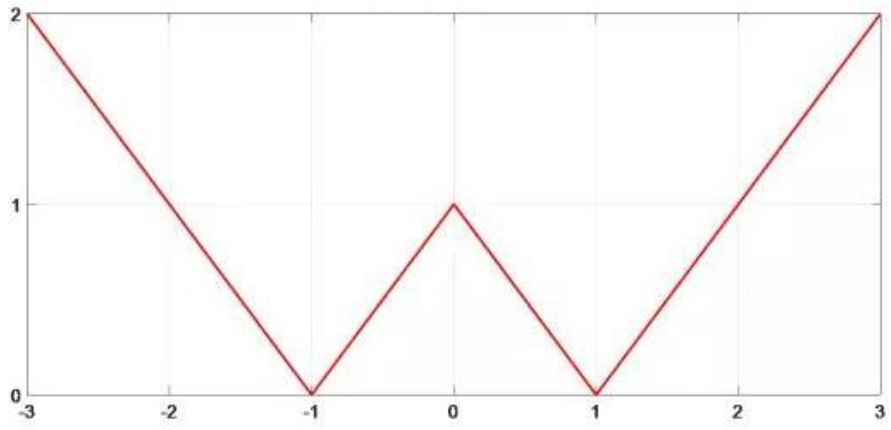

图 4-3 正则项示意图

用公式表示即为：

$$L_2(h_i) = \|h_i - y_i\|_1 \quad \text{(公式 4.3)}$$

且有：

$$h_i = \mathrm{sgn}(y_i) \quad \text{(公式 4.4)}$$

其中y_i即为量化前，由神经网络输出的对应的实数值，这里使用了 L1 范数的距离，因为这样的距离计算更迅速，而且与 L2 范数相比可以产生稀疏的解，也就是使的很多y_i的值与h_i完全相等，另外对于训练过程来说，这样的函数形式，不会像更高级别的范数那样出现饱和的现象（Saturate）使得网络学习变得缓慢甚至停止学习。

加入正则项之后损失函数变为：

$$\begin{aligned} L'(h_i, h_j, s_{ij}) = & \frac{1}{2} s_{ij} \|h_i - h_j\|_2^2 \\ & + \frac{1}{2}(1 - s_{ij}) \max\left(t - \|h_i - h_j\|_2^2, 0\right) \\ & + \alpha(\|h_i - y_i\|_1 + \|h_j - y_j\|_1) \end{aligned} \quad \text{(公式 4.5)}$$

此处的α即为控制正则化程度的参数，这里我们将 4.2 中的$\mathrm{D_h(h_i, h_j)}$使用 L2 范数形式计算，这是因为使用 L1 范数求出的梯度对于不同距离的样本对来说都相等，体现不出距离的差异性。加入正则项之后，大部分的网络输出项被直接限

制在 1 和-1 附近，量化阈值 0 周围的样本很少，因此之前所述的量化误差可以得到极大缓解。

对公式 4.5 中的函数，$L'(h_i, h_j, s_{ij})$对于h_i和h_j的梯度由于 max 函数和 sgn 函数的存在无法直接求解，此处采用次梯度的方法求得当$\|h_i - h_j\|_2^2 <$ t时有：

$$\frac{\partial L'(h_i,h_j,s_{ij})}{\partial h_i} = s_{ij}(h_i - h_j) - (1 - s_{ij})(h_i - h_j) + \alpha\delta(h_i) \quad \text{(公式 4.6)}$$

$$\frac{\partial L'(h_i,h_j,s_{ij})}{\partial h_j} = s_{ij}(h_j - h_i) - (1 - s_{ij})(h_j - h_i) + \alpha\delta(h_i) \quad \text{(公式 4.7)}$$

而当$\|h_i - h_j\|_2^2 \geq$ t时有：

$$\frac{\partial L'(h_i,h_j,s_{ij})}{\partial h_i} = s_{ij}(h_i - h_j) + \alpha\delta(h_i) \quad \text{(公式 4.8)}$$

$$\frac{\partial L'(h_i,h_j,s_{ij})}{\partial h_j} = s_{ij}(h_j - h_i) + \alpha\delta(h_i) \quad \text{(公式 4.9)}$$

上述式中均有：

$$\delta(x) = \begin{cases} 1, -1 \leq x \leq 0 \text{ or } x \geq 1 \\ -1, \text{ Otherwise} \end{cases} \quad \text{(公式 4.10)}$$

4.2.3 样本点语义标签

在[73]中已有实验证明样本点语义标签信息对哈希码生成具有重要的意义，此处就将其加入损失函数中，结合量化误差，样本点分类语义信息和样本标签对相对距离保持三方面的信息共同构成损失函数，这种结合样本点语义标签和样本对语义标签两方面语义信息的思想在[71]中有过体现，该文中将只使用样本对标签的方法称为 CNNH，使用样本对标签和样本点标签两方面信息的进阶方法则称为 CNNH$^+$。CNNH$^+$实现了既保持样本间语义相似性又保留个体语义特征，比前者取得了更好的效果。具体来说，我们对于单个样本点的分类语义标签使用常用的交叉熵损失（Cross-Entropy Loss）函数的形式定义，它同样也适用于多标签场景，在单标签多分类场景中，可以直接使用 Softmax 函数，即为：

$$L_3(h_i) = CELoss(x_i) \quad \text{(公式 4.11)}$$

其中$CELoss(\cdot)$即为交叉熵损失，梯度计算不再详述，则此时总的损失函数变为：

$$L(h_i, h_j, s_{ij}) = L_1 + \alpha L_2 + \beta L_3 \quad \text{(公式 4.12)}$$

其中β是样本点标签这部分的权重因子，考虑所有的样本对可得总的损失函数：

$$L = \sum_{x_i, x_j \in \mathcal{X}} L(h_i, h_j, s_{ij}) \quad \text{(公式 4.13)}$$

值得一提的是，在实际场景中，如 3.2 节所述，有样本点标签一定可以转换成样本对标签，但只有样本对标签的时候却无法得到样本点标签，因此对于包含了样本点语义标签的 DHSR 方法，如果没有或不考虑单个样本点语义标签信息，这时损失函数中L_1和L_2两项仍可以正常计算，因此去掉第三项L_3即可形成该种情况下的损失函数，这种方法称为 DHSR-S 方法。

4.2.4 哈希码生成

经过了神经网络的训练学习之后，我们可以直接通过符号函数 sgn 量化得到训练数据的哈希码，而对于测试集中的数据和之后会遇到的新数据或是用户查询$\mathrm{x}_q \notin \mathcal{X}$，由于训练好的神经网络就相当于是我们学习到的哈希函数，所以我们也可以简单地对样本x_q通过在神经网络中的前向传播得到对应的输出y_q，之后同样通过量化得到哈希码$\mathrm{h_q} = \mathrm{sgn}(\mathrm{y_q})$。

4.3 基于神经网络的复合哈希码层次检索

4.3.1 方法描述

如 2.1.2 节中所述，在得到哈希码之后，有两种最基本的方法来使用哈希码进行最近邻检索：哈希表查找和哈希码排序。在实际应用中，哈希码排序比哈希表查询准确率更好，但搜索效率稍差，总体效果更好，我们的 DHSR 方法采用了基于神经网络的哈希检索策略，结合了两种基本策略的优势。

如图 4-4 所示，与之前已知的所有基于深度学习的哈希算法，将最后一层的输出结果直接量化得到哈希码的方法不同，本文提出的方法利用了两个全连接层

来进行哈希，整个过程分为两步：

1. 利用 FC2 层中的哈希码进行哈希表查找操作，当此处选用的码长较短时，可以保证较好的召回率，以避免 2.1.2.1 节中提到的查找相邻的多个表项，或者构建多个表等加大时间和空间耗费的操作。
2. 利用 FC1 层量化得到的哈希码，对第 1 步中命中的哈希表项中的样本计算汉明距离进行哈希码排序操作，此处选用的码长相对较长，保证较好的准确率，并将此作为最后的排序结果。

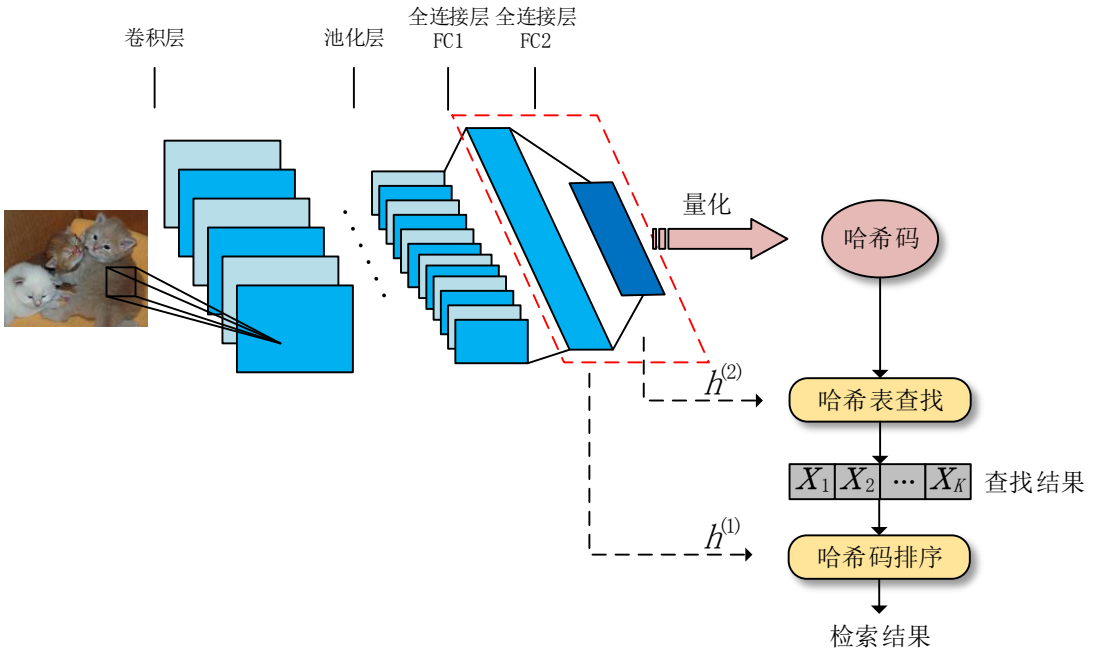

图 4-4 复合哈希码层次检索策略示意图

通过 4.2 节的描述可知，其实我们的方法只对最后一层 FC2 的输出通过损失函数进行限制，使其学习到保持样本间语义相似性和个体语义特征的哈希码表示，而从理论上说，要同时使用 FC1 层和 FC2 层的内容，最直截了当的做法就是在损失函数的目标中也加入 FC1 层的相关优化目标，但是这会造成我们的算法模型更加复杂，而且可能使得神经网络难于训练和优化，而且过多项优化目标反而可能会造成多个目标都不能很好的学习。另一方面，由于神经网络是一个完整的系统，它的学习过程是层次化的从低层到高层逐级抽象提取特征的过程，它的每一层之间都通过输出值的前向传播和梯度值的反向传播这两个过程紧密地联系在一起，

前一层是后一层学习特征的来源，所以对于被损失函数限制的 FC2 层的前一层，FC1 层的作用可以理解为对作为特征提取器的 CNN 网络学习到的特征进行初级提取整合，它的特征抽象层次稍低于 FC2 层。而且在梯度回传学习算法过程中，FC1 层是紧随着 FC2 层的后继层，直接接受从 FC2 层传来的梯度，因此对 FC2 层限制的优化目标也直接影响并指导它提取学习对于 FC2 层有用的初级特征，供 FC2 层进一步整合，所以该层的内容作为更细粒度的特征对于语义相关性也已经有很好的保持。

因此在我们的DHSR方法中，并没有在优化目标中加入对FC1层的直接限制，类似的思想在[73]中也有类似的应用，该文中提出方法的优化目标是分类的 Softmax 损失函数，但选用了之前的一层全连接层来为来源进行量化得到哈希码，其实验部分已经证明这种间接联系和利用优化目标的方法可以取得很好的效果，但由于它的损失函数中缺乏样本对之间具体距离量度的考虑，所以效果仍不理想。

另一种可能的变形是仍然采用之前基于深度学习的哈希方法普遍采用的只使用最后一层全连接层 FC2 层的方式，但为了得到层次化哈希策略需要的复合哈希码，分别将 FC2 层输出设置为较小维度和较大维度，训练两次网络，一次短码长一次长码长，并将它们对应应用于层次化方法的两个步骤中，这种方法的好处是层次化方法中使用的两组哈希码都是与优化目标直接相连而得到的，但是会造成模型训练和数据库中样本哈希码生成过程中的开销加倍，并且实验结果证明其在实际应用中并没有带来准确率上的提升。

4.3.2 与传统方法的对比

本文中的层次哈希策略与单独使用哈希表查找的方法相比，第一步查找哈希表的操作完全相同，但在第二步中哈希表方法需要对第一步中命中的哈希表项中的样本在原始特征空间中进行欧式距离的计算，而我们的方法只需要进行汉明距离的计算，因而可以带来几个数量级级别的提速，而对于整个哈希表查找方法来说，前半部分查找哈希表的操作是常数级别时间复杂度的，最耗时的正是后半部分在原始空间中的欧氏距离计算，因而这样的改进可以大幅提升算法的速度。

另外，其实本文中的层次哈希策略解决了哈希表查找算法中对哈希码长度选

择的两难，在传统哈希表算法中，如果哈希码选择太长会造成召回率过低，为得到足够个数的检索结果，需要采用 2.1.2.1 节中提到的查找相邻的多个表项，或者构建多个表等操作提高召回率，耗费时间和空间，另一方面如果哈希码太短会影响检索的准确率，而本文中的方法由于采用复合哈希码，可以很好地通过两组哈希码不同长度的选取，分别优先考虑准确率和召回率，在准确率和召回率间达到平衡。

在实际的相关检索应用中，需要返回的排序的检索结果的个数常常是会发生变化的，这时如果采用哈希表查找的方法，如果不选用查找相邻表项，或者构建多个表等方法，就需要随之调整哈希码的长度，随之带来的就是对整个模型的重新训练，并需要对数据库中所有参照项样本全部重新生成新的哈希码，而本文采取的方法在使用时可以在第一步中选用较短的哈希码保证较好的召回率，不会发生检索结果个数不够的问题，也因为第二步的存在不用担心由此带来的准确率损失的问题，在实际应用中通用性较强。

另外一方面，这种策略与单独使用哈希码排序的方法相比，在前半部分加入了一步哈希表查找，使原先哈希码排序方法中的穷举汉明距离计算这一步的范围缩小到了命中哈希表同一表项的范围，且省去了最后的欧式距离计算重排序的过程，直接使用汉明排序的结果作为最后结果，大大节省了检索所需的时间。在[75]等基于深度学习的哈希方法的实验部分中也都提到了，哈希码排序的准确率可以在很大程度上体现出哈希码对于语义相似性保持的优劣，本文中实验部分的结果也证明了，采取这样的策略，以本文的 DHSR 方法中模型学习到的哈希码为基础，直接使用汉明排序的结果作为最后的检索排序结果，在提升速度的同时也保持了很好的准确率。

4. 4 本章小结

本章内容是本文的重点，包括本文提出的 DHSR 算法的具体描述，通过对整个算法详尽的细节阐释，说明了算法的设计思路和合理性。在 4.1 节中介绍了作为特征学习器的 CNN 网络的具体结构，并详细列出了每层的具体设置和函数选择，介绍了分块全连接模块对于降低哈希码各位冗余性的作用。4.2 节中具体说

明了本文提出的哈希方法的学习目标和损失函数设计，介绍了这种结合量化误差，样本点分类语义信息和样本标签对相对距离保持三个层面信息的学习目标，并阐述了针对这种方法生成哈希码的过程。4.3 节介绍了本文提出的基于神经网络的复合哈希码层次检索策略，该方法结合了哈希表查找和哈希码排序两种基本方法的优点，之后解释了可能采用的一些相似方案的缺点，并与哈希表查找和哈希码排序两种基本方法分别对比，说明该检索策略的优越性。

第5章 实现细节与优化

本章重点介绍本文提出的DHSR方法实现上的细节和在实现过程中使用的一些优化技巧。

5.1 基本实现

我们的 DHSR 在深度学习平台 Caffe[79]上实现，对于 4.2 节中设计的结合样本对标签和样本点分类标签的损失函数，我们需要在 Caffe 中编写新的损失函数层，此处我们借鉴并修改了[74]对应的开源代码[2]对于样本对标签的部分实现，将 4.2 节中介绍的损失函数的前两部分分别编写新的损失函数层 PairwiseLoss 和 QuantLoss。此处的 PairwiseLoss 层在实现时，由于我们的实验数据集以及大多数情况下，样本对标签都是由样本点标签生成的，所以在我们的设计中，原始的数据层还是只提供点标签 label，将由样本点标签生成样本对标签的过程放在 PairwiseLoss 层中，对当前 batch 中所有样本对根据其样本点标签是否有重合来实现。另外样本点分类标签部分可以直接使用 Caffe 中支持的 Softmax 层来实现多类分类，损失函数中的各项的权重因子通过 Caffe 的损失函数层中支持的 loss_weight 参数来实现，并通过交叉验证的方法来设定最优的权重因子值。

实现的 DHSR 模型结构图如图 5-1 所示，该图中损失函数之前的配置与 4.1.1 节中描述的结构一致，网络中从数据层（Data）开始，训练和测试阶段根据各自需要选用不同的批大小（Batch Size），数据层提供图像的像素信息（data）和标签信息（label），前者进入 CNN 进行特征学习与提取，后者直接进入 PairwiseLoss 和 SoftmaxWithLoss 两个损失函数层通过批量梯度下降的方法来进行训练。在经过如前所述的 CNN 结构提取特征之后，根据 4.3 节所述，FC1 和 FC2 两层的输出都需要被拿来进行量化，因此对这两层均加入量化损失作为正则项，而只对 FC2 层计算保持相似性的损失函数 PairwiseLoss，最后为了样本点的分类标签，添加与分类类别总数相同的 fc_class 层并进行 Softmax 分类。

[2] https://github.com/lhmRyan/deep-supervised-hashing-DSH

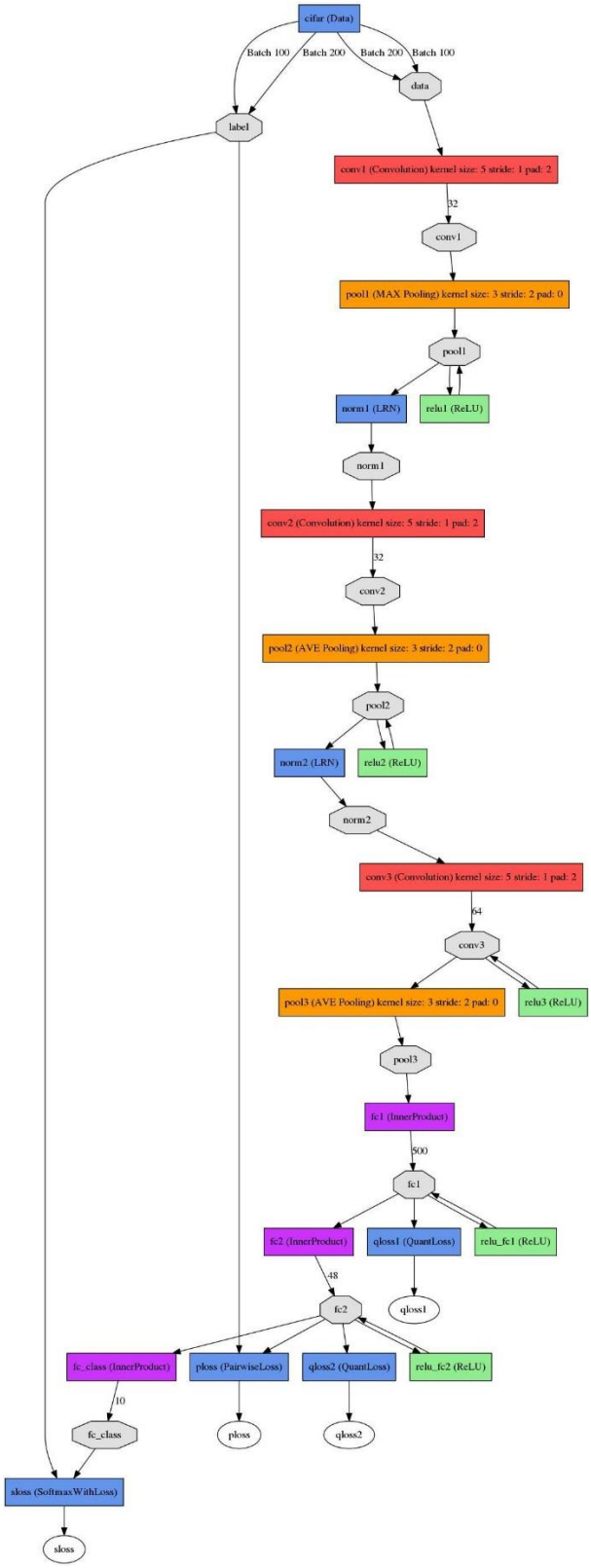

图 5-1 在 Caffe 上实现 DHSR 方法基本示意图

另外，Caffe 本身不支持在输入数据中含有多标签，然而在很多实际场景中，由于每个图像样本的内容很丰富，往往会含有多方面的语义信息，数据中往往都不是简单的单个分类标签，例如图 5-2，该图含有船只，建筑，河流，树林等多个语义信息，无法简单直接地把它归为单独的一个类，这样的场景在机器学习中即为多标签的数据，在我们进行实验的公开数据集中，NUS-WIDE 正是这样一个公开的多标签数据集，每一幅图像可能有多个标签，所以为了实现哈希方法在更广泛的场景中通用，DHSR 方法的实现必须要支持多标签数据。

图 5-2 多语义信息图片

之前有许多相关的工作[74][75]在面临同样问题时采用了重新编写全新的数据层的方式，而我们发现在 Caffe 现有框架下经过合理地变通和复用，完全不修改 Caffe 源码只通过分片层也可以更方便快捷地实现多标签数据层的效果。Caffe 的数据存储单元定义如表 5-1 所示，可以发现其中正常情况下存放标签的“label”定义为 optional，即代表个数为 1 或者 0，造成无法存储多个标签，而另一个“float_data”项则定义为“repeated”，即个数可以为多个，另外经过查找 Caffe 源码，“data”和“float_data”无法同时出现，所以我们放弃正常情况下数据和标签分别存放在“data”和“label”中的方式，而是将它们一起按照先数据后标签的格式存入“float_data”中，

并且在生成数据时，将图片原有的三通道信息顺次在通道（channel）这一维度中存储，即设定通道数为图像所有的像素数，之后在网络定义时再通过分片层（Slice Layer）在通道这一维度上切分，将它们分成数据和标签两部分，再通过重构层（Reshape Layer）将数据部分还原成三通道的形状。

表 5-1 Caffe 数据单元定义

```
message Datum {
  optional int32 channels = 1;
  optional int32 height = 2;
  optional int32 width = 3;
  // the actual image data, in bytes
  optional bytes data = 4;
  optional int32 label = 5;
  // Optionally, the datum could also hold float data.
  repeated float float_data = 6;
  // If true data contains an encoded image that need to be decoded
  optional bool encoded = 7 [default = false];
}
```

另外，对于多数据集中样本点标签的损失函数计算，可以直接将单标签使使用的 SoftmaxWithLoss 层换用为 SigmoidCrossEntropyLoss 层即可，该层支持多标签的数据。

5.2 模型融合

模型融合是在机器学习领域经常用到的综合多个模型的优势来提高模型总体效果的方法，也很早就被用到了神经网络领域，1990 年 Hansen[80]就证明了对神经网络进行融合可以提高总体模型的泛化能力，他证明了在采用绝对多数投票法的情况下，如果单个网络错误率为 p，且网络之间错误不相关，则总的错误率为：

$$\mathrm{p}_{err} = \sum_{k>N/2}^{N} \binom{N}{k} p^k (1-p)^{N-k} \qquad \text{(公式 5.1)}$$

在$p < 1/2$的情况下，p_{err}随 N 的增大而递减，在实际情况下，除了对分类问题常用的绝对多数投票法和相对多数投票法之外，还有在回归问题中常见的（加权）平均法。

在我们的 DHSR 方法中，我们也使用了模型融合的方法来提升整体的实验效果。对于融合的个体，如 4.1 节中所述，CNN 网络在我们提出的只是作为特征学习器，多种不同的 CNN 结构都可以使用，又因为理论上来说越不相关融合的效果越好，所以我们的融合个体可以选择不同的网络结构，这样通过相关性很弱的不同的网络结构可以学习到互补的不同种类的特征；在此基础上，因为神经网络的缺点之一是会陷入局部最优而得不到全局最优的结果，所以我们可以在不同的网络结构中进一步采用不同的随机权值初始化来生成更多模型个体，这样不同的初始化将使神经网络陷入不同的局部最优解，不同的模型差异越大也可以进一步促进模型融合带来的效果提升。

对于个体融合的方法，理论上按标准的方法可以有两种方式：1）直接按单个模型划分哈希码的位数，如有 n 个模型个体，对于 FC1 层的αK位目标哈希码和 FC2 层的K位目标哈希码，将他们分给 n 个个体，这样每个个体模型在 FC1 层设定节点数为$\left\lfloor \frac{\alpha K}{n} \right\rfloor$位，FC2 层设定节点数为$\left\lfloor \frac{K}{n} \right\rfloor$位，最后一个模型设定为相应剩余的所有位数，由每个模型单独决定它分到的部分哈希码；2）每个个体模型仍然设定全部的 FC1 层的αK位和 FC2 层的K位，对于所有位数均采用投票的形式，选择 1 和-1 的结果中的票更多的作为该位哈希码的最终结果。但由于在我们的场景中，方法 2）中不同的模型个体的某一位哈希码的意义并不完全相同，不能按照投票的方式完全对等处理，所以我们选用了方法 1），由每个模型决定部分哈希码，然后直接将它们连接起来合成完整的哈希码作为最终结果。

5. 3 Fine-tuning 两阶段训练法

微调训练（Fine-tuning）借鉴了类似迁移学习的思想，是在深度学习中比较常见的一种加速训练过程以及提升训练效果的方法，在比较复杂的任务以及网络模型中，神经网络中的局部最优解个数也会大幅增加，这样如果数据量没有足够

多，很可能导致神经网络陷入其中无法达到很好的效果，而如果使用先进行预训练（Pre-training），之后再将预训练得到的权重等参数作为初始值，进一步微调训练的方法，可以使得预训练以一个更合理的初始化权值作为优化起点，更可能达到全局最优从而提升效果，另外从特征学习的角度讲，预训练可以得到一些通用的浅层特征，之后再通过微调训练使网络适应于新的任务，而且不用完全重新训练模型也可以提升模型的收敛速度。

在 DHSR 方法的实现过程中，我们采用了两个层面上的微调训练：

1. 标准的微调训练方法。使用在更大规模数据集（常用 Imagenet）上预训练之后的模型作为起始模型，由于本文使用的是目前广泛使用的一些 CNN 结构，而 Caffe 对此在 Model zoo[3]中已经提供了大量对于现有网络在大规模数据集中训练后的模型，可以方便地下载使用，因此本文对使用到的 Cuda-convnet，Alexnet，VGGNet，GoogleNet 等结构均使用该来源的模型。
2. 针对哈希码的微调训练。在需要生成不同位数的哈希码时（例如本文实验部分需要生成不同位数的哈希码比较实验效果），此时可以通过微调训练的方式对网络权重进行复用，先训练较小码长的哈希码，再只修改 FC1 层和 FC2 层的长度进行微调训练，以生成更长的哈希码。

上述两个步骤中，其实都涉及到在微调网络中对模型结构进行部分修改。在第一部分中针对原网络结构，由于数据集不同，原结构中的全连接层 FC1 和 FC2 需要调整以学习目标长度的哈希码，第二部中同样需要修改 FC1 和 FC2 两层的输出大小以适应不同的哈希码长。在这种微调阶段对网络结构也进行了一定调整的情况下，修改的层需要像未经过预训练一样，采用随机的初始化权值，实际应用中往往对于这样修改过的结构单独提升学习速率（Learning Rate），也有方法使得未修改的部分完全停止学习，但实践中设置一个相对较小的速率继续学习效果更好，所以我们的方法中也采用了类似的做法，提升改变的层的学习速率（本方法中设定为其他层的 10 倍），使得修改过的层可以更快地训练得到更好的结果，而

[3] https://github.com/BVLC/caffe/wiki/Model-Zoo

之前的层仍然可以比较好地保留之前学习到的初级特征，帮助新修改的层更好地训练，这样的机制在 Caffe 中可以通过修改该层的 lr_mult 参数实现。

5.4 其他细节

为了使模型达到更好的效果，我们还在实现中加入了一系列优化技巧。

5.4.1 避免过拟合

在机器学习当中，模型如果过于复杂，训练数据量又不够多的话，可能会在训练过程中过度拟合训练数据本身，而没有学习到很好的泛化能力，致使其在没有见到过的数据上表现较差，这种现象就叫做过拟合。在我们的实现中，由于模型融合部分使用了许多不同的模型，而且一些模型结构比较复杂，在部分规模较小的数据集上有过拟合的风险，于是我们采用了一系列措施避免这种现象。

对于数据本身，我们随机对其进行镜面对称，得到像素点信息不同的数据，另外在模型训练时，我们加入了 dropout 层，在每次迭代中以 50%的概率随机隐藏一些神经元，并采用提前停止（early stopping）的方法在测试集效果不再变好时停止训练，还加入了正则项限制模型复杂度。

5.4.2 训练过程优化

为了让模型更稳定迅速地收敛到最优解，我们对数据在每次迭代时均打乱排序，这样不同的批（batch）中的数据不同，训练过程不会出现明显的震荡；另外出于相同的考虑，我们谨慎实验选择了训练中的批大小（batch size），因为选择过小，在整体数据集重复性不强的情况下，会导致无法选到最优的训练方向，而如果选择过大则会加长训练时间，并因为缺少随机性容易陷入局部而非全局最优解；另外对于激活函数我们选用 ReLU，而不是 TanH 或者 Sigmoid 等 S 型曲线函数，以此来避免梯度消失（gradient vanishing）或爆炸的情况，并对于从头开始的随机初始化参数选用了 Xavier 方法替代 Gaussian 方法，通过考虑每个神经元的入度和出度的信息来更合理地初始参数，还使用了动量（momentum）的方法来使训练过程更稳定。

5.5 本章小结

本章介绍了 DHSR 方法的实现细节与优化策略，5.1 节介绍了整个 DHSR 方法的基本实现方法及平台，5.2 节介绍模型融合策略的使用，5.3 节给模型加入预训练和微调训练的两阶段机制，提高训练效果和速度，最后在 5.4 节中介绍了一些优化模型的策略和技巧。

第6章 实验结果与分析

6.1 实验配置

为了测试本文提出的 DHSR 方法的检索效果和时间效率，我们将 DHSR 方法与目前最先进的哈希方法在三个公开的权威图像数据集上进行了一系列实验对比。

6.1.1 实验环境

整个实验在一台搭载 GPU 的单机上运行，且 Caffe 代码在 GPU 模式下运行，该机器的软硬件配置如表 6-1 所示。

表 6-1 实验机器软硬件配置

项目	内容
CPU	Genuine Intel(R) CPU @ 2.60GHz ×32
内存	64GB DDR3 @ 1333MHz
GPU	GeForce GTX Titian X
显存	12GB DDR5, 10Gbps
操作系统	Ubuntu 14.04 LTS
内核版本	GNU/Linux 3.13.0-24-generic x86_64
Cuda	Cuda7.5 with cudnn
数据处理	Python2.7.6, Matlab r2014a, gcc 4.8.4 etc.

6.1.2 实验对比算法

我们将提出的 DHSR 方法，在检索质量相关的一些指标和时间效率等方面，与目前最先进的 12 种哈希方法进行了比较，包括局部敏感哈希（LSH）[1]，两种无监督学习算法迭代量化法（ITQ）[3]，谱哈希（SH）[5]，四种有监督学习算法，二值重建嵌入（BRE）[9]，最小损失哈希（MLH）[42]，典型关联分析迭代量化法（CCA-ITQ）[3]，监督核哈希（KSH）[2]，还有五种最新的基于深度学习的哈希算法，包括 CNNH 方法[71]，CNNH$^+$方法[71]，DLBHC[73]方法，DNNH 方法[72]，

DSH[74]方法。

6.1.3 实验数据

我们的实验在三个权威的公开数据集上进行：

MNIST[81]数据集由 0-9 共 10 个数字的手写图片组成，共有 60000 张训练图像，10000 张测试图像，所有的数字被标准化为 28×28 大小的灰度图片。

CIFAR10[82]数据集包括 10 种对象分类，每个分类包括 6000 张图像，全集共 60000 张图像被分为训练集的 50000 张和测试集的 10000 张，所有的图像被大小为 32×32。

NUS-WIDE[83]数据集是一个公开的网页多标签图片数据集，它包含从 Flickr 上收集的将近 270000 张图像，数据集中共有 81 个语义概念，每张图像有其中的一个或多个语义标签。为了公平的对比，我们采取大部分深度学习哈希方法的实验设定[72]，只选择了其中频率最高的 21 个语义标签（每个标签最少出现在 5000 张以上的图像中），并选择有这些标签的图像子集进行实验，并将其大小调整为 256×256。

具体来说，遵循普遍采用的方法[72]，对于 MNIST 和 CIFAR-10 数据集，我们随机选择了每个类 100 张共计 1000 张图像作为测试检索项集。对于 LSH 和非监督训练方法，我们将剩下的所有图像作为训练集；而对于有监督的方法，我们随机从剩下的图片中选择每个类 500 张共计 5000 张图像作为训练集。样本对标签集合$\mathcal{S}$根据图像的类标签得到，即两张属于同一类的图像相似，反之则不相似。

对于 NUS-WIDE 数据集，我们从选定的 21 个标签中每个随机选择 100 张有该标签的图片组成测试检索项集，共 2100 张，对于 LSH 和非监督训练方法，21 个选定标签相关的所有图像组成训练集；对于有监督的方法，我们从选定的 21 个标签中每个随机选择 500 张图像组成训练集。样本对标签集合$\mathcal{S}$根据图像的语义标签得到，即两张图像只要有至少一个公共的标签，即认为这两张图像相似，反之则不相似。

另外，对于我们的 DHSR 方法和所有基于深度学习的哈希方法，我们直接使用图像像素作为输入，而在传统的哈希方法中，我们用一个 512 维的 GIST 向量

来代表 MNIST 和 CIFAR-10 中的每张图像，用一个 500 维的词袋向量来表示 NUS-WIDE 中的每张图像。

6.1.4 参数设定

我们在单模型实验中使用 Alex Krizhevsky 在 Cuda-convnet[4]中使用的网络结构，在模型融合时使用这种结构和 Alexnet[65]共两种 CNN 网络结构，并直接使用 Caffe 提供的这些网络结构经过大规模数据集上训练之后的模型（对比方法同样使用预训练后的模型），根据哈希码位数改变 FC1 层和 FC2 层结构，由于新的层中参数是从完全随机开始学习，所以在训练时将其学习率设置为未改动网络层的 10 倍。

在训练时，我们使用批量随机梯度下降（Batch Stochastic Gradient Descent）的方法，根据 5.4.2 节中所述经过多次实验选择效果最好的批大小 200。此外，动量（Momentum）设置为 0.9，权重衰减参数（Weight Decay）设为 0.001。由于模型大部分已经过预训练，可被认为已经在训练全程中处于后期，所以选用较小的起始学习速率（Learning rate）为10^{-4}，并利用 Caffe 中的 Multistep 机制，在 60000 次和 70000 次迭代之后将学习速率降低为之前的 10%（共 80000 次迭代），在公式 4.5 中的阈值参数 t，我们根据[74]中的建议，将其设为$t = 2K$，这样可以使得神经网络对不相似的样本在哈希后的哈希码不到 K/2 位不一样的情况加以惩罚，损失函数中的各项损失因子均通过交叉验证的方法选择。

6.2 实验评判标准

为了公平地比较，我们遵循之前相关论文的评判标准。值得一提的是，由于本文 4.3 节中提出的方法是一种综合了哈希表查找和哈希码排序的方法，且使用了两组哈希码，而之前的论文中大多只使用了一组哈希码，且评测标准均是单独针对哈希表查找或哈希码排序一个过程的，所以在实验中我们先将不采用该策略的方法作为 DHSR 方法与其他常见的方法在和之前完全相同的环境和评测标准下进行比较，最后再在完整的检索场景下（即使用相同的总哈希码长前提下，比较

[4] https://github.com/akrizhevsky/cuda-convnet2

返回 topN 检索结果的准确率和检索时长），单独地对比该模块会对整个系统哈希过程带来的改进。

之前的文章中通用的评测标准包括：1）哈希表查找方面是不同长度哈希码时，查找所有汉明距离为 2 之内的表项中结果的准确率；2）哈希码排序方面包括 48 位哈希码时对于不同的检索结果数目的准确率曲线，以及不同长度哈希码时的平均准确率（MAP）。

6.3 实验结果及分析

6.3.1 哈希算法对照实验

采用不同的哈希码长度位数进行哈希码排序，在三个数据集上的平均准确率（MAP）结果分别如表 6-2，表 6-3 和表 6-4 所示，其中对于 NUS-WIDE 数据集，我们采用排序的前 5000 个结果来计算。可以看出，我们提出的 DHSR 方法在三个数据集上，都比目前为止最先进的哈希方法表现出了一定程度的提升。

表 6-2 MNIST 数据集上基于不同位数的哈希码排序的 MAP 值

方法	12 位	24 位	32 位	48 位
LSH	0.187	0.209	0.235	0.243
ITQ	0.388	0.436	0.422	0.429
SH	0.265	0.267	0.259	0.250
BRE	0.515	0.593	0.613	0.634
MLH	0.472	0.666	0.652	0.654
CCA-ITQ	0.659	0.694	0.714	0.726
KSH	0.872	0.891	0.897	0.900
CNNH	0.957	0.963	0.956	0.960
CNNH$^+$	0.969	0.976	**0.971**	0.975
DLBHC	0.961	**0.977**	0.965	0.976
DNNH	0.953	0.963	0.964	0.971
DSH	0.949	0.968	0.967	0.970
DHSR	**0.972**	0.973	0.970	**0.981**

表 6-3 CIFAR-10 数据集上基于不同位数的哈希码排序的 MAP 值

方法	12 位	24 位	32 位	48 位
LSH	0.121	0.126	0.120	0.120
ITQ	0.162	0.169	0.172	0.175
SH	0.131	0.135	0.133	0.130
BRE	0.159	0.181	0.193	0.196
MLH	0.182	0.195	0.207	0.211
CCA-ITQ	0.264	0.282	0.288	0.295
KSH	0.303	0.337	0.346	0.356
CNNH	0.439	0.511	0.509	0.522
$CNNH^{+}$	0.465	0.521	0.521	0.532
DLBHC	0.533	0.550	0.552	0.566
DNNH	0.552	0.566	0.558	0.581
DSH	0.623	0.647	0.660	0.671
DHSR	**0.646**	**0.673**	**0.679**	**0.703**

表 6-4 NUS-WIDE 数据集上基于不同位数的哈希码排序的 MAP 值

方法	12 位	24 位	32 位	48 位
LSH	0.403	0.421	0.426	0.441
ITQ	0.452	0.468	0.472	0.477
SH	0.433	0.426	0.426	0.423
BRE	0.485	0.525	0.530	0.544
MLH	0.500	0.514	0.520	0.522
CCA-ITQ	0.435	0.435	0.435	0.435
KSH	0.556	0.572	0.581	0.588
CNNH	0.611	0.618	0.625	0.608
$CNNH^{+}$	0.623	0.630	0.629	0.625
DLBHC	0.630	0.641	0.655	0.653
DNNH	0.674	0.697	0.713	0.715
DSH	0.660	0.701	0.728	0.732
DHSR	**0.680**	**0.703**	**0.741**	**0.743**

具体来说，相比使用人工特征的最好的传统哈希方法 KSH，DHSR 方法在 MNIST，CIFAR-10 和 NUS-WIDE 三个数据集上分别实现了对不同位数的情况求平均之后，7.8%，33.8%和 14.2%的绝对数值提升。事实上不只是本文提出的 DHSR 方法，表中所有基于深度学习的方法都可以得到比传统方法更好的准确率，这也印证了利用 CNN 网络和深度学习的方法，可以从算法原理设计和特征提取方面促进哈希函数的学习过程。

而对于到目前为止最新提出的基于深度学习的方法 DSH，DHSR 则分别实现了 1.0%，2.5%和 1.2%的平均绝对数值提升。这样的结果说明在模型中综合样本对标签和样本点标签可以得到更好的效果。CNNH$^+$方法中也有这样的想法，但由于它的哈希函数学习过程和特征学习过程是割裂在两个不同的阶段中的，无法相互促进，所以也没有得到很好的效果。而对于 DLBHC 方法来说，只单方面考虑样本点的语义而不针对样本对的距离进行优化，这导致只要没有完全影响分类结果，对不相似的样本对生成相似的哈希码或者对相似样本对生成不相似的哈希码都不会增大损失函数的值，但这些恰恰都会直接对检索结果造成不好的影响。另外对于 DNNH 方法，它虽然考虑了量化误差，但只使用样本三元组标签未结合个体的语义信息，而且在实验数据条件下，数据原始都只有样本点标签信息，样本对标签和三元组标签都是由其生成的，所以它并不能为模型带来更大的信息量，而且更加难于训练，而在点击数据等的处理场景中，这样的标签有可能带来更大的信息量。

可以特别地发现，MNIST 数据集由于比较简单，原先方法已经能达到很高的准确率，所以提升并不明显。而在表中，考虑了样本点信息的 CNNH$^+$，DLBHC 和 DHSR 方法在这样的数据集中取得了更好的效果，说明在较为简单的数据条件下，图像的意义可简单地由单个分类完全表达，这种情况只要能很好地刻画单个样本点的语义即可得到很好的检索效果。

另外，对于 CIFAR-10 数据集上的结果，可以看到我们的 DHSR 方法在 24 位哈希码时的准确率就已超过了其他方法生成 48 位哈希码的准确率，这也说明 DHSR 方法可以生成更加紧致的哈希码。

除此之外，图 6-1 是在 CIFAR-10 数据集中，使用 48 位哈希码时对于不同的检索结果数目的准确率曲线。

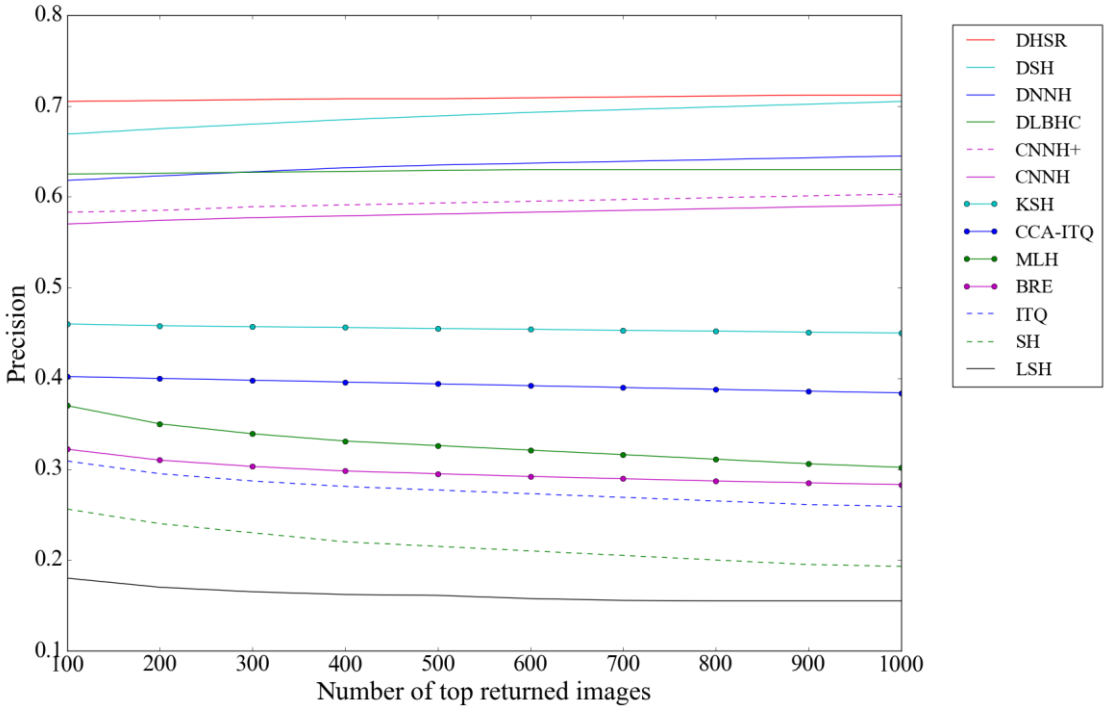

图 6-1 48 位哈希码时对于检索结果数的准确率曲线

从图中可以看出，基于深度学习的哈希方法依然在较大程度上胜过传统的人工特征的方法，体现出了利用深度学习提取特征和进行哈希函数学习的价值。另一方面，基于深度学习的哈希方法在检索结果数增多时，不会像大多数传统哈希方法一样准确率下降，而是可以保持平稳甚至有所提升，这也证明对于较多数返回结果的范围内，基于深度学习的方法在相对靠后的排序位置上依然保持了较高的准确率，而传统方法则有所下降。具体在基于深度学习的方法当中，我们在文中提出的 DHSR 在准确率方面也胜过了已有的基于深度学习的方法，而且随着检索结果数的增长，准确率保持稳定。

另外，哈希码在只进行哈希表查找时的检索效果也非常重要，因为这样的检索过程只需要花费常数复杂度的时间，图 6-2 为在 CIFAR-10 数据集中，使用不同位数哈希码的情况下，汉明距离距检索项为 2 之内的哈希表表项中的结果的准确率曲线，从中可以看出，在哈希码位数较大（48 位）时，汉明距离空间已相当

稀疏，汉明距离为 2 之内的样本数较少，大多数哈希方法已无法保持最接近的哈希码对应于实际相似的结果，即失去了语义层面上的最近邻的相似度保持，而我们的方法则依然取得了比较好的准确率。

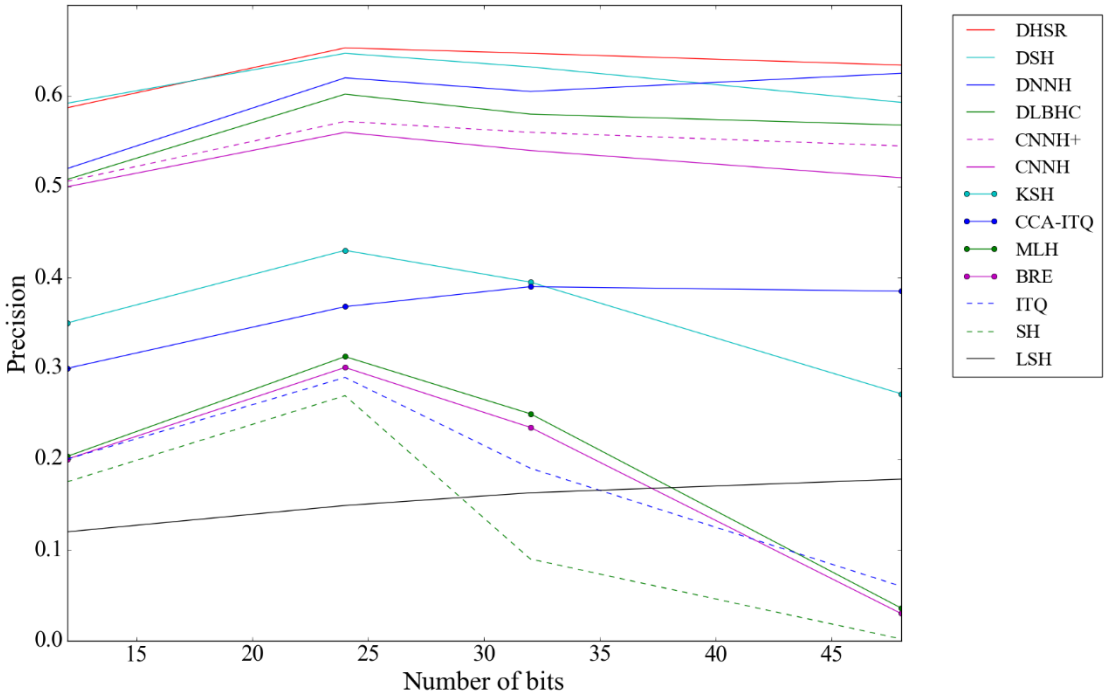

图 6-2 汉明距离为 2 之内的检索结果的准确率曲线

6.3.2 结合多种标签信息

在 4.2.3 节中我们提出了结合样本点语义标签和样本对标签的优化函数，在本节中我们通过实验证明其有效性。不失通用性的，我们在 CIFAR-10 数据集上进行实验，对于结合了样本点标签和样本对标签两者的 DHSR，我们提出两种变体，分别是去掉单个样本点标哈希签信息（$\beta = 0$）的 DHSR-S 和样本对标签信息（$L_1 = 0$）的 DHSR-D，其他实验设置和环境及数据均完全相同。以上操作均可以如 5.1 节中所述，在 Caffe 中通过去掉 SoftmaxWithLoss 和 PairwiseLoss 层方便地实现，分别对他们在不同的哈希码长度情况下求平均准确率，结果如表 6-5 所示。

表 6-5 DHSR 及其两种变体在不同哈希码长度下的平均准确率

方法	12 位	24 位	32 位	48 位
DHSR	**0.646**	**0.673**	**0.679**	**0.703**
DHSR-S	0.643	0.660	0.663	0.675
DHSR-D	0.547	0.572	0.579	0.578

从中可以看出，去掉样本点标签和样本对标签两者中的任何一部分，哈希检索的准确率都会下降，充分说明了这两部分都对提升哈希码的准确率有所贡献。进一步两相比较可以发现，去掉样本对标签信息又比去掉样本点标签信息造成的损失更大，这是因为相比样本点标签，样本对标签的信息更接近于检索任务中重点衡量的相对位置关系，样本点标签的加入可以辅助 CNN 网络学习到更好的保持个体语义的特征，这也与我们在通过交叉验证选取参数β时观察到的现象相一致，即在样本对标签信息占主导地位的情况下，哈希检索的效果最好。

6.3.3 模型融合

如 5.2 节中所述，在很多机器学习的应用场景中，多个模型融合都可以提升整体的最终结果，本节中我们也通过实验探究其在利用深度学习进行哈希这一场景中的作用。在之前的实验中，为了和之前方法的实验公平的对比，我们只使用了单模型作为 DHSR 方法的结果，在此基础上我们对不同哈希码位数的情况下，融合多个模型的结果进行了实验，其中涉及到包括 Cuda-convnet 结构和 Alexnet[65] 共两种 CNN 网络结构，每种网络结构负责一半的哈希码，我们使用 C 代表只单独使用 Cuda-convnet 结构且无模型融合，A 代表只单独使用 Alexnet 且无模型融合，C+E 代表使用 Cuda-convnet 且对不同的哈希位使用不同的权值初始化，A+E 代表使用 Alexnet 且对不同的哈希位使用不同的权值初始化，C+A 代表使用 Cuda-convnet 和 Alexnet，C+A+E 代表使用 Cuda-convnet 和 Alexnet 且对不同的哈希位使用不同的权值初始化。在 CIFAR-10 数据集中使用 48 位哈希码时进行试验，结果如图 6-3 所示。

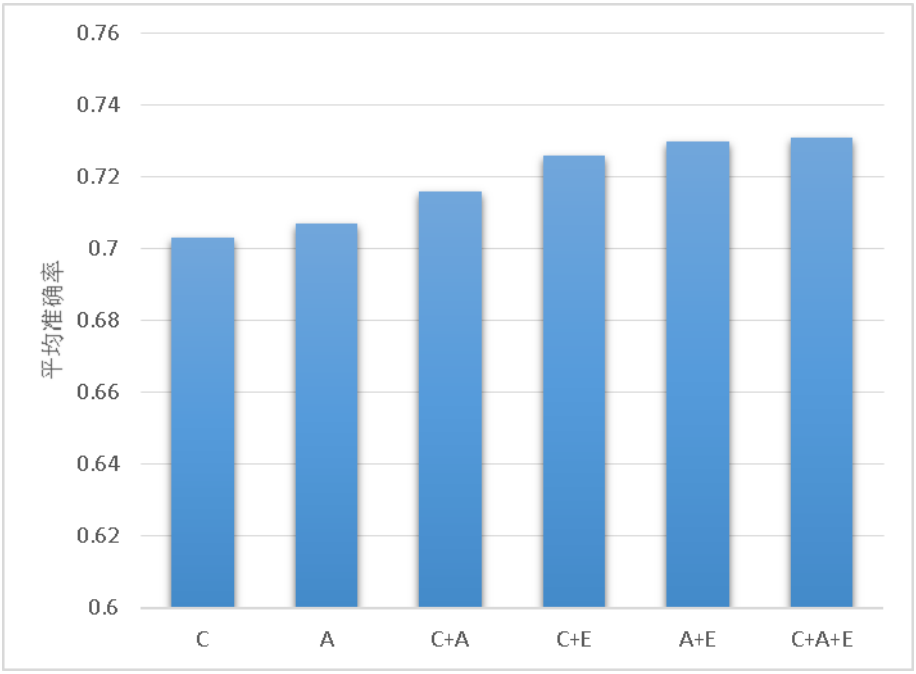

图 6-3 各种网络融合变体在不同长度哈希码条件下的平均准确率

从图中可以看出，由于模型更加复杂，单独使用 Alexnet 比单独使用 Cuda-convnet 效果稍好，但差距不大，而使用了两种模型融合的结果已有比较明显的提升，如果对每个模型内部的各个哈希位再使用不同的权值初始化方式，相当于实现更多种不同模型的融合，检索的平均准确率效果将相比完全不使用模型融合达到最多 2.8%的绝对数值提升，说明使用模型融合的方法确实可以提升系统整体的检索效果，且融合不同网络结构以及对每种结构内部的各个哈希位使用不同的权值初始化这两种融合方式都对模型的效果提升有所帮助。不同的网络结构更可能学习到互补的不同种类的特征，而采用不同的权值初始化则可以使模型综合优化到更好的结果。

6. 3. 4 复合哈希码层次检索

对于 4.3 节中提到的复合哈希码层次检索策略，我们在本节中单独对其通过实验测试效果。在完整的检索场景中，检索系统需要在数据库中搜索出与用户的检索项最为相似的样本，作为检索结果返回给用户。对于使用哈希码进行检索的方法，大多分为通过哈希码排序和哈希表查找两种方式获得最后结果，而在本文

中提出的方法综合使用了这两种方式，并需要使用一共两组哈希码。

在空间有限的条件下，为了公平地比较，在实验中我们设定本文提出方法的两组哈希码长度之和与其他不使用这一方法的单组哈希码长度相等，以保证他们占用完全相同的空间。未采用复合哈希码的方法 DHSR 如之前实验中的设定，加入复合哈希码的方法称为 DHSR+C。我们在 CIFAR-10 数据集上进行实验，类似于 6.3.1 节，我们对比只使用哈希表查找的方法测试使用不同位数哈希码的情况下，汉明距离距检索项为 2 之内的哈希表表项中的结果的准确率曲线如图 6-4 所示。

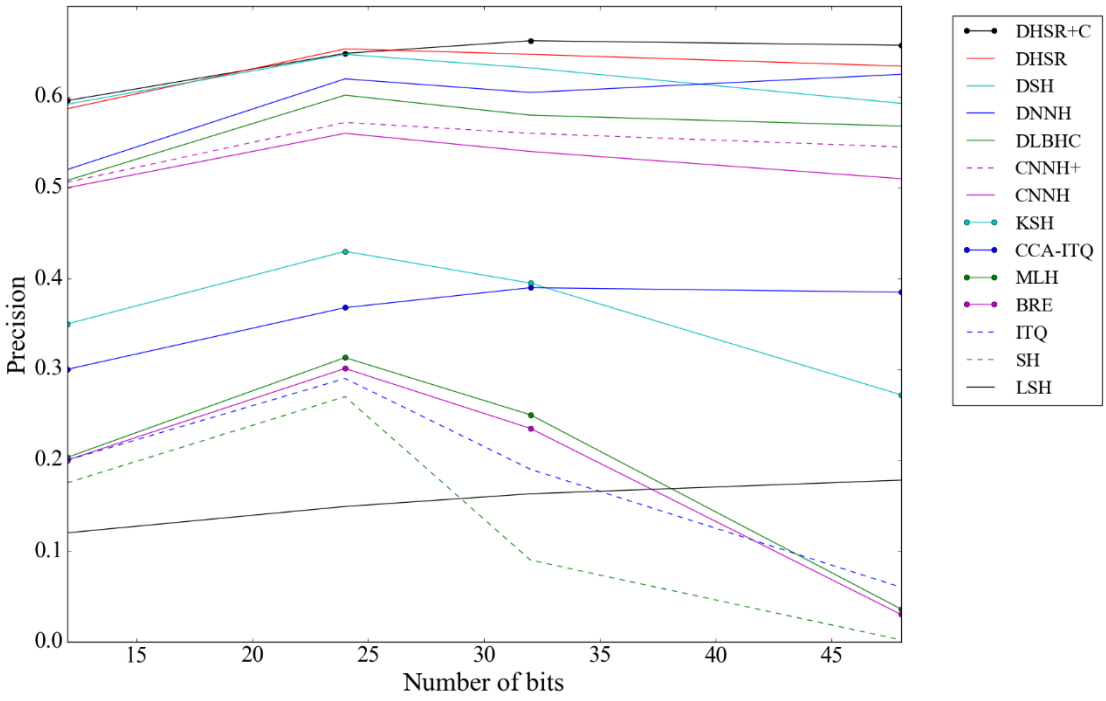

图 6-4 汉明距离为 2 之内的检索结果的准确率曲线

值得一提的是，在传统方法中由于哈希码位数较长时，每个哈希表项中的样本数很少，所以为了足够数量的结果以及提高召回率，必须要将检索范围扩大到汉明距离为 2 或更大的范围之内寻找样本，而复合哈希码方法中由于第一层哈希码的位数较短其实并不需要如此操作就可以得到足够多的返回结果。在实验中，由于我们的复合方法是经过哈希码排序后得到最终结果，也为了更贴近真实的检索场景，我们选取返回相同结果数时的准确率作为比较，即对于不使用复合哈希

码机制的 DHSR 方法，我们使用汉明距离为 2 之内的哈希表表项中的结果统计准确率，而对于 DHSR+C，我们统计同样结果数条件下的准确率。

图中结果显示，加入复合哈希码机制的 DHSR+C 方法相比 DHSR 可以结合哈希表查找和哈希码排序两种方法的优点，取得更好的效果，特别是在位数较多时，可以很好地解决汉明距离空间稀疏条件下，最近邻的相似度保持较差的问题。

另外对比只使用哈希码排序的方法，我们测试在使用 48 位（DHSR+C 为 12 位+36 位）和 128 位（DHSR+C 为 32 位+96 位）哈希码时对于不同的检索结果数目的准确率曲线分别如图 6-5，图 6-6 所示。由于 6.3.1 节中其他作为对比的方法均为在 48 位哈希码条件下的实验结果，而且方法不同无法单独反映出复合哈希码机制的表现，所以我们此处只比较了 DHSR 和 DHSR+C 两种方法的结果。

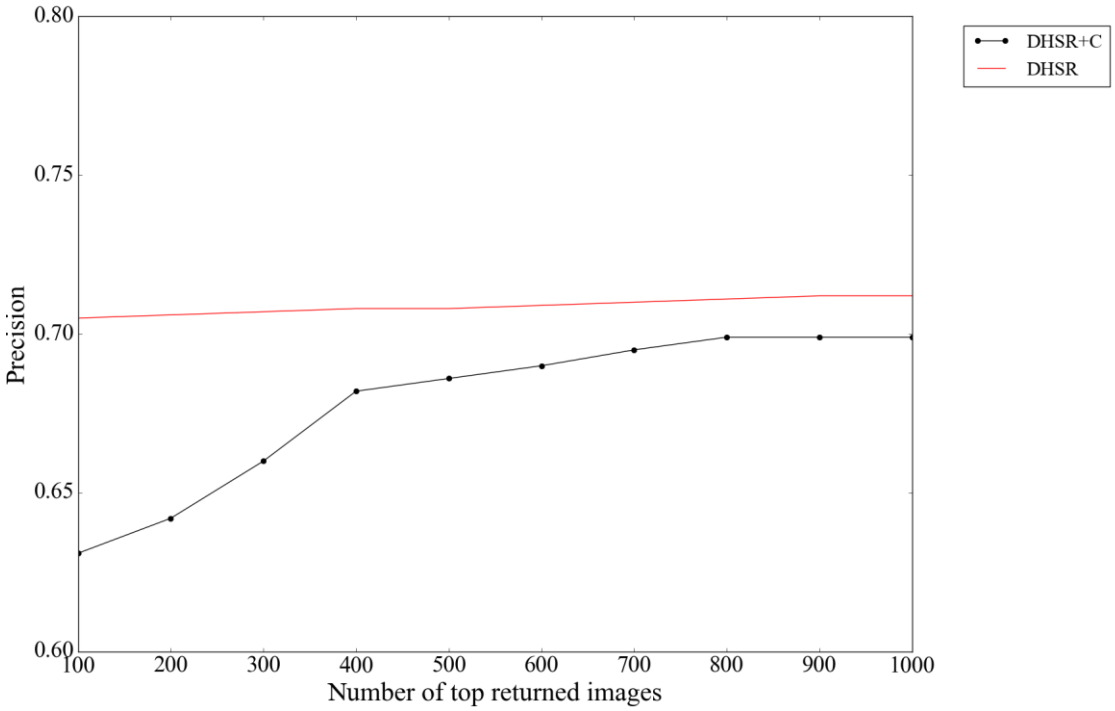

图 6-5 使用 48 位哈希码时对于检索结果数的准确率曲线

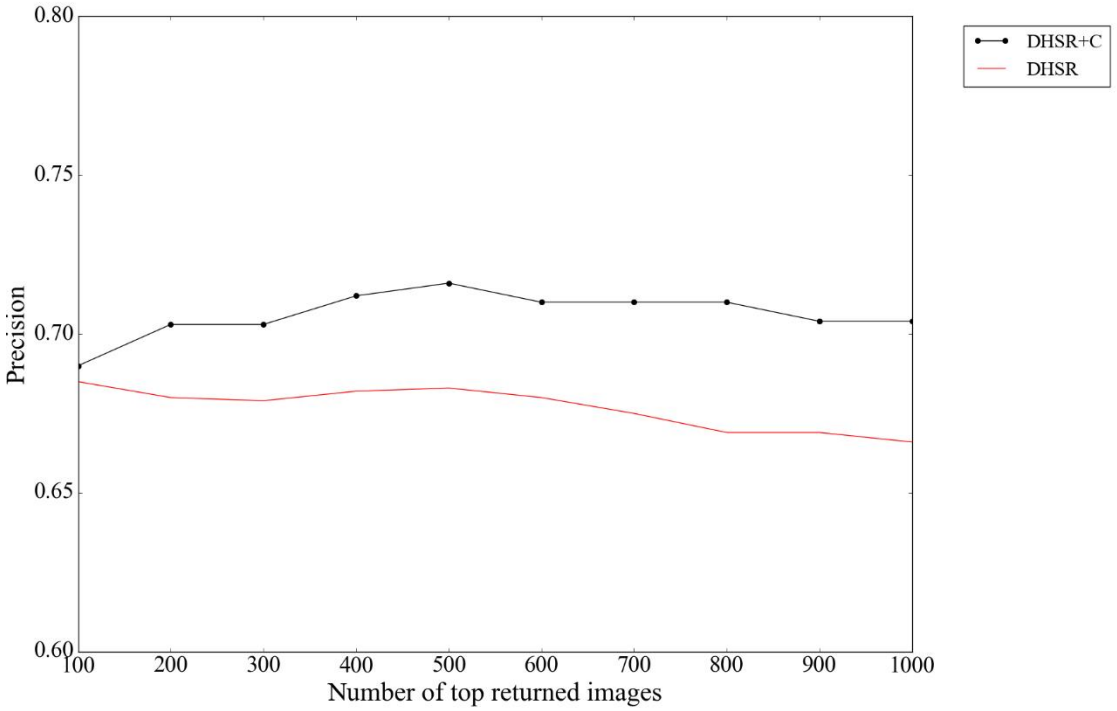

图 6-6 使用 128 位哈希码时对于检索结果数的准确率曲线

图中结果显示，在总位数较低（48 位）时，使用复合哈希码的方法会带来准确微小的下降，但在总位数较高（128 位）时，使用符合哈希码策略可以提升检索的准确率，这是由于神经网络在学习过程中位数较大时可能存在过拟合现象，而且观察图 6-6 可以看出，哈希码位数较高时，对于排序相对靠后的结果，对应汉明距离中等大小的样本已经无法保证可以比汉明距离更大的样本保持更好的相似性，故准确率进一步下降，而使用复合哈希码策略使得这种问题得到缓解。而在位数较少（48 位）时，由于我们的复合策略首先将范围限定为短哈希码（12 位）同表项内的数据，之后在对其进行利用长哈希码（36 位）的排序，相比原先对全局进行更长哈希码排序的方式，损失了精度，而且图中随着返回结果数的增长，准确率也有所提升，说明精度损失更多来源于使用哈希码排序环节使用的哈希码长度较短造成的排序上的不合理。

另一方面，在检索时间上，我们取在 CIFAR-10 数据集上按 6.1.3 节所述设定的测试检索项集，平均得到单个检索项的检索时间如图 6-7 所示。

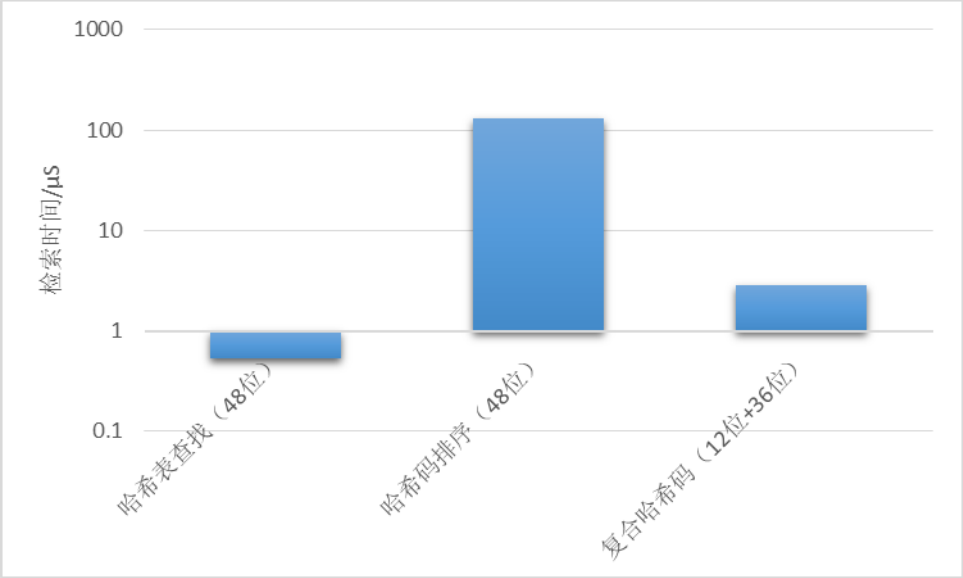

图 6-7 不同算法中单个检索项的检索时间

图中，说明复合哈希码机制在检索时间上略高于只为常数时间复杂度的哈希表查找，但是相比哈希码排序的方法，效率得到了很大提高。

综上可知，复合哈希码层次检索的策略，在使用相同空间的前提下，相比只使用哈希表查找的方法改善了其准确率低的问题；比较只使用哈希码排序的方法，在位数较多时效果更好，在位数较少时，效果有微小的下降，但是其在检索时间上比哈希码排序方法始终具有很大优势。

而如果在空间充足的条件下，从网络结构对等的角度出发，由于本文方法中使用了分块全连接模块，这会对网络结构以及 FC 层节点数造成影响，而在对比方法中只有 DNNH 使用了这种结构，所以此处只对比 DNNH 和未使用复合哈希码机制的 DHSR 以及使用了复合哈希码策略的 DHSR+C 三种方法。在生成 12 位哈希码时，DNNH 中使用的网络结构中最后两个全连接层 FC1 层和 FC2 层长度分别为 600 和 12，当我们采用与之完全相同的网络结构，只是比它多提取利用了 FC1 层的信息，两者的准确率对比如图 6-8 所示。

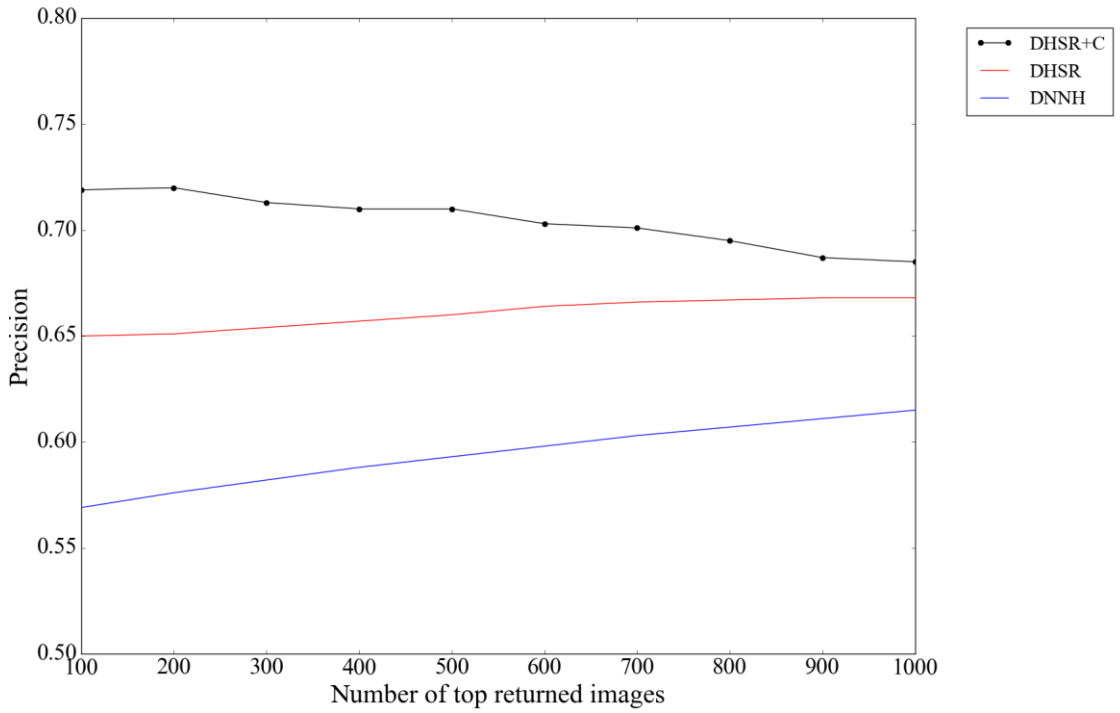

图 6-8 相同网络结构前提下对于检索结果数的准确率曲线

从图中可以看出复合哈希码机制先使用 12 位哈希码进行哈希表查找，之后再使用 600 位哈希码进行哈希码排序，多利用了 FC1 层输出生成的 600 位哈希码信息，取得了更好的效果，证明了这部分输出也可以保持样本的相似性并用来帮助取得更好的检索效果，但同时我们为了与 DNNH 算法完全相同条件下进行比较，将 FC1 层的哈希码位数设置为 600，这样的位数在实验条件和情境下过长，对返回数增加时的准确率有所影响，实际应用中可以根据实际需要选择更为合理的哈希码位数。

6.3.5 哈希码生成时间

除去检索时间，对新数据生成哈希码的时间也是实际应用中必须要考虑的一部分。本节中我们在完全相同的实验环境和设定下比较了在生成 48 位哈希码情况下，本文方法 DHSR，加入模型融合（12 种模型进行融合）的 DHSR+E 和 CNNH，DLBHC，DNNH，DSH 这几种基于深度学习的方法以及传统哈希方法 MLH，CCA-ITQ，KSH 和 BRE 的编码时间，结果如图 6-9 所示。

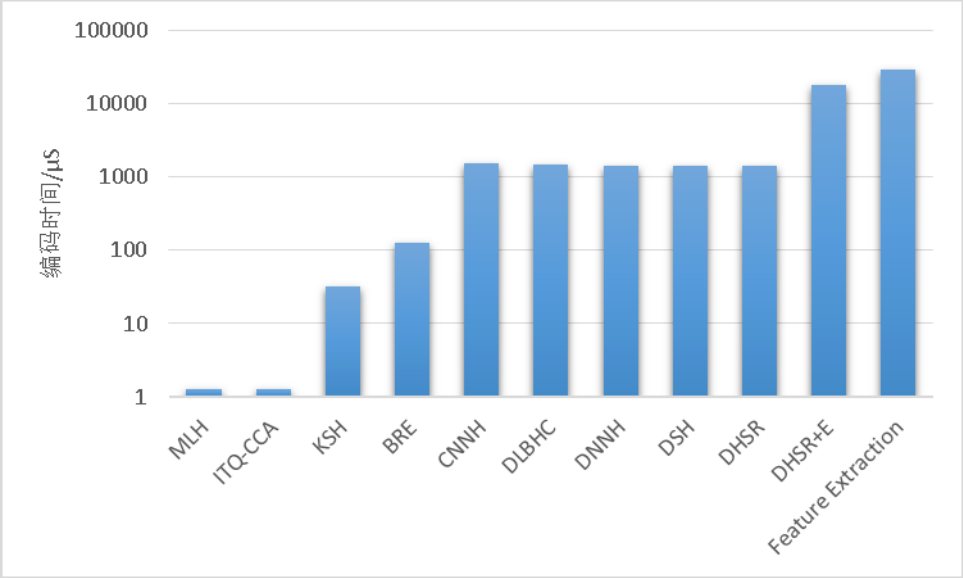

图 6-9 不同算法对于新数据的哈希码生成时间

从图中可以看出基于 CNN 的算法比传统方法纯编码过程用时更多，但如果在传统编码过程中算上提取人工特征的时间，则总时间比基于 CNN 的算法更多；另外，在基于 CNN 的方法中加入模型的融合会在一定程度上增大编码时间，但是仍然比传统编码过程中算上提取人工特征的时间的总时间要短，这也进一步说明了基于深度学习的方法，以及加入模型融合的方法，不仅准确率高于传统方法，对新的数据生成哈希码的整个过程也更加高效。

6.4 本章小结

本章中我们通过实验的方式证实了我们的 DHSR 方法的有效性，从介绍数据集和实验环境与设置开始，在三个公共数据集 MNIST，CIFAR-10，NUS-WIDE 上将我们的算法与目前为止最先进的方法进行了多方面的评测。6.1 节中介绍了实验环境和各种设置，以及实验用到的数据集和作为对比的现有算法。6.2 节中说明了实验采用的评价标准和方法。6.3 节中介绍了 DHSR 方法与 12 种现有哈希算法的比较结果，并对于 DHSR 方法中几个重要的模块单独实验，在检索准确率和用时等方面证明了其有效性。

第7章 总结与展望

7.1 工作总结

本文主要研究在互联网等大规模图像样本检索的场景下，既保持较好准确率，又提升检索效率的哈希检索算法。

具体来说，本文的主要工作可以总结为以下几点：

1. 提出了一种基于深度学习的哈希方法 DHSR，实现了完全端到端的训练学习，通过卷积神经网络的强大表达能力和特征学习能力，将对图像有用特征的提取以及哈希函数的学习在同一个过程中完成，使得两部分可以相互反馈促进。
2. 结合了样本点以及样本对标签两种监督信息，与量化误差一起构成损失函数，一方面通过样本对标签直接学习与检索场景直接相关的相对距离问题，使得在哈希前后，不同样本对的基本位置和距离关系得到很好的保持；另一方面通过结合样本点标签，可以辅助 CNN 网络学习到更好的保持个体语义的特征，帮助提高模型的表达能力和检索性能。
3. 提出了一种基于神经网络的复合哈希码层次检索策略，通过单次训练神经网络生成两组哈希码分别用于层次进行哈希表查找和哈希码排序，并结合这两种基本方法的优点，可根据实际情况灵活地使用提升检索准确率或减少检索时间。
4. 采用分块全连接模块代替全连接部分，生成各位更加不相关的哈希码等策略优化模型结构，并通过一系列优化技巧提升算法的稳定性和准确率。
5. 将本文算法应用于三个公开图片数据集，通过大量实验，与各种已有的传统哈希方法和基于深度学习的哈希方法对比，结果证明文中提出的 DHSR 方法相比现有方法，在准确率和检索时间方面都具有优势。

7.2 未来展望

本文提出了在图像检索场景中，基于深度学习的哈希方法 DHSR，并从使用哈希算法的检索系统涉及的多个方面介绍了该方法的原理和实验效果，在其中的

许多方面，DHSR 方法都还有可以进一步优化扩展的内容。

在 CNN 特征学习方面我们可以使用更为复杂的 CNN 模型来提取更有效的图像特征服务于哈希函数学习，在损失函数的设计方面，对于本文提出的复合哈希码层次检索机制，DHSR 方法并没有对两组哈希码全部直接进行优化而是对其中一组进行隐式地优化，可以进一步探索与这种复合哈希码机制更匹配高效的损失函数，在样本标签信息的利用方面，由于实验用测试集都是标准的样本点标签测试集，所以本文使用的样本点标签和其生成的样本对标签已经可以反映数据集的标签信息，而在在线点击日志系统等真实场景中，可能会出现 3.2 节中所述的三元组标签，即图像I和I^+的相似度要比I和I^-的相似度更大，这时将三元组标签信息设计入优化目标中可以更好地利用数据标签信息。

另外，在其他的优化和实现方面，本文中提出的较为基本的模型融合方法已经收到了较好的效果，今后可以进一步探索通过遗传算法等方式生成融合个体模型等方法，并研究更有效的预训练-微调训练机制。

参考文献

[1] Gionis A, Indyk P, Motwani R. Similarity search in high dimensions via hashing[C]//VLDB. 1999, 99(6): 518-529.

[2] Liu W, Wang J, Ji R, et al. Supervised hashing with kernels[C]//Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012: 2074-2081.

[3] Gong Y, Lazebnik S. Iterative quantization: A procrustean approach to learning binary codes[C]//Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. IEEE, 2011: 817-824.

[4] Kulis B, Grauman K. Kernelized locality-sensitive hashing for scalable image search[C]//2009 IEEE 12th international conference on computer vision. IEEE, 2009: 2130-2137.

[5] Weiss Y, Torralba A, Fergus R. Spectral hashing[C]// Advances in neural information processing systems. 2009: 1753-1760.

[6] Salakhutdinov R, Hinton G E. Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure[C]//AISTATS. 2007: 412-419.

[7] Wang J, Kumar S, Chang S F. Semi-supervised hashing for large-scale search[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(12): 2393-2406.

[8] Wang J, Kumar S, Chang S F. Sequential projection learning for hashing with compact codes[C]//Proceedings of the 27th international conference on machine learning (ICML-10). 2010: 1127-1134.

[9] Kulis B, Darrell T. Learning to hash with binary reconstructive embeddings[C]//Advances in neural information processing systems. 2009: 1042-1050.

[10] Smeulders A W M, Worring M, Santini S, et al. Content-based image retrieval at the end of the early years[J]. IEEE Transactions on pattern analysis and machine intelligence, 2000, 22(12): 1349-1380.

[11] Oliva A, Torralba A. Modeling the shape of the scene: A holistic representation of the spatial envelope[J]. International journal of computer vision, 2001, 42(3): 145-175.

[12] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]// 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). IEEE, 2005, 1: 886-893.

[13] Bentley J L. Multidimensional binary search trees used for associative searching[J]. Communications of the ACM, 1975, 18(9): 509-517.

[14] Ciaccia P，Patella M，Zezula P. M-tree：An Efficient Access Method for similarity search in Metric spaces[C] // Proceedings of the International Conference on Very Large Data Bases．1997

[15] Beygelzimer A, Kakade S, Langford J. Cover trees for nearest neighbor[C]//Proceedings of the 23rd international conference on Machine learning. ACM, 2006: 97-104.

[16] Indyk P, Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality[C]//Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM, 1998: 604-613.

[17] Wang J, Zhang T, Song J, et al. A Survey on Learning to Hash[J]. arXiv preprint arXiv:1606.00185, 2016.

[18] Charikar M S. Similarity estimation techniques from rounding algorithms[C]// Proceedings of the thiry-fourth annual ACM symposium on Theory of computing. ACM, 2002: 380-388.

[19] Johnson W B, Lindenstrauss J. Extensions of Lipschitz mappings into a Hilbert space[J]. Contemporary mathematics, 1984, 26(189-206): 1.

[20] Bingham E, Mannila H. Random projection in dimensionality reduction: applications to image and text data[C]//Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2001: 245-250.

[21] Broder A Z, Glassman S C, Manasse M S, et al. Syntactic clustering of the web[J]. Computer Networks and ISDN Systems, 1997, 29(8): 1157-1166.

[22] Broder A Z. On the resemblance and containment of documents[C]// Compression and Complexity of Sequences 1997. Proceedings. IEEE, 1997: 21-29.

[23] Dasgupta A, Kumar R, Sarlós T. Fast locality-sensitive hashing[C]// Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011: 1073-1081.

[24] Datar M, Immorlica N, Indyk P, et al. Locality-sensitive hashing scheme based on p-stable distributions[C]//Proceedings of the twentieth annual symposium on Computational geometry. ACM, 2004: 253-262.

[25] Motwani R, Naor A, Panigrahy R. Lower bounds on locality sensitive hashing[J]. SIAM Journal on Discrete Mathematics, 2007, 21(4): 930-935.

[26] O'Donnell R, Wu Y, Zhou Y. Optimal lower bounds for locality-sensitive hashing (except when q is tiny)[J]. ACM Transactions on Computation Theory (TOCT), 2014, 6(1): 5.

[27] Panigrahy R. Entropy based nearest neighbor search in high dimensions[C]// Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm. Society for Industrial and Applied Mathematics, 2006: 1186-1195.

[28] Gan J, Feng J, Fang Q, et al. Locality-sensitive hashing scheme based on dynamic collision counting[C]// Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. ACM, 2012: 541-552.

[29] Lv Q, Josephson W, Wang Z, et al. Multi-probe LSH: efficient indexing for high-dimensional similarity search[C]//Proceedings of the 33rd international conference on Very large data bases. VLDB Endowment, 2007: 950-961.

[30] Li P, Church K W, Hastie T J. Conditional random sampling: A sketch-based sampling technique for sparse data[C]//Advances in neural information processing systems. 2006: 873-880.

[31] Ji J, Li J, Yan S, et al. Min-max hash for jaccard similarity[C]//2013 IEEE 13th International Conference on Data Mining. IEEE, 2013: 301-309.

[32] Ji J, Li J, Yan S, et al. Super-bit locality-sensitive hashing[C]//Advances in Neural Information Processing Systems. 2012: 108-116.

[33] Li P, Owen A, Zhang C H. One permutation hashing[C]//Advances in Neural Information Processing Systems. 2012: 3113-3121.

[34] Li P, König C. b-Bit minwise hashing[C]//Proceedings of the 19th international conference on World wide web. ACM, 2010: 671-680.

[35] Li P, Konig A, Gui W. b-Bit minwise hashing for estimating three-way similarities[C]// Advances in Neural Information Processing Systems. 2010: 1387-1395.

[36] Li P, Hastie T J, Church K W. Very sparse random projections[C]//Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2006: 287-296.

[37] Shrivastava A, Li P. Densifying One Permutation Hashing via Rotation for Fast Near Neighbor Search[C]//ICML. 2014: 557-565.

[38] Jegou H, Douze M, Schmid C. Product quantization for nearest neighbor search[J]. IEEE transactions on pattern analysis and machine intelligence, 2011, 33(1): 117-128.

[39] Liu W, Wang J, Kumar S, et al. Hashing with graphs[C]//Proceedings of the 28th international conference on machine learning (ICML-11). 2011: 1-8.

[40] Kong W, Li W J. Isotropic hashing[C]//Advances in Neural Information Processing Systems. 2012: 1646-1654.

[41] Jiang Q Y, Li W J. Scalable graph hashing with feature transformation[C]// Proceedings of IJCAI. 2015.

[42] Norouzi M, Blei D M. Minimal loss hashing for compact binary codes[C]// Proceedings of the 28th international conference on machine learning (ICML-11). 2011: 353-360.

[43] Norouzi M, Fleet D J, Salakhutdinov R R. Hamming distance metric learning[C]// Advances in neural information processing systems. 2012: 1061-1069.

[44] Rastegari M, Farhadi A, Forsyth D. Attribute discovery via predictable discriminative binary codes[C]//European Conference on Computer Vision. Springer Berlin Heidelberg, 2012: 876-889.

[45] Wang J, Kumar S, Chang S F. Semi-supervised hashing for scalable image retrieval[C]// Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010: 3424-3431.

[46] Shen F, Shen C, Liu W, et al. Supervised discrete hashing[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 37-45.

[47] Lin G, Shen C, Suter D, et al. A general two-step approach to learning-based hashing[C]//Proceedings of the IEEE International Conference on Computer Vision. 2013: 2552-2559.

[48] Lin G, Shen C, Shi Q, et al. Fast supervised hashing with decision trees for high-dimensional data[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014: 1963-1970.

[49] Zhang P, Zhang W, Li W J, et al. Supervised hashing with latent factor models[C]//Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. ACM, 2014: 173-182.

[50] Kang W C, Li W J, Zhou Z H. Column Sampling based Discrete Supervised Hashing[C]//Thirtieth AAAI Conference on Artificial Intelligence. 2016.

[51] Wang J, Liu W, Sun A X, et al. Learning hash codes with listwise supervision[C]//Proceedings of the IEEE International Conference on Computer Vision. 2013: 3032-3039.

[52] Li X, Lin G, Shen C, et al. Learning Hash Functions Using Column Generation[C]//ICML (1). 2013: 142-150.

[53] Wang Q, Zhang Z, Si L. Ranking preserving hashing for fast similarity search[C]//Proc. Int. Joint Conf. Artif. Intelli. 2015.

[54] Wang J, Wang J, Yu N, et al. Order preserving hashing for approximate nearest neighbor search[C]//Proceedings of the 21st ACM international conference on Multimedia. ACM, 2013: 133-142.

[55] Ge T, He K, Sun J. Graph cuts for supervised binary coding[C]//European Conference on Computer Vision. Springer International Publishing, 2014: 250-264.

[56] Liu W, Mu C, Kumar S, et al. Discrete graph hashing[C]//Advances in Neural Information Processing Systems. 2014: 3419-3427.

[57] McCulloch W S, Pitts W. A logical calculus of the ideas immanent in nervous activity[J]. The bulletin of mathematical biophysics, 1943, 5(4): 115-133.

[58] Rosenblatt F. The perceptron, a perceiving and recognizing automaton Project Para[M]. Cornell Aeronautical Laboratory, 1957.

[59] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural computation, 2006, 18(7): 1527-1554.

[60] LeCun Y, Boser B, Denker J S, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural computation, 1989, 1(4): 541-551.

[61] Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. Cognitive modeling, 1988, 5(3): 1.

[62] Hubel D H, Wiesel T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex[J]. The Journal of physiology, 1962, 160(1): 106-154.

[63] Fukushima K, Miyake S, Ito T. Neocognitron: A neural network model for a mechanism of visual pattern recognition[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1983 (5): 826-834.

[64] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[65] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[66] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks[C]//European Conference on Computer Vision. Springer International Publishing, 2014: 818-833.

[67] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[68] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern

Recognition. 2015: 1-9.

[69] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[J]. arXiv preprint arXiv:1512.03385, 2015.

[70] Salakhutdinov R, Hinton G. Semantic hashing[J]. International Journal of Approximate Reasoning, 2009, 50(7): 969-978.

[71] Xia R, Pan Y, Lai H, et al. Supervised Hashing for Image Retrieval via Image Representation Learning[C]//AAAI. 2014, 1: 2.

[72] Lai H, Pan Y, Liu Y, et al. Simultaneous feature learning and hash coding with deep neural networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 3270-3278.

[73] Lin K, Yang H F, Hsiao J H, et al. Deep learning of binary hash codes for fast image retrieval[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2015: 27-35.

[74] Liu H, Wang R, Shan S, et al. Deep Supervised Hashing for Fast Image Retrieval[J].

[75] Zhu H, Long M, Wang J, et al. Deep Hashing Network for Efficient Similarity Retrieval[C]//Thirtieth AAAI Conference on Artificial Intelligence. 2016.

[76] Li W J, Wang S, Kang W C. Feature learning based deep supervised hashing with pairwise labels[J]. arXiv preprint arXiv:1511.03855, 2015.

[77] Neyshabur B, Srebro N, Salakhutdinov R R, et al. The power of asymmetry in binary hashing[C]//Advances in Neural Information Processing Systems. 2013: 2823-2831.

[78] Ji J, Yan S, Li J, et al. Batch-orthogonal locality-sensitive hashing for angular similarity[J]. IEEE transactions on pattern analysis and machine intelligence, 2014, 36(10): 1963-1974.

[79] Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional architecture for fast feature embedding[C]//Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014: 675-678.

[80] Hansen L K, Salamon P. Neural network ensembles[J]. IEEE transactions on pattern analysis and machine intelligence, 1990, 12: 993-1001.

[81] LeCun Y, Cortes C, Burges C J C. The MNIST database of handwritten digits, 1998[J]. Available electronically at http://yann. lecun. com/exdb/mnist, 2012.

[82] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[J]. 2009.

[83] Chua T S, Tang J, Hong R, et al. NUS-WIDE: a real-world web image database from National University of Singapore[C]//Proceedings of the ACM international conference on image and video retrieval. ACM, 2009: 48.

攻读硕士学位期间主要的研究成果

[1] Wu S, Ren W, Yu C, et al. Personal recommendation using deep recurrent neural networks in NetEase[C]// IEEE, International Conference on Data Engineering. IEEE Computer Society, 2016:1218-1229.

致谢

从 2010 年 9 月来到浙江大学，六年半的青春时光在求是园中飞逝而过，一路上有挫折与打击，也有收获与欢笑，其间丰富多彩的点点滴滴至今我都历历在目。我要感谢母校浙江大学以及计算机学院为我提供一流的科研条件和学习环境，培养教育我成为更加成熟优秀的浙大学子。求是创新的精神也将在未来的人生路上引领我不断前进，提醒我始终以更高的标准要求自己。

本论文是在实验室多位老师的指导和帮助下完成的，我要感谢实验室的陈刚老师，伍赛老师，陈珂老师，寿黎但老师，你们专业知识渊博，治学态度严谨，对我耐心指导，循循善诱，教会了我许多思考解决问题的方式方法，激励我在学术上不断追求卓越，树立远大的学术理想；而在生活上，你们平易近人，低调朴实，给予了我许多关心和鼓励，也让我懂得了很多为人处世的道理。

另外还要感谢同一实验室的刘伟，金明建，钱宇，周俊林，王伟迪，王改革，李邦鹏，张也，朱华，胡凡，朱清华，吴连坤，于志超，冯杰等各位同学，大家平时在实验室中朝夕相伴，互相合作，在课余时间也互相支持，团结友爱，使得实验室中的学习生活充满了乐趣，还应当特别感谢俞骋超，彭湃，陈鸿翔，李环，骆歆远，陈伟等多位师兄，你们为我的学习和生活各方面提供许多帮助和建议，帮助我排忧解难，度过难关，更快融入实验室的工作当中，顺利地完成自己的科研和学习目标。

最后更要感谢家人和朋友的支持，感谢 Chiong 同学与我相伴校园生活，让我的生活中充满欢笑与快乐，也帮助我度过一个个难关；感谢父母在我的成长路上对我始终如一无条件地支持，是你们的关心和爱护帮助我一次次克服困难勇攀高峰，你们始终是我前进的力量源泉。

任伟超

2017 年 1 月 5 日